NeoNexus

ANÁLISIS DE SENTIEMIENTO EN REDES SOCIALES CON SPARK NLP

Mariam Razmadze

Angélica Simarra

Felipe Gallegos

Paola León

Ariel Graverán

Contenido

Origen e Importancia del Proyecto

En la era digital, las redes sociales se han convertido en una fuente primaria de información sobre la percepción del público. Twitter, en particular, es una plataforma donde los usuarios expresan opiniones en tiempo real, lo que lo convierte en un recurso valioso para estudios de opinión.

Muchas empresas y organizaciones enfrentan un desafío crucial: comprender realmente lo que las personas piensan y sienten sobre su marca o productos. Aunque tienen acceso a grandes volúmenes de datos en redes sociales, foros y reseñas en línea, la falta de herramientas adecuadas para interpretar esta información de manera efectiva puede llevar a decisiones basadas en suposiciones o a perder oportunidades clave de conexión con su audiencia.

Este proyecto nace de la necesidad de transformar esos datos en conocimiento accionable. Creemos que el análisis de sentimientos es una herramienta poderosa para anticipar las necesidades de los clientes, ajustar estrategias y fortalecer la relación entre las marcas y su público. No se trata solo de tecnología, sino de cómo esta puede ayudar a las empresas a ser más humanas, responder de manera empática y tomar decisiones más acertadas.

Además, el análisis de sentimientos puede ser aplicado en otros ámbitos como la política, la investigación académica y el servicio al cliente, permitiendo detectar tendencias de opinión pública, medir el impacto de campañas y mejorar la relación con los usuarios de una marca o servicio.

Descripción del Proyecto- Visión general

El presente proyecto consiste en una plataforma de análisis de sentimientos aplicada a redes sociales, con un enfoque específico en Twitter. Su objetivo principal es permitir a los usuarios conocer la opinión del mercado respecto a un producto o tema determinado, proporcionando un resumen analítico basado en comentarios reales.

Dada la creciente necesidad de comprender la percepción del público en canales digitales, se ha desarrollado un sistema capaz de analizar grandes volúmenes de texto, identificar opiniones positivas o negativas y generar métricas accionables para la toma de decisiones en marketing y relaciones públicas. La solución está diseñada para analizar datos en tiempo real, permitiendo la detección de patrones y tendencias emergentes.

Beneficios y Aplicaciones

El sistema está diseñado con una arquitectura modular y flexible, permitiendo su integración con APIs de redes sociales, bases de datos y otras fuentes externas.

Con esta herramienta, las empresas pueden tomar decisiones basadas en datos, ajustar estrategias de marketing y mejorar la experiencia del cliente. La capacidad de escuchar y actuar en función de las emociones del público puede marcar la diferencia en un entorno competitivo.

Este proyecto no solo representa un avance tecnológico en el campo del análisis de sentimientos, sino que también refuerza la importancia de utilizar la inteligencia artificial para entender y conectar mejor con las personas. En el futuro, esta tecnología podría ampliarse a otras plataformas y mejorar con modelos más avanzados que permitan detectar matices emocionales más complejos y realizar análisis más profundos sobre el comportamiento de los usuarios.

Proceso de Extracción de Datos

El proceso de extracción de datos la dividimos en varios apartados:

a. Investigación inicial sobre las fuentes de datos

   El primer paso fue identificar qué redes sociales serían más relevantes para la extracción de información. Nos enfocamos en plataformas donde los usuarios suelen expresar sus opiniones y emociones de manera abierta. Después de analizar varias opciones, elegimos Twitter como nuestra fuente principal de datos debido a su popularidad y la facilidad para acceder a información pública. Twitter es una plataforma ampliamente utilizada para compartir opiniones en tiempo real, lo que la convierte en una fuente ideal para nuestro análisis de sentimientos.

b. Selección de herramientas para la extracción

   Una vez definida la fuente de datos, investigamos qué herramientas o APIs podíamos utilizar para realizar el scraping. Este paso fue particularmente desafiante, ya que el acceso a la API oficial de Twitter tiene un costo elevado, lo cual no era viable para nuestro proyecto, dado que se trata de una prueba de concepto.

   Después de evaluar varias alternativas, encontramos APIFY, una plataforma que ofrece scrapers preconfigurados para diferentes redes sociales, incluyendo Twitter. APIFY resultó ser la mejor opción porque nos permitió ahorrar tiempo en la configuración y nos ofreció una solución eficiente y económica para extraer los datos necesarios.

c. Definición de la empresa para la prueba de concepto

   El siguiente paso fue decidir qué empresa utilizaremos como caso de estudio para la prueba de concepto. Optamos por enfocarnos en empresas reconocidas a nivel mundial, ya que estas suelen generar un alto volumen de interacciones en Twitter. Después de analizar varias opciones, seleccionamos NVIDIA debido a los eventos significativos que ha experimentado en los últimos años, lo que nos aseguraba una gran cantidad de datos relevantes para analizar.

d. Implementación del scraping

   Realizamos un análisis relacionadas con NVIDIA. Procedimos a implementar el scraping utilizando la librería apify-client de APIFY. Este proceso incluyó:

   - Configuración del scraper de Twitter: Usamos el scraper preconfigurado de APIFY para extraer tweets relacionados con NVIDIA en las fechas seleccionadas.

Configuramos parámetros como palabras clave, hashtags y rangos de fechas para obtener datos relevantes.

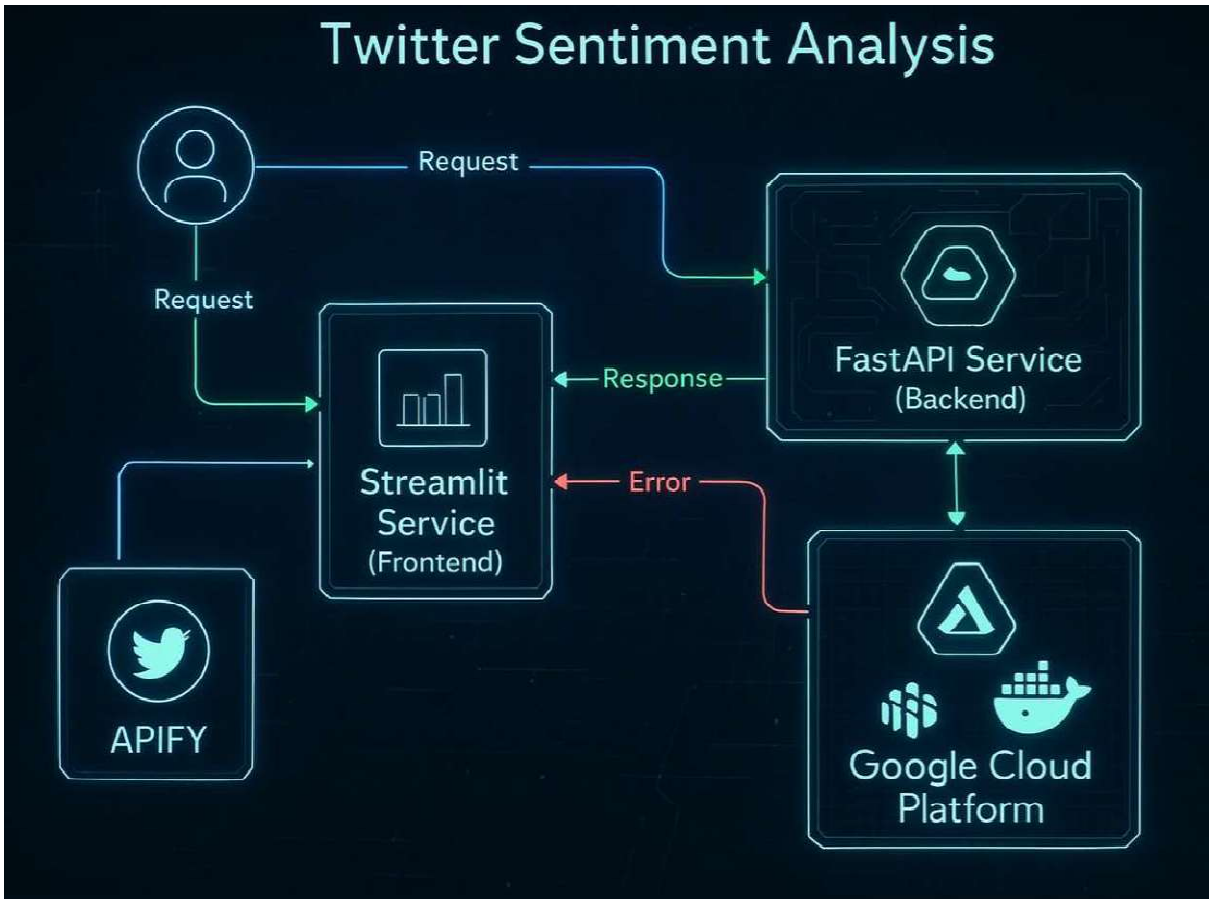
- Extracción de datos: Ejecutamos el scraper para recopilar tweets que mencionan a NVIDIA o estuvieran relacionados con los eventos clave. Los datos extraídos incluyeron texto de los tweets, fechas, autores y métricas como retweets y likes.
- Almacenamiento de los datos: Organizamos los datos extraídos en un formato estructurado (CSV) para facilitar su análisis posterior.

e. Organización del proyecto

Finalmente, integramos todo el proceso en el proyecto, documentando cada paso y asegurándonos de que el script de scraping fuera reutilizable para futuras extracciones.

Esto nos permitió tener una base sólida para realizar el análisis de sentimientos y presentar resultados claros y organizados.

Los componentes principales



- Frontend (Streamlit): Proporciona una interfaz de usuario interactiva para buscar tweets, configurar parámetros y visualizar resultados.
    - Tecnologías: Streamlit.
    - Características:
        - Formulario para ingresar términos de búsqueda y filtros.
        - Visualización de resultados en gráficos y tablas.
        - Descarga de resultados en formato JSON.
- Backend (Scraper y Procesamiento de Datos): Extrae tweets desde Twitter utilizando Apify, procesa los datos y realiza análisis de sentimientos.
- Infraestructura (Pulumi y Google Cloud Platform): Gestiona la infraestructura como código y despliega los servicios en la nube.
    - Tecnologías: Pulumi, Google Cloud Platform (GCP), Docker.

- Características:
  - Despliegue de servicios en Cloud Run.
  - Almacenamiento de imágenes Docker en Artifact Registry.
  - Configuración de permisos IAM y manejo seguro de credenciales.
  - Arquitectura del modelo : Realiza el análisis de sentimientos utilizando representaciones de texto como BERT y Word2Vec.

- Tecnologías: PyTorch, Spark NLP.

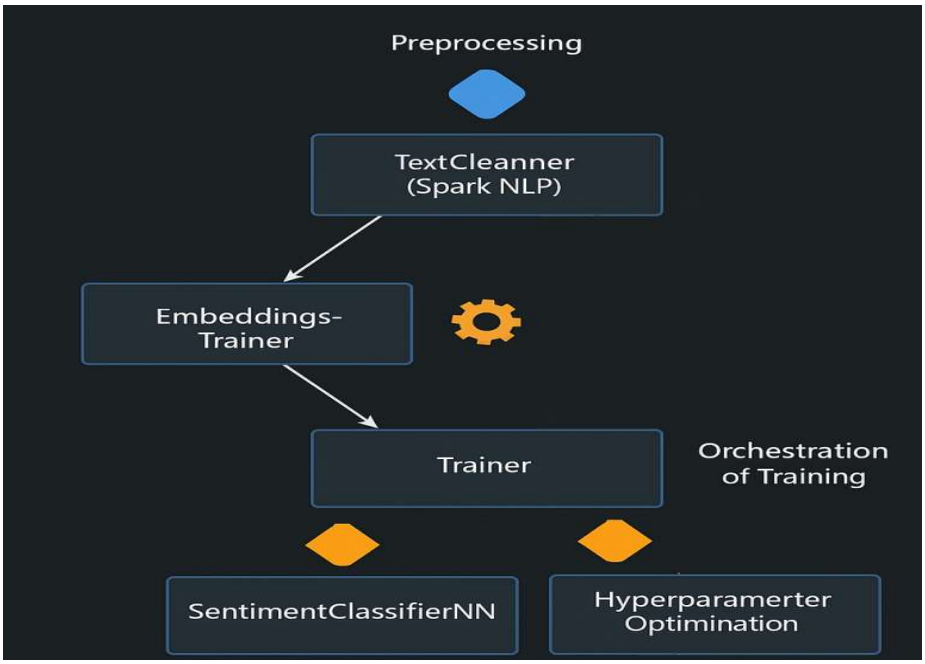
  - Características:
    - Entrenamiento del modelo con datos preprocesados.
    - Optimización de hiper parámetros con Optuna.
    - Inferencia para clasificar sentimientos.

Implementación y desarrollo del modelo

Desde un punto de vista técnico, el proyecto utiliza Procesamiento del Lenguaje Natural (NLP) y Aprendizaje Automático (Machine Learning) para clasificar y analizar sentimientos en textos. Se apoya en Apache Spark NLP, que proporciona una arquitectura distribuida eficiente para el procesamiento de grandes volúmenes de datos. Para el entrenamiento del sistema de clasificación binaria, se utilizó el dataset Sentiment140, que contiene aproximadamente 1.6 millones de registros.

Uno de los principales retos fue la carga computacional asociada al procesamiento de los embeddings. Para optimizar el rendimiento y evitar desbordamientos de memoria, se optó por un enfoque de procesamiento por lotes, dividiendo los datos en fragmentos de 200,000 registros. Esto permitió una ejecución escalable y controlada del flujo de procesamiento. Google Colab fue elegido como entorno de desarrollo debido a su accesibilidad, soporte de GPU y flexibilidad para experimentación iterativa.

Adicionalmente, se implementaron técnicas de limpieza y normalización del texto, como la eliminación de stopwords, tokenización y lematización, con el fin de mejorar la calidad de los datos procesados y obtener un análisis más preciso.

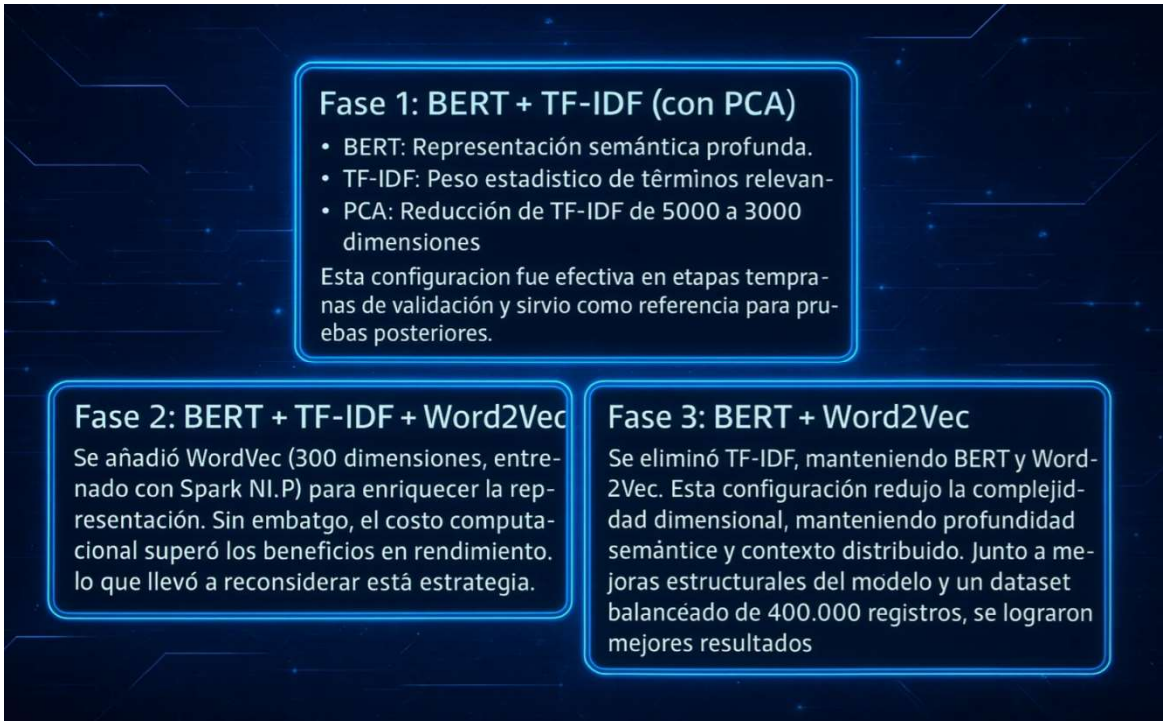


Arquitectura técnica:

- TextCleaner (Spark NLP)**:** Realiza la limpieza y normalización del texto de entrada.
- EmbeddingsTrainer**:** Calcula los embeddings con BERT y Word2Vec, preparando los datos para el modelo.
- SentimentClassifierNN: Red neuronal profunda que fusiona ambos vectores y ejecuta la clasificación binaria.
- SparkNLPProcessor**:** Inicializa Spark con configuraciones optimizadas y gestiona la carga del dataset.
- Trainer**:** Entrena el modelo en PyTorch, implementando estrategias como *Early Stopping*.
- HyperparameterOptimization (Optuna): Optimiza los hiperparámetros para mejorar el rendimiento del modelo.

Combinaciones evaluadas y evolución metodológica

Durante el desarrollo, se exploraron distintas combinaciones de representaciones vectoriales con el objetivo de maximizar el rendimiento sin comprometer la eficiencia computacional.



Fase 1: BERT + TF-IDF (con PCA)

- BERT: Representación semántica profunda.
- TF-IDF: Asigna pesos estadísticos a términos relevantes.
- PCA: Reduce la dimensionalidad de TF-IDF de 5000 a 3000 dimensiones.

Esta configuración resultó efectiva en las primeras etapas de validación y sirvió como base para pruebas posteriores.

Fase 2: BERT + TF-IDF + Word2Vec

Se añadió Word2Vec (300 dimensiones, entrenado con Spark NLP) para enriquecer la representación. Sin embargo, el costo computacional superó los beneficios en rendimiento, lo que llevó a reconsiderar esta estrategia.

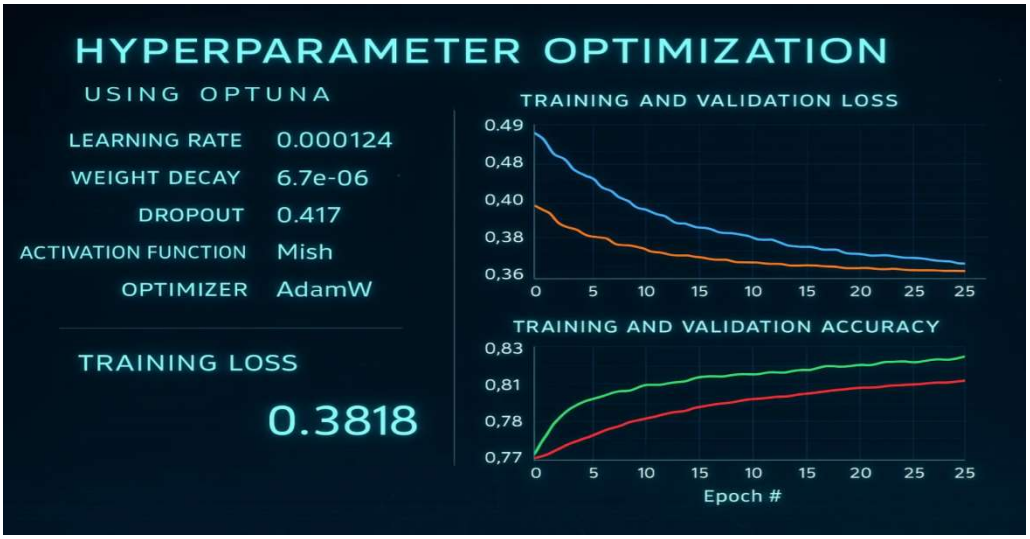
Fase 3: BERT + Word2Vec

Se eliminó TF-IDF, manteniendo únicamente BERT y Word2Vec**.** Esta configuración redujo la complejidad dimensional, conservando la profundidad semántica y el contexto distribuido. Con mejoras en la estructura del modelo y un dataset balanceado de 400.000 registros, se lograron resultados superiores.

Adicionalmente, se evaluaron modelos de *Machine Learning* clásicos, como Regresión Logística (LR), LightGBM y XGBoost**.** Sin embargo, el desempeño de estos modelos no superó al del enfoque basado en *Deep Learning*. En particular, XGBoost obtuvo un F1-score de 0.7776**,** quedando por debajo del modelo neuronal, lo que reafirma la elección de este último como solución principal.

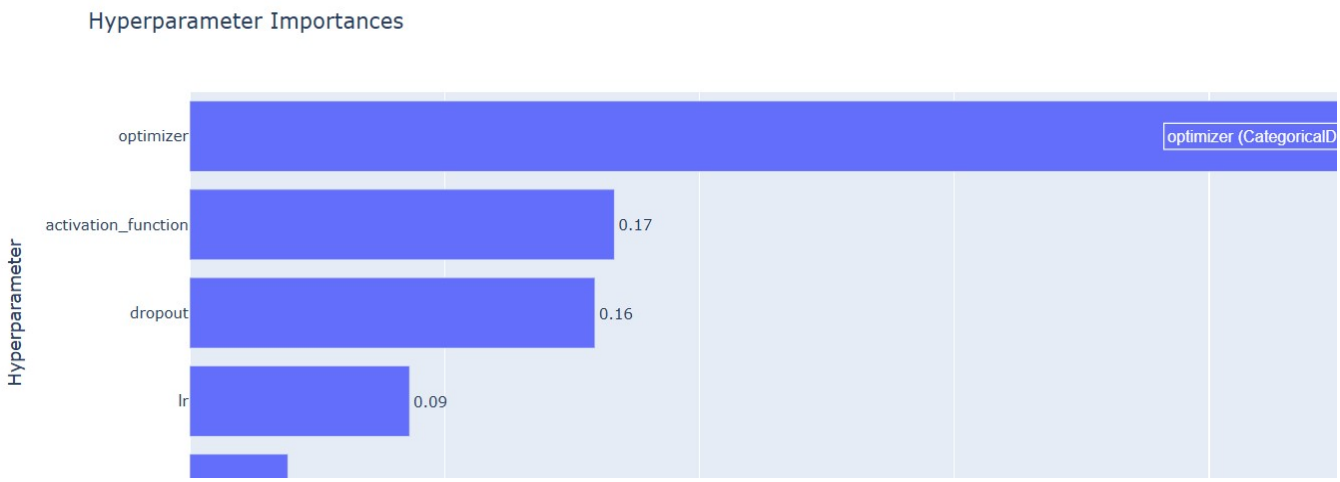
Optimización de hiperparámetros con Optuna

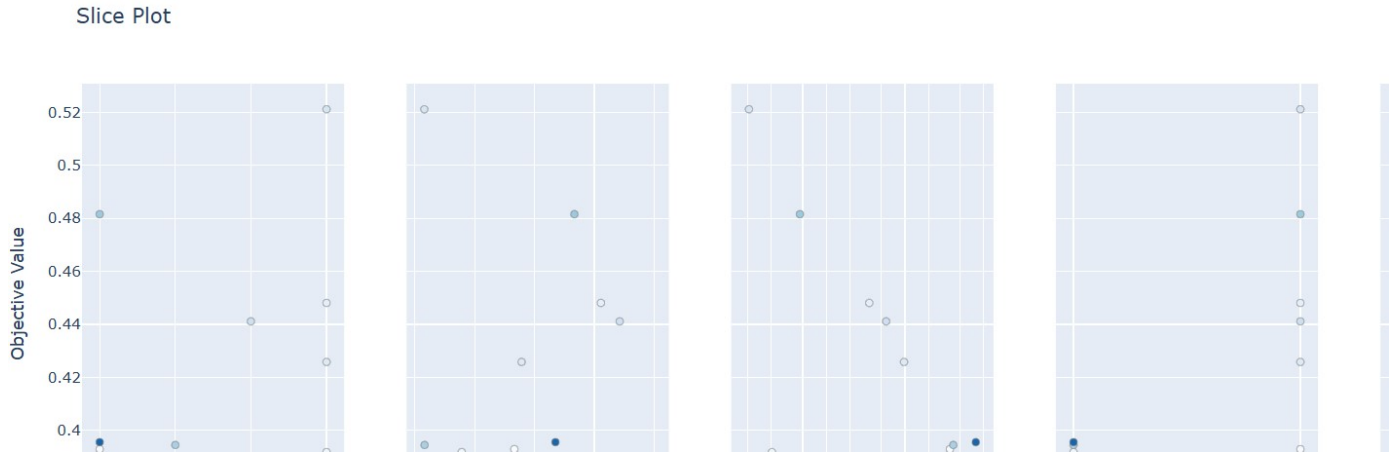
Optuna identificó la mejor combinación de hiperparámetros para mejorar el rendimientoEsta configuración, en conjunto con los ajustes estructurales y el entrenamiento con el dataset balanceado, ofreció mejoras en estabilidad y rendimiento**.**



Pérdida en entrenamiento: 0.3818

Como el estudio de train se registro con optuna, pudimos evaluar los mejores espacios de valores para los hyperparametros y sus pesos:





Reentrenamiento con dataset ampliado y mejoras arquitectónicas

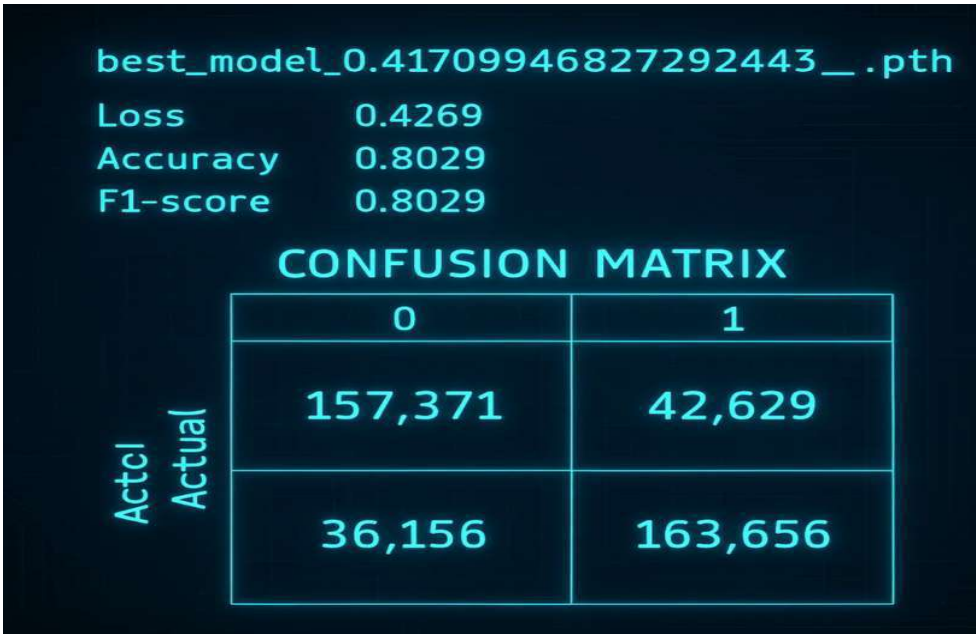
Dado que el modelo mostraba un F1-score alto pero artificialmente inflado, se identificó un problema de desbalance en la clase 1. Se tomaron las siguientes acciones:

- Ampliación del dataset a 400.000 registros balanceados.
- Pruebas con dos enfoques:
    - Congelar los embeddings (BERT y Word2Vec) y entrenar solo la capa combinadora.
    - Ajustar la arquitectura, añadiendo capas intermedias sin congelar pesos.

Los resultados mostraron una mejora en recall para ambas clases**,** lo que indica una mejor comprensión del contexto y reducción del sesgo del modelo.

MODEL NAME	F1-SCORE	ACCURACY	RECALL CLASS 0	LASL 1	STRUCTURE
best_model_0.417099468279443_pth	0,8029	0,8029	0,7869	0,8195	Sentiment
best_model_0.417053810332437_pth	0,8029	0,8029	0,7873	0,8246	Sentiment
best_model_0.417973405104036_pth	0,8024	0,8028	0,7807	0,8245	Sentiment
best_model_0.419087691088377_pth	0,8024	0,8028	0,7802	0,8706	Sentiment
best_model_0.417270347679399_pth	0,8023	0,8028	0,7818	0,8244	Sentiment
best_model_0.416872384039581_pth	0,8023	0,8028	0,7804	0,8235	Sentiment
best_model_0.419353783763525_pth	0,8023	0,8028	0,7810	0,8710	Sentiment
best_model_0.417076307020645_pth	0,8022	0,8023	0,7902	0,8233	Sentiment

El mejor modelo presentaba esta matriz de confusión:



Ensamblado de modelos y estrategia de Stacking

Dado que los modelos individuales mostraban diferentes niveles de rendimiento, se implementó un esquema de stacking, donde un meta-modelo combina las predicciones de los 5 mejores modelos base.

Modelos seleccionados para stacking:

Regresión Logística (Stacking):

F1-score: 08025, Accuracy: 0.8025 y la Matriz de Confusión: [[121173 28827] [30403 119456]]

XGBoost (Stacking):

F1-score**:** 0.8043 y Accuracy: 0.8043

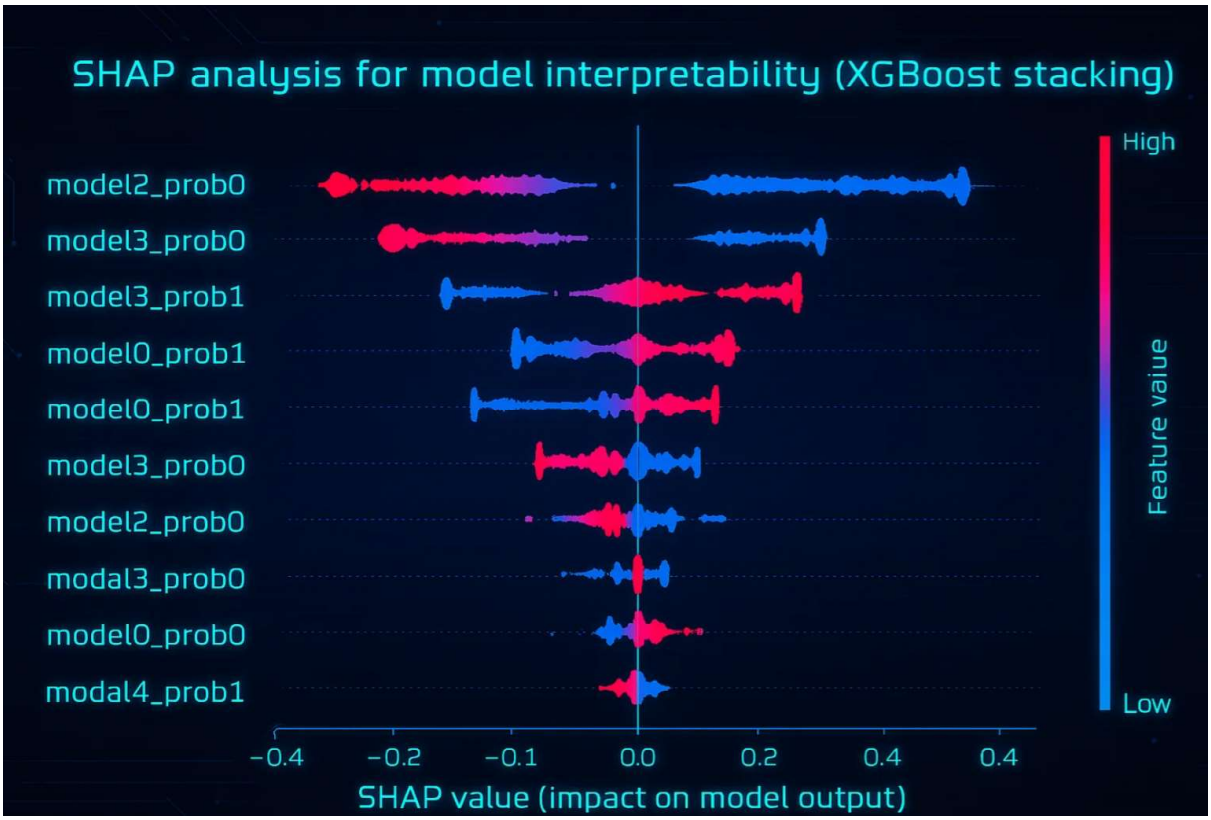
XGBoost ofreció una leve mejora gracias a su capacidad para capturar interacciones no lineales y ponderar de forma más flexible las predicciones de cada modelo base.

Ambos modelos demostraron consistencia y estabilidad, consolidando la estrategia de *stacking* como la solución más efectiva para la clasificación.

Interpretabilidad del modelo: análisis con SHAP

Para entender la importancia de cada modelo dentro del *stacking*, se aplicó SHAP (SHapley Additive exPlanations) sobre el clasificador XGBoost**.**

El análisis reveló que algunos modelos base aportaban poco valor predictivo**,** sugiriendo la necesidad de un proceso de selección de expertos más refinado**.**



Como se aprecia en la imagen el model2 tenía un peso significativo en la predicción, mientras que models 4 y 1 podrían ser reevaluados debido a su baja contribución.

Selección optimizada de expertos con estrategia Greedy

Se refinó el ensamblado de modelos con un método greedy**,** donde solo se añadían expertos que mejoraran el F1-score**.** Para esto, se implementaron:

- GreedyExpertSelectorXGB (XGBoost como meta-modelo).
- GreedyExpertSelector (Regresión Logística como meta-modelo).

Cada experto se representaba mediante dos features (softmax de clase 0 y 1)**.** El meta-modelo se entrenaba iterativamente y solo se añadían expertos que mejoraran la métrica final

Resultados finales

Tras la optimización del mejor modelo conseguimos los siguientes resultados:

Ambos enfoques lograron rendimiento competitivo, con XGBoost ligeramente superior. No obstante, la Regresión Logística es más interpretable y menos costosa computacionalmente, lo que la convierte en una alternativa viable en ciertos escenarios.

Después, elaboramos un dataset de 500 registros procesando los tweets de forma manual. Realizamos la última prueba del modelo, obteniendo un F1-score de 0.89. El resultado es mejorable en lo que respecta a la clasificación de la parte negativa.

Conclusión y futuras mejoras

Este proyecto ha demostrado ser una solución eficiente y escalable para el análisis de sentimientos en redes sociales, especialmente en plataformas como Twitter. La capacidad de procesar grandes volúmenes de datos en tiempo real utilizando Big Data, Machine Learning y Cloud Computing permite obtener resultados rápidos y precisos, lo que es esencial para empresas y analistas que buscan comprender la percepción pública de manera inmediata. El uso de modelos como XGBoost y técnicas de embeddings como TF-IDF ha sido clave para garantizar la calidad y relevancia del análisis.

Sin embargo, se han identificado áreas clave para futuras mejoras. En primer lugar, una de las posibles optimizaciones sería la inclusión de un análisis de sentimiento neutral, además de los tradicionales sentimientos positivos y negativos, ya que no todos los tweets tienen una polaridad clara, lo que puede dificultar la clasificación en este tipo de datos. Los tweets, a diferencia de las reseñas, no siempre contienen una expresión definida de satisfacción o insatisfacción, lo que puede dificultar la identificación clara de una polaridad positiva o negativa.

Para futuras mejoras, se propone explorar modelos adicionales que fortalezcan el sistema de expertos, optimizar el espacio de búsqueda de hiperparámetros en XGBoost y evaluar embeddings TF-IDF con reducción de dimensionalidad t-SNE en hardware más potente. Además, la incorporación de métricas adicionales como ROC-AUC y Precision-Recall curve podría mejorar la validación del modelo.

Otra posible mejora sería realizar un análisis de palabras frecuentes en los tweets clasificados como positivos y negativos. Esto permitiría identificar patrones lingüísticos característicos de cada categoría y refinar aún más el modelo. Con estos avances, el sistema podría proporcionar resultados más precisos y matizados, abordando mejor la complejidad del lenguaje en redes sociales.