Modelo predictivo del precio del oro en función del impacto de noticias

Jose David Ortiz

Juan Pablo Sánchez

Dayana Henao

Luis Vera

Michael Tarazona

David Alava

Sebastian Agudelo

Talento Tech

2025

Introducción

Nos planteamos las siguientes preguntas: ¿Es posible que el precio del oro sea afectado por las noticias? Tiene sentido hacerse esta pregunta debido a la forma en la que funcionan los precios de las divisas y/o activos. Estos precios funcionan debido a las compras y ventas que se hacen con los mismos. Por tanto, hay que tener en cuenta el comportamiento humano, pues muchas personas basan sus decisiones de compra o venta en sus emociones y juicios al momento de la transacción, juicios y emociones que podrían estar influenciados por las noticias del momento, a través de un comportamiento reactivo, afectando el alza o la baja del precio del activo.

Por otro lado, en los últimos años se han desarrollado tecnologías que permiten crear modelos predictivos por medio de inteligencia artificial. Algunos de estos modelos son los modelos NNW, redes convolucionales, etc. Con estos métodos, se pueden crear modelos predictivos para predecir el comportamiento de ciertos sistemas en función de sus características en el tiempo; en particular, se pueden aplicar al precio de activos o divisas, como por ejemplo el oro.

Así, el siguiente proyecto pretende generar un modelo predictivo que tenga en cuenta la relación del precio del oro con las noticias del momento. Se realizará *web scraping* de las noticias de los últimos años de los noticieros de la BBC y WSJ. Adicionalmente, se extraerá el precio del oro en dólares AUX/USD de Dukascopy. Luego se revisarán los datos, se limpiarán, filtrarán y se hará un modelo predictivo en función de las noticias que puedan impactar los precios. Adicional a esto, se analizará cuáles noticias son fuera de lo común en función de los sentimientos de un modelo preentrenado.

Así, el trabajo tiene como objetivo realizar un modelo predictivo para el precio del oro en función del impacto que pueden llegar a tener las noticias y se comparará con una tendencia esperada sin tener presente los noticieros

Extracción de los datos

Para extraer los datos se usaron diversos métodos y/o librerías según la naturaleza de cada fuente, por ejemplo se usó request para usar la API REST, Selenium para poder acceder a contenido dinámico de JavaScript, y Scrapy para poder extraer los datos de las páginas en formato HTML estático. Este proceso se dividió en tres partes:

Precio del oro en horas: Se usó la API REST de la página Dukascopy para descargar datos tick-by-tick de la relación del oro contra el dólar, también conocida como XAU-USD. El proceso consistió en realizar peticiones HTTP por horas cada día, utilizando la librería request. Los datos se reciben en formato JSON comprimido usando diferencias incrementales, por tanto, fue necesario decodificarlos y construir los precios reales, sumando los deltas al precio inicial. Luego se usó pandas para agregar los ticks en barras OHLCV de 1 segundo mediante un resampling temporal. Por último se utilizó JSON para poder parsear los datos, que estos fueran legibles para su posterior uso y se guardó en formato CSV, de tal forma que tuviera el formato de los precios de apertura, precio más alto, precio más bajo, precio de cierre (todos en USD), y volumen en millones equivalentes en XAU/USD tramitados en dicho intervalo de tiempo**.**

Extracción de noticias de WSJ: Para extraer noticias del Wall Street Journal se utilizó Selenium, pues el sitio carga el contenido de forma dinámica mediante JavaScript. WSJ tiene la característica de tener un repositorio con los hipervínculos de las noticias por día. El header tiene como común lo siguiente: “wsj.com/news/archive/{fecha}”. Luego se hace un tratamiento con BeautifulSoup para sacar todas las noticias; para que no se tardara tanto, se hace una ejecución en paralelo con una librería de ThreadPoolExecuter. Por último, se hace un tratamiento para entregar los datos de forma limpia, es decir, sin duplicados o datos erróneos.

Extracción de noticias de BBC: Para BBC se utilizó Scrapy. Pero hubo que hacer una variación respecto a los header. Este periódico no tiene los header comunes, así que hay que optar por otro método. Investigando encontramos que existe una página que archiva páginas; esta se llama Wayback Machine y toma snapshots de páginas en el tiempo. Para encontrar los header de las páginas a scrapear se usará la API de dicha página. Luego se usará Scrapy para scrapear los HTML y buscar los títulos de las noticia

Análisis de sentimientos:

El análisis de sentimientos se ha consolidado como una herramienta clave en la investigación financiera moderna. Comprender si las noticias transmiten un tono favorable, adverso o neutral permite estudiar posibles patrones en los mercados, anticipar reacciones de los inversores o enriquecer modelos predictivos de precios.

Sin embargo, los modelos genéricos de NLP no suelen capturar adecuadamente el vocabulario del ámbito financiero. Para superar esta limitación, el presente trabajo emplea FinBERT, un modelo entrenado específicamente con documentos financieros y capaz de reconocer matices de sentimiento dentro de este dominio.

Preparación del Entorno y Carga de Datos:

El análisis se desarrolla en un entorno basado en Jupyter Notebook, con soporte exclusivo en CPU. Se instalan librerías clave para manipulación de datos (Pandas, NumPy) , Visualización (Matplotlib, Seaborn y Plotly) , Procesamiento con modelos Transformers (PyTorch y Hugging Face)

Tras configurar las rutas del proyecto, se cargan noticias previamente procesadas y almacenadas en un archivo CSV. El conjunto abarca un rango temporal de enero de 2016 a octubre de 2025, lo que permite analizar múltiples ciclos económicos, periodos de volatilidad y eventos relevantes para el mercado del oro.

FinBERT: Modelo Especializado para Finanzas:

Se emplea el modelo ProsusAI/finbert, un modelo derivado de BERT pero ajustado con textos financieros. FinBERT clasifica cada texto en tres categorías: positivo , negativo y neutral.

Antes de procesar el dataset completo, se realiza una prueba con varios titulares simulados. Los resultados demuestran que el modelo identifica correctamente patrones frecuentes en el lenguaje financiero, como repuntes de precios, caídas motivadas por el dólar o señales de confianza por parte de bancos centrales.

Procesamiento por Lotes y Evaluación del Sentimiento:

Dado el volumen de datos, el análisis de sentimientos se realiza mediante un sistema de procesamiento por lotes, lo cual optimiza el rendimiento en CPU. Se procesan los titulares en bloques de 16 textos, registrando tanto la etiqueta asignada como el nivel de confianza del modelo.

Una vez obtenido el sentimiento de cada noticia, se integran los resultados al DataFrame de trabajo y se mapean las etiquetas originales a una versión estandarizada en español: *Positivo*, *Neutral* y *Negativo*.

Resultados Estadísticos de la distribución de Sentimientos:

La distribución global de sentimientos revela una preponderancia de noticias neutrales y negativas como se observa en la siguiente tabla:

Sentimiento	Cantidad	Porcentaje
Neutro	8007	42.6 %
Negativo	6684	35.6%
Positivo	4085	21.7%

Este patrón es coherente con la narrativa característica del periodismo financiero, donde abundan reportes descriptivos y alertas sobre riesgos.

Intensidad y Confianza del Modelo:

Se analizan también los puntajes de confianza (scores) asociados a cada predicción: Los sentimientos negativos presentan la mayor confianza promedio (0.79), lo que indica que FinBERT identifica con claridad expresiones de preocupación o advertencia. Los titulares neutrales también exhiben alta confianza (0.78), acorde a su frecuencia y los positivos tienen la menor confianza relativa (0.74), reflejando que el optimismo financiero suele expresarse con mayor ambigüedad.

Modelo LSTM

El modelo LSTM (Long Short-Term Memory) es un tipo de red neuronal diseñada especialmente para trabajar con datos que cambian en el tiempo, como series financieras, señales o lenguaje. A diferencia de las redes recurrentes tradicionales, que suelen “olvidar” información cuando las secuencias son largas, las LSTM incorporan un mecanismo de memoria que les permite recordar patrones importantes durante más tiempo.

La idea central es que, cuando un modelo intenta predecir una serie temporal, no solo necesita ver el punto inmediatamente anterior, a veces los eventos pasados, incluso lejanos, influyen en lo que ocurrirá después. Las LSTM fueron creadas para manejar justamente ese tipo de dependencias.

Una LSTM utiliza una estructura interna compuesta por “puertas”, que actúan como filtros capaces de decidir cuánta información debe recordarse, cuánta debe olvidarse y cuánta debe actualizarse en cada paso. Estas puertas funcionan de manera análoga a interruptores que se activan o desactivan según lo que convenga al aprendizaje [1,2].

Componentes Principales

1. Puerta de Olvido (Forget) → Decide qué información descartar
2. Puerta de Entrada (Input) → Decide qué nueva información guardar
3. Puerta de Salida (Output) → Decide qué información pasa al output

En conjunto, estas tres puertas permiten que la LSTM mantenga una “memoria regulada”, recordando solo lo necesario y olvidando lo que dejó de ser útil. Gracias a este mecanismo, este tipo de red es especialmente eficaz para predecir series financieras como el precio del oro, donde las tendencias pasadas influyen en el presente, pero no toda la información histórica es igual de importante [1,2].

Entrenamiento del modelo

Manejo de los datos

Después de cargar el archivo CSV con los precios del oro (XAU/USD) en intervalos de una hora, se realizó una limpieza inicial del campo de fecha eliminando el texto “UTC” y convirtiendo la columna a formato datetime, para luego establecerla como índice temporal del DataFrame. Con el índice ya en formato de fecha, se analizó la disponibilidad de información contabilizando cuántos registros horarios había por día y comparándolos con el valor esperado (24 por día),

identificando así que todos los días estaban incompletos. Por esto, la serie horaria se remuestreó a una frecuencia diaria utilizando las reglas estándar de agregación OHLCV: el primer valor como apertura, el máximo como el *high*, el mínimo como el *low*, el último como cierre y la suma del volumen. Finalmente, se aplicó un *forward fill* a las columnas de precios para completar días con datos faltantes y garantizar la continuidad temporal necesaria antes de entrenar el modelo.

División del Dataset

Para el entrenamiento, se seleccionaron todas las observaciones comprendidas entre 2016 y 2023, abarcando un período amplio y relativamente estable del comportamiento del precio del oro. Por otro lado, el conjunto de prueba se construyó exclusivamente con los datos posteriores a 2023, es decir, un período más reciente y no visto por el modelo. Esta división permite evaluar el desempeño del modelo en un escenario estrictamente futuro, respetando el orden temporal y evitando cualquier fuga de información (*data leakage*).

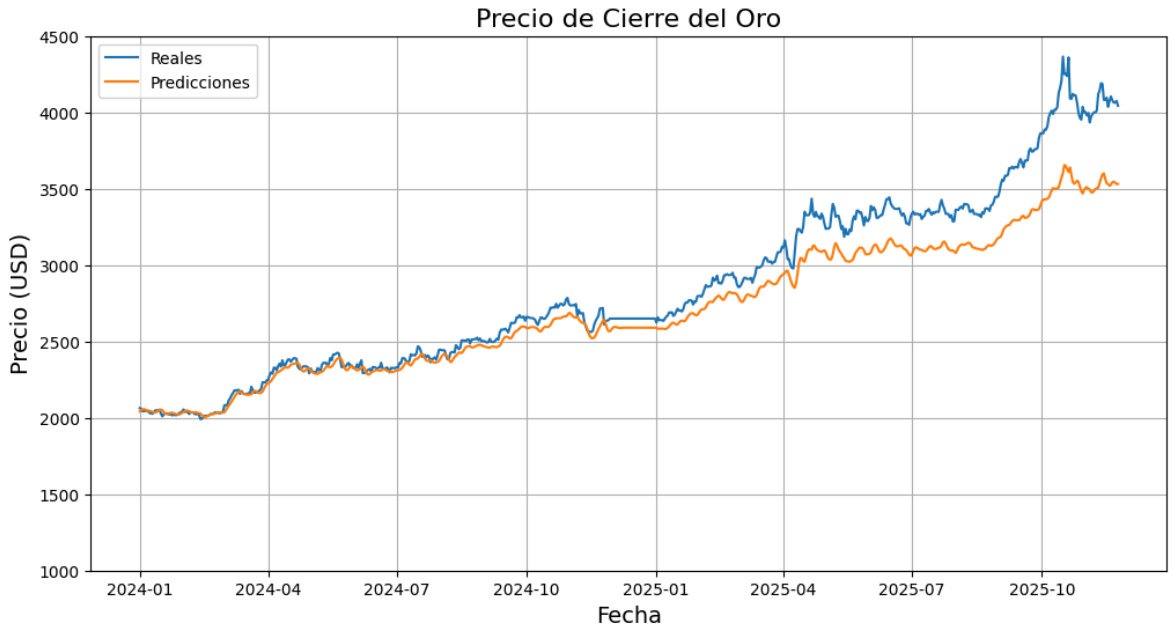
Construcción del modelo

Para construir el modelo predictivo, primero se aplicó un escalado Min-Max a la serie de precios de cierre del conjunto de entrenamiento, normalizando los valores dentro del rango [0, 1]. Este paso es fundamental en redes neuronales como las LSTM, ya que su desempeño depende de que las entradas estén en una escala numérica controlada. Luego, se generaron las secuencias temporales necesarias para el entrenamiento: cada muestra de entrada está formada por los 85 días previos de precios escalados, mientras que la etiqueta corresponde al precio del día siguiente. Estas secuencias se reorganizaron en un formato tridimensional, tal como exige la arquitectura LSTM (muestras, pasos temporales, características).

Con los datos preparados, se diseñó un modelo secuencial compuesto por tres capas LSTM con 256, 128 y 64 unidades respectivamente. Las dos primeras capas devuelven secuencias completas para permitir un aprendizaje profundo de patrones temporales, mientras que la última solo entrega el estado final. Entre cada capa se incorporaron capas Dropout que desactivan el 20% de las neuronas de manera aleatoria para reducir el sobreajuste. Finalmente, se añadió una capa densa con una única unidad para generar el valor predicho. El modelo se compiló utilizando el optimizador Adam y la función de pérdida MSE, junto con métricas adicionales como MAE y MAPE.

El entrenamiento se realizó durante un máximo de 32 épocas, empleando un tamaño de batch de 32 muestras y un callback de *early stopping*, configurado para detener el entrenamiento automáticamente si no se observaban mejoras después de varias iteraciones. Esto evitó el sobreajuste y permitió conservar los mejores pesos obtenidos. Una vez entrenado, el modelo fue guardado en formato Keras para su uso posterior.

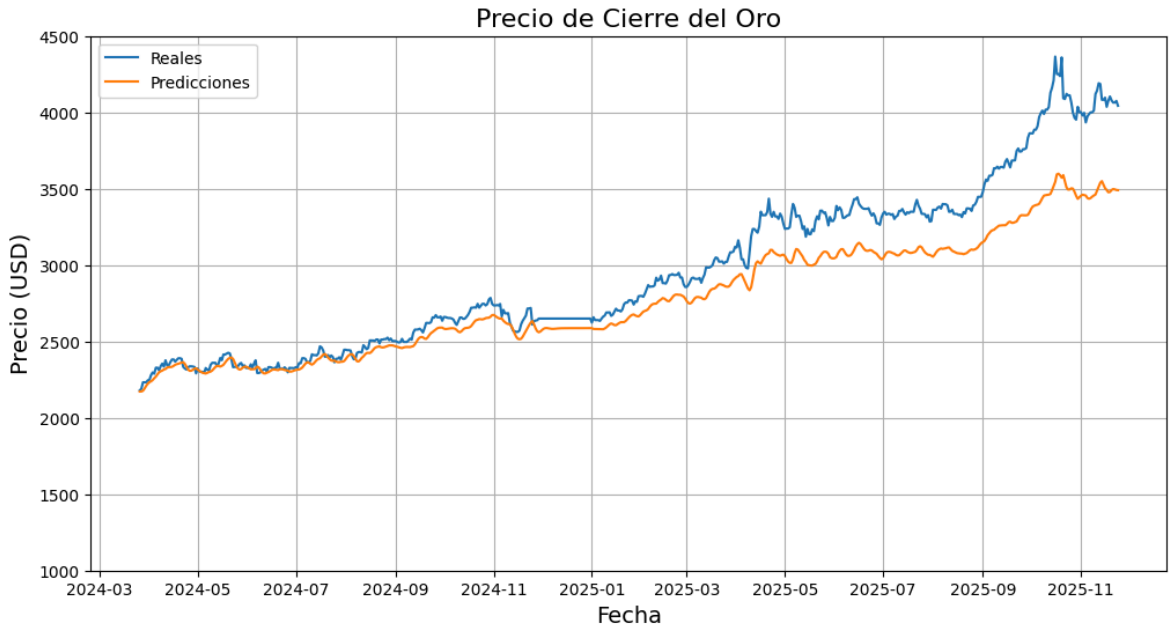
Para realizar predicciones sobre el conjunto de prueba, se extrajeron los últimos valores necesarios para generar las ventanas de entrada, añadiendo los 85 días previos correspondientes. Estos valores se transformaron nuevamente con el escalador Min-Max utilizado en el entrenamiento, asegurando coherencia en la escala antes de alimentar el modelo. Con esto, el sistema quedó preparado para generar predicciones sobre el período no visto por la red.



El modelo LSTM muestra un buen entendimiento de la tendencia general del precio del oro, algo que se refleja en su R^2 de 0.87, indicando que capta gran parte del comportamiento de la serie. También logra un MAPE del 4.27%, que significa que, en promedio, el error relativo no es muy alto. Sin embargo, cuando se mira el error en dólares, se tiene un RMSE de unos 209 USD y un MAE de 142 USD muestran que, aunque el modelo sigue bien la forma general de la curva, puede equivocarse por más de cien dólares al predecir un valor puntual. Esto vuelve el modelo útil para analizar si la tendencia va al alza o a la baja, pero no lo hace suficientemente preciso para predicciones exactas o decisiones de trading delicadas.

Construcción del modelo con múltiples características

En la versión multivariable del modelo se siguió el mismo procedimiento, pero incluyendo también el volumen como segunda característica. Primero se escalaron ambas columnas de características (Close y Volume), usando un Min-MaxScaler entrenado únicamente con los datos de entrenamiento. Luego, se construyeron ventanas de 85 días donde cada entrada contenía la secuencia conjunta de precios y volúmenes. Estas secuencias se reorganizaron en el formato requerido por la LSTM, que ahora recibía dos características por paso temporal. Finalmente, el modelo generó las predicciones del precio de cierre y, para volver a la escala original, se reconstruyó una matriz con la predicción del precio y una columna ficticia de volumen, permitiendo aplicar correctamente la transformación inversa del escalador. Esto permitió evaluar si incluir el volumen mejoraba la capacidad predictiva del modelo.



Los resultados del modelo multivariado muestran que, aunque se agregó una segunda característica (el volumen), el desempeño empeoró en comparación con el modelo que usaba únicamente el precio de cierre. El RMSE y el MAE aumentaron de forma notable, lo que indica que los errores absolutos y cuadrados crecieron. El MAPE también subió, pasando de alrededor del 4.27% al 5.22%, lo que significa que las predicciones son proporcionalmente menos precisas. Además, el R^2 disminuyó a 0.79, mostrando que el modelo explica menos variabilidad del precio real del oro.

Esto demuestra que el modelo con más características no necesariamente aprende mejor, y en este caso, el volumen parece no aportar información útil para predecir el precio futuro del oro a una escala diaria.

Conclusiones

El modelo predictivo basado en redes LSTM logró aprender bastante bien cómo se había comportado históricamente el precio del oro. Mostró un MAPE relativamente bajo y un R^2 alto, lo que indica que sí entendió los patrones del pasado. Sin embargo, cuando se observan métricas como el RMSE y el MAE, se nota que los errores absolutos siguen siendo altos, lo cual refleja que, aunque el modelo captura la tendencia general, tiene dificultades para aproximarse con precisión a los valores reales.

La causa principal parece estar en la naturaleza de los datos. Al observar la gráfica que separa el conjunto de entrenamiento y el de prueba, se puede observar que entre 2016 y 2023 el precio del oro tuvo un crecimiento moderado y bastante estable, mientras que en el período de prueba aparece una subida abrupta y completamente distinta al comportamiento histórico. Ese salto tan fuerte representa un cambio que no depende de las variables utilizadas, y por lo tanto es algo que ningún modelo puede anticipar sólo a partir de datos pasados.

Básicamente, el modelo aprendió de datos estables, pero se le pidió predecir un escenario completamente diferente. Aun así, el modelo logró seguir de forma razonable la dirección de la tendencia, aunque no alcanzó la magnitud real del incremento.

Bibliografía

[1] Codificando Bits. (s. f.). ¿Qué son las Redes LSTM? Codificando Bits. https://codificandobits.com/blog/redes-lstm/

[2] GeeksforGeeks. (2025, octubre 7). What is LSTM – Long Short Term Memory? GeeksforGeeks. https://www.geeksforgeeks.org/deep-learning/deep-learning-introduction-to-long-short-term-memory/