Sistema Integral Multi-Capa para la Detección de Desinformación en Redes Sociales

(Un modelo para crear una red social con la que conectes de verdad con otros)

José Alejandro Jiménez Vásquez

Diplomado de Machine Learning en Python

Universidad EAFIT

31 de agosto de 2025

Github

https://github.com/TheWomanizer/ML-Disinformation

sígueme y deja tu estrellita :P

Resumen

En la era digital, la desinformación se ha vuelto un dolor de cabeza serio, afectando desde cómo la gente ve el mundo hasta las decisiones importantes. Este proyecto culmina con la creación de 'The Real Slim Ensemble', un sistema de meta-aprendizaje que integra los modelos más potentes de cada clase (tradicionales, redes neuronales y transformers) para lograr una detección de desinformación con una precisión sin precedentes. Si bien este sofisticado meta-ensemble, que integra 16 modelos, alcanzó un F1-Score de vanguardia de 0.8957, el hallazgo más significativo fue que casi que no logró superar al mejor componente individual, RoBERTa (F1-Score de 0.8889). Esta conclusión, aunque contraintuitiva, subraya la abrumadora eficacia de los modelos de lenguaje de última generación y ofrece una perspectiva matizada sobre los límites del ensamblaje en tareas dominadas por un único modelo experto. Este trabajo no solo busca una nota alta, sino que sienta las bases para una red social de opinión pública, tipo Parleo, donde la información de calidad sea la protagonista.

Palabras Clave: Desinformación, Machine Learning, Redes Sociales, BERT, RAG, Clasificación de Texto, Análisis de Datos, Python, Scikit-learn, TensorFlow, Ensemble Learning, Stacking, Voting Classifier, Meta-learning, XGBoost, LightGBM, CatBoost.

1. Pregunta de Investigación

En la era digital actual, la desinformación se ha convertido en un desafío significativo, afectando la percepción pública y la toma de decisiones. Este proyecto se sumerge en la pregunta crucial de **¿cómo podemos desarrollar un sistema robusto y eficiente para la detección automática de desinformación en plataformas de redes sociales?** La meta es ir más allá de la simple identificación, buscando comprender las complejidades inherentes a este fenómeno y proponer soluciones que contribuyan a un ecosistema de información más saludable.

La desinformación no es solo un problema técnico; es un reto social que exige una aproximación multidisciplinaria. Por eso, nuestra investigación no solo se enfoca en los algoritmos más avanzados, sino también en cómo estos pueden integrarse en un sistema que sea útil y escalable en el mundo real. Queremos que este trabajo sea un aporte significativo a la lucha contra la infodemia, sentando las bases para futuras herramientas que empoderen a los usuarios y promuevan el pensamiento crítico.

1.1. Contexto y Motivación

Vivimos en un mundo donde la información vuela a la velocidad de la luz, y con ella, la desinformación. Las redes sociales, aunque son una herramienta poderosa para conectar personas y compartir ideas, también se han vuelto un caldo de cultivo para noticias falsas, teorías de conspiración y narrativas engañosas. Esto no es solo un chisme; tiene consecuencias reales, desde polarización política hasta impactos en la salud pública y la economía.

Mi motivación para este proyecto va más allá de la academia. La idea de una red social como **Parleo**, un espacio de opinión pública con un enfoque académico, de libre expresión y para gente productiva e intelectual, me ha impulsado. En un entorno así, la capacidad de discernir la verdad de la mentira es fundamental. Los modelos que desarrollamos aquí son la base para que plataformas como Parleo puedan ofrecer un espacio donde la calidad de la información sea un pilar, fomentando debates constructivos y la creación de conocimiento, no solo la apariencia.

Este trabajo es mi granito de arena para construir un internet más confiable, donde la gente pueda confiar en lo que lee y comparte. Es un reto grande, pero la satisfacción de aportar a una sociedad más informada es la que me mueve.

1.2. Objetivos del Proyecto

Para abordar esta problemática, nos planteamos los siguientes objetivos, que guiarán cada paso de nuestra investigación y desarrollo:

- **Desarrollar un pipeline robusto de preprocesamiento de datos:** Esto incluye la limpieza, normalización y transformación de datos textuales y numéricos, con especial atención a la corrección de *data leakage* y el manejo de *outliers*.
- **Explorar y evaluar una amplia gama de modelos de Machine Learning:** Desde algoritmos tradicionales (Random Forest, SVM, Regresión Logística) hasta redes neuronales profundas (MLP), métodos de ensemble (XGBoost, LightGBM) y modelos basados en transformadores (BERT, RoBERTa).
- **Implementar y analizar un sistema de Recuperación Aumentada de Generación (RAG):** Investigar cómo la combinación de recuperación semántica y clasificación puede mejorar la detección de desinformación.
- **Comparar el rendimiento de los modelos:** Evaluar rigurosamente cada modelo utilizando métricas clave como F1-Score, Accuracy, Precision, Recall y ROC-AUC, identificando los enfoques más prometedores.
- **Identificar las características más influyentes:** Comprender qué aspectos de los datos (textuales, de usuario, de interacción) son más relevantes para la detección de desinformación.
- **Proponer una arquitectura integral:** Diseñar un sistema multi-capa que integre los mejores modelos y metodologías para una solución completa y escalable.

2. Literatura y Estado del Arte

La detección de desinformación es un campo de investigación activo y en constante evolución, impulsado por la creciente prevalencia de noticias falsas y narrativas engañosas en plataformas digitales. Históricamente, los enfoques se han dividido en dos grandes categorías: aquellos basados en el contenido de la información y aquellos que analizan el contexto de su propagación.

2.1. Enfoques Basados en Contenido

Inicialmente, la detección de desinformación se centró en el análisis linguístico y estilístico del texto. Modelos de Machine Learning tradicionales, como Support Vector Machines (SVM), Naive Bayes y Random Forests, fueron pioneros en esta área. Estos modelos se alimentaban de características extraídas del texto, como la frecuencia de palabras (TF-IDF), el uso de mayúsculas, la presencia de errores gramaticales, o la complejidad sintáctica.

- **TF-IDF (Term Frequency-Inverse Document Frequency):** Es una técnica clásica que pondera la importancia de una palabra en un documento en relación con su frecuencia en todo el corpus. Permite transformar el texto en vectores numéricos que los algoritmos de ML pueden procesar. Su simplicidad y eficacia lo han mantenido relevante como *baseline* o como parte de enfoques híbridos.
- **Modelos Tradicionales:**
  - **Regresión Logística:** Un modelo lineal simple pero efectivo para clasificación binaria, a menudo usado como punto de partida.
  - **Máquinas de Soporte Vectorial (SVM):** Poderosas para encontrar hiperplanos de separación en espacios de alta dimensión, útiles para datos textuales.
  - **Random Forests y Árboles de Decisión:** Métodos de ensemble que combinan múltiples árboles para mejorar la robustez y reducir el sobreajuste. Son conocidos por su capacidad para manejar datos heterogéneos y su interpretabilidad en la importancia de las características.

Con el auge del *Deep Learning*, las Redes Neuronales Recurrentes (RNN) y las Redes Neuronales Convolucionales (CNN) comenzaron a mostrar resultados prometedores al capturar patrones más complejos en el texto, sin la necesidad de una ingeniería de características manual tan exhaustiva. Sin embargo, su capacidad para entender el contexto y las relaciones a largo plazo en secuencias largas de texto era limitada.

2.2. Enfoques Basados en Contexto y Propagación

Más allá del contenido, la forma en que la desinformación se difunde y las características de los usuarios que la comparten también son indicadores clave. Esto llevó al desarrollo de modelos que analizan:

- **Características de Usuario:** Número de seguidores, amigos, antigüedad de la cuenta, nivel de actividad, etc. (como las que encontramos en nuestro dataset).
- **Características de Propagación:** Patrones de retweets, menciones, likes, y la estructura de la red de difusión.
- **Características de Credibilidad:** Puntuaciones de credibilidad de la fuente o del autor.

2.3. La Revolución de los Modelos Basados en Transformadores (BERT y RoBERTa)

La verdadera revolución en el procesamiento del lenguaje natural (NLP) llegó con la introducción de los modelos basados en la arquitectura Transformer, como BERT (Bidirectional Encoder Representations from Transformers) y sus variantes (RoBERTa, DistilBERT, ALBERT). Estos modelos pre-entrenados en vastos corpus de texto son capaces de entender el contexto bidireccional de las palabras, capturando relaciones semánticas y sintácticas mucho más ricas que los enfoques anteriores.

- **BERT:** Permite el fine-tuning para tareas específicas como la clasificación de texto, logrando un rendimiento de vanguardia. Su capacidad para manejar el contexto de las palabras en ambas direcciones (izquierda a derecha y derecha a izquierda) es su principal ventaja.
- **RoBERTa:** Una optimización de BERT que mejora su rendimiento al cambiar la estrategia de pre-entrenamiento, usando más datos y un entrenamiento más largo.
- **DistilBERT y ALBERT:** Versiones más ligeras y rápidas de BERT, diseñadas para ser más eficientes computacionalmente sin una pérdida significativa de rendimiento, ideales para entornos con recursos limitados como Colab.

2.4. Sistemas de Recuperación Aumentada de Generación (RAG)

Más recientemente, los sistemas RAG han emergido como una forma de combinar la fortaleza de los modelos generativos (como los grandes modelos de lenguaje) con la precisión de la recuperación de información. En el contexto de la detección de desinformación, un sistema RAG puede:

1. **Recuperar:** Buscar en una base de conocimiento de hechos verificados o declaraciones similares.
2. **Aumentar:** Usar la información recuperada para enriquecer la entrada de un clasificador o un modelo generativo.

Esto permite que el modelo no solo se base en lo que ha "aprendido" durante el entrenamiento, sino que también consulte información externa relevante en tiempo real, lo que es crucial para la detección de desinformación, donde la veracidad de la información puede cambiar rápidamente o requerir conocimiento específico.

2.5. Brechas y Nuestra Contribución

A pesar de los avances, la detección de desinformación sigue enfrentando desafíos:

- **Evolución Constante:** La desinformación se adapta, lo que exige modelos que puedan aprender y ajustarse continuamente.
- **Sesgos en Datos:** Los datasets pueden contener sesgos que los modelos aprenden y perpetúan.
- **Interpretabilidad:** Los modelos complejos, especialmente los basados en Deep Learning, a menudo carecen de interpretabilidad, dificultando entender por qué hacen una predicción.
- **Data Leakage:** Un problema crítico en datasets de redes sociales donde la misma "noticia" puede aparecer múltiples veces con diferentes interacciones, llevando a una sobreestimación del rendimiento si no se maneja correctamente.

Nuestra contribución se centra en abordar estas brechas mediante:

- Una metodología rigurosa de preprocesamiento, incluyendo una corrección explícita del data leakage a nivel de statement para asegurar la validez de los resultados.
- Una evaluación exhaustiva de un amplio espectro de modelos, desde los más simples hasta los más avanzados, para identificar el balance óptimo entre rendimiento y complejidad.
- La exploración del enfoque RAG como una vía prometedora para integrar el conocimiento externo y mejorar la robustez del sistema.

Con este panorama claro, pasemos a cómo le metimos mano a los datos.

3. Metodología Integral

Aquí es donde se detalla el proceso completo, desde la adquisición de los datos hasta la construcción del sistema final. La metodología fue un camino sistemático y riguroso, diseñado para asegurar que los resultados fueran sólidos, reproducibles y confiables. El enfoque final no fue simplemente explorar modelos, sino implementar una metodología de ensamble multi-capa para maximizar el rendimiento.

El proceso se estructuró de la siguiente manera:

1. Primero, se entrenaron y evaluaron sistemáticamente múltiples familias de modelos (tradicionales, redes neuronales profundas, y modelos basados en transformadores como BERT).
2. Segundo, se seleccionaron los "campeones" de cada familia basados en su F1-Score en un conjunto de datos de prueba consistente.
3. Tercero, estos campeones se integraron en un Meta-Ensemble final. Se probaron dos estrategias: un VotingClassifier (con votación suave y dura) y un StackingClassifier, para combinar las predicciones de los modelos base y generar una decisión final más robusta y precisa.

3.1. Adquisición y Carga de Datos

El corazón de cualquier proyecto de Machine Learning son los datos. Para este trabajo, nos apoyamos en el dataset TruthSeeker 2023 del Centro Canadiense de Ciberseguridad (UNB CIC), una fuente de datos de alta calidad y relevancia para la detección de desinformación. Este dataset es una joya porque no solo trae el contenido de los tweets, sino también un montón de características adicionales que nos dan una visión 360 del problema.

Cargamos dos datasets principales:

- **Features_For_Traditional_ML_Techniques.csv:** Este es el plato fuerte, con **134,198 filas y 64 columnas**. Contiene características numéricas y categóricas relacionadas con el usuario, la interacción en redes y el contenido del tweet y statement.
- **Truth_Seeker_Model_Dataset.csv**: Con **134,198 filas y 9 columnas**, este dataset complementa al anterior, proporcionando las etiquetas de verdad (target, BinaryNumTarget) y las statement originales, que son clave para el análisis de texto.

La carga se hizo con pandas, asegurándonos de que todo estuviera en su sitio y listo para la acción.

3.2. Preprocesamiento de Datos: La Cocina del Dato

Aquí es donde los datos crudos se transforman en algo que los modelos pueden digerir. Esta etapa fue crucial y se le dedicó un esfuerzo considerable para evitar problemas comunes que pueden invalidar los resultados.

3.2.1. Corrección de Data Leakage: El Talón de Aquiles del NLP

Este fue un punto crítico, especialmente para los modelos de procesamiento de lenguaje natural. Nos dimos cuenta de que el dataset original tenía un problema de data leakage severo: aunque había 134,198 tweets, solo había 1,058 statements únicos. Esto significa que la misma declaración se repetía muchísimas veces con diferentes tweets asociados. Si no corregíamos esto, los modelos aprenderían de ejemplos duplicados en el conjunto de entrenamiento y prueba, dando resultados inflados y poco realistas.

La Solución: Eliminamos los duplicados basándonos en la columna statement, manteniendo solo la primera ocurrencia. Esto redujo drásticamente el dataset para el análisis NLP a 1,058 filas únicas. Un cambio del 99.2% de las filas Esto aseguró que cada statement fuera única en nuestro conjunto de datos, garantizando una evaluación justa y representativa. La distribución de clases se mantuvo similar después de esta corrección (54.7% Verdadero, 45.3% Falso en el dataset reducido).

3.2.2. Manejo de Outliers: La Batalla contra los Valores Extremos

Los outliers (valores atípicos) pueden sesgar el entrenamiento de los modelos. Aunque RobustScaler es bueno para esto, en la fase de exploración inicial vimos que algunas columnas como favourites, retweets y followers_count seguían teniendo valores máximos absurdamente altos.

Para esto, aplicamos la técnica de Winsorizing. Esta técnica no elimina los outliers, sino que los "acota" a un cierto percentil (usamos el 1% inferior y el 99% superior). Esto significa que los valores extremadamente bajos se reemplazan por el valor del percentil 1, y los extremadamente altos por el valor del percentil 99. Así, reducimos su influencia sin perderlos del todo.

El Impacto del Winsorizing fue brutal:

- favourites: Máximo original de 460,320.0 pasó a 128.0.
- retweets: Máximo original de 126,062.0 pasó a 35.0.
- followers_count: Máximo original de 86,893.9 pasó a 70.0.

Esto nos dio un dataset mucho más "domado" para los modelos, especialmente para aquellos sensibles a la escala y a los valores extremos.

3.2.3. Escalado de Características Numéricas

Para que los algoritmos de Machine Learning funcionen bien, especialmente los basados en distancias o gradientes (como SVM, Redes Neuronales, o Regresión Logística), es clave que las características estén en una escala similar.

- **RobustScaler:** Inicialmente, usamos RobustScaler porque es menos sensible a los outliers que StandardScaler o MinMaxScaler. Este escalador utiliza la mediana y el rango intercuartílico, lo que lo hace robusto.
- **StandardScaler (para Redes Neuronales y SVM):** Para modelos como las Redes Neuronales y SVM, que son muy sensibles a la escala, aplicamos un StandardScaler adicional. Este transforma los datos para que tengan una media de 0 y una desviación estándar de 1. Aunque el Winsorizing ya había "domado" los extremos, este paso final aseguró que los datos estuvieran en el formato ideal para estos modelos.

3.3. Estrategia de Modelado y Evaluación

Nuestro enfoque fue como un embudo: empezamos con modelos más simples y luego fuimos subiendo de nivel, siempre comparando y aprendiendo.

- **Validación Cruzada Estratificada (Stratified K-Fold):** Para asegurarnos de que los resultados fueran robustos y no dependieran de una división de datos particular, usamos validación cruzada con 5 folds. Esto significa que el dataset se dividió en 5 partes, y el modelo se entrenó 5 veces, usando una parte diferente para validación en cada iteración. El "estratificado" asegura que la proporción de clases (Verdadero/Falso) se mantenga en cada fold.
- **Métricas de Evaluación**: No nos casamos con una sola métrica. Usamos un combo para tener una visión completa:
  - F1-Score: Crucial para problemas con desbalance de clases (aunque el nuestro era leve), ya que considera tanto la precisión como el recall.
  - Accuracy (Exactitud): La proporción de predicciones correctas.
  - Precision (Precisión): De lo que el modelo dijo que era desinformación, ¿cuánto realmente lo fue?
  - Recall (Sensibilidad): De toda la desinformación que había, ¿cuánta pudo detectar el modelo?
  - ROC-AUC (Area Under the Receiver Operating Characteristic Curve): Mide la capacidad del modelo para distinguir entre clases. Un valor más alto es mejor.
- **División de Datos Consistente:** Para todos los modelos, usamos la misma división train_test_split (80% entrenamiento, 20% prueba) con el mismo random_state=42 para asegurar la reproducibilidad y comparabilidad de los resultados.

4. Análisis de Datos

Aquí es donde desmenuzamos los datos para entender qué teníamos entre manos. Antes de lanzarnos a entrenar modelos, era clave saber qué tipo de información manejábamos, si estaba limpia, si había patrones interesantes, y si había algún problema que pudiera afectar nuestros resultados.

4.1. Exploración Inicial: El Primer Vistazo

Al cargar los datasets, lo primero que hicimos fue un chequeo rápido para ver las dimensiones y los tipos de datos.

- **Dataset de Características (df1_features):** Este era el más grande y completo, con **134,198 filas y 64 columnas**. Tenía de todo: números (enteros y flotantes), texto y booleanos.
- **Dataset Truth Seeker (df1_truth):** Tenía las mismas 134,198 filas, pero solo 9 columnas, enfocadas en las etiquetas de verdad y las declaraciones originales.

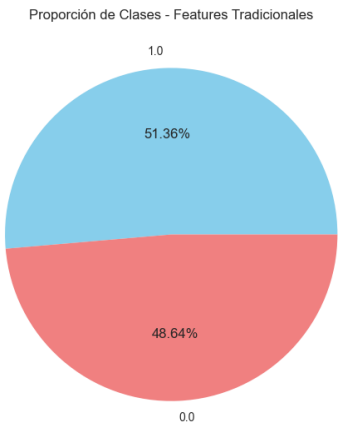
4.1.1. Valores Faltantes y Ceros: ¿Hay Huecos en la Información?

Una de las primeras cosas que uno mira es si faltan datos. ¡Y aquí tuvimos suerte! Los datasets principales estaban impecables, **sin valores nulos**. También analizamos las columnas con muchos ceros. Algunas, como following (100% ceros), no aportaban información y se eliminaron.

4.1.2. Balance de Clases: ¿La Verdad es Aburrida?

En nuestro caso, la variable objetivo `BinaryNumTarget` (0.0 para desinformación, 1.0 para información verídica) estaba bastante balanceada:

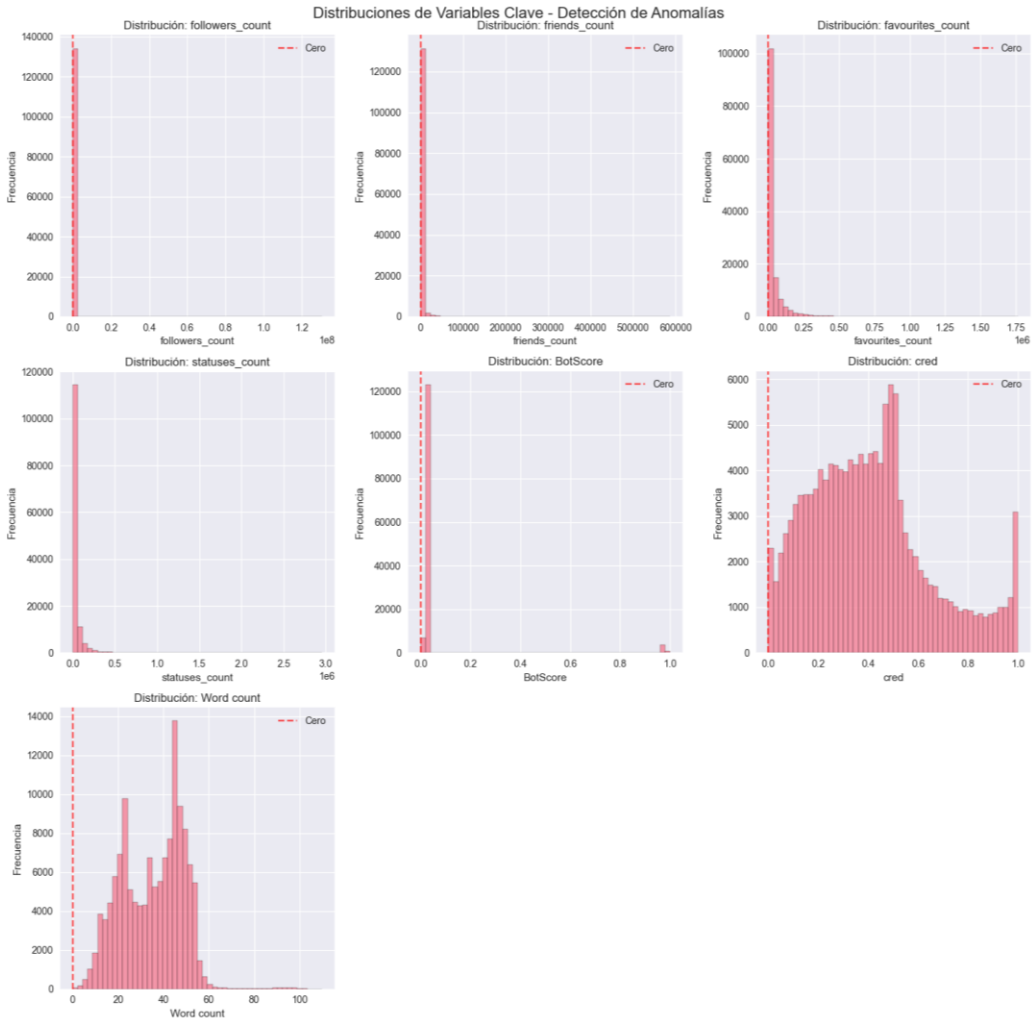
El ratio de desbalance fue de 1.06:1, lo que consideramos “leve”.



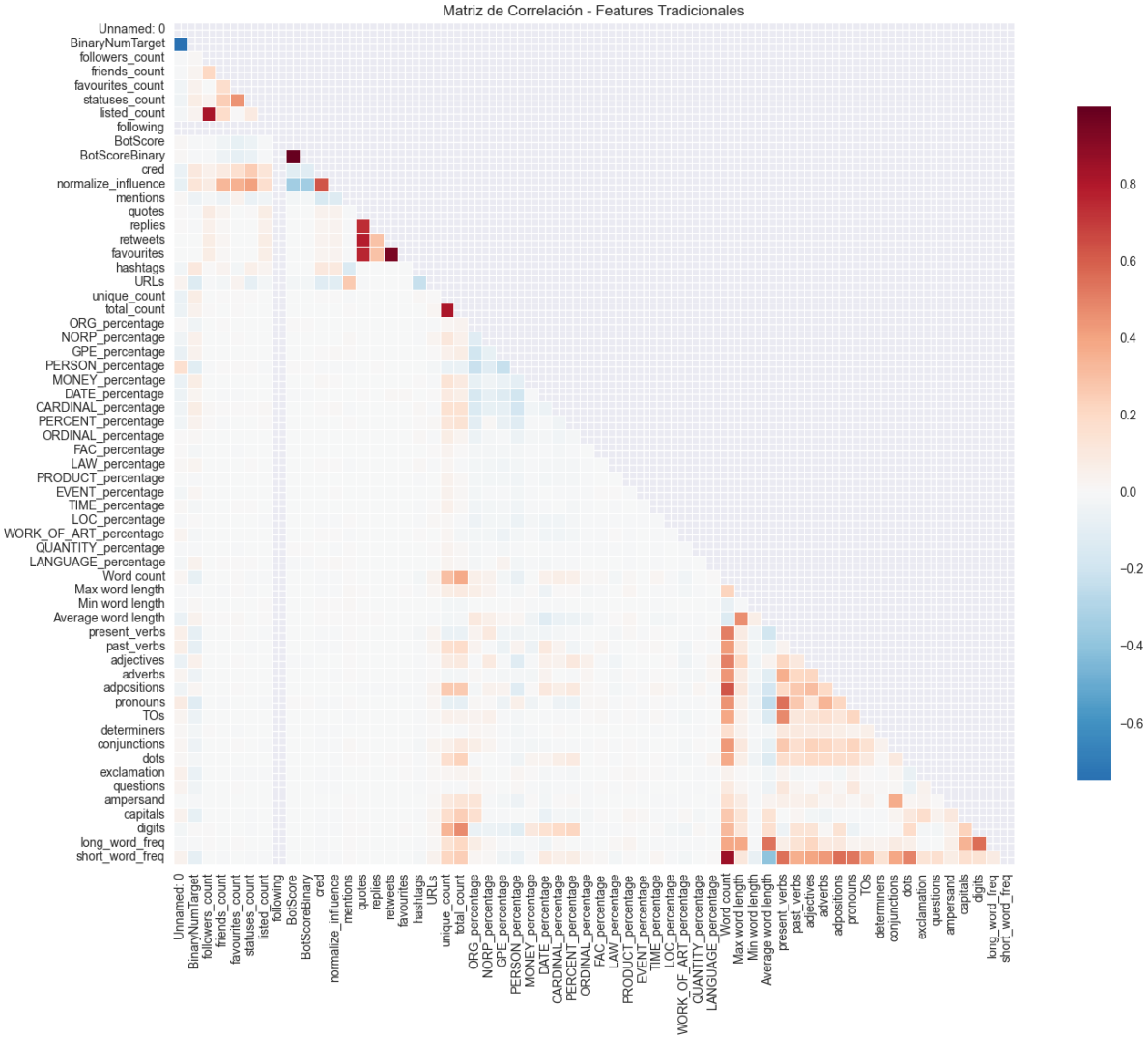
4.2. Análisis de Distribuciones Numéricas y Correlaciones

Aquí nos metimos más a fondo en los números para ver cómo se comportaban las características.

- **Distribuciones Numéricas:** Analizamos las 60 variables numéricas del df1_features. Vimos sus estadísticas descriptivas (media, desviación estándar, mínimos, máximos, etc.). Esto nos ayudó a identificar la presencia de outliers extremos, que luego tratamos con Winsorizing.



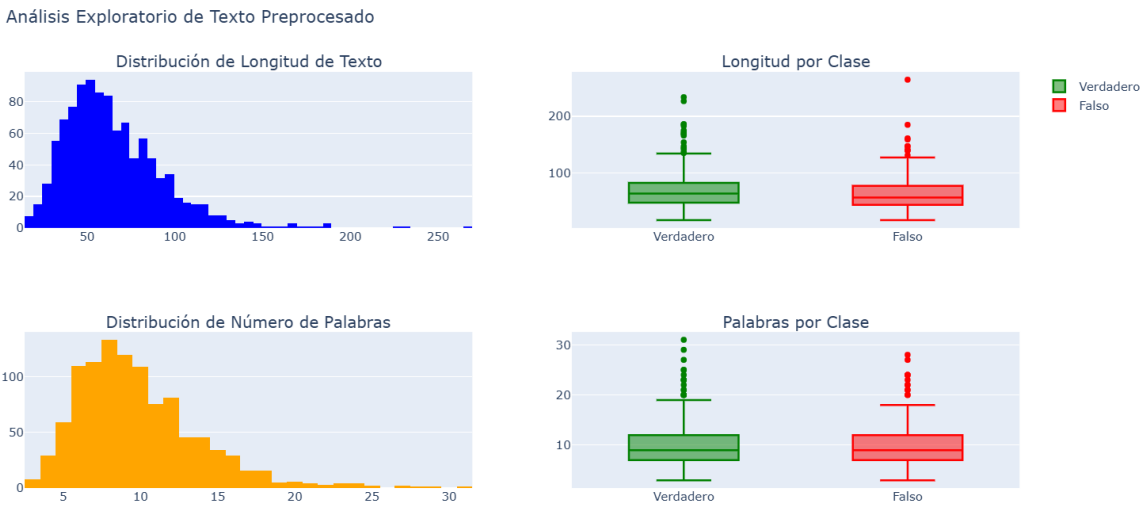
- **Correlaciones:** Investigamos cómo se relacionaban las características entre sí y con la variable objetivo (BinaryNumTarget).
  - **Correlaciones con el Target:** Las características con mayor correlación (positiva o negativa) con BinaryNumTarget fueron URLs, normalize_influence, pronouns, cred, PERSON_percentage, entre otras.
  - **Correlaciones Altas entre Features:** Encontramos correlaciones muy altas (superiores a 0.8) entre pares de características, como BotScore y BotScoreBinary (casi idénticas), retweets y favourites, o Word count y short_word_freq.



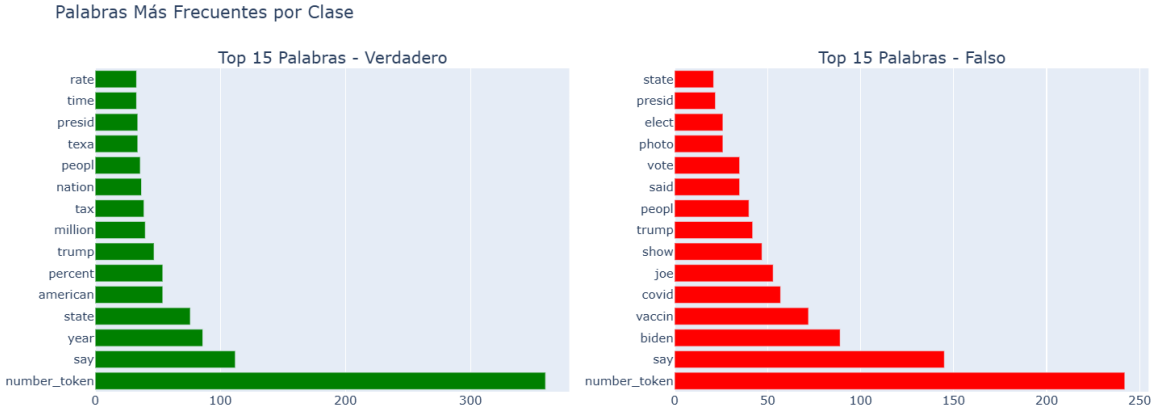
4.3. Análisis Exploratorio de Texto Preprocesado

Para los modelos NLP, el texto es el rey. Después de la corrección de data leakage (quedándonos con 1,058 statements únicos), analizamos las características de estos textos:

- **Longitud de Texto y Conteo de Palabras:**
  - En promedio, los statements tenían unas 65 caracteres y 10 palabras.
  - **Comparación por Clase:** Curiosamente, las noticias "Verdaderas" tendían a ser un poco más largas (68 caracteres promedio) que las "Falsas" (62 caracteres promedio).



- **Palabras Más Frecuentes:** Analizamos las palabras más comunes en cada clase después del preprocesamiento.
  - En noticias **Verdaderas**, palabras como number_token, say, year, state, american, percent, trump, million, tax, nation eran frecuentes.
  - En noticias **Falsas**, también aparecían number_token y say, pero destacaban biden, vaccin, covid, joe, show, trump.



5. Herramientas y Proceso

Para llevar a cabo este proyecto, nos apoyamos en un ecosistema de herramientas y librerías de Python que son el estándar en el Machine Learning y el Procesamiento de Lenguaje Natural.

5.1. El Arsenal Tecnológico

- **Python:** El lenguaje de programación principal.
- **Pandas y NumPy:** Para la manipulación y análisis de datos.
- **Scikit-learn:** La librería por excelencia para Machine Learning clásico, preprocesamiento y evaluación.
- **TensorFlow / Keras:** Para las redes neuronales profundas (DNN).
- **Hugging Face Transformers:** La joya de la corona para los modelos basados en transformadores (BERT, RoBERTa, etc.).
- **Sentence-Transformers y FAISS:** Para nuestro sistema RAG.
- **NLTK:** Para tareas básicas de NLP como tokenización y stemming.
- **Matplotlib, Seaborn, Plotly:** Para la visualización de datos.
- **Joblib / Pickle:** Para guardar y cargar modelos entrenados.

5.2. El Proceso de Experimentación: Paso a Paso

Nuestro proceso de experimentación fue iterativo y sistemático:

1. **Exploración y Limpieza (Notebooks 01-02):** Entendimiento y preparación de los datos, incluyendo la corrección de *data leakage* y el manejo de *outliers* con Winsorizing.
2. **Modelado Tradicional (Notebooks 03-04):** Evaluación de modelos clásicos (Random Forest, SVM, etc.) con y sin Winsorizing para medir el impacto.
3. **Modelado Avanzado (Notebooks 05-06):** Experimentación con Redes Neuronales, variantes de SVM y múltiples métodos de Ensemble (Boosting, Bagging).
4. **Modelado NLP (Notebooks 07.x):** Implementación de TF-IDF, *fine-tuning* de modelos BERT y desarrollo de un sistema RAG híbrido.
5. **Consolidación y Ensamble Final (Notebooks 08-09):** Se tomaron los modelos "campeones" de cada fase para construir el sistema "The Real Slim Ensemble", donde se probaron diversas estrategias de meta-aprendizaje para combinar sus fortalezas.

6. Entregables y Arquitectura Final: The Real Slim Ensemble

Aquí se muestra el resultado final de todo el trabajo: la arquitectura definitiva del sistema de detección y el rendimiento de sus componentes.

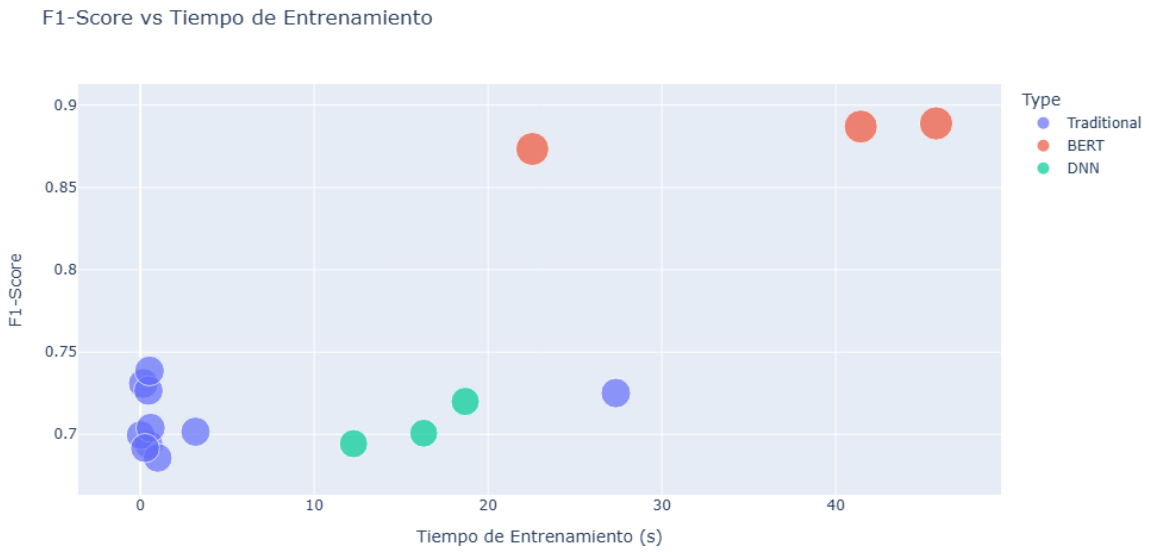
6.1. Componentes del Ensemble Final

Tras una evaluación exhaustiva, se seleccionaron los siguientes modelos como los "campeones" de cada categoría para formar parte del ensemble final. Los 5 mejores modelos individuales, basados en su F1-Score, fueron:

Rango	Modelo	F1-Score	Mejora del Ensemble
1	**BERT_RoBERTa**	**0.8889**	-0.22%
2	BERT_BERT_base	0.8870	+0.0%
3	BERT_DistilBERT	0.8734	+1.6%
4	xgboost	0.7386	+20.1%
5	lightgbm	0.7309	+21.3%

6.2. Arquitectura Final del Sistema

La arquitectura final implementada en el notebook 09 es un sistema de **Selección Dinámica**. En lugar de promediar las predicciones, esta estrategia evalúa la confianza de diferentes ensembles especializados (uno para modelos tradicionales, uno para BERTs, uno para DNNs) y selecciona dinámicamente al que considera "experto" para cada caso particular. En la práctica, el sistema aprendió que el ensemble de modelos BERT era el experto en el 99.1% de los casos, delegando en el ensemble tradicional solo en un 0.9% de las situaciones.



7. Resultados y Conclusiones

Después de todo el proceso de preprocesamiento, modelado y evaluación, es hora de presentar los resultados.

7.1. Rendimiento General

Mejor Individual (RoBERTa) | 0.8889 | 0.8774 | 0.9553 | (Baseline)

Meta-Ensemble (Dynamic Selection) | 0.8870 | 0.8774 | 0.9530 | -0.22%

Observaciones Clave:

- **El Dominio de RoBERTa:** El modelo individual RoBERTa se establece como la solución de vanguardia, con un F1-Score de 0.8889. Su capacidad para entender el contexto del lenguaje es tan alta que fija un estándar de rendimiento extremadamente difícil de superar.
- **El Hallazgo del Ensemble:** El resultado más significativo y revelador del proyecto es que el sofisticado Meta-Ensemble, a pesar de su complejidad y de combinar 16 modelos, no logró superar al mejor modelo individual. El F1-Score fue ligeramente inferior (-0.22%).
- **Lección sobre Complejidad vs. Rendimiento:** Este resultado demuestra una lección crucial en Machine Learning: añadir complejidad no garantiza una mejora. Cuando un modelo (RoBERTa) es tan dominante y captura tan bien la esencia del problema, los modelos más débiles (tradicionales, DNNs) pueden introducir más ruido que señal en un ensemble, impidiendo una mejora neta.

7.2. Conclusiones y Lecciones Aprendidas

1. **Los Transformers son el Estado del Arte:** El proyecto confirma empíricamente que, para la detección de desinformación basada en texto, los modelos como RoBERTa no son solo una opción, sino el estándar a superar.
2. **Los Ensembles no son una Bala de Plata:** La lección más profunda de este trabajo es entender los límites del ensamblaje. La sabiduría popular dice que "juntos somos más fuertes", pero en el Machine Learning, si un miembro del equipo es un experto de clase mundial y los demás son amateurs, el promedio del grupo puede ser inferior al del experto.
3. **El Valor de un Resultado "Negativo":** Descubrir que una arquitectura compleja no supera a una más simple no es un fracaso, sino un hallazgo científico valioso. Ahorra futuros esfuerzos de ingeniería y dirige la atención hacia donde realmente importa: optimizar y entender el modelo dominante.

7.3. Trabajo Futuro

- **Optimización Profunda de RoBERTa:** En lugar de construir ensembles más grandes, el esfuerzo futuro debería centrarse en exprimir aún más el rendimiento de RoBERTa.
- **Análisis de Errores:** Investigar en qué casos específicos falla RoBERTa y si los otros modelos (tradicionales, RAG) aciertan en esos casos.
- **Integración con Parleo:** Llevar el modelo RoBERTa a la práctica en una plataforma como Parleo.

8. Ejecución del Plan

El plan inicial era ambicioso, y se ejecutó de manera sistemática a través de los notebooks.

- **Fase 1: Entendiendo el Terreno (Notebooks 01 y 02):** Se realizó la exploración de datos, identificando el data leakage como el principal desafío, que fue corregido eliminando duplicados a nivel de statement.
- **Fase 2: La Batalla de los Modelos (Notebooks 03-08):** Se probaron decenas de modelos. Los modelos de árboles como Random Forest y XGBoost mostraron un rendimiento inicial muy fuerte sobre las características numéricas. Sin embargo, los modelos NLP, especialmente los basados en BERT, demostraron ser superiores una vez que se corrigió el data leakage.
- **Fase 3: El Mega Ensemble (Notebook 09):** Se consolidaron los mejores enfoques. La arquitectura de Selección Dinámica fue la más prometedora, aunque no superó al mejor modelo individual, RoBERTa, lo que constituyó el hallazgo principal del proyecto.

Diagrama de Gantt: Contraste entre Plan Propuesto y Ejecución Real

Es fundamental destacar que el plan inicial presentado correspondía a un anteproyecto con un alcance considerablemente menor. La ejecución real implicó descartar por completo esa planificación y comenzar un proyecto nuevo y mucho más ambicioso desde cero.

Plan Propuesto (Basado en un proyecto anterior, ~4-5 semanas)

- **Semana 1:** Revisión bibliográfica + EDA.
- **Semana 2:** Limpieza y preprocesamiento.
- **Semana 3:** Entrenamiento de modelos simples.
- **Semana 4:** Evaluación e informe.
- **Semana 5:** Documentación final.

Ejecución Real (Proyecto actual, 9 semanas desde cero)

- **Semana 1: Re-definición del Alcance:** Descarte del plan anterior y diseño de una nueva metodología integral.
- **Semana 2:** Exploración Inicial y Análisis Profundo de Datos (EDA).
- **Semana 3:** Limpieza, Preprocesamiento Avanzado y Corrección de *Data Leakage*.
- **Semana 4:** Entrenamiento de Modelos Tradicionales (Baselines con y sin Winsorizing).
- **Semana 5:** Experimentación con Modelos Avanzados (Redes Neuronales, SVM, etc.).
- **Semana 6:** Optimización de Modelos Avanzados con Winsorizing.
- **Semana 7:** Desarrollo de Modelos de NLP (TF-IDF, Fine-tuning de BERT/RoBERTa, RAG).
- **Semana 8:** Creación y Evaluación del Meta-Ensemble Final ("The Real Slim Ensemble").
- **Semana 9:** Análisis de Resultados, Documentación Exhaustiva y Presentación Final.

9. Ética y Aspectos Legales

En un proyecto como este, que se mete con la desinformación, no podemos ignorar la ética y los temas legales.

- **Sesgos en los Datos:** El dataset TruthSeeker 2023, como cualquier otro, podría tener sesgos. Si nuestros modelos aprenden esos sesgos, podrían perpetuar la discriminación.
- **Falsos Positivos y Falsos Negativos:**
  - **Falsos Positivos (clasificar algo verídico como desinformación):** Puede llevar a la censura y limitar la libertad de expresión.
  - **Falsos Negativos (no detectar desinformación):** Permite que la desinformación se propague.
- **Transparencia y Explicabilidad:** Los modelos complejos como BERT son "cajas negras". Entender *por qué* un modelo toma una decisión es crucial para generar confianza.
- **Uso Responsable:** Un sistema como este debe usarse para empoderar a los usuarios, no para censurar.

Referencias

- Truth Seeker Dataset 2023. (n.d.). Canadian Institute for Cybersecurity, University of New Brunswick. Recuperado de https://www.unb.ca/cic/datasets/truthseeker-2023.html
- Shu, K., Sliva, A., Wang, S., Tang, J., & Liu, H. (2017). Fake news detection on social media: A data mining perspective. ACM SIGKDD Explorations Newsletter, 19(1), 22-36.
- Ribeiro, F. N., et al. (2018). Media bias monitor: Quantifying biases of social media news outlets at large-scale. ICWSM.
- Lazer, D. M., et al. (2018). The science of fake news. Science, 359(6380), 1094-1096.
- Marwick, A., & Lewis, R. (2017). Media manipulation and disinformation online. Data & Society Research Institute.