



Tecnológico de Monterrey

Reporte de resultados obtenidos

Desarrollo de aplicaciones avanzadas de ciencias computacionales

Presenta

Alan Fernando Razo Peña A01703350

Grupo 301

25 de mayo del 2025

Introducción

En este proyecto, el uso del aprendizaje automático (Machine Learning) para analizar y sintetizar información sobre artículos de noticias resulta sumamente útil. Combinado con una búsqueda eficiente, nos permite revisar grandes volúmenes de texto, identificar patrones lingüísticos sospechosos y discriminar de forma rápida entre afirmaciones basadas en hechos y desinformación, lo que ahorra una gran cantidad de tiempo en comparación con una revisión manual por parte de verificadores humanos.

Sin embargo, aunque el aprendizaje automático es una herramienta poderosa para esta tarea, no es infalible. Puede presentar errores, especialmente en sus versiones gratuitas o limitadas, al analizar matices del lenguaje, sarcasmo, o noticias que imitan el estilo de fuentes legítimas. Por ello, una vez que se han clasificado los casos sospechosos de noticias falsas, es fundamental revisar manualmente el contenido implicado, contrastando con fuentes confiables y utilizando criterio profesional, sin depender ciegamente de los resultados generados por el modelo.

Obtener, generar o aumentar un set de datos

El Dataset utilizado para el proyecto fue obtenido en la plataforma Kaggle. Posteriormente se almacenó en una carpeta de Google Drive para su manipulación dentro del modelo de Machine Learning (ML). En este caso escogí el dataset de “**Real & Fake News**” debido a que me pareció un tema bastante interesante, muy relevante en la actualidad.

Entender la diferencia entre una noticia falsa de una real es crucial para que no estén circulando libremente en medios de difusión masiva como la prensa escrita, la radio, la televisión, el cine y, más recientemente, Internet y las redes sociales.

Los datos que vienen en el dataset son los siguientes:

- title - El titular del artículo
- text - El cuerpo completo de la noticia
- subject - La categoría o tema (por ejemplo, política, noticias mundiales, etc.)
- date - La fecha de publicación

Separación de los sets de prueba y entrenamiento

Usando Scikit-learn, conocido como sklearn, que es una biblioteca de código abierto para aprendizaje automático en Python, se usó la herramienta de “train_test_split” para separar la información contenida en el dataset para entrenar y probar el modelo de Machine Learning.

La división de los datos se realizó de la siguiente manera:

- Entrenamiento: 80%
- Prueba: 20%

Técnicas de escalamiento y preprocesado de datos

Los métodos que se utilizaron para analizar los datos fueron: el escalamiento estándar de sklearn y la vectorización de texto con TF-IDF con TfidfVectorizer. Estas técnicas estadísticas son fundamentales para el desarrollo de nuestro modelo del ML. Los beneficios más importantes son que se puede clasificar y ponderar la importancia de las palabras en los archivos de texto.

El software utilizado para ejecutar estas herramientas fue Jupyter Notebook (junto con Python), que sirve para crear documentos de cuaderno interactivos que pueden contener código en vivo, ecuaciones, visualizaciones, medios y otros resultados computacionales.

Implementación del modelo

Para la primera implementación del modelo se utilizó un modelo de red neuronal (NN) donde las capas se apilan secuencialmente (una tras otra). El modelo usado fue el Sequential ya que fue visto durante las clases de Inteligencia Artificial y mencionado en artículos de investigación como **Advanced machine learning techniques for fake news detection: A comprehensive analysis** y **Analysis and Detection of Fake News Using Machine Learning**.

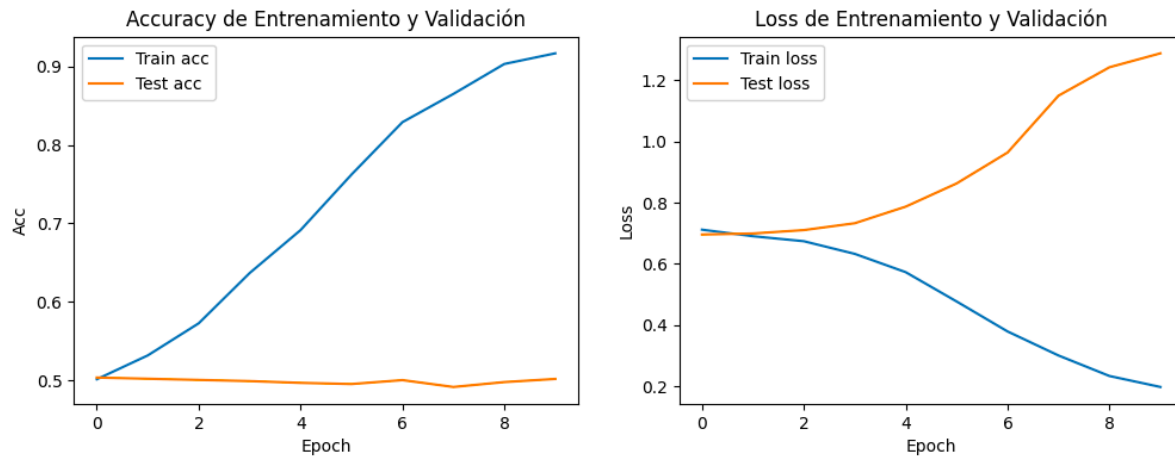
Esta selección del modelo está respaldada por distintos artículos sobre el estado del arte del proyecto. Se utilizaron algoritmos de preprocesamiento natural (NLP) y características como la lematización, las palabras vacías (stop-words) y la vectorización TF-IDF para entrenar el modelo de aprendizaje automático.

El optimizador Adam se empleó para minimizar la función de pérdida, destacando por su buen rendimiento en diversas tareas y su menor sensibilidad a cambios en los hiperparámetros.

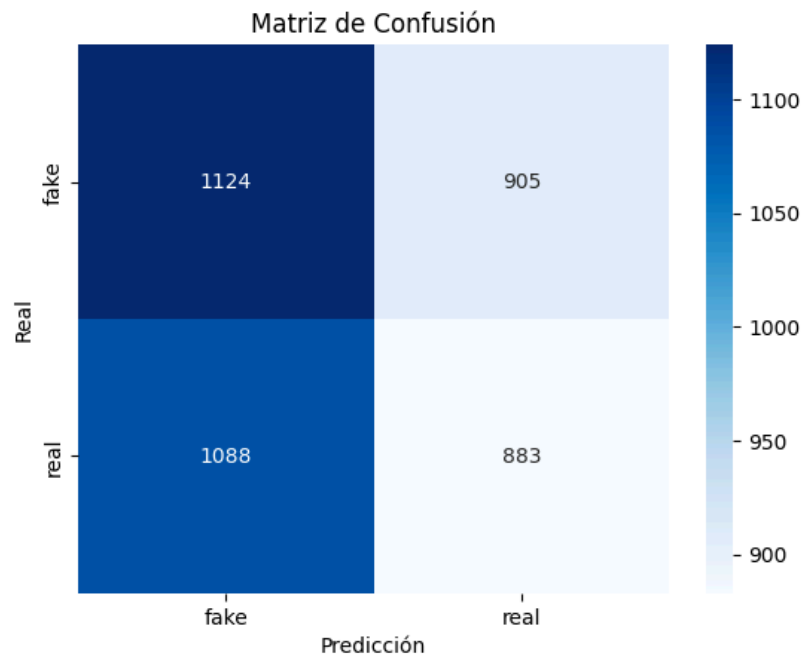
Las métricas, como la accuracy, se utilizan para medir el rendimiento del modelo al calcular el porcentaje de predicciones correctas. En este caso, se emplea accuracy debido a que se trabaja con un dataset balanceado y es la primera vez que se aprende a hacer un modelo de Machine Learning.

Evaluación inicial del modelo

Para esta primera evaluación del modelos, se demostró por medio de las métricas de “accuracy” y “loss” que el modelo está sobreajustado (Overfitting). Esto se debe a una variación alta de los datos y un sesgo bajo.



Usando una matriz de confusión, se pudo observar que para la primera implementación del modelo mostraba mayormente un falso positivo en las predicciones que deberían ser verdaderas.



Para resolver esta situación se debe hacer una regularización para reducir la complejidad del modelo y simplificación del modelo. También se va a realizar una implementación del modelo de regresión logística para comparar su desempeño en contraste con el Secuencial de redes neuronales.

Refinamiento del modelo

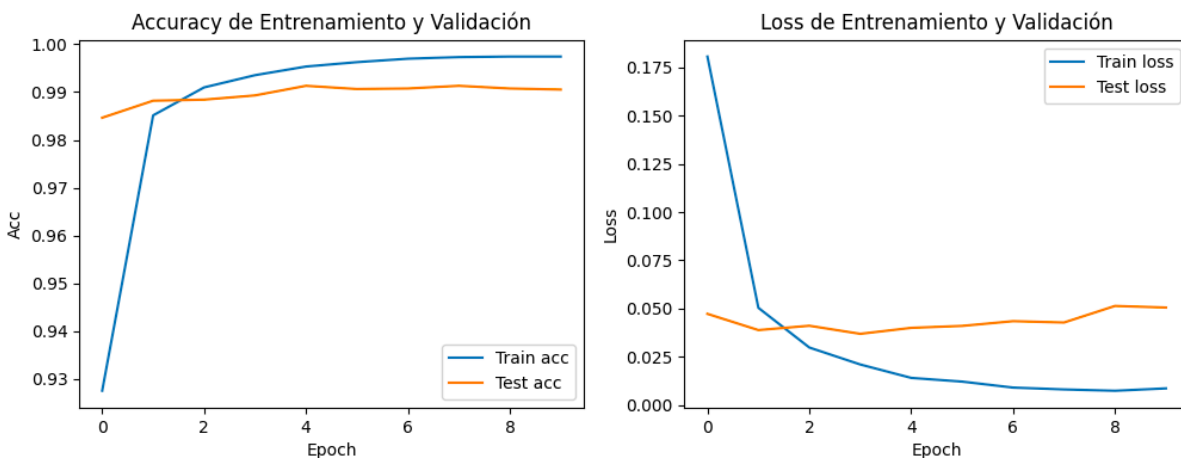
En la **segunda** evaluación se investigó más a profundidad la raíz de porque estaba arrojando falsos positivos. En este caso se debía a la información del dataset. Investigando en distintas implementaciones con dicho dataset en Kaggle, se observó que la precisión de la predicción del modelo no superaba el 0.52.

Classification Report:				
	precision	recall	f1-score	support
0	0.53	0.52	0.52	2029
1	0.51	0.51	0.51	1971
accuracy			0.52	4000
macro avg	0.52	0.52	0.52	4000
weighted avg	0.52	0.52	0.52	4000

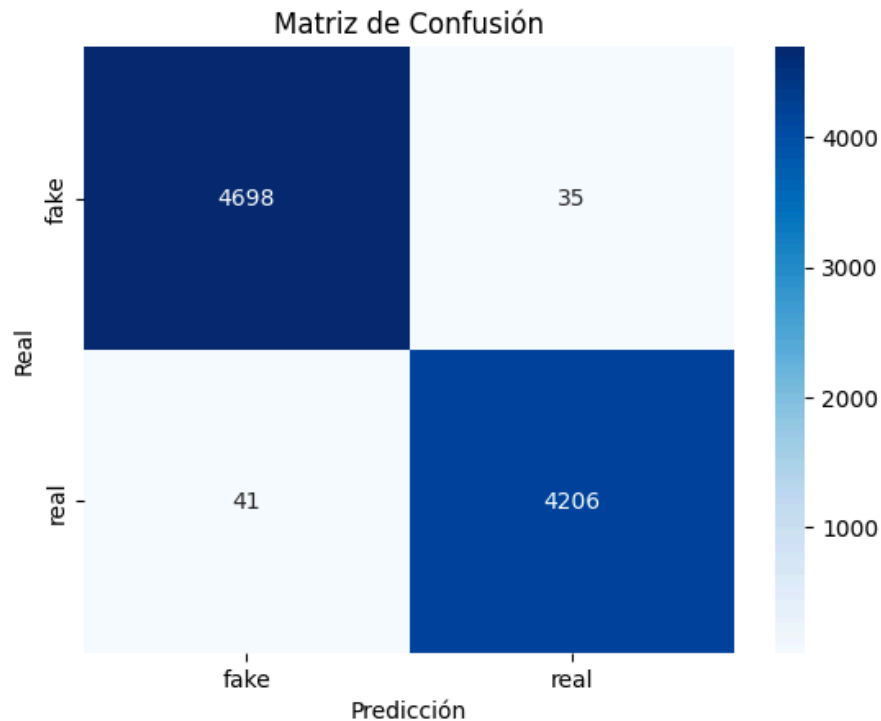
Debido a esto, decidí cambiar el dataset original que contaba con 20,000 registros. En la misma plataforma de Kaggle encontré uno dividido en dos archivos .csv separados (Fake, True). Concatenando la información y etiquetarlos de acuerdo si eran verdaderos o falsos, se llegó a tener un dataframe con 44,000 registros. El contar con más del doble de información resultaría más beneficioso para el aprendizaje automático por medio de redes neuronales.

Las correcciones en los hiper-parámetros de la arquitectura del modelo fueron la reducción en el número de neuronas en las capas densas y el aumento en el porcentaje de Dropout de 0.3 a 0.5. Esto con el fin de simplificar el modelo para evitar que aprenda demasiado rápido y evitar el sobreajuste (Overfitting).

Para la **tercera** implementación se usaron estas modificaciones mencionadas para que el modelo resultara más eficaz para la detección de noticias falsas. Se alimentó el modelo refinado con un nuevo dataset con las mismas características del anterior, con el doble de la información del original. Estas fueron las métricas de accuracy y loss al momento del entrenamiento y validación.




En la siguiente matriz se muestran los resultados de la clasificación durante la validación con una drástica mejora en comparación con el modelo inicial.




Correcciones

Estos fueron los resultados de la predicción al probar el modelo entrenado con dos noticias reales. Los artículos fueron obtenidos de fuentes confiables como Sky News y CNN para asegurar su viabilidad en la prueba. Los resultados fueron los siguientes:

Archivos reales

1/1  **0s 65ms/step**
Archivo: /content/real_example.txt
Probabilidad de ser Real: 0.8693
Predicción: Verdadero
'real'

1/1  **0s 44ms/step**
Archivo: /content/real_example2.txt
Probabilidad de ser Real: 0.5092
Predicción: Verdadero
'real'

Sin embargo existen casos en los que el modelo no puede procesar correctamente una noticia verdadera en cuestiones y temas más específicos. Este es un ejemplo en donde se esperaba un resultado verdadero en una noticia de un tema particular:

```
1/1 _____ 0s 48ms/step
Archivo: /content/article_real.txt
Probabilidad de ser Real: 0.0000
Predicción: Falso
'fake'
```

Por lo tanto fue necesario revisar el preprocesado de la información para los datos de entrada y para verificar que el texto no estuviera demasiado filtrado sobreajustado al momento de pasarlo a la predicción del modelo.

La corrección particular que se realizó fue separar en diferentes celdas la ejecución del programa que hace la predicción y así se evitó el sobreprocesamiento de la información con las stopwords y el lemmatizer.

A pesar de que se realizaron las mejoras correspondientes, aún existen discrepancias en cuanto a los resultados que arroja el modelo. Esto debido a que existen diferentes tipos de redacciones y temas muy objetivos, puede ser difícil obtener una predicción acertada cuando el modelo no conoce los temas con los que fue entrenado. Para ello se requeriría entrenar la Inteligencia Artificial con aún más información para que pueda aprender con datos más reales y de diversas fuentes para enriquecer su conocimiento.

Bibliografía

[1]

P. Kumar, P. Suthanthiradevi, C. A. Stephen, E. Abishek B, S. Sivakumar, and M. Mathiyarasu, “Analysis and Detection of Fake News Using Machine Learning,” May 2024, doi: 10.1109/aiiot58432.2024.10574761

[1]

I. Jahan *et al.*, “Advanced machine learning techniques for fake news detection: A comprehensive analysis,” *Magna Scientia Advanced Research and Reviews*, vol. 12, no. 2, pp. 203–212, Dec. 2024, doi: 10.30574/msarr.2024.12.2.0198

[1]

S. Velumuru, “Fake News Detection System Using Machine Learning and Deep Learning,” *Indian Scientific Journal Of Research In Engineering And Management*, Apr. 2024, doi: 10.55041/ijsrem30766