Detección de rumores en tweets a partir de características estructurales y linguísticas de la red

Marian Aguilar Tavier

Enero 2026

Contents

List of Figures

List of Tables

Introducción

Hoy en día, la difusión de información ocurre en cuestión de segundos y, muchas veces, es imposible distinguir qué información es realmente cierta. La propagación de rumores y noticias falsas genera confusión, alarma y consecuencias sociales significativas, lo que hace fundamental contar con herramientas que ayuden a evaluar la veracidad de los mensajes. En este contexto, la inteligencia artificial juega un rol fundamental en la detección de desinformación en las redes sociales.

El objetivo de este proyecto es determinar si es posible identificar la desinformación únicamente a partir de las propiedades de la red y de los atributos lingüísticos presentes en los mensajes. Para ello, se evaluarán modelos de distinta naturaleza y se analizarán los factores más determinantes para esta clasificación, con el fin de explorar si las características estructurales y de estilo son suficientes para predecir la veracidad de la información.

Dataset utilizado

Para la detección de rumores en este caso se utilizó el dataset de Twitter **Pheme**.

Este dataset contiene tweets reales relacionados con una serie de acontecimientos ocurridos entre 2014 y 2015: Charlie Hebdo, Ferguson, Germanwings Crash, Ottawa Shooting, Sydney Siege, Ebola Essien, entre otros [Figura 1].

Tiene dos tipos de mensajes, el mensaje fuente (iniciador del posible rumor) y las reacciones (mensajes de respuesta de los usuarios al tweet fuente o a otra reacción)[Figura 2].

Este dataset facilita la obtención de características estructurales ya que proporciona información sobre el hilo completo del tweet inicial, cantidad de seguidores, número de retweets, favoritos y autenticidad de las cuentas, entre otros. Por otro lado, proporciona el texto del mensaje original del tweet, así como número de hashtags[Figura 3] y URLs[Figura 4], lo cual brinda información sobre el estilo del mismo.

Además los rumores aparecen etiquetados según su veracidad ("true", "false", "unverified")

(Ver otras visualizaciones en el apéndice de figuras)

Metodología

Exploración, limpieza y transformación

El dataset tiene un total de 95 columnas, por lo que se llevó a cabo un proceso de exploración para determinar las características que realmente tienen relevancia para la solución del problema planteado.

Se descartaron los features relacionados con imágenes, personalizaciones de las cuentas de los usuarios, así como columnas duplicadas, con todos sus valores faltantes o valores invariables. Luego las columnas que inicialmente se seleccionaron para el análisis se muestran en la tabla 3:

Es importante destacar que el dataset presenta una gran desproporción entre el número de rumores y no rumores dentro de los mensajes fuente, por lo que esto debe tenerse en cuenta a la hora de entrenar modelos para evitar sesgos en la clasificación[Figura 5].

Creación de nuevos features

Para la detección de rumores en este caso, se decidió no utilizar explícitamente el texto del mensaje, sino intentar clasificarlo solamente a partir de características de estilo y las interacciones en la red. Para ello se analizó la creación de nuevos features basado en el comportamiento de los datos observados y los citados en trabajos previos entre ellos la cantidad de palabras del texto[Figura], la presencia o no de menciones[Figura 7], la presencia de emojis, signos de exclamación[Figura 8] e interrogación.

Además se utilizó TextBlob para análisis de sentimiento en los tweets y comprobar si este factor tiene influencia o no en los rumores[Figura 9].

El análisis de sentimiento se realizó tanto en los tweets fuente como en las reacciones de los usuarios[Figura].

La lista de todos los features empleados en el dataset final aparecen en la tabla 4.

Transformación de datos

Como parte del proceso de transformación para el entrenamiento se convirtieron las variables categóricas a numéricas utilizando *label encoding*. Además se normalizaron las variables numéricas mediante la técnica de *StandardScaler*, lo que permitió que todas las características tuvieran una escala comparable.

Luego se dividió el dataset en conjunto de entrenamniento y de prueba utilizando el parámetro *stratify* durante la partición, garantizando que la distribución de las clases sea proporcional en ambos conjuntos.

Modelos utilizados

Para la detección de rumores y determinar los features que mejor lo predicen, en este caso, se aplicaron cuatro modelos: **Decision Tree, Random Forest,**

Regresión Logística y XGBoost.

Se seleccionaron estos modelos debido a su alto grado de interpretabilidad a la hora de determinar qué tan relevante es una determinada característica para el modelo. En cada uno de ellos se fijó una semilla (`random_seed = 42`) con el fin de garantizar la reproducibilidad de los resultados. Una vez entrenados los modelos se evaluó la importancia de las características: en los modelos basados en árboles (Decision Tree, Random Forest y XGBoost) se utilizó la medida de importancia derivada de la reducción de impureza o de la ganancia; mientras que en el caso de la Regresión Logística se analizaron los coeficientes del modelo como indicador de relevancia.

Para optimizar el rendimiento de cada modelo se aplicó *Randomized Search* sobre un espacio de hiperparámetros predefinido. Con el objetivo de evitar sesgos y asegurar una evaluación robusta, se empleó validación cruzada estratificada (*StratifiedKFold*), lo que garantiza que la proporción de clases se mantenga en cada partición.

Las métricas utilizadas para evaluar el desempeño fueron *accuracy*, *precision*, *recall* y *F1-score*, todas calculadas en su versión macro para evitar que la clase mayoritaria domine la evaluación. Para el ajuste de hiperparámetros se utilizó específicamente el *F1-score*. Esta métrica combina precisión y recall en una sola medida armónica, evitando que el modelo favorezca únicamente a la clase mayoritaria y promoviendo un equilibrio entre ambos indicadores.

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

Además, se calculó el área bajo la curva ROC (ROC-AUC) como métrica adicional de discriminación.

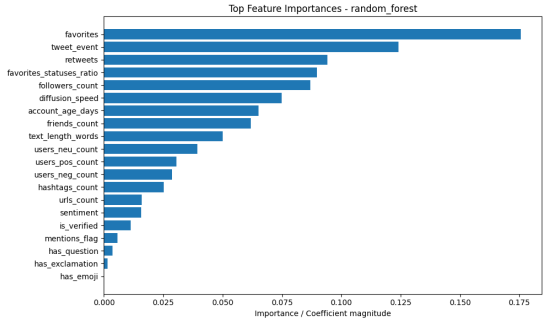
Resultados

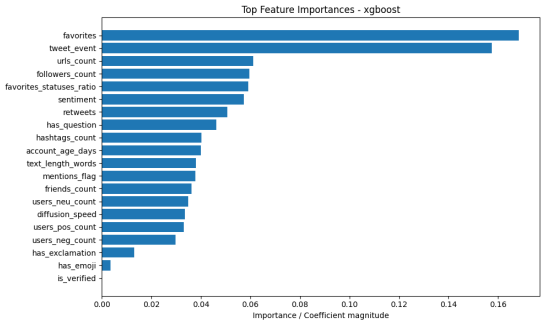
Para la búsqueda de los mejores hiperparámetros se realizaron un total de 30 iteraciones con 5 folds por cada uno, por lo que en total por cada modelo se entrenaron 150 instancias diferentes del modelo con distintos hiperparámetros, luego para cada modelo los mejores hiperparámetros son los siguientes:

Modelo	Best Score	Best Params
Decision Tree	0.6871	min_samples_split = 10 min_samples_leaf = 2 max_depth = 10 criterion = entropy
Logistic Regression	0.6487	solver = liblinear penalty = l1 max_iter = 1000 class_weight = balanced C = 10
Random Forest	0.7450	n_estimators = 300 min_samples_split = 10 min_samples_leaf = 4 max_depth = None class_weight = balanced
XGBoost	0.7400	subsample = 1.0 n_estimators = 300 max_depth = 3 learning_rate = 0.1 colsample_bytree = 1.0

Table 1: Mejores hiperparámetros y puntuaciones obtenidas mediante Randomized Search.

Luego, con respecto a los features que cada modelo considera importante, se ratifican los mencionados en la literatura por lo que se puede decir que las características estructurales y linguísticas que mejor permiten determinar la veracidad de un mensaje son: número de favoritos del tweet, evento asociado, cantidad de retweets, número de seguidores,la relación entre favoritos y cantidad de publicaciones, longitud del texto, velocidad de difusión, usuarios seguidos y sentimiento.





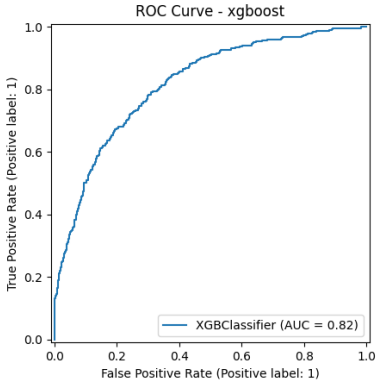
En cuanto al desempeño de los modelos, se utilizó la regresión logística como *baseline* para evaluar mejoras relativas. Los resultados muestran diferencias claras entre los algoritmos.

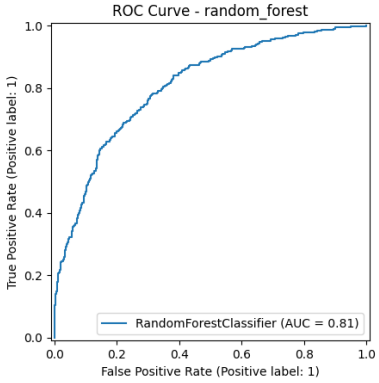
La regresión logística obtuvo el rendimiento más bajo, lo cual era esperable dada su naturaleza lineal y la complejidad no lineal del problema. El Decision Tree mejoró ligeramente estos resultados, aunque mantuvo un desempeño moderado. Por su parte, Random Forest presentó un salto significativo en todas las métricas macro, evidenciando la ventaja de los métodos de ensamble para este tipo de tareas.

Modelo	**Accuracy**	**Macro Precision**	**Macro Recall**	**Macro F1**
Decision Tree	0.6833	0.6643	0.6681	0.6658
Logistic Regression	0.6350	0.6309	0.6397	0.6274
Random Forest	0.7471	0.7298	0.7292	0.7295
XGBoost	**0.7572**	**0.7421**	**0.7293**	**0.7341**

Table 2: Comparación de desempeño entre modelos

Finalmente, el modelo con mejor rendimiento global fue XGBoost, que superó al resto en *accuracy*, *precision*, *recall* y *F1-score* macro. La diferencia con Random Forest fue sutil, pero consistente en todas las métricas.





Conclusiones

La realización de este proyecto demostró que el dataset utilizado contiene la información de interacciones sociales y linguísticas necesarias para la detección de rumores.

Los modelos empleados lograron distinguir la veracidad de los mensajes sin analizar directamente el contenido textual. Los modelos de mayor precisión fueron Random Forest y XGBoost, destacando como características importantes el número de retweets, favoritos, evento asociado, sentimiento, velocidad de difusión, cantidad de palabras del mensaje entre otros.

Cabe destacar que a pesar de los resultados obtenidos, estos pueden mejorarse si se tienen en cuenta el contenido semántico de los mensajes, como lo hacen algunos trabajos en la literatura.

Apéndice

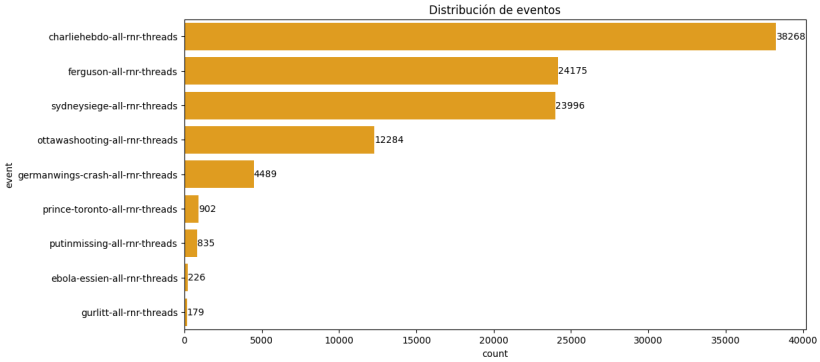


Figure 1: Tweets por eventos

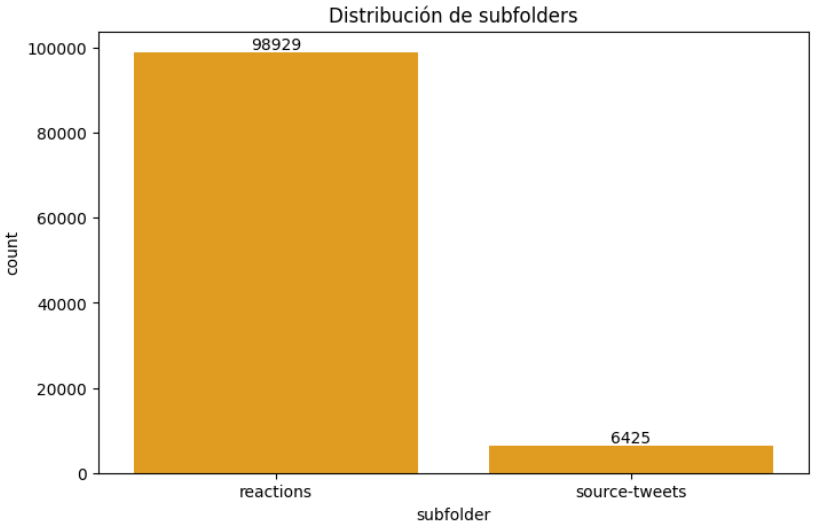


Figure 2: Tweets por tipo

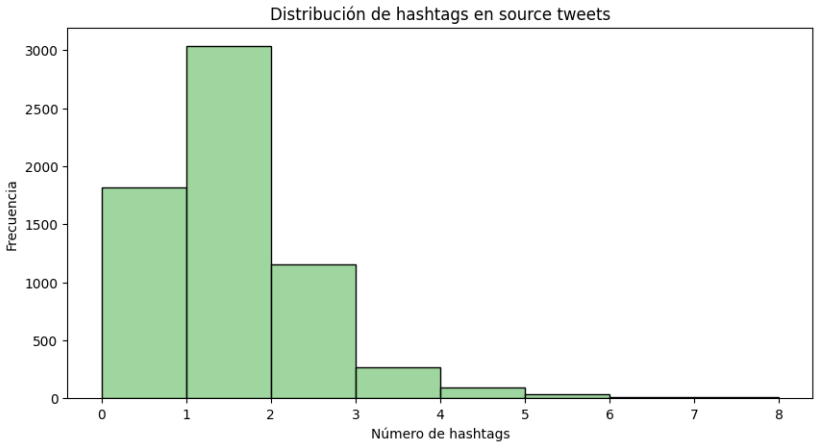


Figure 3: Análisis de hashtags en los tweets

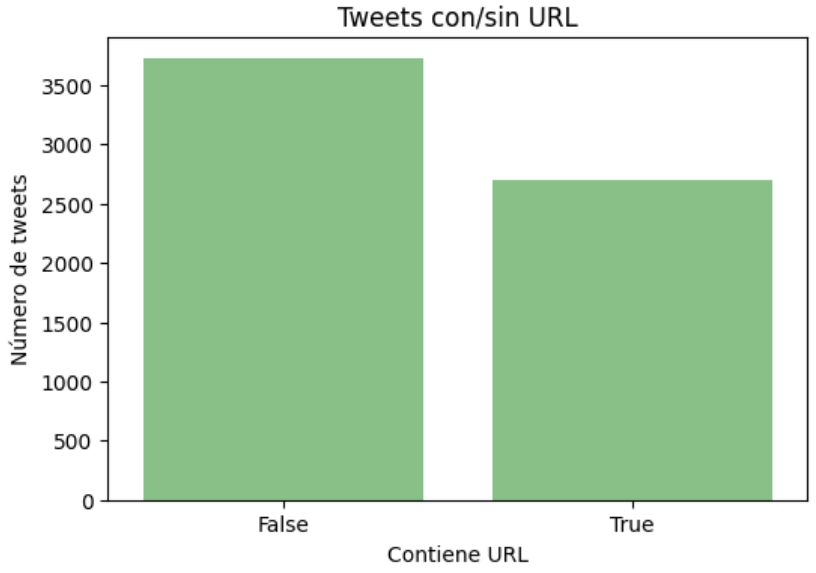


Figure 4: Análisis de urls en los tweets

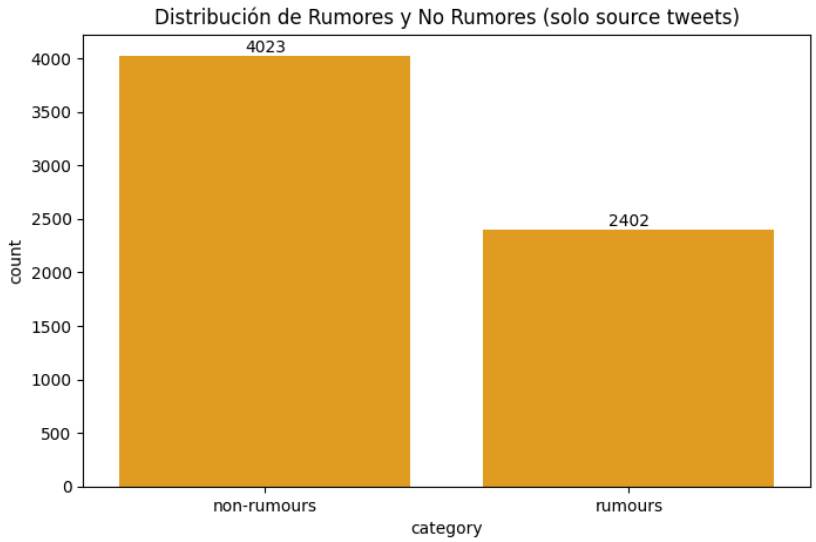


Figure 5: Distribución de rumores

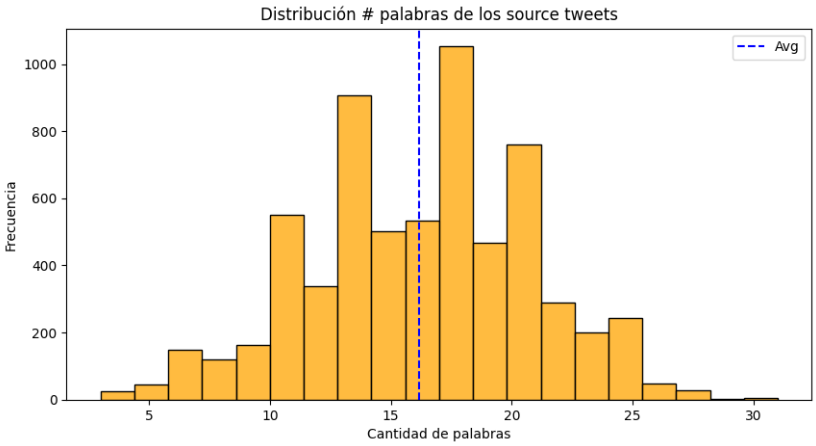


Figure 6: Textos según su longitud

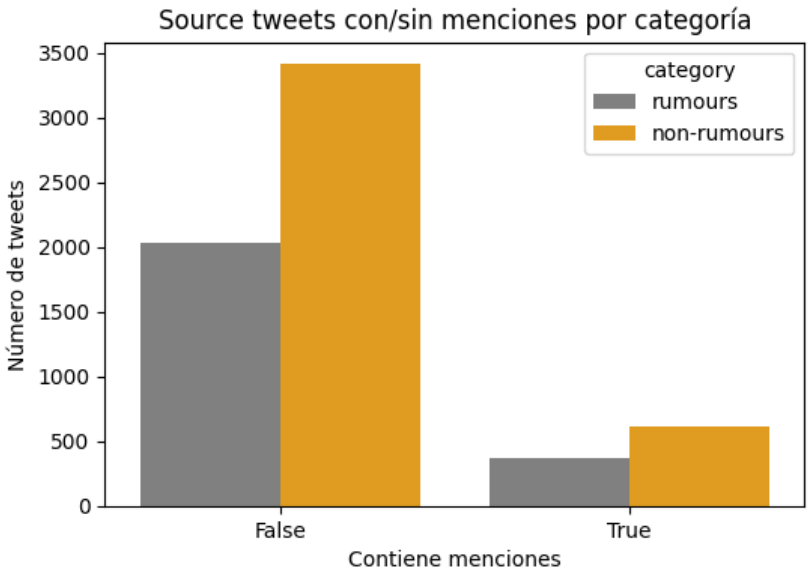


Figure 7: Menciones en los tweets

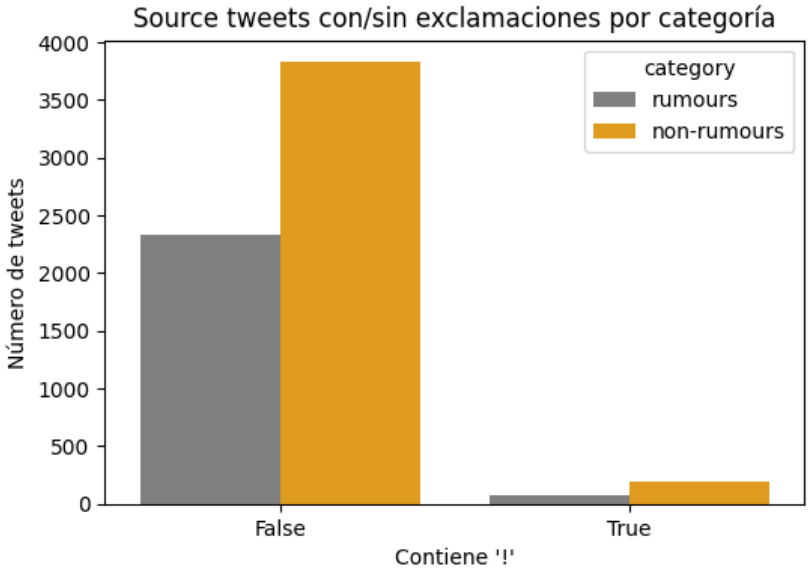


Figure 8: Mensajes con y sin signos de exclamación

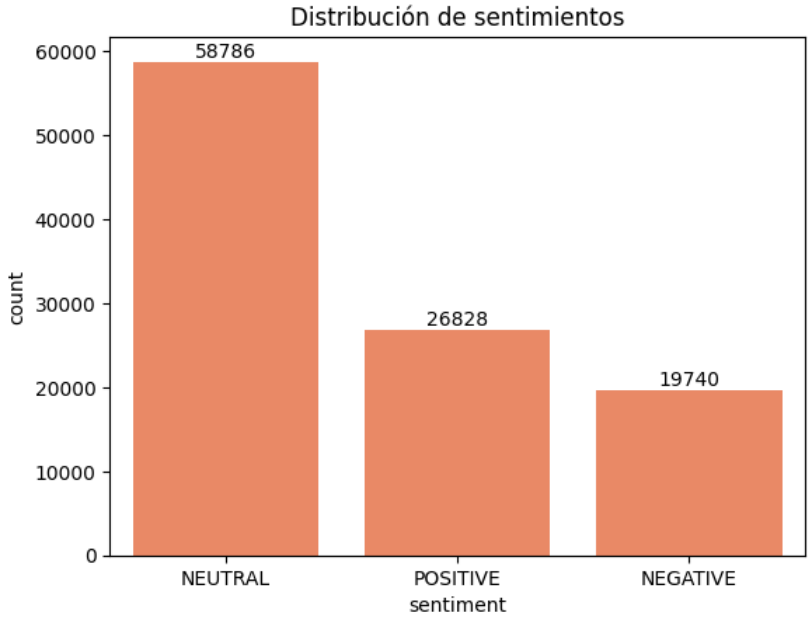


Figure 9: Distribución de sentimientos en los tweets

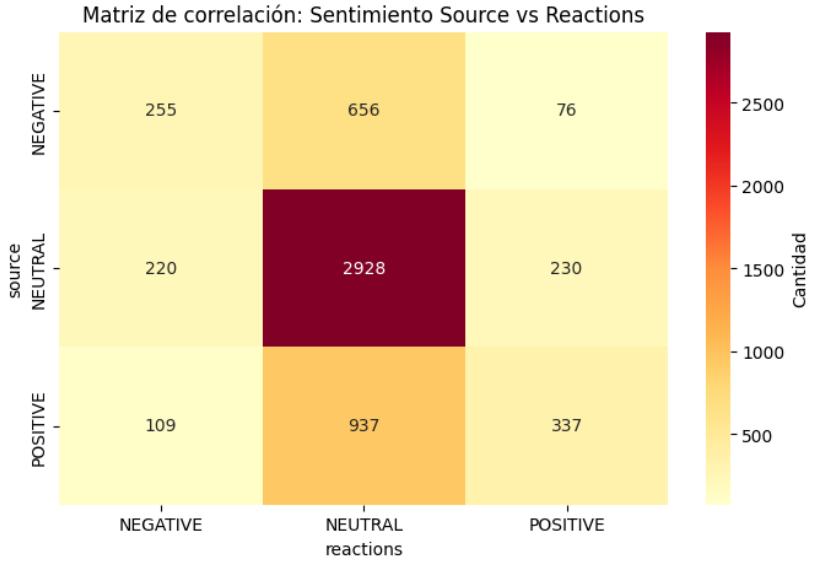


Figure 10: Sentimientos en reacciones

Figure 11: Wordcloud no rumores

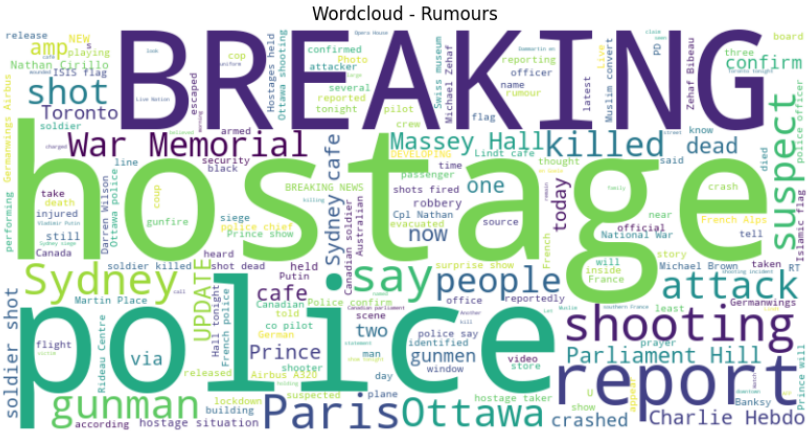


Figure 12: Wordcloud rumores

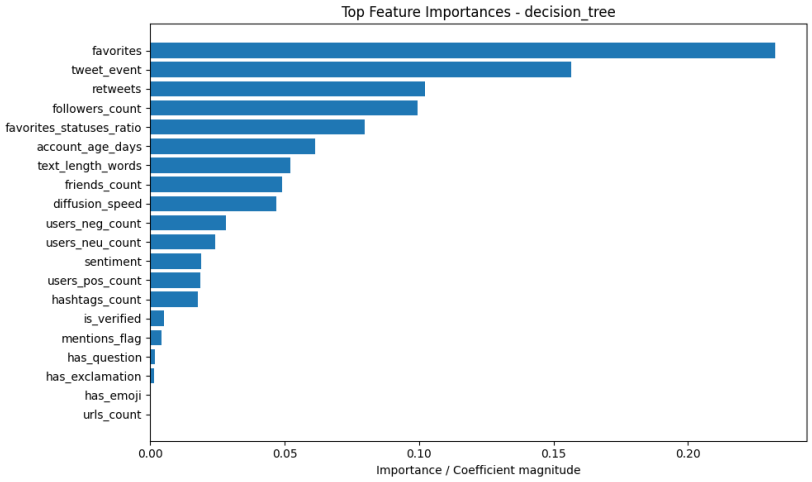


Figure 13: Feature importance Decision Tree

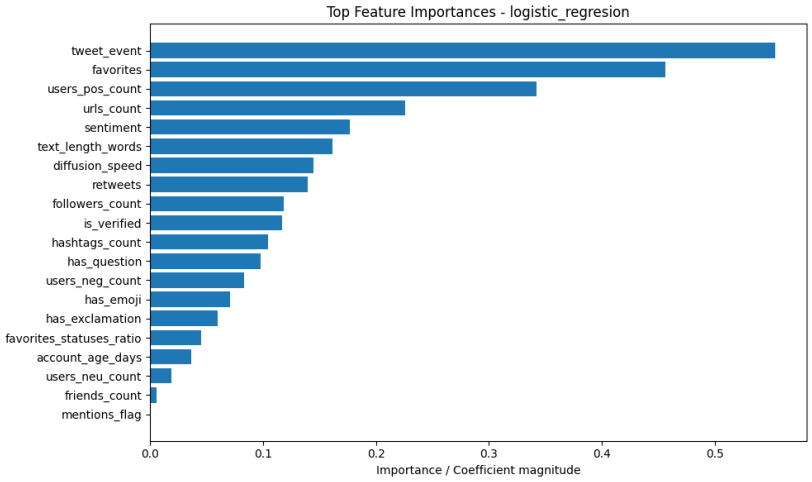


Figure 14: Feature importance Regresión Logística

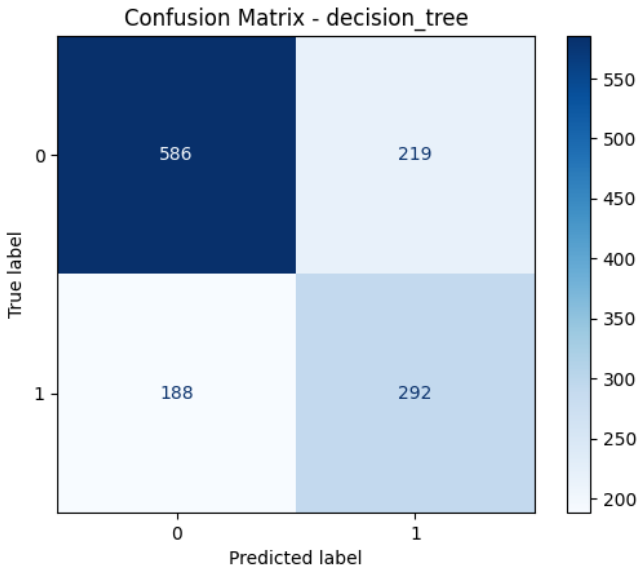


Figure 15: Matriz de confusión de Decision Tree

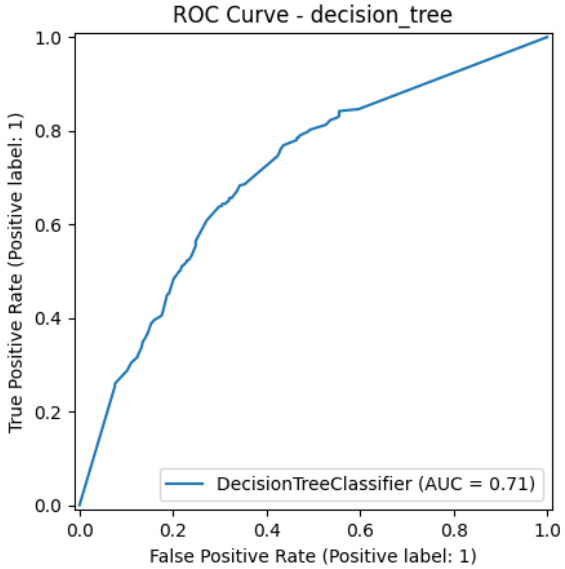


Figure 16: Curva de ROC Decision Tree

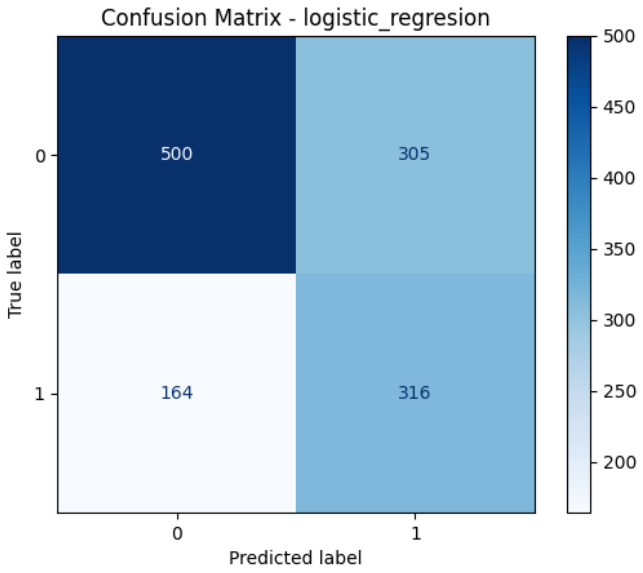


Figure 17: Matriz de confusión de Regresión Logística

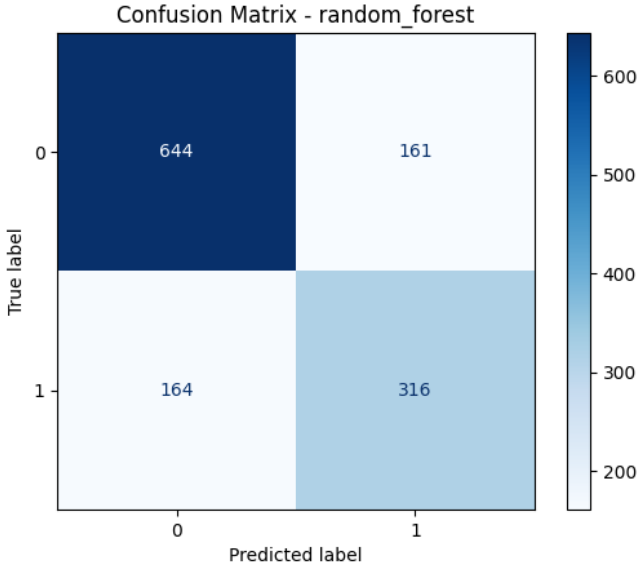


Figure 18: Matriz de confusión de Random Forest

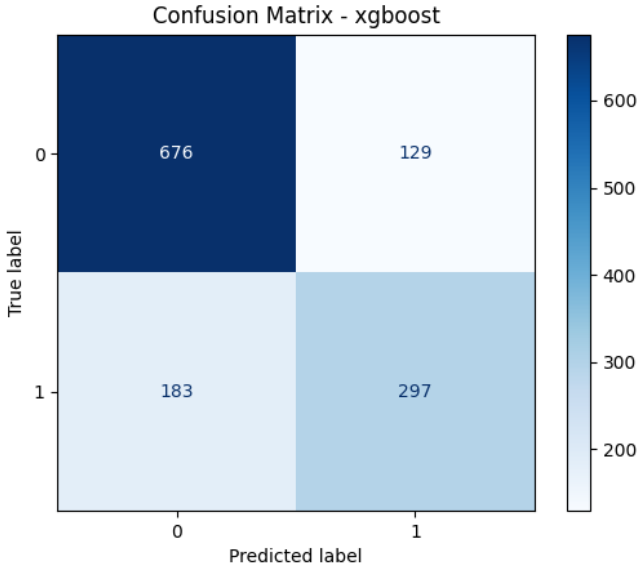


Figure 19: Matriz de confusión de XGBoost

Variable	**Significado**
text	Texto del tweet
in_reply_to_status_id	ID del tweet al que responde
id	ID del tweet
favorite_count	Número de favoritos
retweeted	¿Es un retweet? (booleano)
entities_hashtags	Hashtags presentes en el tweet
entities_urls	URLs presentes en el tweet
retweet_count	Número de retweets
in_reply_to_user_id	ID del usuario al que responde
user_id	ID del usuario que publicó el tweet
user_verified	¿Cuenta verificada?
user_followers_count	Número de seguidores del usuario
user_statuses_count	Número total de tweets publicados por el usuario
user_friends_count	Número de cuentas que sigue el usuario
user_favourites_count	Número de tweets que el usuario ha marcado como favoritos
user_created_at	Fecha de creación de la cuenta del usuario
lang	Idioma del tweet
created_at	Fecha de publicación del tweet
event	Evento al que está asociado el tweet
category	Categoría del tweet (ej. rumor/no rumor)
thread_id	ID del hilo de conversación
subfolder	Carpeta/subconjunto dentro del dataset
thread_veracity	Veracidad del hilo (verdadero, falso, incierto)

Table 3: Columnas seleccionadas de Pheme

Feature	**Explicación**
text	Texto del tweet
account_age_days	Días desde la creación de la cuenta del usuario
text_length_words	Longitud del tweet en número de palabras
has_exclamation	Indica si el tweet contiene signos de exclamación
has_question	Indica si el tweet contiene signos de interrogación
has_emoji	Indica si el tweet contiene emojis
hashtags_count	Número de hashtags presentes en el tweet
urls_count	Número de URLs presentes en el tweet
mentions_flag	Indica si el tweet contiene menciones a otros usuarios
retweets	Número de retweets del tweet
favorites	Número de favoritos del tweet
is_verified	Indica si la cuenta del usuario está verificada
tweet_event	Evento al que está asociado el tweet
followers_count	Número de seguidores del usuario
friends_count	Número de cuentas que sigue el usuario
favorites_statuses_ratio	Relación entre favoritos y número total de tweets publicados
diffusion_speed	Velocidad de difusión del tweet (ej. tiempo hasta la primera respuesta)
sentiment	Sentimiento del tweet (positivo, negativo, neutro)
avg_reply_sentiment	Promedio de sentimiento en las respuestas al tweet
users_pos_count	Número de usuarios con respuestas de sentimiento positivo
users_neg_count	Número de usuarios con respuestas de sentimiento negativo
users_neu_count	Número de usuarios con respuestas de sentimiento neutro
tweet_type	Tipo de tweet (ej. original, respuesta, retweet)
classification	Clasificación del tweet (ej. rumor, no rumor)
language	Idioma del tweet

Table 4: Características del dataset creadas