Universidad Europea

UNIVERSIDAD EUROPEA DE MADRID

ESCUELA DE ARQUITECTURA, INGENIERÍA Y DISEÑO

MÁSTER UNIVERSITARIO EN ANÁLISIS DE DATOS MASIVOS (BIG DATA)

TRABAJO FIN DE MÁSTER

ANÁLISIS DE MODELOS PREDICTIVOS EN BOLSA

ROBERTO PRIETO PRIETO

Dirigido por

Ingeniero SAMUEL GARCÍA SABOYA

CURSO 2024-2025

TÍTULO: ANÁLISIS DE MODELOS PREDICTIVOS EN BOLSA

AUTOR: ROBERTO PRIETO PRIETO

TITULACIÓN: MÁSTER UNIVERSITARIO EN ANÁLISIS DE DATOS MASIVOS (BIG DATA)

DIRECTOR DEL PROYECTO: INGENIERO SAMUEL GARCÍA SABOYA

FECHA: SEPTIEMBRE DE 2025

RESUMEN

La alta volatilidad de los mercados financieros dificulta la generación de predicciones bursátiles consistentes. Este proyecto aborda esta problemática mediante una comparativa rigurosa de enfoques estadísticos tradicionales (ARIMA, Prophet y SARIMAX), modelos de aprendizaje automático (Random Forest y XGBoost) y modelos de aprendizaje profundo (LSTM y Transformer), aplicados al caso de NVIDIA, empresa líder en inteligencia artificial y con la mayor capitalización bursátil de la historia.

Para ello, se construyó un data warehouse ligero en MongoDB, integrando datos históricos, indicadores técnicos, cotizaciones de otras Big Tech, índices bursátiles globales, indicadores económicos y análisis de sentimiento basado en noticias y Google Trends.

Los modelos fueron entrenados y evaluados en el periodo 2015–2025 mediante validación estática y backtesting, utilizando como referencia un predictor base (aleatorio y de persistencia). El mejor rendimiento global se obtuvo con un modelo SARIMAX, que superó al predictor base en un 17%. En regresión, el mejor resultado correspondió a XGBoost (+2%), mientras que en clasificación destacó un Transformer (+30%).

El proyecto concluye que, dado que algunos modelos han logrado superar al predictor base, los avances recientes en técnicas de inteligencia artificial permiten identificar patrones explotables que cuestionan parcialmente la hipótesis de mercados eficientes. Asimismo, demuestra la viabilidad técnica, económica y legal de una solución íntegramente desarrollada con software libre.

Como entrega final, se desarrolló una aplicación web en Flask que realiza una simulación de operativa bursátil sobre un período histórico de dos meses, estableciendo así las bases para su futura integración en sistemas de negociación automatizada.

Palabras clave: predicción bursátil, NVIDIA, series temporales, aprendizaje automático, aprendizaje profundo, análisis de sentimiento

ABSTRACT

The high volatility of financial markets makes it difficult to generate consistent stock market predictions. This project addresses this challenge through a rigorous comparison of traditional statistical approaches (ARIMA, Prophet, and SARIMAX), machine learning models (Random Forest and XGBoost), and deep learning models (LSTM and Transformer), applied to the case of NVIDIA, a global leader in artificial intelligence and the company with the highest market capitalization in history.

To this end, a lightweight data warehouse was built in MongoDB, integrating historical data, technical indicators, stock prices of other Big Tech companies, global stock indices, economic indicators, and sentiment analysis based on news and Google Trends.

The models were trained and evaluated over the period 2015–2025 using static validation and backtesting, with a baseline predictor (random and persistence) as a reference. The best overall performance was achieved by a SARIMAX model, which outperformed the baseline by 17%. In regression tasks, the best result was obtained with XGBoost (+2%), while in classification a Transformer achieved the highest improvement (+30%).

The project concludes that, since some models managed to outperform the baseline, recent advances in artificial intelligence techniques make it possible to identify exploitable patterns that partially challenge the Efficient Market Hypothesis. It also demonstrates the technical, economic, and legal feasibility of a solution developed entirely with open-source software.

The final deliverable is a Flask web application that simulates daily trading over a two-month historical period, setting the stage for future integration into automated trading systems.

Keywords: stock market prediction, NVIDIA, time series, machine learning, deep learning, sentiment analysis

AGRADECIMIENTOS

Quisiera expresar mi más sincero agradecimiento a todas las personas que han contribuido de alguna manera a la realización de este Trabajo Fin de Máster. En especial, agradezco la dedicación, orientación y apoyo de mi tutor don Samuel, cuya experiencia y disponibilidad han sido fundamentales en cada fase del proyecto. Su acompañamiento ha supuesto una guía clave tanto en los aspectos técnicos como en los conceptuales del proyecto.

También deseo agradecer a mi familia, por su paciencia, comprensión y apoyo incondicional durante todo el proceso, así como a mis amigos, que han estado siempre cerca aportando ánimo, motivación y valiosas conversaciones en los momentos más exigentes. Sin su presencia, este recorrido habría sido mucho más difícil. A todos, gracias por formar parte de este camino.

"*Sin datos, solo eres otra persona con una opinión*." **W. Edwards Deming**

TABLA RESUMEN

	DATOS
Nombre y apellidos:	ROBERTO PRIETO PRIETO
Título del proyecto:	ANÁLISIS DE MODELOS PREDICTIVOS EN BOLSA
Directores del proyecto:	SAMUEL GARCÍA SABOYA
El proyecto ha implementado un producto: (esta entrada se puede marcar junto a la siguiente)	SI (APLICACIÓN DE SIMULACIÓN QUE UTILIZA LOS MEJORES MODELOS Y CONJUNTO DE DATOS, DESPLEGADA EN CONTENEDORES QUE SIMULA LA PREDICCIÓN DIARIA SOBRE UN PERIODO HISTÓRICO DE DOS MESES)
El proyecto ha consistido en el desarrollo de una investigación o innovación: (esta entrada se puede marcar junto a la anterior)	SI
Objetivo general del proyecto:	DESARROLLAR UN SISTEMA PREDICTIVO QUE PREDIGA EL PRECIO DE CIERRE DIARIO DE LAS ACCIONES DE NVIDIA, EVALUANDO SI PUEDE SUPERAR A UN PREDICTOR BASE MEDIANTE TÉCNICAS ESTADÍSTICAS AVANZADAS Y DE INTELIGENCIA ARTIFICIAL, COMPARANDO MODELOS E INTEGRANDO DIVERSAS FUENTES DE DATOS.

Índice

Índice de Figuras

Índice de Tablas

Capítulo 1. RESUMEN DEL PROYECTO

1.1 Contexto y justificación

Este proyecto se enmarca en la intersección entre inteligencia artificial y análisis financiero, impulsado por una inquietud personal del autor y una motivación académica orientada a explorar si los métodos tradicionales y las técnicas recientes pueden superar la aparente aleatoriedad de los mercados bursátiles. Su principal aporte es una comparativa de modelos predictivos y conjuntos de datos aplicados a NVIDIA, con el objetivo de identificar la combinación más eficaz para la predicción diaria del precio de cierre.

1.2 Planteamiento del problema

El análisis del estado del arte muestra que, a pesar de los avances en metodologías predictivas, desde modelos estadísticos hasta técnicas de aprendizaje profundo (Deep Learning) [1], la complejidad y aleatoriedad de los mercados financieros impiden obtener predicciones bursátiles fiables. En este contexto, persiste la ausencia de una comparativa sistemática y completa que evalúe de forma objetiva el rendimiento de distintos modelos predictivos aplicados a un mismo activo financiero y utilizando múltiples fuentes de información. Para cubrir esta brecha, el presente proyecto realiza una evaluación estructurada de modelos predictivos sobre datos diversos aplicados a NVIDIA. La *Tabla 2* ilustra cómo este proyecto se diferencia de estudios anteriores, al ofrecer una comparativa que no había sido abordada hasta ahora en la literatura.

1.3 Objetivos del proyecto

El objetivo es desarrollar un sistema predictivo que prediga el precio de cierre diario de las acciones de NVIDIA, evaluando si puede superar a un predictor base mediante técnicas estadísticas avanzadas y aprendizaje automático y profundo. Se comparan distintos modelos y se analiza el impacto de incorporar conjuntos de datos heterogéneos: indicadores técnicos y económicos, Big Tech, índices bursátiles y análisis de sentimiento. Como resultado, se elabora una tabla comparativa del rendimiento de todos los modelos y se construye una aplicación que simula la predicción diaria. El proyecto también ha constituido una experiencia formativa para el autor, que inició el trabajo con conocimientos limitados en el ámbito financiero.

1.4 Resultados obtenidos

Los resultados, comparados con el predictor base, muestran que en condiciones realistas de backtesting el modelo SARIMAX ofrece el mejor rendimiento global: empata en regresión y mejora un 17% en clasificación. De forma aislada, en regresión XGBoost alcanza una mejora del 2% y en clasificación el Transformer logra un incremento del 30%. Los conjuntos de datos más relevantes son los directos, seguidos del análisis de sentimiento e indicadores económicos, después los índices bursátiles y, en último lugar, los indicadores técnicos y las Big Tech. Finalmente, se ha desarrollado una aplicación en Flask y Docker que simula la predicción diaria sobre un período histórico de dos meses utilizando los mejores modelos identificados.

Capítulo 2. ANTECEDENTES / ESTADO DEL ARTE

Este capítulo recoge los antecedentes y el estado del arte en la predicción bursátil, y establece el marco teórico en el que se fundamenta este proyecto. Se abordan los avances metodológicos e históricos, así como la contextualización y la formulación del problema que orienta el estudio.

2.1 Estado del arte

El análisis del estado del arte en la predicción de precios bursátiles está estrechamente relacionado con la evolución de los algoritmos de predicción de series temporales. En este apartado se revisan aspectos clave para entender la evolución y el estado actual.

2.1.1 Teorías económicas clásicas

En la década de 1970, diversos estudios sentaron las bases de teorías que siguen vigentes en la actualidad:

- **Teoría de los Mercados Eficientes (HME):**
  En 1970, el estudio *"Efficient Capital Markets: A Review of the Theory and Empirical Work"* [2] definió la Teoría de los Mercados Eficientes. Esta teoría se ha erigido como un pilar fundamental para comprender la dinámica de los mercados financieros, postulando que los precios de los activos reflejan toda la información disponible. En consecuencia, es imposible obtener beneficios sistemáticos mediante el análisis de información histórica o pública.

Figura 1 - Ciclo de la teoría de Mercados Eficientes (fuente: elaboración propia).

- **Teoría del Camino Aleatorio:**
  En 1973, la obra *"A Random Walk Down Wall Street"* [3], introdujo la Teoría del Camino Aleatorio, una de las primeras aproximaciones para explicar el comportamiento de los precios bursátiles. Según esta teoría, los movimientos de los precios siguen un proceso aleatorio, en el que las variaciones futuras son independientes de las observadas en el pasado. Esta hipótesis implica limitaciones a la hora de predecir futuros movimientos basándose exclusivamente en datos históricos.

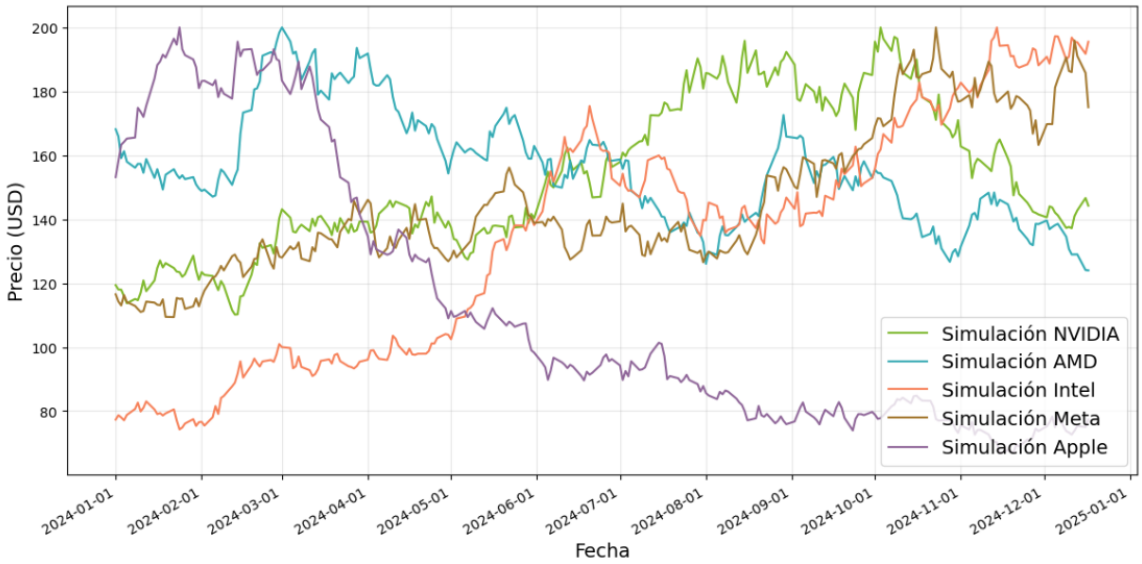


Figura 2 - Camino Aleatorio simulando precio de acciones (fuente: elaboración propia).

De las teorías expuestas se deduce que la autorregulación y el carácter inherentemente aleatorio de los mercados financieros dificultan el desarrollo de un mecanismo de predicción completamente fiable. La elevada complejidad del entorno y la presencia de variables desconocidas complican la elaboración de predicciones prácticas y válidas, sin descartar la posible existencia de patrones subyacentes.

2.1.2 Evolución de algoritmos en predicción de series temporales

En las últimas décadas, las técnicas empleadas en la predicción de series temporales han experimentado una transformación notable, motivada principalmente por avances en computación, disponibilidad de datos y nuevas metodologías en inteligencia artificial. Inicialmente dominadas por métodos estadísticos clásicos, estas técnicas han evolucionado progresivamente hacia modelos más sofisticados capaces de gestionar relaciones complejas. La irrupción del aprendizaje profundo y, más recientemente, la aparición de arquitecturas basadas en transformadores y modelos de lenguaje a gran escala, ha ampliado significativamente las capacidades predictivas, integrando información contextual de diversas fuentes. Esta evolución se esquematiza en la *Figura 3*.

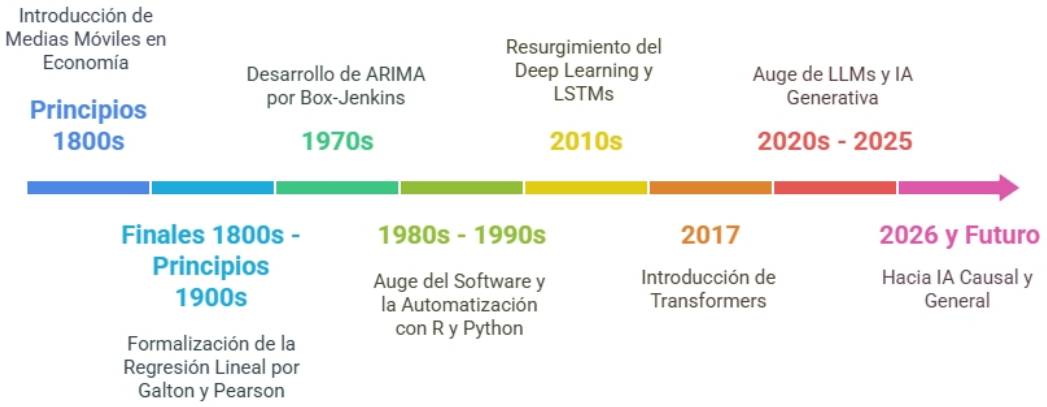


Figura 3 - Evolución de los principales métodos de predicción de series temporales (fuente: elaboración propia).

A continuación, se presentan brevemente tres grandes bloques de técnicas que han marcado esta evolución:

1. **Métodos estadísticos tradicionales:**
   En un primer momento, los modelos estadísticos clásicos constituyeron la base principal del análisis predictivo. Destacan especialmente técnicas simples como la media móvil [4] y regresión lineal, así como métodos más complejos y utilizados como el modelo autorregresivo integrado de promedios móviles (ARIMA) [5]. Estos modelos parten de la hipótesis de que los patrones históricos explican de manera significativa los movimientos futuros del mercado bajo una estructura predominantemente lineal. Actualmente, variantes avanzadas como SARIMA (ARIMA estacional) y SARIMAX (SARIMA con variables exógenas) continúan vigentes, aunque con limitaciones claras frente a relaciones complejas no lineales.
   Un artículo sobre ARIMA y predicción bursátil puede consultarse en [6].

2. **Aprendizaje profundo:**
   La popularización del aprendizaje profundo ha supuesto un punto de inflexión clave en la capacidad predictiva de modelos financieros. Las redes neuronales recurrentes (RNN) [7] y especialmente las redes Long Short-Term Memory (LSTM) [8], han mostrado una capacidad sobresaliente para identificar patrones no lineales complejos y capturar dependencias temporales significativas. Estas técnicas permiten incorporar, además de datos numéricos tradicionales, información procedente de fuentes externas, tales como noticias financieras o sentimiento en redes sociales, proporcionando una ventaja significativa frente a enfoques anteriores.
   Un artículo destacado del uso de LSTM en análisis financiero puede encontrarse en la *Tabla 1* [9].

3. **Transformers y modelos de lenguaje a gran escala (LLM):**
   Más recientemente, las arquitecturas basadas en transformadores, que incluyen mecanismos avanzados de atención [10], han revolucionado la capacidad de modelado predictivo. Estos modelos pueden integrar eficazmente grandes volúmenes de información textual y contextual, mejorando considerablemente tanto la precisión como la generalización en series temporales financieras. Además, la aplicación de modelos de lenguaje a gran escala (LLM) [11] a tareas financieras ha demostrado una capacidad adicional para gestionar datos complejos y heterogéneos, ofreciendo resultados predictivos aún más robustos y adaptables a diferentes contextos de mercado.
   Un artículo destacado del uso de Transformers en análisis financiero se presenta en la *Tabla 1* [12].

Finalmente, es importante señalar que la próxima frontera en el campo del análisis predictivo de series temporales financieras probablemente incluirá una mayor integración de técnicas centradas en la identificación explícita de relaciones causales, buscando combinar la interpretabilidad de los métodos estadísticos tradicionales con la potencia predictiva de las redes neuronales y los Transformers.

Este enfoque híbrido promete mejorar aún más la precisión, interpretabilidad y robustez de las predicciones.

2.1.3 Estudios recientes y actividades relacionadas

Actualmente se observa un desarrollo de nuevos sistemas de predicción bursátil sin precedentes: la publicación de artículos especializados se ha acelerado gracias a la disponibilidad de GPUs de uso general, repositorios de código abiertos y conjuntos de datos cada vez más accesibles.

La *Tabla 1* sintetiza esta actividad recogiendo los trabajos más influyentes y cercanos al presente proyecto, ordenados del más reciente al más antiguo.

Fecha	Título, referencia y descripción
2025-02	***"FinMamba: Market-Aware Graph Enhanced Multi-Level Mamba for Stock Movement Prediction"*** [13] Combina datos individuales de acciones con sus correlaciones mediante un marco Mamba-GNN [14] de grafos dinámicos y atención selectiva, capaz de capturar dependencias entre activos y patrones a varias escalas pese al alto ruido del mercado. El código aún no es público, aunque los autores anuncian su futura liberación.
2024-02	***"TimeXer: Empowering Transformers for Time Series Forecasting with Exogenous Variables"*** [15] Este trabajo propone TimeXer, un enfoque que incorpora variables exógenas (información externa relacionada) para mejorar la precisión de la predicción. A través de mecanismos avanzados de atención, TimeXer combina eficazmente información interna y externa, aprendiendo representaciones globales que capturan relaciones causales. El código de TimeXer [16] está disponible en GitHub.
2023-10	***"TimeGPT-1"*** [17] TimeGPT se presenta como el primer modelo base para series temporales capaz de generar predicciones precisas en conjuntos de datos heterogéneos, incluso sin haber sido entrenado específicamente en ellos. El código de TimeGPT [18] está disponible en GitHub.
2022-10	**"*A Time Series is Worth 64 Words: Long-term Forecasting with Transformers*"** [12] PatchTST es un diseño eficiente que utiliza modelos Transformer para la predicción de series temporales multivariadas y el aprendizaje de representaciones auto-supervisadas, basado en la segmentación de las series en parches y la independencia de canales. Esta estrategia retiene la información semántica local, reduce el uso de memoria en los mapas de atención y permite analizar historiales más extensos. El código de PatchTST [19] está disponible en GitHub.
2019-09	***"Stock closing price prediction based on sentiment analysis and LSTM"*** [9] La predicción bursátil es un reto por el ruido y la volatilidad. Se propone un enfoque de aprendizaje profundo que integra el sentimiento de inversores, usando EMD para abordar la no linealidad y un LSTM con atención para optimizar las predicciones.

Tabla 1 - Lista de publicaciones y trabajos más relevantes de los últimos años.

Asimismo, destacan diversas competiciones en el ámbito bursátil que impulsan el desarrollo de sistemas de inversión automatizados basados en algoritmos. Por ejemplo:

- "**Robotrader**": [20]
  Desde 2010, la Universidad Politécnica de Madrid organiza una competición centrada en el diseño de sistemas algorítmicos para operar en bolsa, en la que se imparten charlas técnicas y se facilitan contactos con expertos en la materia.

- "**Numerai Tournament**": [21]
  De forma continua, científicos de datos de todo el mundo desarrollan modelos predictivos para generar estrategias de inversión basadas en datos financieros actualizados regularmente.

2.1.4 Comparativa con trabajos existentes

En la *Tabla 2*, se comparan las publicaciones descritas en la *Tabla 1* con este proyecto.

Estudio	Métodos	Datos	Predicción
2025 ***"Análisis de modelos predictivos en bolsa"***	✅ Estadísticos tradicionales ✅ Aprendizaje automático ✅ Aprendizaje profundo	✅ Directos ✅ Indicadores técnicos ✅ Big Tech e índices bursátiles ✅ Indicadores económicos ✅ Análisis de sentimiento	✅ NVIDIA ❌ Generalista
2025 ***"FinMamba: Market-Aware Graph Enhanced Multi-Level Mamba for Stock Movement Prediction"***	❌ Estadísticos tradicionales ✅ Aprendizaje automático ❌ Aprendizaje profundo	✅ Directos ❌ Indicadores técnicos ✅ Big Tech e índices bursátiles ✅ Indicadores económicos ❌ Análisis de sentimiento	❌ NVIDIA ✅ Generalista
2024 ***"TimeXer: Empowering Transformers for Time Series Forecasting with Exogenous Variables"***	❌ Estadísticos tradicionales ❌ Aprendizaje automático ✅ Aprendizaje profundo	✅ Directos ❓ Indicadores técnicos ❓ Big Tech e índices bursátiles ❓ Indicadores económicos ❌ Análisis de sentimiento	❌ NVIDIA ❓ Generalista
2023 ***"TimeGPT-1"***	❌ Estadísticos tradicionales ✅ Aprendizaje automático ❌ Aprendizaje profundo	✅ Directos ❌ Indicadores técnicos ✅ Big Tech e índices bursátiles ✅ Indicadores económicos ❌ Análisis de sentimiento	❌ NVIDIA ✅ Generalista
2022 ***"A Time Series is Worth 64 Words: Long-term Forecasting with Transformers"***	❌ Estadísticos tradicionales ❌ Aprendizaje automático ✅ Aprendizaje profundo	✅ Directos ❌ Indicadores técnicos ❌ Big Tech e índices bursátiles ❌ Indicadores económicos ❌ Análisis de sentimiento	❌ NVIDIA ✅ Generalista
2019 ***"Stock closing price prediction based on sentiment analysis and LSTM"***	❌ Estadísticos tradicionales ✅ Aprendizaje automático ❌ Aprendizaje profundo	✅ Directos ❌ Indicadores técnicos ❌ Big Tech e índices bursátiles ❌ Indicadores económicos ✅ Análisis de sentimiento	❌ NVIDIA ✅ Generalista

Tabla 2 - Comparativa entre estudios previos y el presente proyecto.

2.1.5 Contribución y novedad del presente proyecto

A la luz del análisis anterior, puede observarse que la mayoría de publicaciones en el ámbito de la predicción bursátil se centran en un único enfoque metodológico o en un conjunto de datos específico, limitando la posibilidad de realizar comparaciones directas entre distintas técnicas y fuentes de información. En este contexto, el presente proyecto aporta un enfoque diferencial al proponer una comparativa sistemática entre distintos tipos de modelos predictivos, incluyendo métodos estadísticos tradicionales, algoritmos de aprendizaje profundo y arquitecturas basadas en Transformers, aplicados a varios conjuntos de datos heterogéneos.

A diferencia de estudios previos más generalistas, este proyecto se enfoca exclusivamente en la predicción del precio de cierre diario de las acciones de NVIDIA empresa que el 9 de julio de 2025 alcanzó la mayor capitalización bursátil de la historia [22]. Este enfoque permite profundizar en el análisis de un activo altamente representativo del sector tecnológico y del comportamiento bursátil contemporáneo.

La principal contribución de este proyecto radica en la unificación de estos enfoques dentro de un mismo estudio, con el objetivo de predecir el precio de cierre del día siguiente. Esta evaluación cruzada permite determinar, de forma empírica y objetiva, qué combinación de modelo y conjunto de variables ofrece un mejor rendimiento, facilitando así la selección óptima para futuras aplicaciones de predicción diaria automatizada.

Hasta la fecha, no se han identificado investigaciones que integren una comparativa tan amplia, sistemática y enfocada específicamente en una única acción tecnológica de referencia. Por tanto, el presente proyecto representa un avance significativo en términos de exhaustividad metodológica y utilidad práctica, al proporcionar conclusiones basadas en resultados numéricos objetivos y comparables entre distintos enfoques.

2.2 Contexto y justificación

El presente proyecto se enmarca en el contexto de la creciente intersección entre la inteligencia artificial y el análisis financiero. En los últimos años, la rápida evolución de los algoritmos de aprendizaje profundo ha abierto nuevas posibilidades en la predicción de series temporales, especialmente en mercados tan volátiles y complejos como el bursátil. Un ejemplo de esta interacción se produjo el 27 de enero de 2025, cuando la presentación del modelo de inteligencia artificial (IA) gratuito de China, DeepSeek R1 [23], provocó una caída del 17 % en el precio de las acciones de NVIDIA [24], lo que generó pérdidas estimadas en 590 000 millones de dólares y afectó significativamente al índice compuesto NASDAQ.

La idea de este proyecto surge de la inquietud personal de su autor, que desde temprana edad ha seguido de cerca el desarrollo de la inteligencia artificial y que ha realizado otros proyectos

como *"Pac-Man Evolution"* [25], y de una motivación académica centrada en evaluar si los métodos tradicionales y, más recientemente, las técnicas basadas en aprendizaje profundo (como LSTM y Transformers) son capaces de superar la aleatoriedad inherente a los mercados financieros. A ello se suma la necesidad de crear una comparativa completa y estructurada sobre distintos modelos aplicados a la predicción del precio de cierre de las acciones de NVIDIA.

Este proyecto aporta varios aspectos al campo de estudio:

- **Análisis comparativo de metodologías de predicción:**
  Al evaluar y comparar modelos como ARIMA, Prophet, Random Forest, XGBoost, LSTM y Transformers, se determinará cuál es el más adecuado para capturar patrones complejos y dinámicos en los datos financieros.

- **Integración de múltiples fuentes de información:**
  Se explorará el impacto de incorporar variables adicionales, como datos históricos de NVIDIA, datos de las Big Tech, índices bursátiles globales, indicadores económicos claves y eventos relevantes, incluyendo análisis de sentimiento, lo que puede aumentar la precisión predictiva.

- **Contribución al análisis financiero aplicado a una empresa de referencia**:
  Los hallazgos del proyecto podrían ayudar al desarrollo de herramientas de análisis que faciliten la toma de decisiones en el entorno financiero, demostrando que, a pesar de la alta complejidad y el carácter caótico de los mercados, es posible obtener mejoras sustanciales en la predicción mediante enfoques bien estructurados basados en datos diversos y técnicas avanzadas.

En resumen, este proyecto se justifica por la necesidad de abordar el desafío que plantea la alta incertidumbre del mercado bursátil, con un enfoque centrado en la acción de una empresa clave, aportando nuevas perspectivas metodológicas que integren lo último en inteligencia artificial para optimizar la predicción de precios y, en última instancia, contribuir al avance del análisis financiero.

2.3 Planteamiento del problema

El análisis del estado del arte ha revelado que, a pesar del desarrollo y aplicación de diversas metodologías predictivas, desde modelos estadísticos tradicionales hasta técnicas avanzadas de aprendizaje profundo, como LSTM y Transformers, existe una falta de consenso sobre qué enfoque resulta más eficaz en contextos reales. La alta complejidad y el carácter inherentemente aleatorio de los mercados financieros siguen siendo retos fundamentales para la obtención de predicciones bursátiles fiables. Teorías fundamentales como la de los Mercados Eficientes y la del Caos indican que, incluso cuando se identifican patrones subyacentes, la

evolución de los sistemas financieros puede volverse impredecible, lo que dificulta la tarea de realizar predicciones precisas y consistentes.

Además, eventos disruptivos como el impacto observado el 27 de enero de 2025 tras la presentación del modelo DeepSeek R1 ponen de manifiesto la sensibilidad del mercado ante innovaciones tecnológicas y variables externas, evidenciando la necesidad de modelos que integren no solo datos históricos sino también señales contextuales y exógenas.

En este contexto, se detecta la necesidad de realizar una comparativa sistemática entre diferentes aproximaciones predictivas, integrando múltiples fuentes de información (datos históricos, indicadores técnicos, precios de otras Big Tech, índices bursátiles, indicadores económicos y análisis de sentimiento) para evaluar su rendimiento relativo y su viabilidad práctica en la predicción del mercado.

El presente proyecto se orienta a desarrollar, evaluar y comparar distintos tipos de modelos: estadísticos tradicionales, aprendizaje profundo y Transformers, para predecir el precio de cierre diario de las acciones de NVIDIA, empresa de referencia en el sector tecnológico. El objetivo principal es determinar si alguna de estas metodologías puede superar el rendimiento de un predictor base (aleatorio y de persistencia), contrarrestando así la aleatoriedad estructural del mercado financiero.

Para ello, se utilizarán datos comprendidos entre el 5 de enero de 2015 y el 23 de mayo de 2025. El conjunto de test estará compuesto por los últimos 60 días del periodo (aproximadamente tres meses), reservados para la evaluación final.

Cada modelo será entrenado y evaluado utilizando todos los conjuntos de variables disponibles (directas, indicadores técnicos, Big Tech, índices bursátiles, indicadores económicos y de sentimiento) con el fin de identificar la combinación que ofrezca el mejor rendimiento dentro de cada enfoque. Finalmente, se seleccionará el modelo y conjunto de datos que, en conjunto, logren los mejores resultados predictivos para la estimación del precio de cierre diario.

Como parte final del proyecto, se desarrollará una pequeña aplicación que simula la predicción diaria del modelo seleccionado, generando resultados para un periodo histórico de dos meses.

Se espera que los resultados sirvan como base para futuros desarrollos de herramientas predictivas y aplicaciones automatizadas, así como una contribución metodológica relevante al campo de la predicción bursátil.

Capítulo 3. OBJETIVOS

El objetivo de este proyecto es desarrollar un sistema predictivo que prediga el precio de cierre diario de las acciones de NVIDIA, evaluando si puede superar a un predictor base mediante técnicas avanzadas de series temporales y aprendizaje automático y profundo.

3.1 Objetivos generales

Los objetivos generales son:

- Preparar los datos y establecer un sistema base como referencia para la evaluación.
- Entrenar y evaluar los modelos de predicción.
- Comparativa final y desarrollo de aplicación de simulación de predicción diaria.

3.2 Objetivos específicos

A continuación, se detallan los objetivos específicos para cada objetivo general:

- **Preparar los datos y establecer un sistema base como referencia para la evaluación**:
  - Reunir y procesar datos históricos desde 2015 hasta mayo de 2025:
    - Precios de apertura, cierre, volumen e indicadores técnicos de NVIDIA.
    - Cotizaciones de las Big Tech (las "Siete Magníficas" [26]: Google, Amazon, Apple, Meta, Microsoft, NVIDIA y Tesla), junto con AMD e Intel.
    - Principales índices bursátiles: S&P500, NASDAQ, EURO STOXX 50, etc
    - Indicadores económicos y análisis de sentimiento.
    - Creación de un data warehouse ligero [27] basado en MongoDB.
  - Aplicar técnicas de preprocesamiento: detección de valores faltantes, tratamiento de valores atípicos, análisis de distribución, escalado y normalización.
  - Implementar un predictor base: basado en uno aleatorio y de persistencia.

- **Entrenar y evaluar los modelos de predicción**:
  - Entrenar y evaluar modelos tradicionales: ARIMA, Prophet y SARIMAX.
  - Entrenar y evaluar modelos de aprendizaje automático: Random Forest y XGBoost.
  - Entrenar y evaluar modelos basados en aprendizaje profundo: LSTM y arquitecturas Transformer.
  - Probar cada modelo con distintas combinaciones de datos.
  - Validación sobre el conjunto de test y una simulación realista del último mes.

- **Comparativa final y desarrollo de aplicación de simulación de predicción diaria:**
  - Identificar la mejor combinación de modelo y datos.

- Crear una tabla comparativa con todos los modelos y configuraciones de datos y destacar aquellos que superan el predictor base.
- Desarrollar una aplicación que utilice el mejor modelo y conjunto de datos desplegada con contenedores que simule la predicción diaria sobre un período histórico de dos meses.

3.3 Beneficios del proyecto

Este proyecto tiene el potencial de aportar valor a distintos actores del ámbito financiero, tecnológico y académico. Sus principales beneficios son:

- **Identificación del mejor enfoque predictivo:**
  Permite determinar qué combinación de modelo y conjunto de datos ofrece mejores resultados al predecir el precio de cierre de las acciones de NVIDIA, una de las compañías más relevantes del panorama tecnológico actual.

- **Desarrollo de herramientas automatizadas:**
  La comparativa realizada sienta las bases para la construcción de un sistema que se entrene y actualice automáticamente, generando predicciones diarias en tiempo real, útil tanto en entornos de investigación como de inversión.

- **Aplicabilidad práctica en diferentes ámbitos:**
  - **Empresas financieras y de inversión:**
    Firmas de análisis bursátil, fondos de inversión o *fintechs* podrían utilizar los hallazgos para mejorar sus estrategias de negociación algorítmica o análisis de riesgos.

  - **Universidades y centros de investigación:**
    El enfoque comparativo y la metodología pueden ser utilizados como material de estudio, inspiración para tesis o como punto de partida para investigaciones en inteligencia artificial aplicada a series temporales.

  - **Inversores particulares y desarrolladores independientes:**
    El modelo y la aplicación de simulación pueden servir como guía o herramienta base para inversores minoristas interesados en predecir movimientos del mercado utilizando herramientas accesibles y replicables.

En conjunto, este proyecto no solo aporta una solución técnica, sino que ofrece una referencia clara y sistemática para evaluar modelos predictivos bursátiles en un contexto real y complejo como es el de NVIDIA

Capítulo 4. DESARROLLO DEL PROYECTO

Dada la estructura del equipo de desarrollo, compuesto por un único miembro, la ejecución de las tareas se ha llevado a cabo de forma secuencial. Un equipo de desarrollo más amplio habría posibilitado la paralelización de algunas actividades.

4.1 Planificación del proyecto

A continuación, en la *Tabla 3* se listan todas las actividades, duración, fecha de comienzo y fecha de fin.

Las actividades están divididas en 7 fases principales:

- **Fase de concepción**:
  Se estudia la teoría bursátil, el estado del arte y las tecnologías disponibles, definiendo la idea original del proyecto y sus objetivos estratégicos.

- **Fase de planificación:**
  Se concreta el alcance, se elabora el diagrama de Gantt y se asignan recursos y tareas, dejando documentada la hoja de ruta del proyecto.

- **Fase de diseño:**
  Se define la arquitectura de datos y modelos, se seleccionan fuentes, variables e indicadores, y se establece la metodología de evaluación que guiará las pruebas.

- **Fase de implementación:**
  Se codifican, entrenan y documentan los predictores (aleatorio, persistencia, ARIMA, Prophet, Random Forest, XGBoost, LSTM y Transformers).

- **Fase de pruebas:**
  Se validan y comparan los modelos usando el conjunto de test y una simulación realista con métricas de regresión y clasificación.

- **Desarrollo de la aplicación de simulación:**
  Se implementa y prueba una aplicación web que consume los mejores modelos y reproduce la predicción diaria del precio de cierre de NVIDIA sobre un período histórico de dos meses.

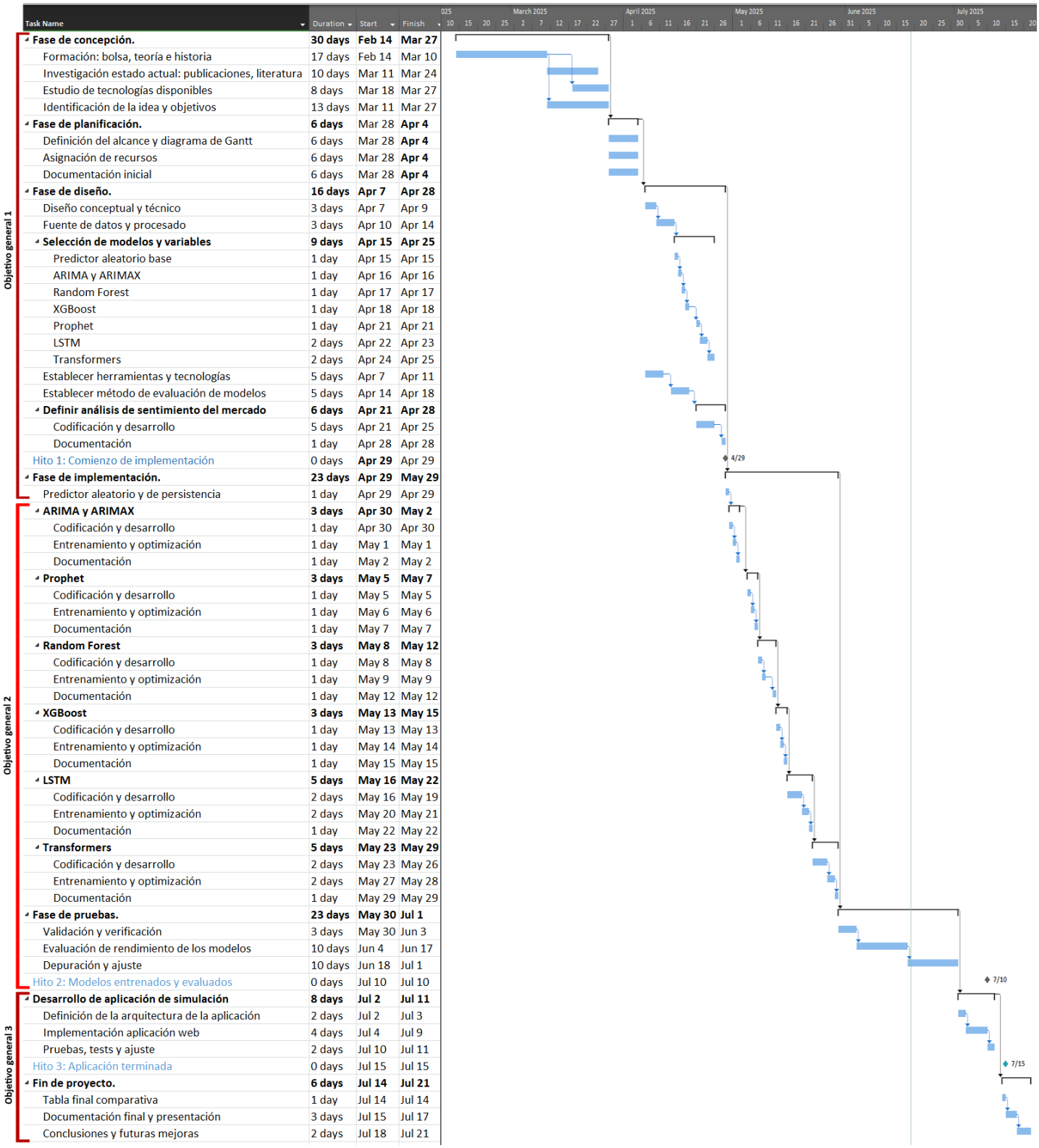
- **Fin de proyecto:**
  Se elabora la tabla comparativa final, la documentación y las conclusiones, dejando planteadas líneas de mejora y trabajos futuros.

Tarea	Duración	Comienzo	Fin
Fase de concepción.	30 días	14-Feb	27-Mar
Formación: bolsa, teoría e historia	17 días	14-Feb	10-Mar
Investigación estado actual: publicaciones, literatura	10 días	11-Mar	24-Mar
Estudio de tecnologías disponibles	8 días	18-Mar	27-Mar
Identificación de la idea y objetivos	13 días	11-Mar	27-Mar
Fase de planificación.	6 días	28-Mar	4-Apr
Definición del alcance y diagrama de Gantt	6 días	28-Mar	4-Apr
Asignación de recursos	6 días	28-Mar	4-Apr
Documentación inicial	6 días	28-Mar	4-Apr
Fase de diseño.	16 días	7-Apr	28-Apr
Diseño conceptual y técnico	3 días	7-Apr	9-Apr
Fuente de datos y procesado	3 días	10-Apr	14-Apr
Selección de modelos y variables	9 días	15-Apr	25-Apr
Predictor base (aleatorio y de persistencia)	1 día	15-Apr	15-Apr
ARIMA y SARIMAX	1 día	16-Apr	16-Apr
Random Forest	1 día	17-Apr	17-Apr
XGBoost	1 día	18-Apr	18-Apr
Prophet	1 día	21-Apr	21-Apr
LSTM	2 días	22-Apr	23-Apr
Transformers	2 días	24-Apr	25-Apr
Establecer herramientas y tecnologías	5 días	7-Apr	11-Apr
Establecer método de evaluación de modelos	5 días	14-Apr	18-Apr
Definir análisis de sentimiento del mercado	6 días	21-Apr	28-Apr
Codificación y desarrollo	5 días	21-Apr	25-Apr
Documentación	1 día	28-Apr	28-Apr
Hito 1: Comienzo de implementación	0 días	29-Apr	29-Apr
Fase de implementación.	23 días	29-Apr	29-May
Predictor base (aleatorio y de persistencia)	1 día	29-Apr	29-Apr
ARIMA y SARIMAX	3 días	30-Apr	2-May
Codificación y desarrollo	1 día	30-Apr	30-Apr
Entrenamiento y optimización	1 día	1-May	1-May
Documentación	1 día	2-May	2-May
Prophet	3 días	5-May	7-May
Codificación y desarrollo	1 día	5-May	5-May
Entrenamiento y optimización	1 día	6-May	6-May
Documentación	1 día	7-May	7-May

Random Forest	3 días	8-May	12-May
Codificación y desarrollo	1 día	8-May	8-May
Entrenamiento y optimización	1 día	9-May	9-May
Documentación	1 día	12-May	12-May
XGBoost	3 días	13-May	15-May
Codificación y desarrollo	1 día	13-May	13-May
Entrenamiento y optimización	1 día	14-May	14-May
Documentación	1 día	15-May	15-May
LSTM	5 días	16-May	22-May
Codificación y desarrollo	2 días	16-May	19-May
Entrenamiento y optimización	2 días	20-May	21-May
Documentación	1 día	22-May	22-May
Transformers	5 días	23-May	29-May
Codificación y desarrollo	2 días	23-May	26-May
Entrenamiento y optimización	2 días	27-May	28-May
Documentación	1 día	29-May	29-May
Fase de pruebas.	23 días	30-May	1-Jul
Validación y verificación	3 días	30-May	3-Jun
Evaluación de rendimiento de los modelos	10 días	4-Jun	17-Jun
Depuración y ajuste	10 días	18-Jun	1-Jul
Hito 2: Modelos entrenados y evaluados	0 días	10-Jul	10-Jul
Desarrollo de aplicación de simulación	8 días	2-Jul	11-Jul
Definición de la arquitectura de la aplicación	2 días	2-Jul	3-Jul
Implementación aplicación web	4 días	4-Jul	9-Jul
Pruebas, tests y ajuste	2 días	10-Jul	11-Jul
Hito 3: Aplicación terminada	0 días	15-Jul	15-Jul
Fin de proyecto.	6 días	14-Jul	21-Jul
Tabla final comparativa	1 día	14-Jul	14-Jul
Documentación final y presentación	3 días	15-Jul	17-Jul
Conclusiones y futuras mejoras	2 días	18-Jul	21-Jul

Tabla 3 - Descripción de tareas del proyecto.

Finalmente, la *Figura 4* muestra el diagrama de Gantt, donde se reflejan la secuencia temporal de las tareas, sus dependencias y su agrupación conforme a los objetivos generales. Esta representación facilita un seguimiento riguroso de los plazos y asegura una ejecución ordenada y coherente del proyecto.



	Task Name	Duration	Start	Finish
Objetivo general 1	**Fase de concepción.**	**30 days**	**Feb 14**	**Mar 27**
	Formación: bolsa, teoría e historia	17 days	Feb 14	Mar 10
	Investigación estado actual: publicaciones, literatura	10 days	Mar 11	Mar 24
	Estudio de tecnologías disponibles	8 days	Mar 18	Mar 27
	Identificación de la idea y objetivos	13 days	Mar 11	Mar 27
	Fase de planificación.	**6 days**	Mar 28	**Apr 4**
	Definición del alcance y diagrama de Gantt	6 days	Mar 28	**Apr 4**
	Asignación de recursos	6 days	Mar 28	**Apr 4**
	Documentación inicial	6 days	Mar 28	**Apr 4**
	Fase de diseño.	**16 days**	**Apr 7**	**Apr 28**
	Diseño conceptual y técnico	3 days	Apr 7	Apr 9
	Fuente de datos y procesado	3 days	Apr 10	Apr 14
	Selección de modelos y variables	**9 days**	**Apr 15**	**Apr 25**
	Predictor aleatorio base	1 day	Apr 15	Apr 15
	ARIMA y ARIMAX	1 day	Apr 16	Apr 16
	Random Forest	1 day	Apr 17	Apr 17
	XGBoost	1 day	Apr 18	Apr 18
	Prophet	1 day	Apr 21	Apr 21
	LSTM	2 days	Apr 22	Apr 23
	Transformers	2 days	Apr 24	Apr 25
	Establecer herramientas y tecnologías	5 days	Apr 7	Apr 11
	Establecer método de evaluación de modelos	5 days	Apr 14	Apr 18
	Definir análisis de sentimiento del mercado	**6 days**	**Apr 21**	**Apr 28**
	Codificación y desarrollo	5 days	Apr 21	Apr 25
	Documentación	1 day	Apr 28	Apr 28
	Hito 1: Comienzo de implementación	0 days	**Apr 29**	Apr 29
	Fase de implementación.	**23 days**	**Apr 29**	**May 29**
	Predictor aleatorio y de persistencia	1 day	Apr 29	Apr 29
Objetivo general 2	**ARIMA y ARIMAX**	**3 days**	**Apr 30**	**May 2**
	Codificación y desarrollo	1 day	Apr 30	Apr 30
	Entrenamiento y optimización	1 day	May 1	May 1
	Documentación	1 day	May 2	May 2
	Prophet	**3 days**	**May 5**	**May 7**
	Codificación y desarrollo	1 day	May 5	May 5
	Entrenamiento y optimización	1 day	May 6	May 6
	Documentación	1 day	May 7	May 7
	Random Forest	**3 days**	**May 8**	**May 12**
	Codificación y desarrollo	1 day	May 8	May 8
	Entrenamiento y optimización	1 day	May 9	May 9
	Documentación	1 day	May 12	May 12
	XGBoost	**3 days**	**May 13**	**May 15**
	Codificación y desarrollo	1 day	May 13	May 13
	Entrenamiento y optimización	1 day	May 14	May 14
	Documentación	1 day	May 15	May 15
	LSTM	**5 days**	**May 16**	**May 22**
	Codificación y desarrollo	2 days	May 16	May 19
	Entrenamiento y optimización	2 days	May 20	May 21
	Documentación	1 day	May 22	May 22
	Transformers	**5 days**	**May 23**	**May 29**
	Codificación y desarrollo	2 days	May 23	May 26
	Entrenamiento y optimización	2 days	May 27	May 28
	Documentación	1 day	May 29	May 29
	Fase de pruebas.	**23 days**	**May 30**	**Jul 1**
	Validación y verificación	3 days	May 30	Jun 3
	Evaluación de rendimiento de los modelos	10 days	Jun 4	Jun 17
	Depuración y ajuste	10 days	Jun 18	Jul 1
	Hito 2: Modelos entrenados y evaluados	0 days	Jul 10	Jul 10
Objetivo general 3	**Desarrollo de aplicación de simulación**	**8 days**	**Jul 2**	**Jul 11**
	Definición de la arquitectura de la aplicación	2 days	Jul 2	Jul 3
	Implementación aplicación web	4 days	Jul 4	Jul 9
	Pruebas, tests y ajuste	2 days	Jul 10	Jul 11
	Hito 3: Aplicación terminada	0 days	Jul 15	Jul 15
	Fin de proyecto.	**6 days**	**Jul 14**	**Jul 21**
	Tabla final comparativa	1 day	Jul 14	Jul 14
	Documentación final y presentación	3 days	Jul 15	Jul 17
	Conclusiones y futuras mejoras	2 days	Jul 18	Jul 21

Figura 4 – Diagrama Gantt del proyecto.

4.2 Descripción de la solución, metodologías y herramientas empleadas

Este proyecto sigue una adaptación del ciclo "*Cross-Industry Standard Process for Data Mining*" (CRISP-DM) [28], metodología estructurada en seis fases, cuya implementación específica se resume en la *Figura 5*.

- **Entendimiento del negocio**.
  Revisión del estado del arte, identificación de necesidades y definición de objetivos que orientan el resto del trabajo.

- **Entendimiento de los datos**.
  Inventario de las fuentes (precios bursátiles, indicadores técnicos, Big Tech, índices bursátiles, indicadores económicos y análisis de sentimiento) y evaluación de su idoneidad.

- **Preparación**.
  Limpieza, imputación de valores faltantes, análisis de distribución, normalización y estudio de correlaciones para generar un conjunto de datos coherente y listo para modelar.

- **Modelado**.
  Entrenamiento de seis familias de modelos (ARIMA, Prophet, Random Forest, XGBoost, LSTM y Transformers) sobre los datos preparados.

- **Evaluación**.
  Comparación mediante métricas de regresión (MAE, RMSE, R^2) y clasificación (Accuracy, F1-Score, ROC-AUC), selección de la combinación modelo-datos con mayor precisión y robustez para la predicción diaria del cierre de NVIDIA.

- **Despliegue**.
  Contenedores con Docker y exposición vía API Flask, lo que permite ejecutar el mejor modelo y comprobar su rendimiento en tiempo real (simulación).

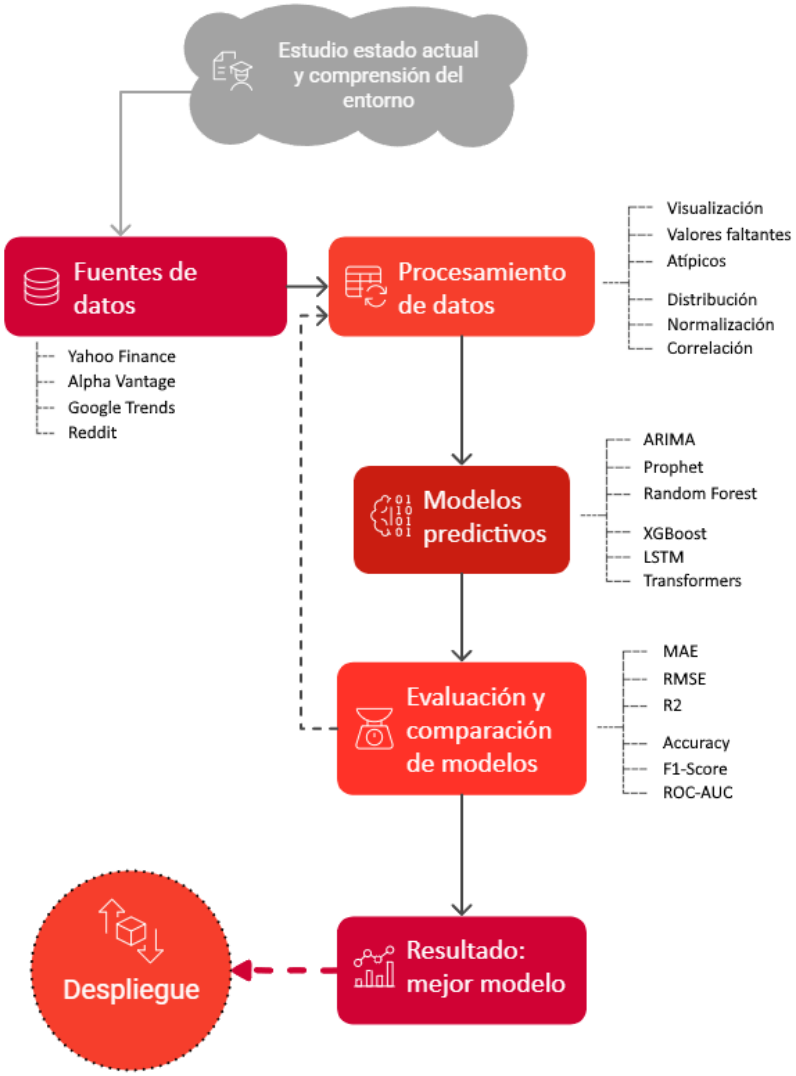


Figura 5 - Adaptación de este proyecto al ciclo CRISP-DM (fuente: elaboración propia).

Este ciclo se implementa en la arquitectura mostrada en la *Figura 6*.

La solución se despliega íntegramente en un equipo local Windows 11 + WSL 2 (AMD Ryzen 9800X3D, 64 GB DDR5 y una GPU NVIDIA RTX 5080), lo que evita dependencias con infraestructuras en la nube y resulta suficiente para el volumen manejado (~2600 entradas con 40 variables con un tamaño total de ~2 Megabytes). Se exploró la ejecución distribuida con PySpark en el clúster Laboratorio Optimizado de Recursos de Computación Avanzada o LORCA [29] de la Universidad Europea de Madrid, sin embargo, la latencia añadida superó el beneficio de paralelizar un conjunto de datos relativamente pequeño, por lo que se optó por el procesamiento local.

A continuación, se detallan los componentes principales de la arquitectura:

- **Componentes principales de la arquitectura (contenedores Docker):**
  - **FinanceDB**: Instancia MongoDB con la base de datos AMPB, organizada en colecciones: "market_sentiment", "tickers", "app_usuarios", etc.
  - **GetData**: Servicio que extrae diariamente (23:00h) noticias de Alpha Vantage, Yahoo Finance y Reddit, almacenándolas en la colección "market_sentiment". Cada hora, extrae los valores económicos de Yahoo Finance y Alpha Vantage almadeándolos en "indicadores_economicos", "índices_bursatiles" y "tickers".
  - **SA-Analyzer**: Procesa las noticias pendientes con FinBERT [30] para asignar la métrica de sentimiento (ejecución diaria, 23:15h).
  - **App-Sim:** Aplicación que simula la predicción diaria del precio de cierre.
- **Entorno de ciencia de datos (Python + Jupyter Notebook):**
  - Ingesta y preprocesado.
  - Entrenamiento y validación de modelos.
  - Selección del mejor modelo y despliegue en App-Sim.

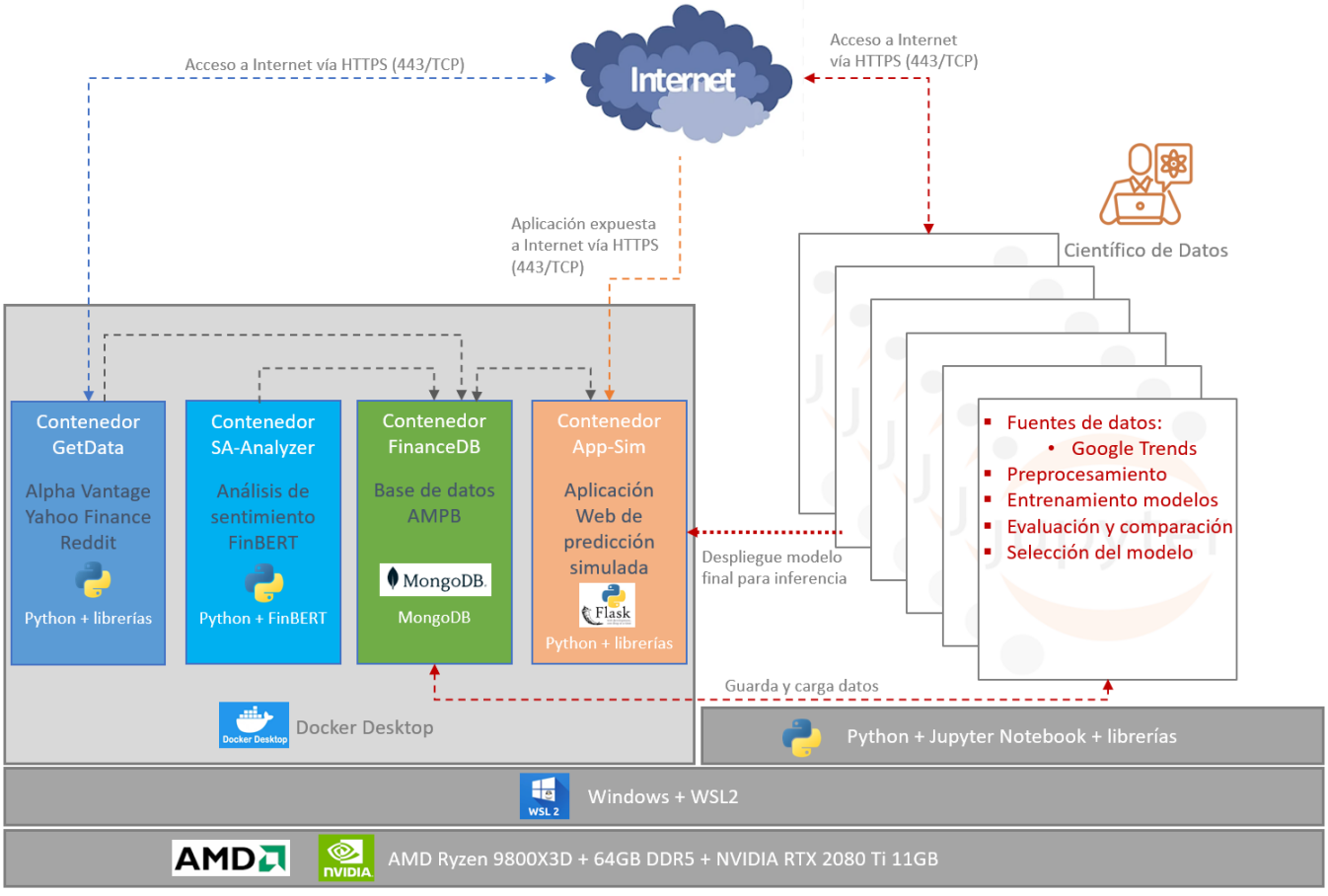


Figura 6 - Arquitectura del sistema AMPB (fuente: elaboración propia).

Todos los componentes se despliegan desde el repositorio de GitHub [31]: un archivo docker-compose.yml instancia los contenedores y deja los cuadernos Jupyter listos para su ejecución y ajuste. La estructura del repositorio de código en GitHub se describe en el *Anexo 9.2*.

El código de cada modelo se ha diseñado con un enfoque modular y reutilizable, estructurado en secciones comunes que aseguran homogeneidad y trazabilidad en el proceso:

- Importación de librerías necesarias para la ejecución.
- **Sección 0**: definición de parámetros configurables dependientes del modelo, soporte para ejecución desde línea de comandos e integración con el orquestador, además de la impresión de la cabecera con el nombre y configuración del modelo.
- **Sección 1**: carga de datos.
- **Sección 2**: división de los datos en conjuntos de entrenamiento y prueba.
- **Sección 3**: procesamiento de datos, incluyendo transformaciones, escalados, alineaciones y análisis de calidad.
- **Sección 4**: entrenamiento del modelo, adaptado a cada caso particular, con funciones auxiliares, búsqueda de hiperparámetros y entrenamiento final.
- **Sección 5**: evaluación del modelo en validación estática y generación del panel visual de resultados.
- **Sección 6 (opcional)**: validación cruzada bajo esquema estático, con adaptaciones específicas según el modelo y el procedimiento de predicción.
- **Sección 7**: evaluación del modelo en backtesting y creación del panel visual de resultados correspondiente.

A modo de referencia rápida, en el *Anexo 9.3*, se muestra el código completo para el modelo SARIMAX.

Además, se ha desarrollado un módulo en Python denominado AMPBLib, que integra todas las funciones auxiliares empleadas tanto por los modelos como por la aplicación. Este módulo constituye un componente crítico para la ejecución del proyecto, ya que permite la reutilización de código y la unificación de los distintos modelos con la aplicación. Con más de mil líneas de código, se organiza en varios grupos funcionales:

- **Configuración y gestión del sistema**: centraliza la configuración global del proyecto (clase AMPBConfig), incluyendo parámetros de puntuación, métricas baseline,

credenciales de APIs externas (Alpha Vantage, Reddit, MongoDB) y opciones de salida visual. También proporciona métodos para el establecimiento de semillas aleatorias (*resetRandomSeeds*) y la configuración optimizada de TensorFlow en entornos CPU/GPU (*setupTensorflowDevice*).

- **Gestión de base de datos**: funciones específicas para la conexión, autenticación y manipulación de colecciones en MongoDB (*getMongoClient*, *getMongoCollection*, *deleteMongoCollection*, *closeMongoClient*), que permiten el almacenamiento persistente de resultados experimentales y datos históricos.

- **Ingeniería de características financieras**: incluye la adquisición de datos (*getTickerData*) integrando diversas fuentes (Yahoo Finance, CSV locales) y el cálculo automatizado de indicadores técnicos mediante *calculateTechnicalIndicators* (SMA, EMA, RSI, MACD, Bandas de Bollinger, ATR). También incorpora análisis de sentimiento a través de *googletrends_add* y herramientas para la selección modular de variables exógenas (*getExogVars*).

- **Preprocesamiento y transformación de datos**: un pipeline completo implementado en *processData*, que abarca transformaciones matemáticas (logarítmica, Box-Cox, Yeo-Johnson), escalado (StandardScaler, MinMaxScaler), winsorización y alineación temporal. Incluye además funciones específicas para secuencias temporales (*createSequences*, *createTFDataset*, *createTimeSeriesCV*) y control de calidad de datos (*analyzeDataQuality*, *checkDataLoss*).

- **Evaluación y métricas**: funciones dedicadas a la evaluación de regresión (*printMetricsRegression*) y clasificación binaria (*printMetricsClassification*), junto con métricas de validación cruzada temporal (*printMetricsCV*), fundamentales para la robustez de los experimentos.

- **Visualización e informes**: generación de paneles gráficos (*printPredictionGraph*) que comparan valores reales y predichos, y producción de paneles visuales consolidados mediante *createReport*, basados en métricas e índices comparativos.

- **Utilidades de predicción**: funciones de reversión de transformaciones (*undoPredictionTransformation*, *reverseTransformPredictions*, *sanitizePrediction*), esenciales para recuperar valores en su escala original, y mecanismos de actualización de variables exógenas (*updateNextDayExog*).

En definitiva, este proyecto combina una aproximación investigadora basada en la metodología CRISP-DM con el desarrollo de un caso de uso práctico, integrando tecnologías de inteligencia artificial, bases de datos y despliegue ágil de arquitecturas en entornos de simulación.

4.3 Recursos requeridos

Para abordar el desarrollo, implementación y análisis de los sistemas predictivos, y garantizar una infraestructura adecuada para la experimentación y evaluación de modelos en el ámbito financiero, se han utilizado los siguientes recursos:

- **Dispositivos y equipos de trabajo:**
  - MacBook Pro.
  - Workstation con una potente GPU.

- **Sistemas operativos y software de apoyo:**
  - macOS.
  - Microsoft Windows 11.
  - Microsoft Office (Word, Excel, PowerPoint)
  - Microsoft Project.
  - Notepad++.
  - Paint.Net.
  - WinMerge.
  - PowerBI.
  - Docker.

- **Entorno de desarrollo y herramientas de programación:**
  - Microsoft Visual Studio Code.
  - Python con librerías de código abierto como: Pandas, NumPy, Matplotlib, etc (ver *Tabla 11*).

- **Gestión y control de código:**
  - TortoiseSVN (Subversion).
  - GitHub.

- **Fuentes de datos**:
  - Yahoo Finance [32].
  - Alpha Vantage [33].
  - Reddit [34].
  - Google Trends [35].
  - Exploradas pero descartadas:
    - Marketstack [36]
    - Finnhub [37]

4.4 Presupuesto

La siguiente tabla detalla el presupuesto estimado para el desarrollo del proyecto, considerando los recursos humanos, tecnológicos y materiales necesarios (incluyendo IVA otros impuestos).

Descripción	Cantidad	Precio Unidad	Subtotal
Equipo de trabajo portátil MacBook Pro	1	EUR 2.500,0	EUR 2.500,0
Equipo de trabajo fijo Workstation	1	EUR 3.000,0	EUR 3.000,0
Licencia Microsoft Windows 11	2	EUR 50,0	EUR 100,0
Licencia Microsoft Office (Word, Excel, Powerpoint)	1	EUR 149,0	EUR 149,0
Licencia Microsoft Project	1	EUR 200,0	EUR 200,0
Licencia Microsoft Visual Studio Code	1	EUR 0,0	EUR 0,0
Licencia Python con librerías Open Source	1	EUR 0,0	EUR 0,0
Donación a arXiv por el acceso a publicaciones	1	EUR 50,0	EUR 50,0
Científico de Datos (Ingeniero) **(*)**	238	EUR 125,0	EUR 29.750,0
Arquitecto y tutor (Ingeniero) **(*)**	20	EUR 180,0	EUR 3.600,0
Electricidad **(*)**	238	EUR 0,2	EUR 47,6
Alquiler sala de trabajo **(**)**	5	EUR 300,0	EUR 1.500,0
Gastos en transporte **(**)**	5	EUR 70,0	EUR 350,0
Internet **(**)**	5	EUR 45,0	EUR 225,0
* *cantidad en horas* ** *cantidad en meses*	**TOTAL (€):**		**EUR 41.471,6**

Tabla 4 – Presupuesto del proyecto.

4.5 Viabilidad

El proyecto presenta una viabilidad favorable desde las perspectivas económica, técnica y legal, lo que permite su desarrollo sin barreras significativas y con alto potencial de aplicabilidad:

- **Viabilidad económica**:
  El proyecto se basa íntegramente en herramientas y librerías de código abierto, ejecutadas en un equipo local, evitando costes asociados a infraestructuras externas.

- **Viabilidad técnica:**
  El proyecto utiliza el ecosistema consolidado de Python y sus librerías especializadas para el análisis y procesado de datos y modelos de inteligencia artificial. Esto garantiza la eficiencia en los cálculos, escalabilidad y sostenibilidad técnica a largo plazo, facilitando posibles ampliaciones o adaptaciones futuras.

- **Viabilidad legal:**
  Las tecnologías y fuentes de datos utilizadas, cuentan con licencias libres o de uso público. Este hecho asegura el cumplimiento normativo y permite libremente su utilización, modificación y difusión posterior, sin restricciones de carácter legal.

4.6 Resultados del proyecto

Conforme a los objetivos planteados, los resultados del proyecto pueden resumirse en tres grandes bloques: preparación de datos, evaluación de modelos y desarrollo de una aplicación de simulación funcional que demuestre los potenciales usos de la investigación realizada en este proyecto.

En primer lugar, se ha construido un ecosistema de datos integrado en un data warehouse ligero basado en MongoDB, con más de 2.500 registros diarios entre 2015 y 2025 que incluyen precios de NVIDIA, indicadores técnicos, cotizaciones de Big Tech, índices bursátiles, indicadores económicos y análisis de sentimiento. Este conjunto ha sido sometido a un proceso de preprocesamiento que incluyó imputación de valores faltantes, tratamiento de valores atípicos, análisis de correlación y definición de transformaciones y escalados posibles (a elección de cada modelo), asegurando la calidad y consistencia de los datos.

En segundo lugar, se han entrenado y evaluado modelos representativos de tres enfoques distintos: estadísticos (ARIMA, Prophet, SARIMAX), de aprendizaje automático (Random Forest, XGBoost) y de aprendizaje profundo (LSTM, Transformer). Se han probado más de 2.200 configuraciones, combinando 63 conjuntos de variables exógenas, distintas transformaciones y técnicas de escalado. Los resultados muestran que el modelo SARIMAX obtiene el mejor rendimiento global en condiciones realistas de backtesting, superando al predictor base en clasificación con una mejora del 17%. Por su parte, XGBoost ofrece la mejor regresión aislada (+2%) y Transformer la mejor clasificación (+30%), aunque sin mejorar el balance global de SARIMAX. Los conjuntos de datos más relevantes son los directos, seguidos del análisis de sentimiento e indicadores económicos, después los índices bursátiles y, en último lugar, los indicadores técnicos y las Big Tech. Estos resultados se han documentado de forma estructurada en paneles visuales y ficheros CSV, garantizando la trazabilidad y reproducibilidad experimental.

En tercer lugar, se ha desarrollado una aplicación de simulación en Flask, desplegada con Docker, que simula la predicción diaria del precio de NVIDIA utilizando un ensamblaje de tres modelos.

Durante el desarrollo del proyecto se introdujeron cambios respecto al planteamiento inicial. Uno de los más importantes fue pasar de evaluar únicamente seis combinaciones acumulativas de conjuntos de datos a explorar las 63 combinaciones posibles, lo que incrementó la carga computacional, pero permitió identificar configuraciones óptimas no evidentes. Asimismo, se diseñó un sistema de orquestación automática que permitió ejecutar miles de pruebas con capacidad de paralelización, reinicio y automatización de salida de resultados, lo que resultó clave para la escalabilidad del proyecto.

En conclusión, los resultados obtenidos cumplen satisfactoriamente los objetivos propuestos y aportan una evaluación sistemática de distintas metodologías de predicción bursátil. El trabajo demuestra que, si bien los mercados mantienen un alto componente aleatorio, el uso de modelos multivariantes avanzados puede extraer patrones relevantes y aportar valor como sistemas de apoyo a la toma de decisiones.

4.7 Datos y fuentes

En este proyecto se emplean exclusivamente fuentes de datos abiertas y gratuitas, principalmente orientadas al valor bursátil de NVIDIA y otros indicadores relevantes. Aunque estos datos presentan un posible desfase de hasta varias horas, esta limitación no afecta al análisis, ya que se trabaja con el precio de cierre diario para predecir el cierre del día siguiente.

NVIDIA cotiza en el mercado estadounidense, concretamente en el índice NASDAQ de Nueva York, cuya operativa se limita a días laborables. En este proyecto se toma como referencia únicamente el horario regular de cotización, excluyendo los periodos de preapertura y post mercado, como se muestra en la *Tabla 5*.

Sesión	Horario Nueva York	Horario España
Preapertura	4:00 AM - 9:30 AM	10:00 AM - 3:30 PM (UTC+2)
Horario regular	9:30 AM - 4:00 PM	3:30 PM - 10:00 PM (UTC+2)
Post mercado	4:00 PM - 8:00 PM	10:00 PM - 2:00 AM (UTC+2)

Tabla 5 - Horario de cotización de NVIDIA.

Las siguientes fuentes de datos fueron evaluadas para la realización de este proyecto:

- **Yahoo Finance**: Ofrece información detallada sobre valores bursátiles, noticias económicas, análisis de mercado, herramientas de gestión de carteras y datos históricos de hasta 40 años. Es reconocida por su accesibilidad, fiabilidad y facilidad de integración con Python mediante la librería *yfinance*. Es libre y gratuita.

- **Marketstack**: Esta fuente destaca por ofrecer acceso gratuito a datos históricos de más de 170,000 etiquetas de cotización en 70 bolsas globales. Su API es fácil de integrar y proporciona datos confiables para análisis financieros. Esta fuente permite hasta 100 peticiones al mes de forma libre y gratuita.

- **Alpha Vantage**: Popular entre desarrolladores y pequeños equipos, esta API gratuita ofrece datos bursátiles en tiempo real, indicadores técnicos y métricas sectoriales. Esta fuente permite hasta 25 peticiones al día de forma libre y gratuita.

- **Finnhub**: Proporciona una amplia variedad de datos financieros en tiempo real, incluyendo cotizaciones, datos históricos, noticias del mercado y reportes de resultados. Esta fuente está muy limitada de forma gratuita y requiere una suscripción.

Todas las fuentes combinan accesibilidad, facilidad de uso e integración directa con herramientas como Python, permitiendo un flujo de información automatizado y eficiente. En este proyecto se integran seis conjuntos de datos en un data warehouse ligero en MongoDB, que sirve como base estructurada para el entrenamiento y evaluación de los modelos predictivos.

4.7.1 Datos directos (conjunto de datos 1)

Los datos directos, también conocidos por las siglas OHLCV (Open, High, Low, Close, Volume), incluyen información diaria sobre apertura, máximo, mínimo, cierre y volumen negociado, como se ejemplifica en la *Tabla 6*.

Date	Open	High	Low	Close	Volume
2025-05-22	132.23	134.25	131.55	132.83	187344000.00
2025-05-23	130.00	132.68	129.16	131.29	198821300.00
2025-05-27	134.15	135.66	133.31	135.50	192953600.00
2025-05-28	136.03	137.25	134.79	134.81	304021100.00
2025-05-29	142.25	143.49	137.91	139.19	369241900.00

Tabla 6 - Ejemplo de datos directos de NVIDIA (5 días).

Yahoo Finance fue seleccionada como fuente principal para los datos directos debido a su fiabilidad, cobertura histórica y disponibilidad de acceso libre y gratuito.

4.7.2 Indicadores técnicos (conjunto de datos 2)

A partir de los datos directos se calculan diversos indicadores técnicos (*Tabla 7*), como la media móvil simple (SMA) o las bandas de Bollinger. Estos indicadores permiten capturar tendencias, volatilidad y condiciones de sobrecompra o sobreventa.

Indicador	Descripción
Simple Moving Average (SMA)	Promedios móviles simples calculados en ventanas que suavizan las fluctuaciones del precio y revelan tendencias. **Unidad**: Dólares. **Frecuencia**: Diario de últimos 20, 50 y 200 días.
Exponential Moving Average (EMA)	Promedio móvil exponencial que otorga mayor peso a los datos más recientes, capturando cambios de tendencia de forma más sensible. **Unidad**: Dólares. **Frecuencia**: Diario de últimos 20 días.
Relative Strength Index (RSI)	El Índice de Fuerza Relativa mide el impulso del precio y ayuda a identificar condiciones de sobrecompra o sobreventa. **Unidad**: Porcentual [0-100], >70% indica sobrecompra, <30% indica sobreventa. **Frecuencia**: 14 días.
Moving Average Convergence Divergence (MACD)	Indicadores de convergencia/divergencia de medias móviles que facilitan la detección de cambios en la dirección de la tendencia. **Unidad**: Valor a comparar. **Frecuencia**: Diario de últimos 26 días.
Average True Range (ATR)	El Rango Verdadero Promedio, mide la volatilidad promedio de un activo en un período específico calculando el rango verdadero (diferencia entre máximo y mínimo, ajustado por gaps de precios) y lo suaviza con una media móvil. **Unidad**: Dólares.

	Frecuencia: Valor diario de últimos 14 días.
Bollinger Bands (BB)	Las Bandas de Bollinger miden la volatibilidad y los niveles de sobrecompra y sobreventa en un periodo dado. **Unidad**: Dólares. **Frecuencia**: Valor diario de últimos 20 días.
Daily Price Range (Range)	La diferencia entre los precios máximos y mínimos del día, que proporciona una medida directa de la amplitud del movimiento. **Unidad**: Dólares. **Frecuencia**: Valor diario.
Open to Close Percent Change (OC_Change)	Variación porcentual entre los precios de apertura y cierre (comportamiento intradía) **Unidad**: Porcentual [0-100]. **Frecuencia**: Diario.
Chaikin Oscillator (Chaikin_Osc)	Oscilador de Chaikin, indicador que evalúa la acumulación y distribución mediante el análisis del volumen. **Unidad**: Valores positivos indican acumulación (los compradores dominan, posible inicio de tendencia alcista) y valores negativos indican distribución (los vendedores dominan, posible inicio de tendencia bajista). **Frecuencia**: Valor diario.

Tabla 7 – Indicadores técnicos o variables calculadas.

Estos datos se calculan con los datos directos por lo que indirectamente también provienen de **Yahoo Finance**.

4.7.3 Big Tech (conjunto de datos 3)

Se incorporó la cotización bursátil de las principales tecnológicas estadounidenses:

- Google
- Amazon
- Apple
- Meta
- Microsoft
- Tesla
- AMD
- Intel

El objetivo es analizar posibles relaciones entre estas empresas y NVIDIA. Estos datos también se han extraído de **Yahoo Finance** por las mismas razones que los datos directos y para reutilizar el sistema de captura de datos.

4.7.4 Índices bursátiles (conjunto de datos 4)

Se incluyeron índices bursátiles representativos de diferentes mercados globales (*Tabla 8*).

Indicador	Descripción
S&P 500 (SP500)	Índice de referencia de las 500 principales empresas de Estados Unidos. Es un indicador representativo del comportamiento general del mercado bursátil. **Frecuencia**: Valor diario.
NASDAQ-100 (NASDAQ100)	Índice compuesto por las 100 mayores empresas no financieras del Nasdaq, con una marcada presencia del sector tecnológico (incluye a NVIDIA) **Frecuencia**: Valor diario.
EURO STOXX 50 (EuroStoxx50)	Índice que agrupa a 50 de las empresas más representativas de la zona euro, sirviendo como indicador del rendimiento económico y bursátil en Europa **Frecuencia**: Valor diario.
Nikkei 225 (Nikkei225)	Índice bursátil japonés que incluye a las 225 principales empresas de la Bolsa de Tokio, reflejando el comportamiento del mercado en Japón **Frecuencia**: Valor diario.
Shanghai Composite (ShanghaiComposite)	Índice representativo del mercado bursátil chino. **Frecuencia**: Valor diario.

Tabla 8 - Índices bursátiles mundiales.

Estos datos también se han extraído de **Yahoo Finance**.

4.7.5 Indicadores económicos (conjunto de datos 5)

También se incorporan Indicadores económicos de Estados Unidos junto con el índice volatilidad y precios del petróleo y oro como se detalla en la *Tabla 9*.

Indicador	Descripción
Consumer Price Index (CPI)	Índice de Precios al Consumidor, que mide la inflación en Estados Unidos. **Unidad**: Cambio porcentual desde 1982-1984 (100%). **Frecuencia**: Mensual.
Real GDP Trending (GDP_Real)	La Tendencia del Producto Interno Bruto real, ajustado a la inflación, indica el crecimiento económico. **Unidad**: Miles de millones de dólares. **Frecuencia**: Trimestral.
Real GDP per Capita (GDP_per_Capita)	El Producto Interno Bruto real por habitante, refleja el ingreso promedio ajustado por inflación. **Unidad**: Dólares por habitante. **Frecuencia**: Valor trimestral.
Treasury Yield (Treasury):	Rendimiento de los bonos del Tesoro. **Unidad**: Miles de millones de dólares. **Frecuencia**: Valor diario para bono trimestral y 10 años.

Volatility Index (VIX)	Mide la expectativa de volatilidad del mercado, llamado "índice del miedo". **Unidad**: Desviación típica anualizada. **Frecuencia**: Valor diario.
Crude Oil Brent (Brent_Oil)	Precio del petróleo a nivel internacional, cuya variación impacta en el coste energético. **Unidad**: Dólares por barril. **Frecuencia**: Valor diario.
Precio del oro (Gold)	Activo refugio en tiempos de incertidumbre. Se relaciona inversamente con el mercado bursátil. **Unidad**: Una décima parte de onza troy de oro. **Frecuencia**: Valor diario.

Tabla 9 - Indicadores económicos.

4.7.6 Análisis de sentimiento (conjunto de datos 6)

Para capturar el impacto de eventos externos y percepciones del mercado, se han explorado las siguientes fuentes adicionales: **Google News** [38] y **CNBC** [39]. Aunque se lograron desarrollar scripts funcionales, las restricciones de acceso en las versiones gratuitas limitaron el histórico de datos útiles. Solo **Alpha Vantage** permitió recuperar un volumen adecuado de noticias desde marzo de 2022, mediante el uso escalonado de varias cuentas.

Además, se emplearon fuentes complementarias como **Reddit** y **Google Trends**. Reddit resultó útil para identificar tendencias emergentes, aunque su API impone un límite de 1000 publicaciones, restringiendo el alcance temporal. **Google Trends**, por su parte, facilitó la detección de patrones de búsqueda relevantes asociados al comportamiento bursátil de NVIDIA.

Finalmente, se utilizan las dos siguientes variables:

- googletrends_NVDA: recoge la tendencia de búsquedas en Google desde el 2015 hasta 2025 aunque su creación no ha sido posible automatizarla (ver *Normalización y escalado de datos* para más información).

- av_nvidia: recoge el análisis de sentimiento de noticias de **Alpha Vantage** desde el 2022.

4.8 Preprocesamiento de datos

En este apartado se describen las etapas de análisis, limpieza y transformación implementadas sobre el conjunto de datos, con el objetivo de garantizar su calidad y adecuación para ser utilizados en los modelos predictivos. Las figuras mostradas en este apartado se han generado utilizando el Notebook "*Preprocesamiento de datos.ipynb*" que se encuentra disponible en el repositorio de GitHub y el panel interactivo sobre la visualización general de análisis de sentimiento de ha creado con la herramienta PowerBI (fichero "*SA-Dashboard.pbix*").

4.8.1 Visualización general de los datos directos

En este subapartado, se representa gráficamente la evolución a lo largo del tiempo de la totalidad de los valores de cierre y en una segunda gráfica, el último año con las variables más relevantes (precio de cierre, volumen, máximos y mínimos diarios). Estas visualizaciones globales permiten identificar patrones, tendencias y comportamientos atípicos a lo largo del tiempo, facilitando la detección de posibles anomalías en la serie.

La *Figura 7* presenta la evolución histórica del precio de cierre de las acciones de NVIDIA desde el 5 de enero de 2015 hasta el 23 de mayo de 2025. Se aprecia un crecimiento casi continuo durante los primeros años, con valores inferiores a 10 dólares hasta mediados de 2020. A partir de entonces, el precio experimenta una apreciable aceleración, alcanzando picos cercanos a 150 dólares en 2024. Asimismo, se observan episodios de elevada volatilidad, especialmente tras la corrección de finales de 2021 y las fuertes oscilaciones registradas entre 2024 y 2025. Esta gráfica es compatible con el hecho de que NVIDIA ha crecido de una forma asombrosa y que durante la primera parte de 2025 se encuentra en un estado de retroceso debido a la creciente competencia en el ámbito del hardware dedicado a la inteligencia artificial y a los recientes aranceles del presidente de EEUU, Donald Trump.

Figura 7 - Evolución histórica del precio de cierre de NVIDIA (fuente: elaboración propia).

La *Figura 8* muestra, entre mayo 2024 y mayo 2025, la evolución del precio de cierre y el volumen de negociación de NVIDIA. La línea azul refleja el precio diario y el área sombreada su rango intradía, evidenciando fases de alta volatilidad (rango más ancho) y periodos de consolidación (rango más estrecho). El histograma inferior representa el volumen, coloreado según el cierre al alza o a la baja, lo que permite vincular picos de negociación con movimientos bruscos del precio. Destacaron 2 eventos: la fuerte caída tras el anuncio de DeepSeek R1 (enero 2025), mencionado en el apartado 2.2, con un volumen superior a 750 millones de operaciones, y la posterior caída y recuperación (abril 2025) causadas por el anuncio de los aranceles estadounidenses al resto del mundo y especialmente a productos tecnológicos chinos.

Figura 8 – Evolución del precio, rango y volumen de NVIDIA en el último año (fuente: elaboración propia).

4.8.2 Visualización general de análisis de sentimiento

En este subapartado se analiza visualmente la evolución conjunta del precio de cierre de NVIDIA y las variables de sentimiento disponibles. Debido a las limitaciones de histórico comentadas en el apartado 4.7, únicamente se han integrado dos fuentes consistentes: el sentimiento derivado de Alpha Vantage (**av_nvidia**) y la intensidad de búsqueda en Google Trends (**googletrends_NVDA**). Los flujos procedentes de Reddit, Yahoo Finance, Google News y CNBC han sido descartados por falta de datos históricos.

La *Figura 9* presenta una captura del panel interactivo creado con PowerBI donde:

- El eje primario representa el precio de cierre (línea azul).
- El eje secundario superponen los valores de sentimiento: los triángulos morados para **av_nvidia** y los rombos naranjas para **googletrends_NVDA** con escala 0 a 1.
- Los selectores inferiores permiten filtrar por año, trimestre y mes, ofreciendo la opción de aislar periodos concretos y también de mostrar un valor de sentimiento, los dos o ninguno.

Al filtrar los años 2022-2025 se aprecia una correlación visual: los valores de **googletrends_NVDA** anteceden o coinciden con fases de subida pronunciada en el precio, mientras que los descensos de interés suelen alinearse con correcciones del valor. El indicador **av_nvidia**, más disperso y más difícil de interpretar de forma visual, no obstante, la densidad de valores bajos (<0.5) parece disminuir en 2024 y 2025 en comparación con 2022 y 2023.

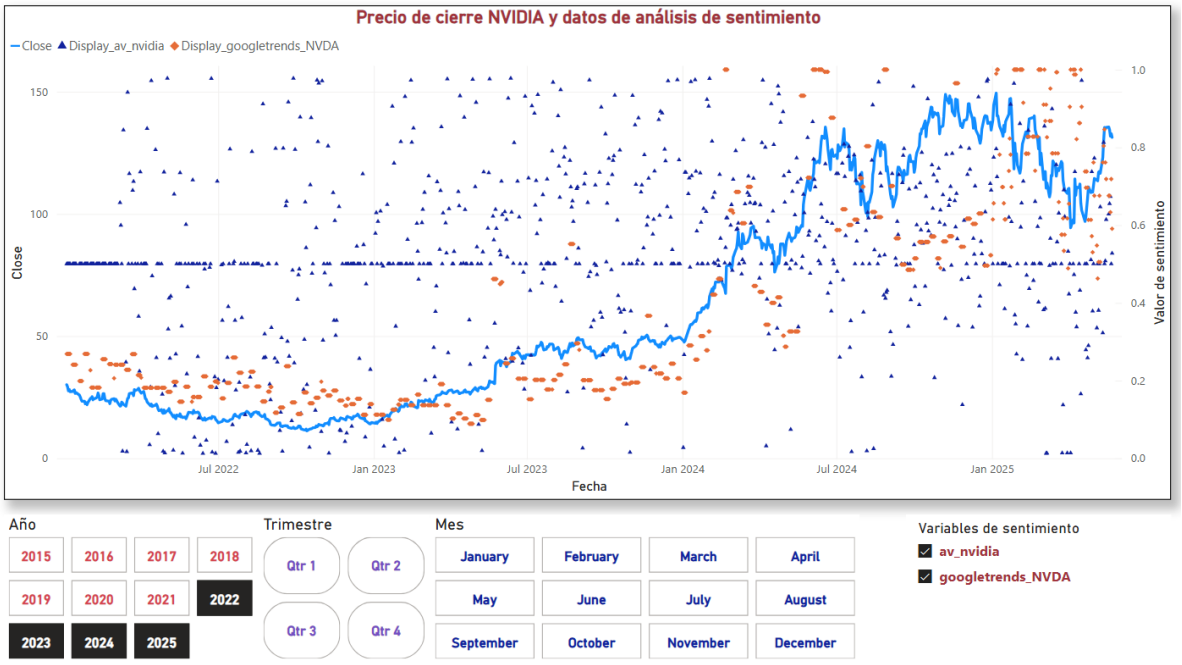


Figura 9 - Panel interactivo del precio de cierre y datos de análisis de sentimiento (fuente: elaboración propia).

4.8.3 Detección de valores faltantes y duplicados

La presencia de valores faltantes o nulos pueden originarse por diversas razones, incluyendo errores en la recolección de datos, interrupciones en la disponibilidad de información, la diferente periodicidad de las series temporales combinadas (algunos datos son mensuales, trimestrales y no diarios) o resultado de algunas de las variables calculadas o indicadores (por ejemplo, medias). El primer paso consiste en cuantificar la magnitud de los valores faltantes o nulos por cada variable. Se utilizan dos visualizaciones gráficas para facilitar esta tarea:

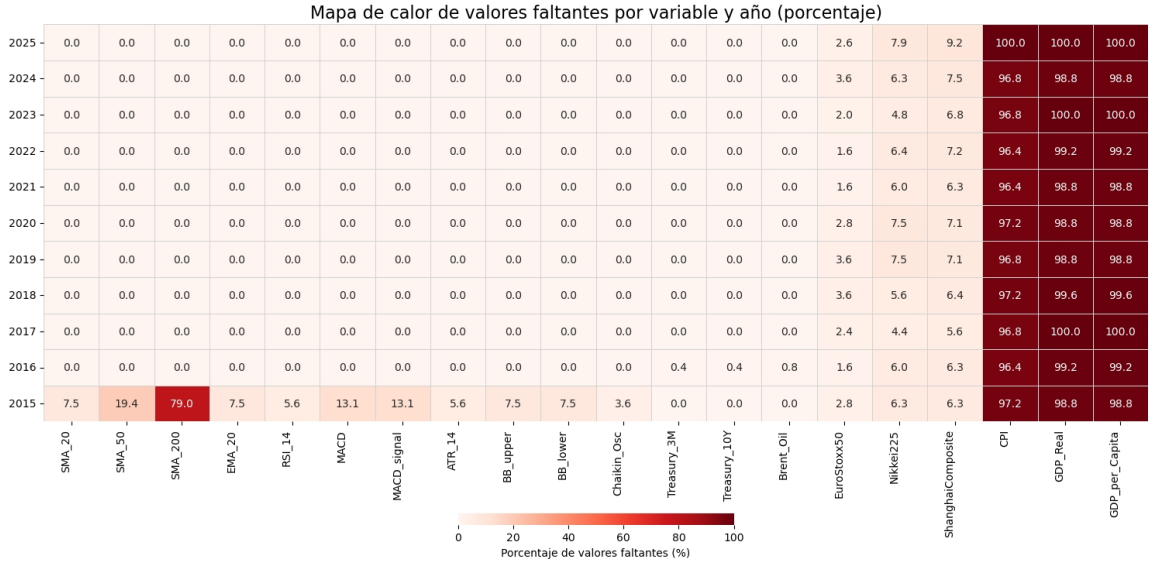


Figura 10 - Mapa de calor de valores faltantes por variable y año (pre) (fuente: elaboración propia).

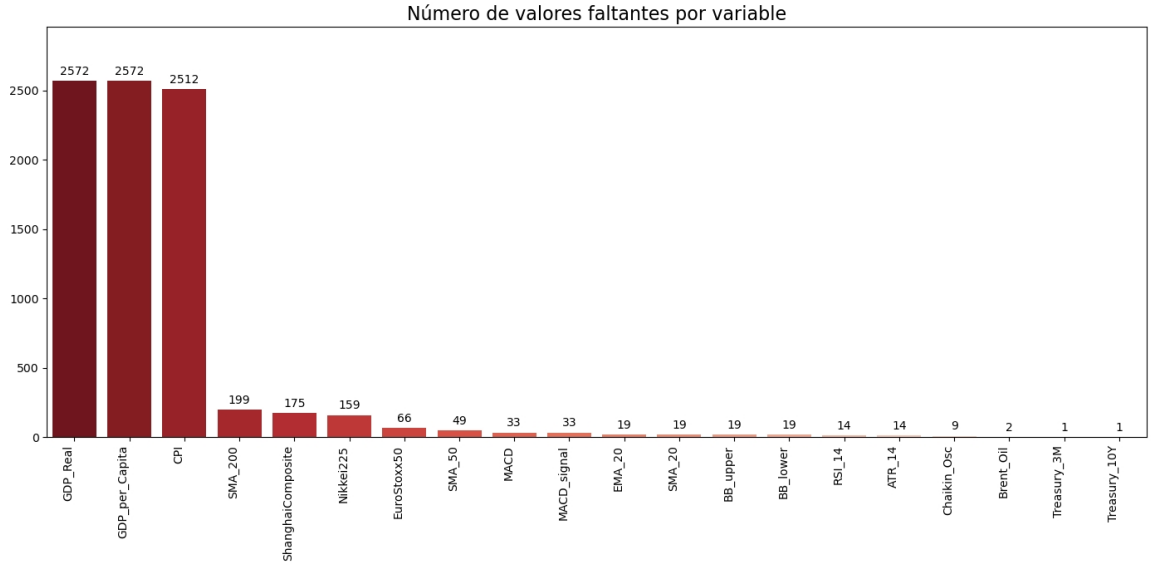


Figura 11 - Número de valores faltantes por variable (pre) (fuente: elaboración propia).

A raíz de las figuras anteriores, se identifican 20 variables con valores faltantes y se procede a investigar la causa clasificando estas variables en 4 grupos:

- **Diferente periodicidad (no diario):**
  En este apartado se encuentran **GDP_Real**, **GDP_Per_Capita** y **CPI** que como ya se indica en la *Tabla 9*, son valores mensuales o trimestrales. En este caso se aplica una estrategia de relleno hacia delante "forward fill" [40]. Este método propaga la última observación válida conocida a los períodos siguientes hasta que se dispone de un nuevo valor.
- **Resultado de variables calculadas o indicadores:**
  Aquí se encuentran variables como **SMA_200**, **SMA_50**, **SMA_20**, **MACD**, **MACD_signal**, **EMEA_20**, **BB_upper**, **BB_lower**, **RSI_14, ATR_14** y **Chaikin_Osc** que debido a cómo son calculadas, requieren varios días anteriores por lo que en las primeras fechas no es posible calcular estos valores. En este caso, no se realiza ninguna operación ni eliminado de estos valores faltantes, cada modelo predictivo según su mejor criterio tomará una acción u otra, esto se detalla en el apartado Modelos predictivos.

- **Índices bursátiles de otros países:**
  La referencia base de este proyecto es la bolsa de EEUU con sus días festivos específicos, el resto de bolsas de otros países como **ShanghaiComposite** (China), **Nikkei225** (Japón) y **EuroStoxx50** (Europa) tienen valores faltantes por tener días de apertura de la bolsa distintos a EEUU. En este caso también se aplica un método "forward fill".

- **Errores en la ingestión**:
  Finalmente, para **Treasury_3M**, **Treasury_10Y** y **Brent_Oil**, se detectaron algunos valores faltantes donde también se aplica el método de "forward fill":
  - 10 de octubre de 2016: falta para **Brent_Oil**.
  - 11 de noviembre de 2016: falta para los 3.

Al terminar de aplicar estos cambios, los datos quedan del siguiente modo:

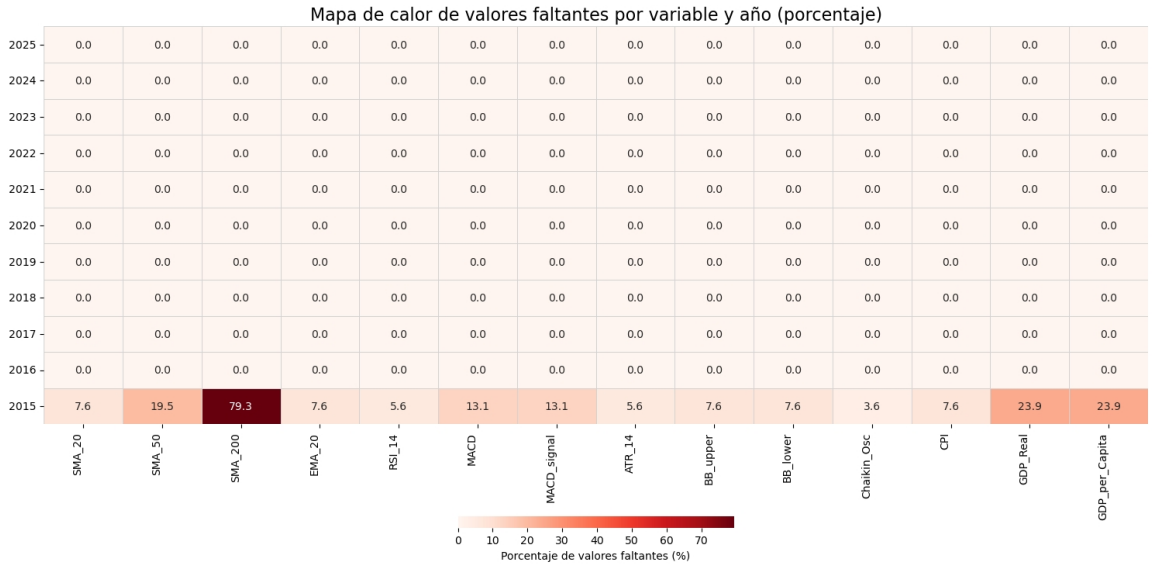


Figura 12 - Mapa de calor de valores faltantes por variable y año (post) (fuente: elaboración propia).

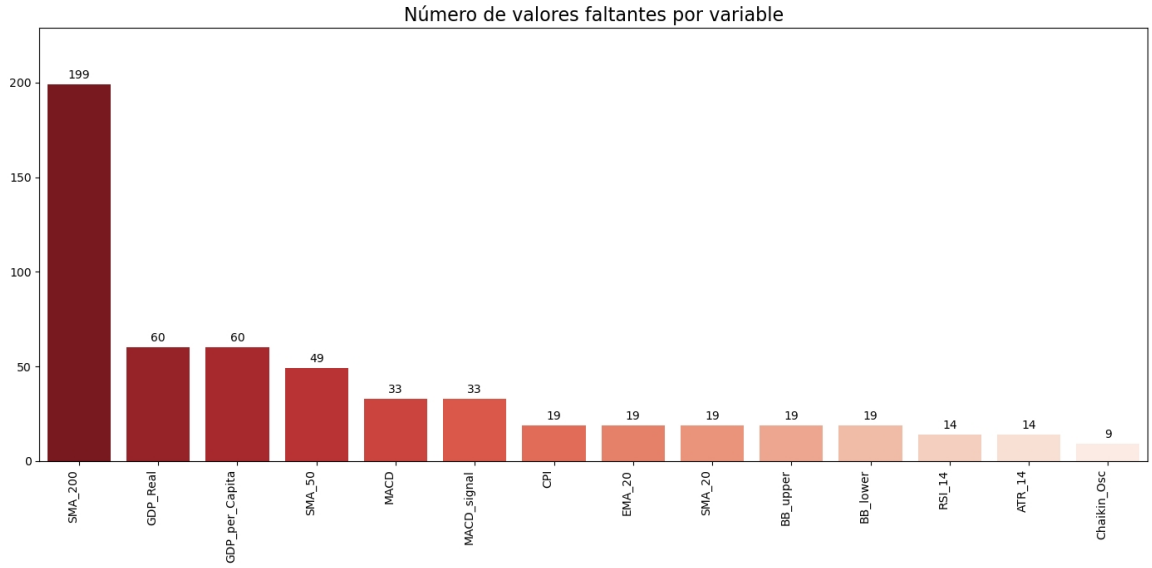


Figura 13 - Número de valores faltantes por variable (post) (fuente: elaboración propia).

Por último, el análisis de valores duplicados no detecta ninguno y tiene sentido ya que los datos son series temporales con índice de fecha.

Sobre los datos de noticias y análisis de sentimiento solo se utilizan los procedentes de Alpha Vantage con existencia de datos desde marzo de 2022, en este caso para todos los valores faltantes y previos a marzo de 2022, se ha aplicado un "fillna" [41] que consiste en rellenar todos los valores faltantes con un 0.5 que indica sentimiento neutro. Para los datos de redes sociales y tendencias solo se utiliza Google Trends y como todos sus valores son calculados y normalizados, no existe ningún valor faltante.

4.8.4 Distribuciones e histogramas

A continuación, se realiza un análisis visual de la distribución de las variables cuantitativas mediante histogramas y estimaciones de densidad kernel (KDE). El objetivo es comprender la distribución de cada variable, identificando características relevantes como:

- **Forma general de la distribución**:
  - Evalúa si los datos son simétricos (forma de campana, como la distribución normal), asimétricos (sesgo) o uniformes (valores igualmente probables).
  - Si hay sesgo, hacia la derecha (cola larga a la derecha, valores extremos altos son raros) o hacia la izquierda (cola larga a la izquierda, valores extremos bajos son raros).

- **Picos (Modas):**
  - Señala los valores o rangos donde la frecuencia es máxima.
  - Identifica si la distribución es unimodal (un solo pico), bimodal (dos picos) o multimodal (varios picos).

- **Dispersión (Visual):**
  - Mide el ancho del conjunto de datos:
    - Estrecha: la mayoría de los valores están concentrados alrededor del (los) pico(s).
    - Ancha: hay una gran variabilidad, con observaciones muy alejadas del (los) pico(s).

Esta inspección visual es útil para detectar patrones inesperados y puede informar sobre la necesidad de aplicar transformaciones a ciertas variables. En este proyecto, si el modelo predictivo lo requiere, se puede aplicar alguna de las siguientes transformaciones:

- **Transformación logarítmica (log(x)):**
  Muy habitual para datos financieros con sesgo positivo. Comprime las escalas, reduciendo el impacto de los valores extremadamente altos y haciendo la distribución más simétrica. Requiere que los datos sean positivos.

- **Diferenciación y retornos logarítmicos ($\log(\frac{P_t}{P_{t-1}})$)**:
  Los retornos logarítmicos, calculados como la diferencia del logaritmo de los precios entre períodos consecutivos, tienden a presentar propiedades estadísticas más deseables y una distribución más cercana a la normal, en comparación con los niveles de precios brutos. Requiere que los datos sean positivos.

- **Transformación Box-Cox:**

Es una familia de transformaciones parametrizadas que busca un exponente óptimo para estabilizar la varianza y acercar la distribución de los datos a la normalidad. Requiere que los datos sean positivos.

- **Transformación Yeo-Johnson:**
  Es una generalización de Box–Cox que soporta ceros y valores negativos.

La estrategia de transformación de variables no es universal y depende de cada modelo predictivo y el conjunto de variables elegidas. La estrategia elegida se especifica en el apartado de Modelos predictivos.

En primer lugar, se analizan los histogramas y densidad de los datos directos donde se observa:

- **Forma general de la distribución**:
  - Son asimétricas con un fuerte sesgo a la derecha.

- **Picos (Modas):**
  - Son unimodales. En el caso de las variables de precio, la moda se sitúa en el rango de precios más bajos.

- **Dispersión (Visual):**
  - Son anchas lo que indica gran variabilidad.

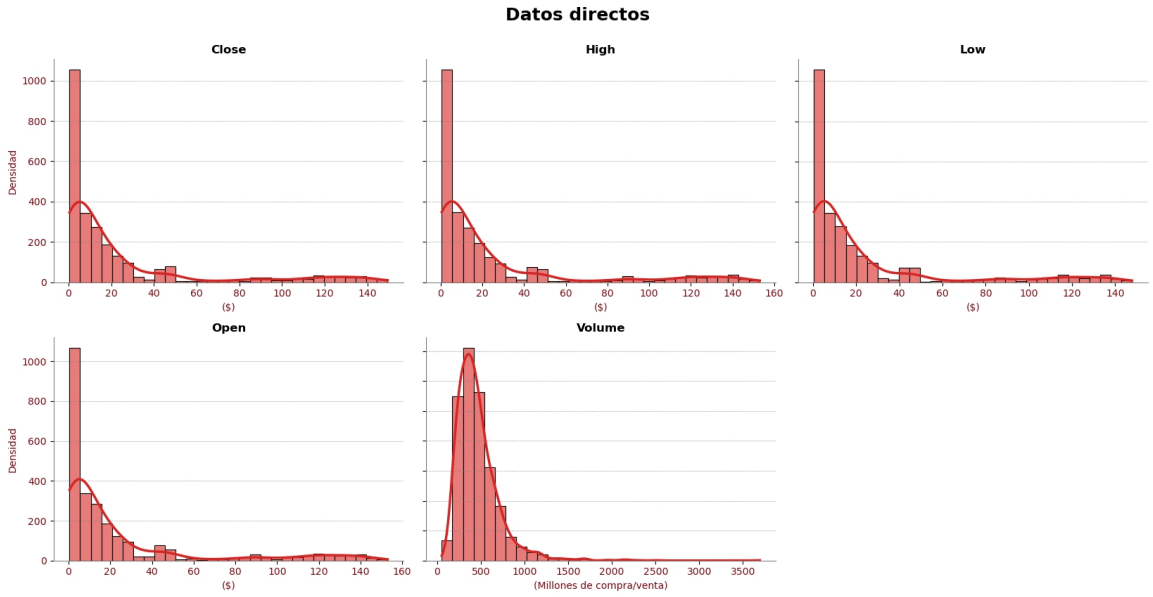


Figura 14 - Histograma y densidad KDE de datos directos (fuente: elaboración propia).

El siguiente análisis es de los histogramas y densidad de los indicadores técnicos (calculados a partir de los datos directos) donde se observa:

- **Forma general de la distribución**:
  - Los indicadores basados en precios y derivados directamente de los datos directos **SMA_200, SMA_50, SMA_20**, **EMA_20**, **ATR_14**, **BB_upper, BB_lower** y **Range**, heredan las asimétricas con un fuerte sesgo a la derecha.
  - Los indicadores de momento y osciladores como **MACD**, **MACD_signal**, **Chaikin_Osc**, **RSI_14** y **OC_Change** muestran una distribución más simétrica que se asemeja a una distribución normal.

- **Picos (Modas):**
  - Todas son unimodales.
  - En el caso de los indicadores basados en precios, la moda se sitúa en el rango de precios más bajos.
  - Los indicadores de momento y osciladores están más centrados indicando la mayor parte del tiempo están en niveles neutrales.

- **Dispersión (Visual):**
  - Para los indicadores basados en precios, son anchas lo que indica gran variabilidad.
  - Los indicadores de momento y osciladores están más concentrados en los valores de la zona central.

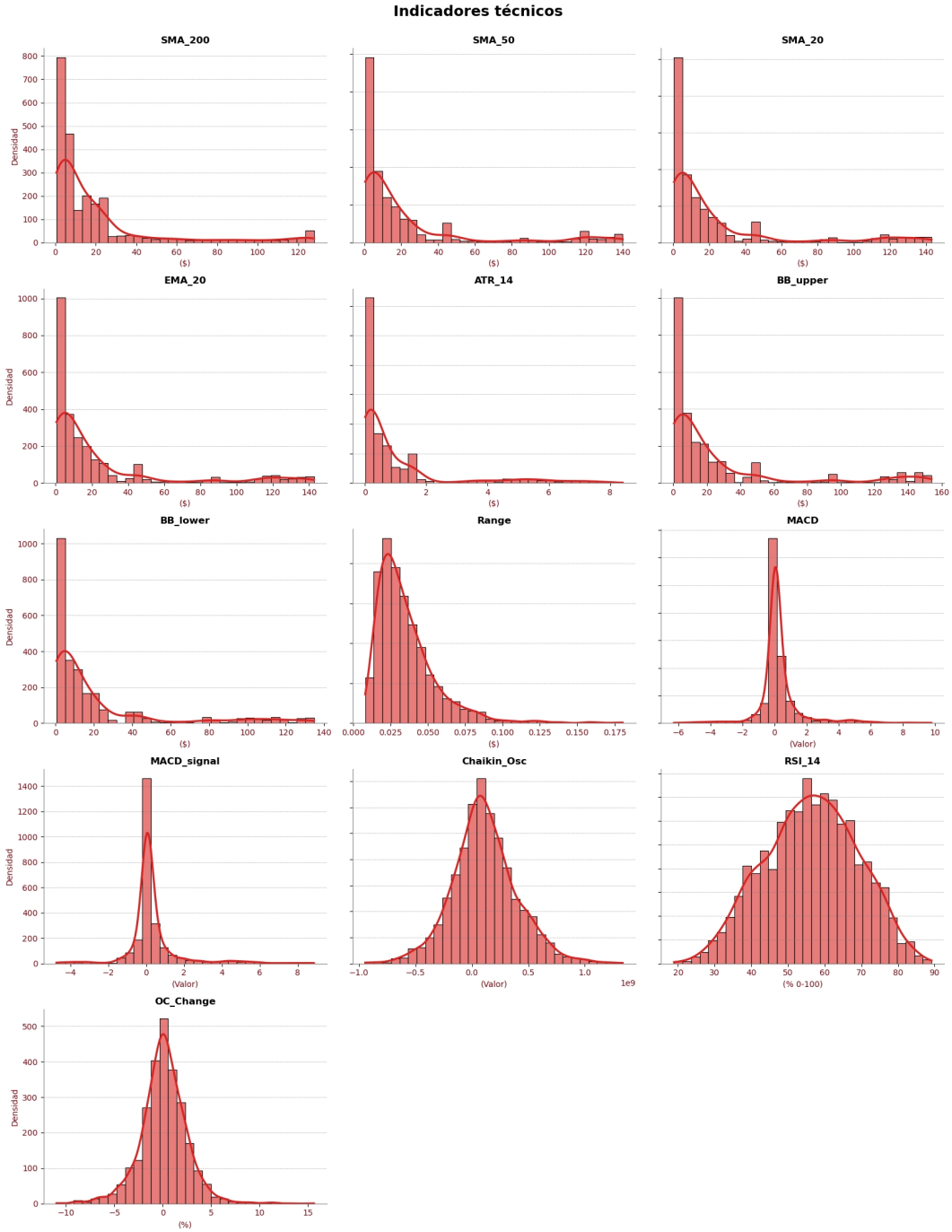


Figura 15 - Histograma y densidad KDE de indicadores técnicos (fuente: elaboración propia).

A continuación, se analizan los histogramas y densidad de las Big Tech donde se observa características distribucionales notablemente diferentes y más complejas en comparación con los datos directos o los indicadores técnicos:

- **Forma general de la distribución**:
  - Todas las distribuciones de este grupo son asimétricas.

- **Tesla** exhibe un fuerte sesgo a la derecha. **Apple**, **Meta**, **Microsoft** y **AMD** también presentan un sesgo a la derecha, pero no tan fuerte.
- **Google**, **Amazon**, e **Intel**, debido a su naturaleza multimodal (ver siguiente punto), presentan formas más complejas que un simple sesgo.

- **Picos (Modas):**
  - **Apple**, **Meta**, **Microsoft**, **Tesla** y **AMD** muestran distribuciones unimodales.
  - **Google** e **Intel** son distribuciones más bimodales, mientras que **Amazon** se puede considerar una distribución multimodal.

- **Dispersión (Visual):**
  - La dispersión es elevada, indicando una alta variabilidad en los precios.
  - **Meta** y **Tesla**, muestran la mayor dispersión. **Intel** muestra la menor dispersión.

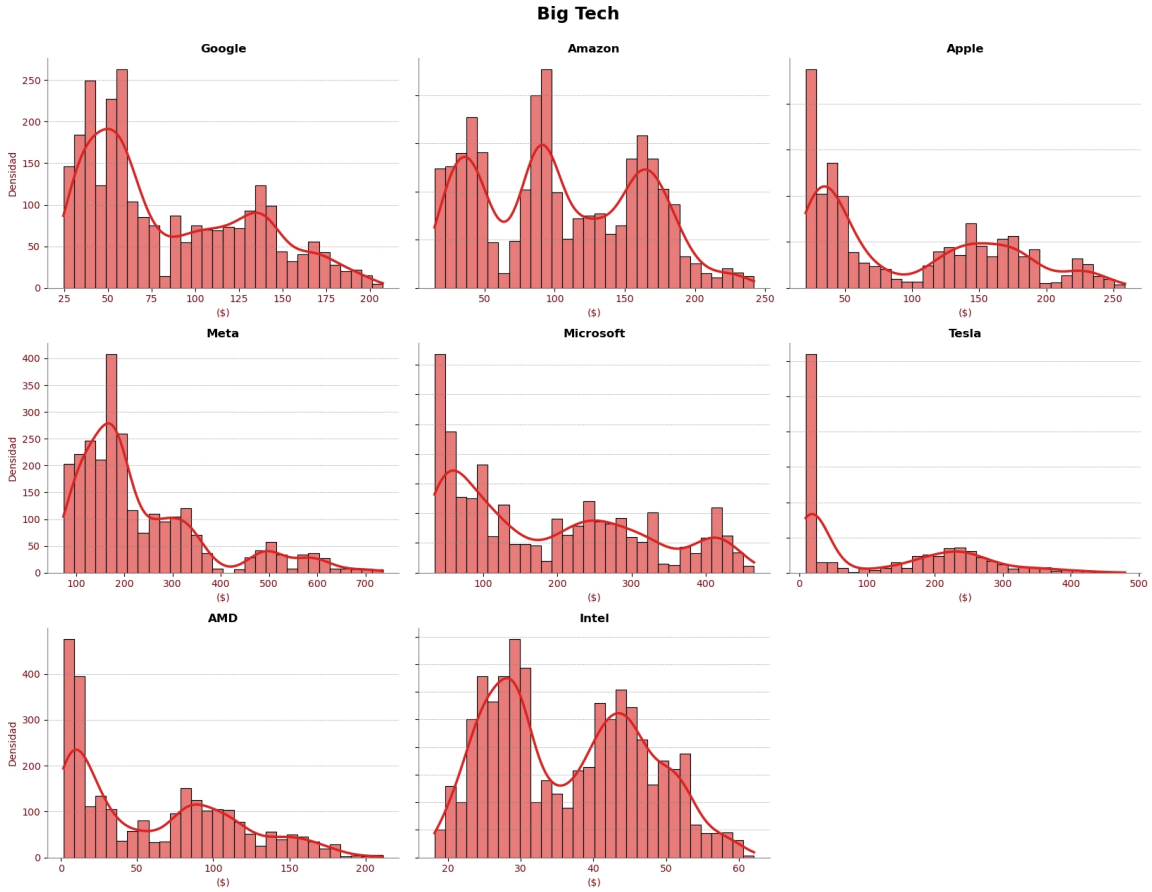


Figura 16 - Histograma y densidad KDE de las Big Tech (fuente: elaboración propia).

El siguiente análisis es de los histogramas y densidad de los índices bursátiles donde se observa:

- **Forma general de la distribución**:
  - El índice **ShanghaiComposite** muestra una distribución fuertemente asimétrica con un claro sesgo a la derecha.

- Los índices **SP500**, **NASDAQ100**, **EuroStoxx50** y **Nikkei225** presentan distribuciones asimétricas sin mostrar un sesgo simple y claro hacia un solo lado.

- **Picos (Modas):**
  - El índice **ShanghaiComposite** es unimodal, con su único pico situado en la parte baja de su rango de valores.
  - Similar a las acciones Big Tech, la multimodalidad es una característica en el resto de índices. Muestran dos o más picos indicando que estos índices han tendido a concentrarse en diferentes niveles a lo largo del período analizado.

- **Dispersión (Visual):**
  - Todos los índices muestran una dispersión considerable, cubriendo amplios rangos de puntos.
  - El índice **ShanghaiComposite** tiene una larga cola hacia la derecha, aunque la mayoría de los valores están en el extremo izquierdo. Comparativamente, su rango absoluto de puntos parece menor que el de otros índices.

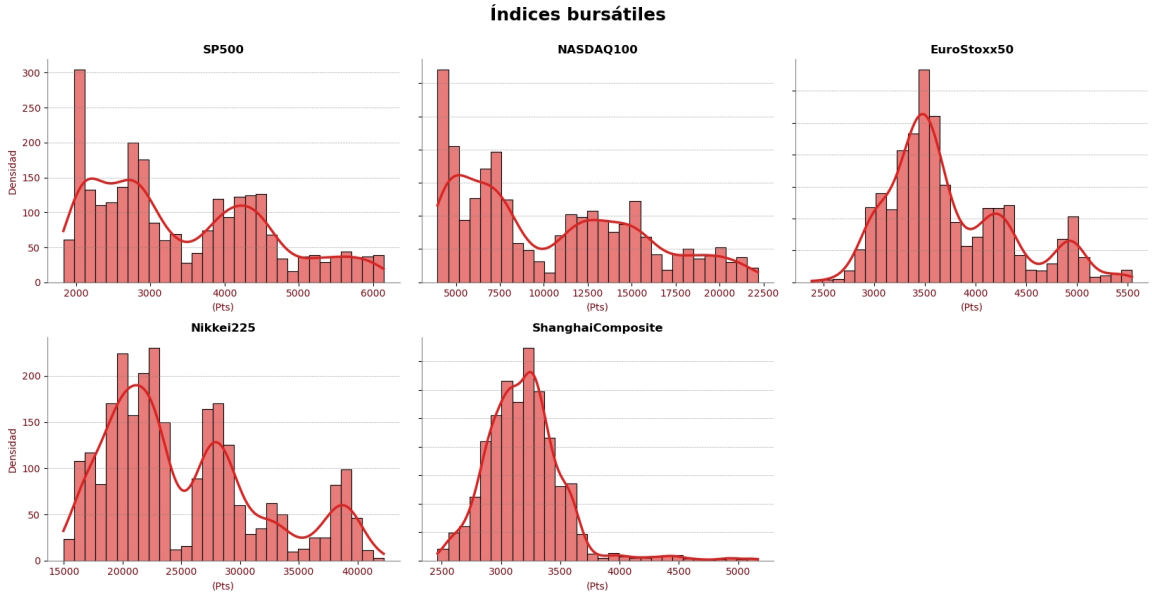


Figura 17 - Histograma y densidad KDE de los índices bursátiles (fuente: elaboración propia).

A continuación, se analizan los histogramas y densidad de los indicadores económicos donde se observa:

- **Forma general de la distribución**:
  - **VIX** muestra una fuerte asimétrica con sesgo a la derecha.

- **Treasury_3M**, **Treasury_10Y**, **Gold** y **CPI** muestran una asimetría multimodal.
- **Brent_Oil** se puede considerar simétrica: se parece a una distribución normal.
- **GDP_Real** y **GDP_per_Capita** se pueden considerar uniformes.

- **Picos (Modas):**
  - **VIX** y **Brent_Oil** son predominantemente unimodales.
  - El resto se pueden considerar multimodales.

- **Dispersión (Visual):**
  - **Brent_Oil** muestra una dispersión moderada alrededor de su pico central.
  - **GDP_Real** y **GDP_per_Capita** tienen una dispersión relativamente contenida dentro de los rangos mostrados en los ejes.
  - El resto, tienen una amplia dispersión.

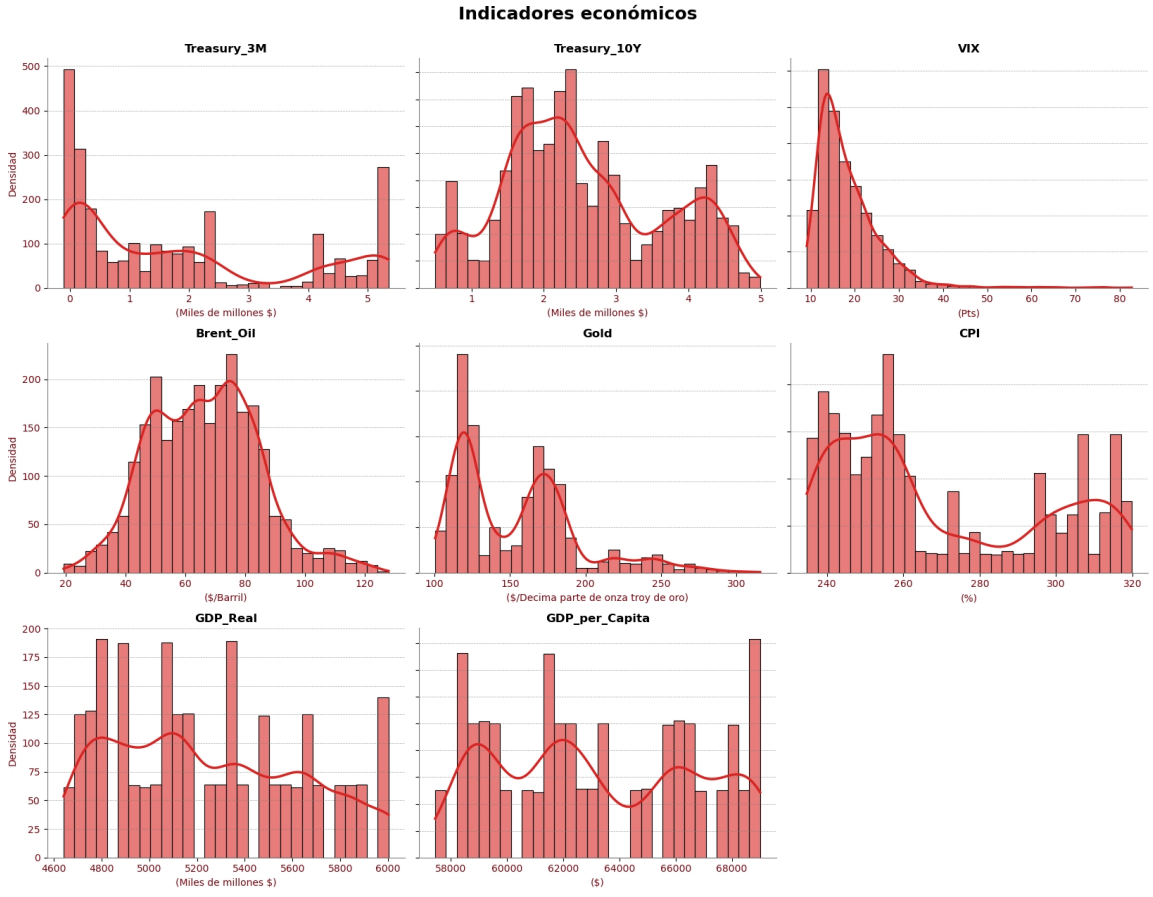


Figura 18 - Histograma y densidad KDE de los indicadores económicos (fuente: elaboración propia).

Finalmente se analizan los histogramas y densidad de las variables de análisis de sentimiento donde se observa:

- **Forma general de la distribución**:

- **googletrends_NVDA** muestra una fuerte asimétrica con sesgo a la derecha.
- **av_nvidia** es casi simétrica alrededor de 0.5 con colas muy finas.

- **Picos (Modas):**
  - **googletrends_NVDA** es unimodal.
  - **av_nvidia** es unimodal.

- **Dispersión (Visual):**
  - **googletrends_NVDA** tiene algunos valores dispersos hasta cerca de 1.0, aunque con frecuencia muy baja. Se observan algunos picos secundarios alrededor de 0.6 y 0.9, posiblemente por eventos puntuales.
  - **av_nvidia** tiene una dispersión extremadamente baja: la mayoría de los valores están entre 0.48 y 0.52, con unos pocos extremos.

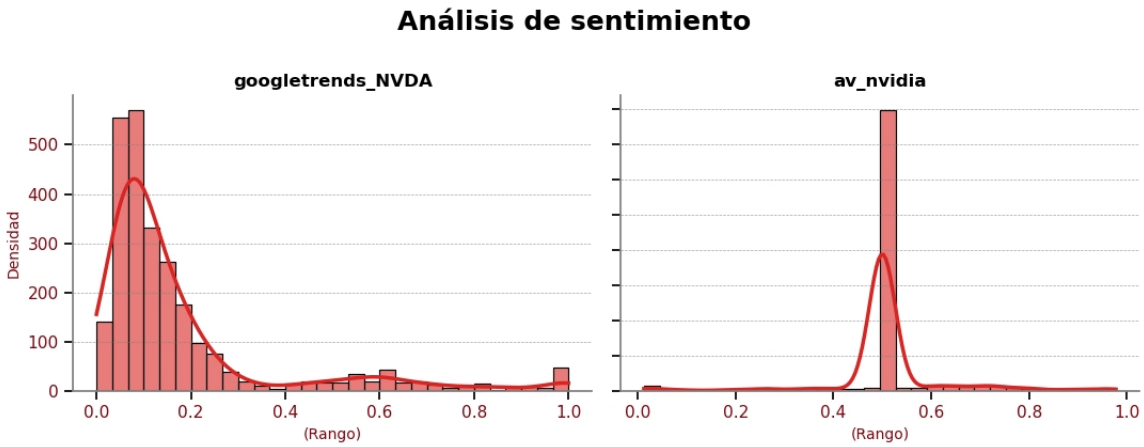


Figura 19 - Histograma y densidad KDE de las variables de análisis de sentimiento (fuente: elaboración propia).

4.8.5 Análisis de valores atípicos

Después de examinar la forma general de las distribuciones, el siguiente paso es la detección e inspección de valores atípicos (outliers). Los valores atípicos son observaciones que se desvían significativamente del patrón general del resto de los datos. Su presencia puede deberse a errores de medición, eventos extraordinarios genuinos en el fenómeno estudiado (por ejemplo, caídas o subidas abruptas del mercado, crisis económicas), o simplemente representar la cola extrema de una distribución muy sesgada.

La identificación de valores atípicos es crucial, ya que pueden tener una influencia desproporcionada en:

- Estadísticas descriptivas (especialmente la media y la desviación estándar).
- El ajuste y el rendimiento de los modelos predictivos.
- La interpretación de los resultados.

En este subapartado, se emplea el método basado en el Rango Intercuartílico (IQR) para la detección visual de valores atípicos, utilizando diagramas de caja. En un diagrama de caja, los puntos que caen más allá de los "bigotes" definidos como 1.5 veces el IQR por encima del tercer cuartil (Q3) o por debajo del primer cuartil (Q1), se consideran valores atípicos potenciales.

Dado que el criterio IQR es más efectivo en distribuciones aproximadamente normales y la mayoría de las variables analizadas presentan colas largas, el método identifica numerosos valores extremos que reflejan movimientos reales del mercado. Para preservar la interpretabilidad económica de precios y volúmenes, los datos se mantienen en su escala original (sin transformaciones logarítmicas).

A partir de estas premisas, se analiza la presencia y naturaleza de los valores atípicos dentro de cada grupo de variables.

En primer lugar, se analizan los diagramas de cajas de los datos directos donde es necesario interpretar los valores atípicos identificados. Los umbrales se calculan basándose en la distribución global de toda la serie histórica. En presencia de una fuerte tendencia alcista, como la observada desde 2015 hasta la actualidad, los valores recientes (precios en el rango 100-140) son significativamente más altos que los cuartiles calculados sobre el histórico completo (que incluyen los valores mucho más bajos de los primeros años, con medianas en torno a 10-15).

Como resultado, estos valores recientes, aunque legítimos y representativos de la dinámica actual, son marcados como valores atípicos superiores. Estos puntos no deben interpretarse como errores o anomalías que requieran eliminación, sino como una manifestación de la no estacionariedad de la serie.

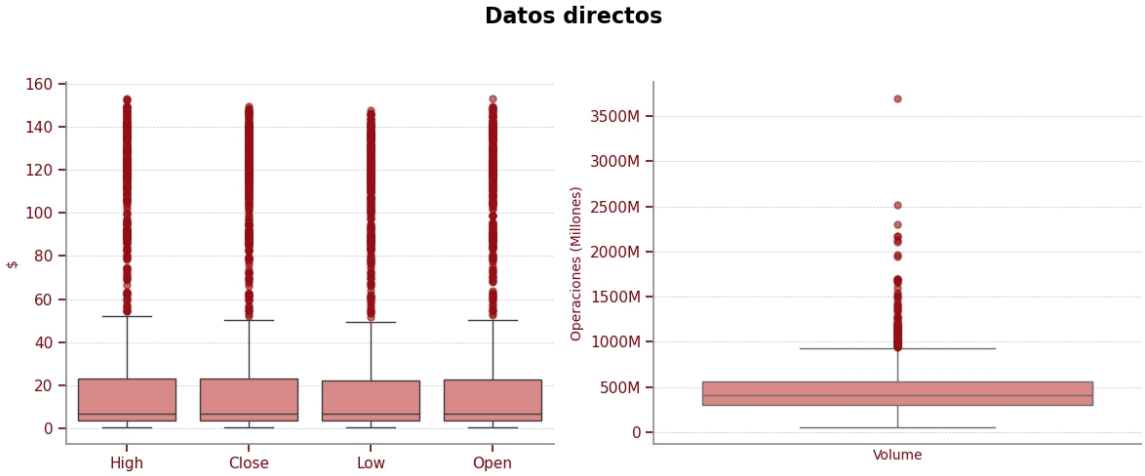


Figura 20 - Detección de valores atípicos en datos directos (fuente: elaboración propia).

El siguiente análisis de diagramas de cajas sobre los indicadores técnicos muestra diversos valores atípicos, pero aplica la misma lógica que con los datos directos por lo que no se interpreta ningún valor como un error o anomalía:

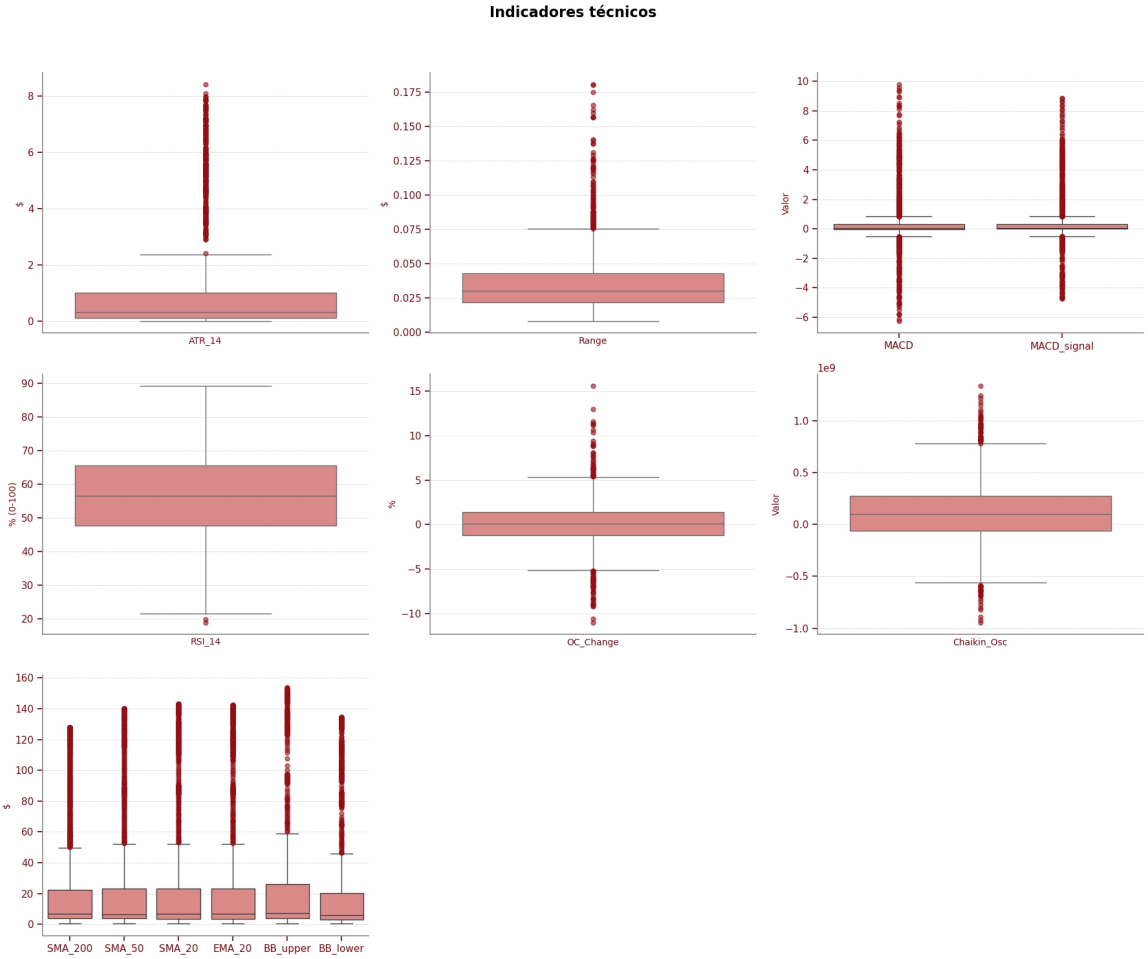


Figura 21 - Detección de valores atípicos en indicadores técnicos (fuente: elaboración propia).

A continuación, se analizan los diagramas de cajas para las Big Tech donde se observa que, para todas las empresas menos **Meta**, no se encuentran valores atípicos. La razón de los valores atípicos para **Meta** se explica del mismo modo que los precios de NVIDIA: su histórico contiene muchos valores bajos, pero recientemente su valor se ha elevado por lo que según el sistema de detección IQR, aparecen como potenciales errores, pero no es el caso y así se puede este crecimiento en la gráfica del histórico de **Meta**.

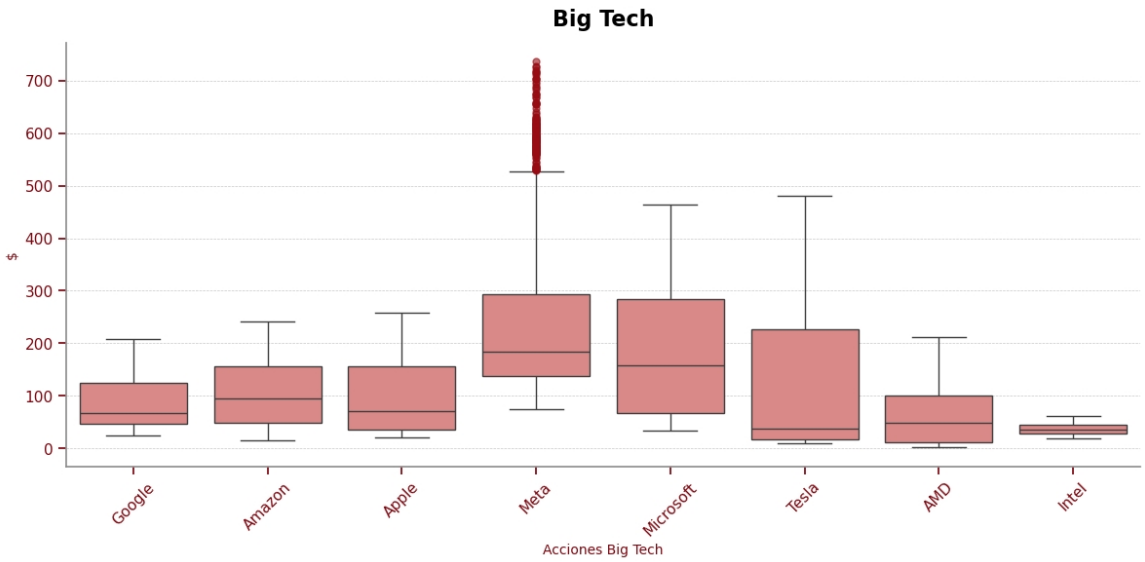


Figura 22 - Detección de valores atípicos en las Big Tech (fuente: elaboración propia).

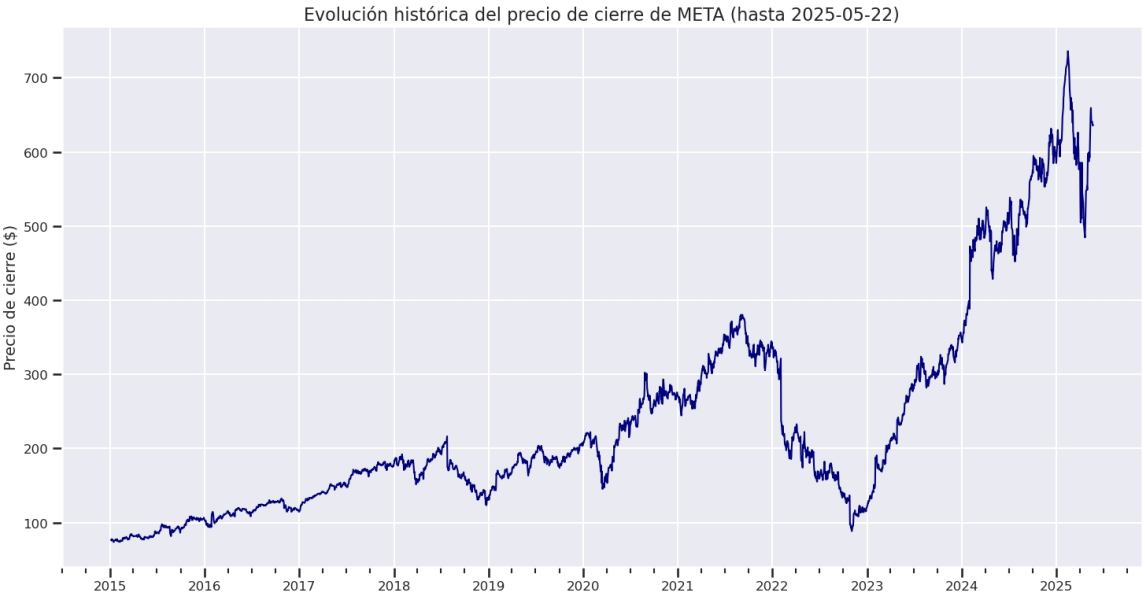


Figura 23 – Evolución histórica del precio de cierre de META (fuente: elaboración propia).

El siguiente análisis corresponde a los diagramas de cajas de los índices bursátiles donde se observan algunos valores atípicos en **EuroStoxx50**, **ShanghaiComposite** y **Nikkei225**. En estos casos se recurre a mostrar la evolución de estos índices en el tiempo para comprobar si se trata del mismo caso que **Meta** (una evolución creciente del valor mostrado) y se observa que es el caso por lo que tampoco los valores atípicos detectados no son errores.

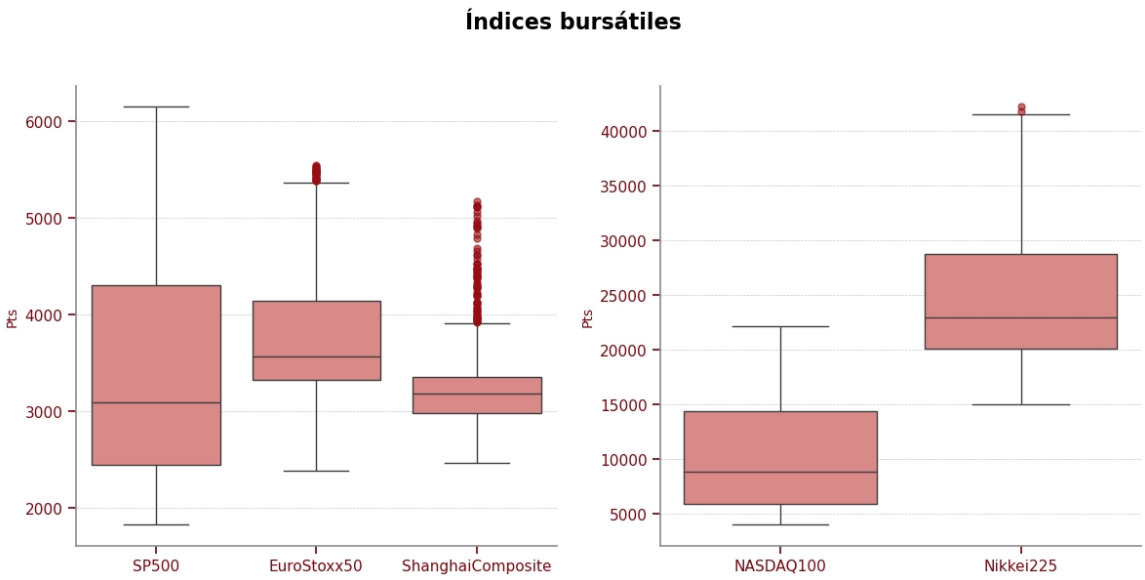


Figura 24 - Detección de valores atípicos en los índices bursátiles (fuente: elaboración propia).

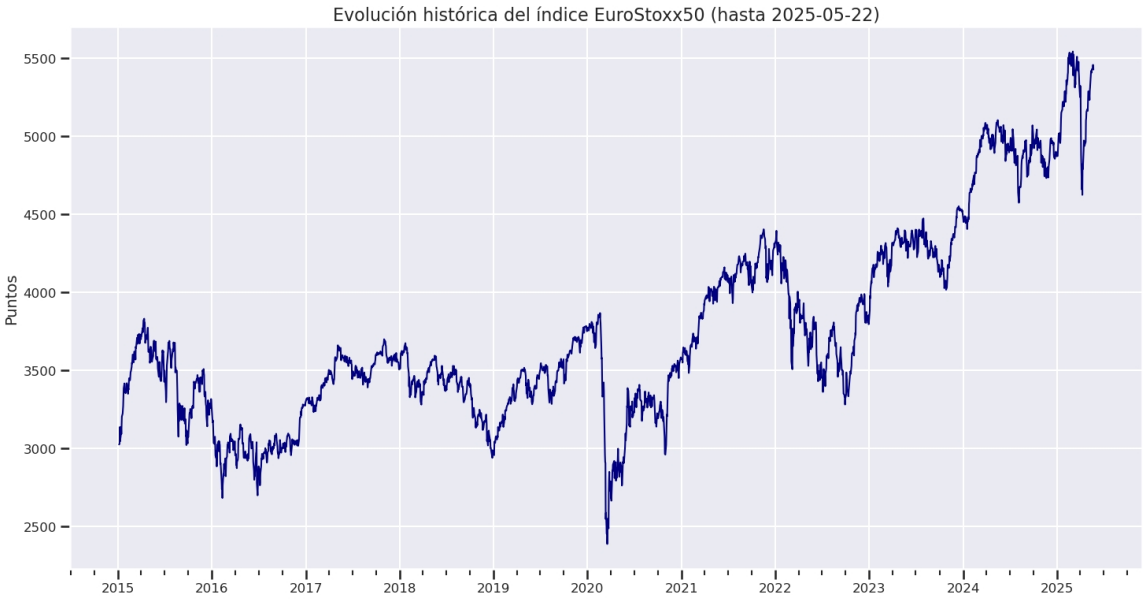


Figura 25 – Evolución histórica del índice EuroStoxx50 (fuente: elaboración propia).

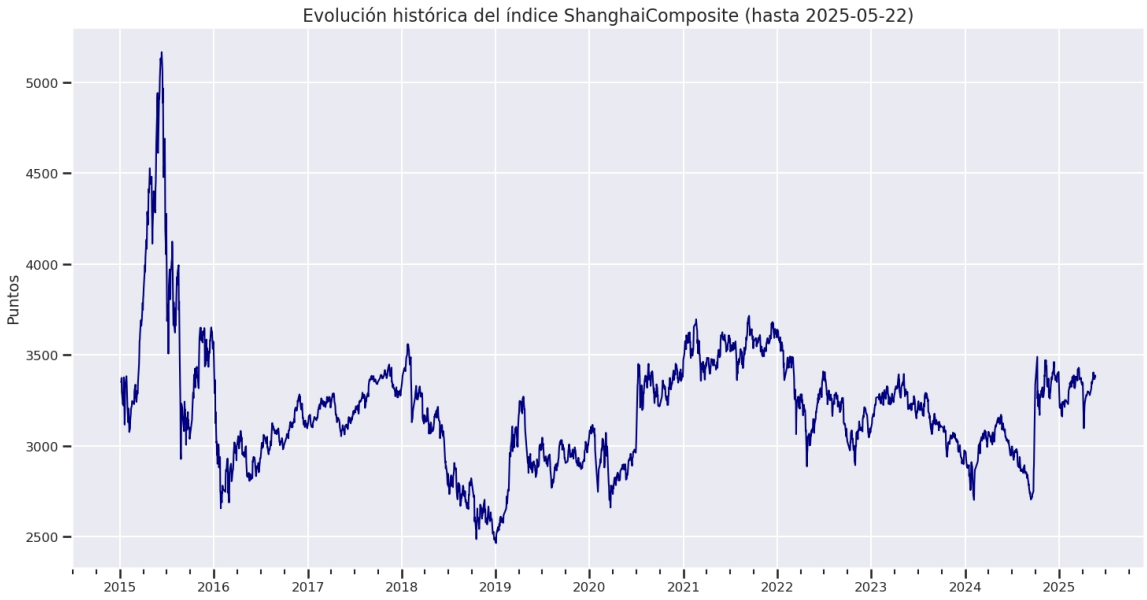


Figura 26 – Evolución histórica del índice ShanghaiComposite (fuente: elaboración propia).

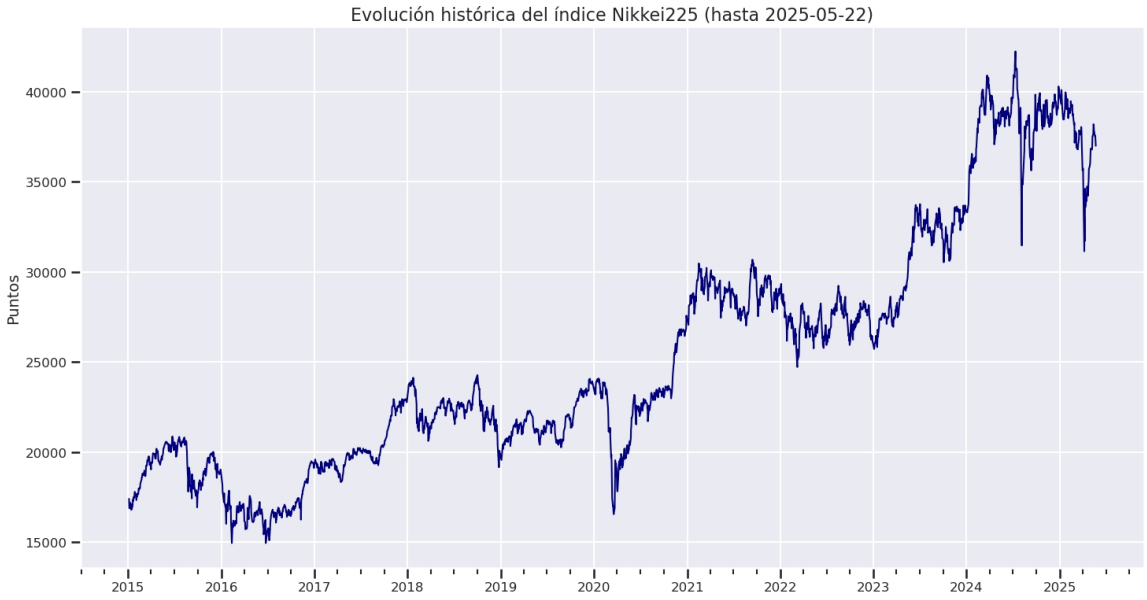


Figura 27 – Evolución histórica del índice Nikkei225 (fuente: elaboración propia).

A continuación, se analizan los diagramas de cajas de los indicadores económicos donde se observan algunos potenciales valores atípicos, pero se descartan con el siguiente análisis:

- **VIX**:
  Dado que esta variable indica la volatilidad esperada del mercado, tiende a permanecer en valores bajos durante periodos de calma (la mayoría del tiempo) y en momentos de

alta incertidumbre como la crisis del 2008 o el COVID-19 (eventos puntuales), se dispara a valores elevados. Por lo tanto, estos valores atípicos detectados no son errores, sino que representan estos eventos genuinos de extrema volatilidad.

- **Brent_Oil**:
  El precio del petróleo es sensible a una multitud de factores. Los valores atípicos detectados probablemente corresponden a periodos históricos de precios del petróleo altos impulsados por eventos como recortes significativos de producción por la OPEP, conflictos geopolíticos en regiones productoras, o picos de demanda. Se descarta que sean errores.

- **Gold**:
  El precio del oro desde 2015 hasta finales de 2024 permanecía en el rango entre 100 y 200 con una ligera tendencia alcista, pero en el 2025, se ha disparado por encima de los 300 por lo que los valores atípicos detectados corresponden a esta evolución creciente y acelerada en los últimos años (mismo patrón que con **Meta** y algunos índices bursátiles). Por lo tanto, estos valores atípicos no son errores si no el efecto de esta tendencia de crecimiento reciente acelerado.

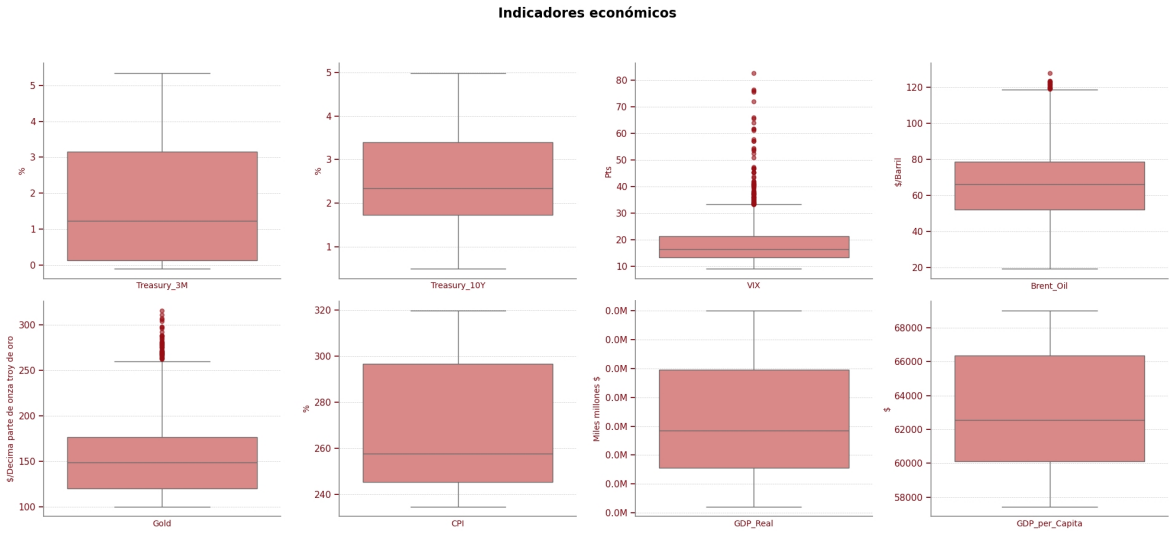


Figura 28 - Detección de valores atípicos en los indicadores económicos (fuente: elaboración propia).

Finalmente, se analizan los diagramas de cajas de las variables de análisis de sentimiento donde se observan potenciales valores atípicos, pero se descartan con el siguiente análisis:

- **googletrends_NVDA**:
  Hay una gran cantidad de puntos por encima del bigote superior que corresponden a picos reales de búsqueda probablemente durante eventos clave relacionados con

NVIDIA (presentaciones, resultados, noticias tecnológicas). En este contexto, no son valores atípicos, sino que reflejan comportamientos extremos pero significativos.

- **av_nvidia**:
  Tiene una línea estrecha en el valor 0.5 y después una nube simétrica de puntos arriba y abajo pero no se trata de valores atípicos, la explicación es que gran parte de los datos fueron rellenados con 0.5 lo que crea una distribución artificial y los otros valores son realmente los que añaden un valor significativo.

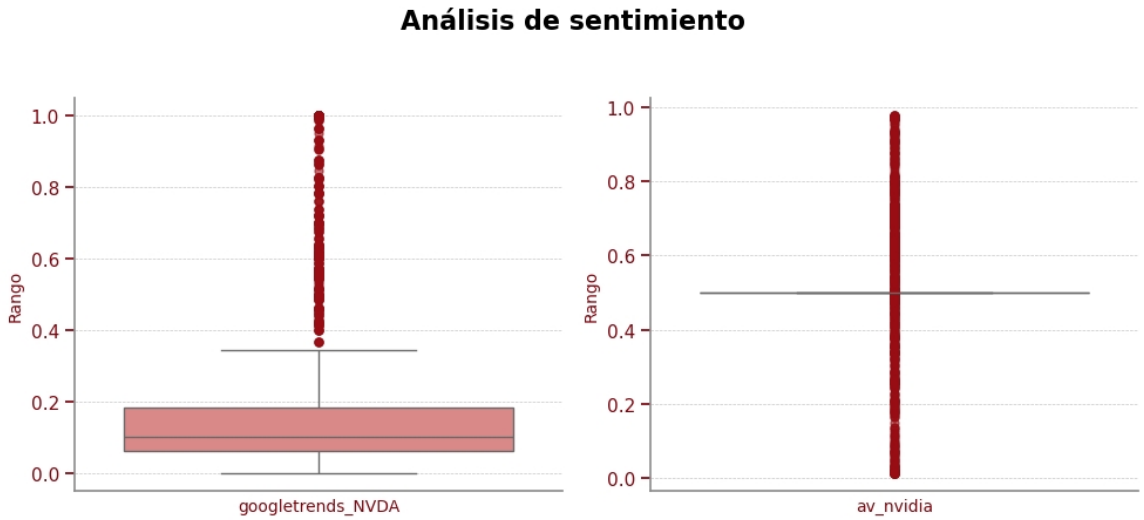


Figura 29 - Detección de valores atípicos en las variables de análisis de sentimiento (fuente: elaboración propia).

En conclusión, en el análisis de valores atípicos se concluye que no se ha detectado ningún error y que los valores detectados, tienen una explicación coherente con el tipo de dato asociado. En cualquier caso, algunos modelos podrán emplear la técnica estadística de Winsorización [42] para manejar estos valores reemplazándolos por los percentiles más cercanos, en lugar de eliminarlos.

4.8.6 Correlaciones entre variables

La correlación mide la intensidad y dirección de la relación lineal entre dos variables. En este análisis, se calcula la correlación entre el precio de cierre (**Close**) y el resto de variables para identificar cuáles están más relacionadas con el objetivo. Los valores de correlación oscilan entre -1 y 1: un valor cercano a 1 indica una relación positiva fuerte, cercano a -1 una relación negativa fuerte, y cercano a 0 una ausencia de relación lineal.

Dado el gran número de variables, a continuación, se muestran las figuras con los mapas de calor por grupos, con el resumen de todas las variables con **Close** y de todas las variables.

En concreto, en la *Figura 36* se observa que **Shanghaicomposite**, **RSI_14**, **VIX**, **Chainkin_Osc** y **OC_Change** tienen un valor muy cercano a 0 indicando que apenas hay alguna correlación con **Close**. Probablemente pueden añadir más información al combinarse con otras variables, este hecho se tendrá en cuenta a la hora de entrenar los modelos.

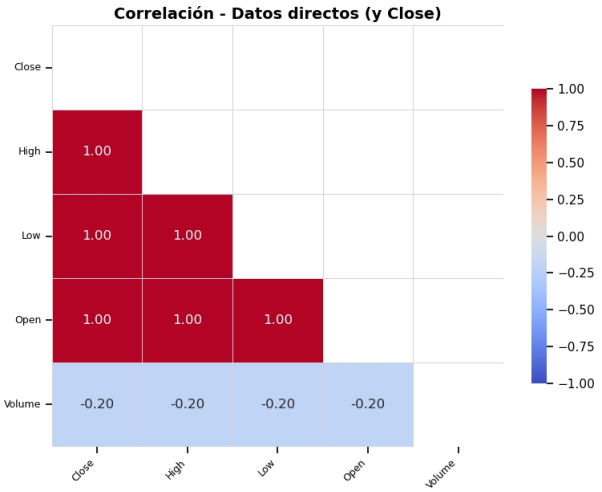


*Figura 30- Correlación de datos directos y **Close** (fuente: elaboración propia).*

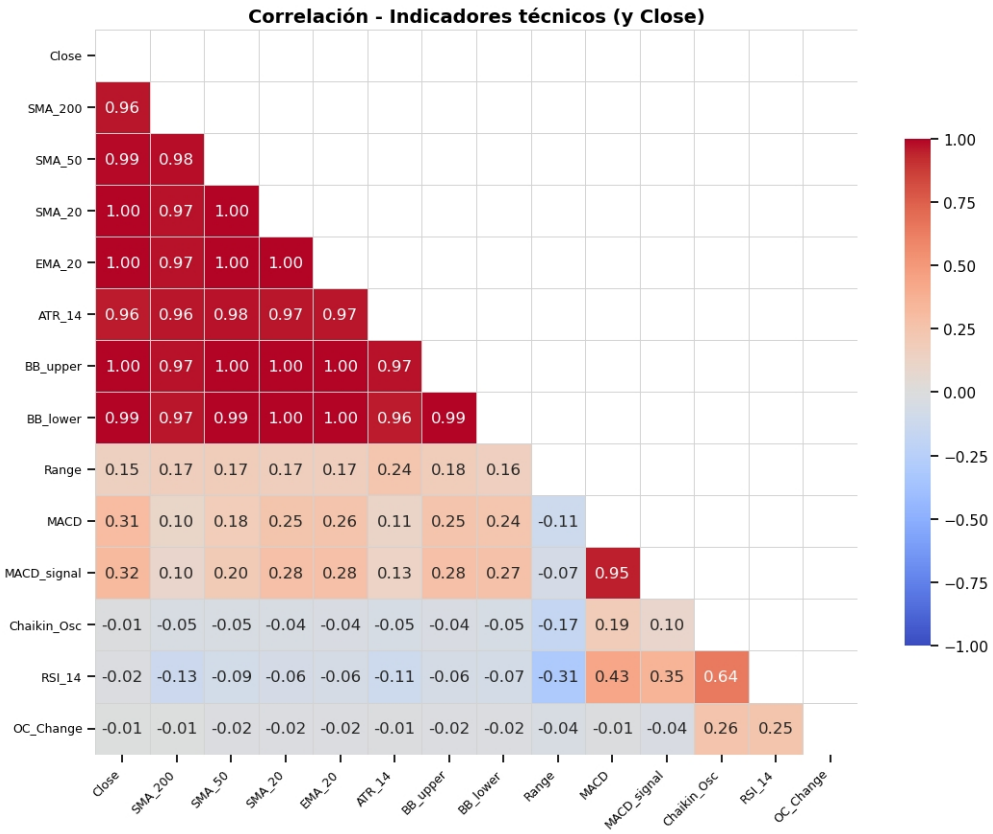


*Figura 31- Correlación de indicadores técnicos y **Close** (fuente: elaboración propia).*

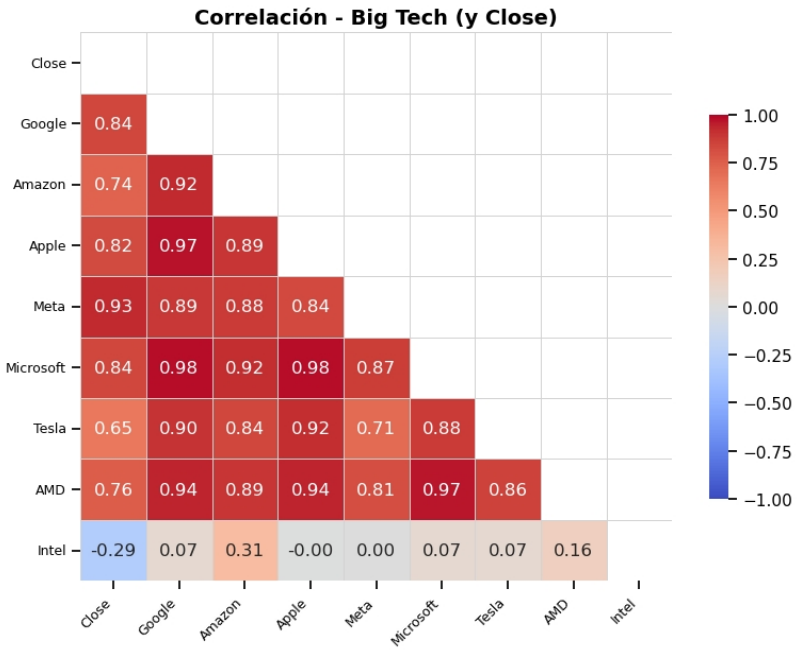


*Figura 32 - Correlación de las Big Tech y **Close** (fuente: elaboración propia).*

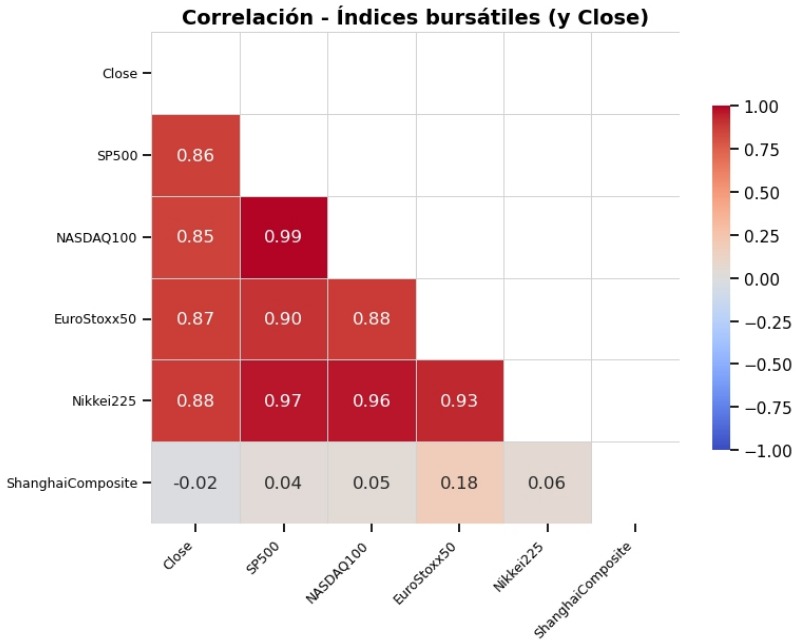


*Figura 33 - Correlación de índices bursátiles y **Close** (fuente: elaboración propia).*

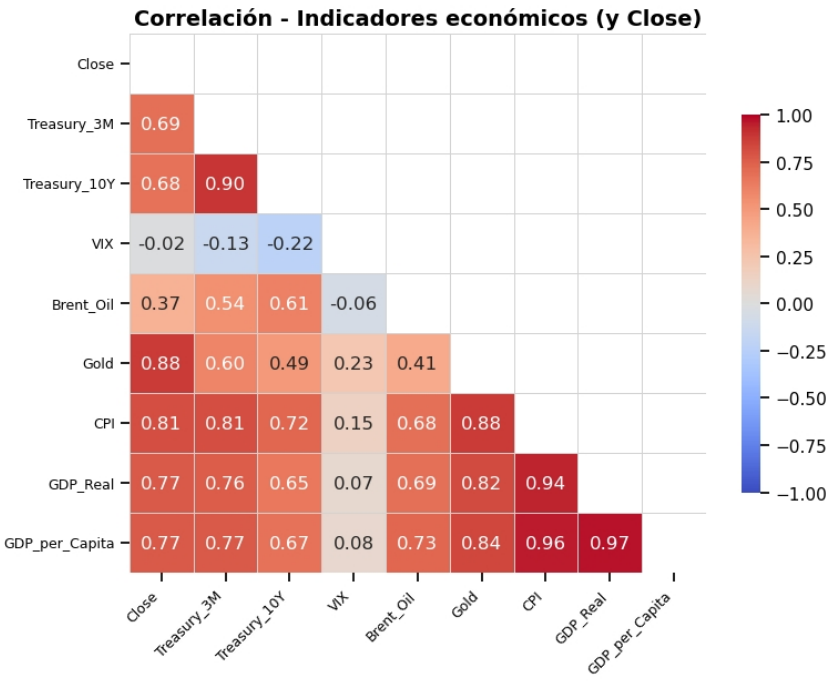


*Figura 34- Correlación de indicadores económicos y **Close** (fuente: elaboración propia).*

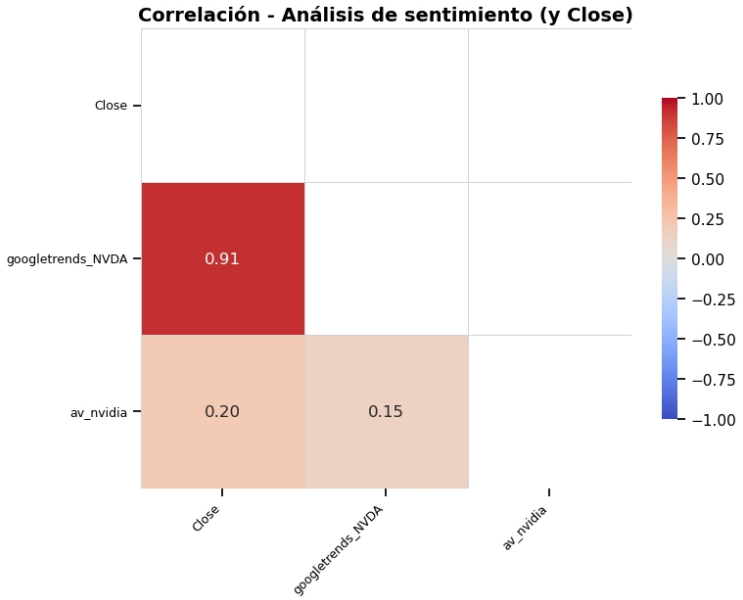


*Figura 35- Correlación de variables de análisis de sentimiento y **Close** (fuente: elaboración propia).*

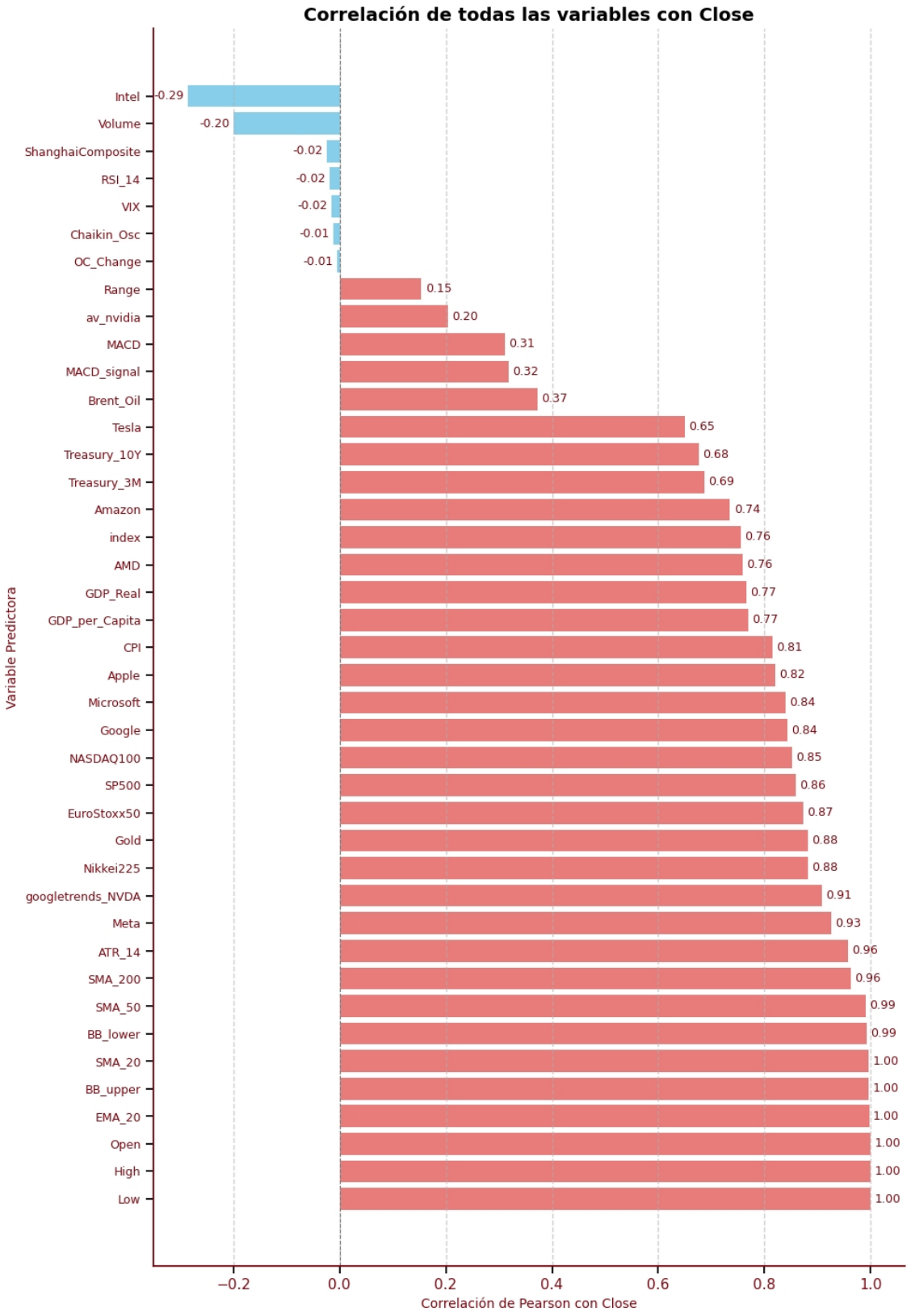


*Figura 36 - Correlación de todas las variables con **Close** (fuente: elaboración propia).*

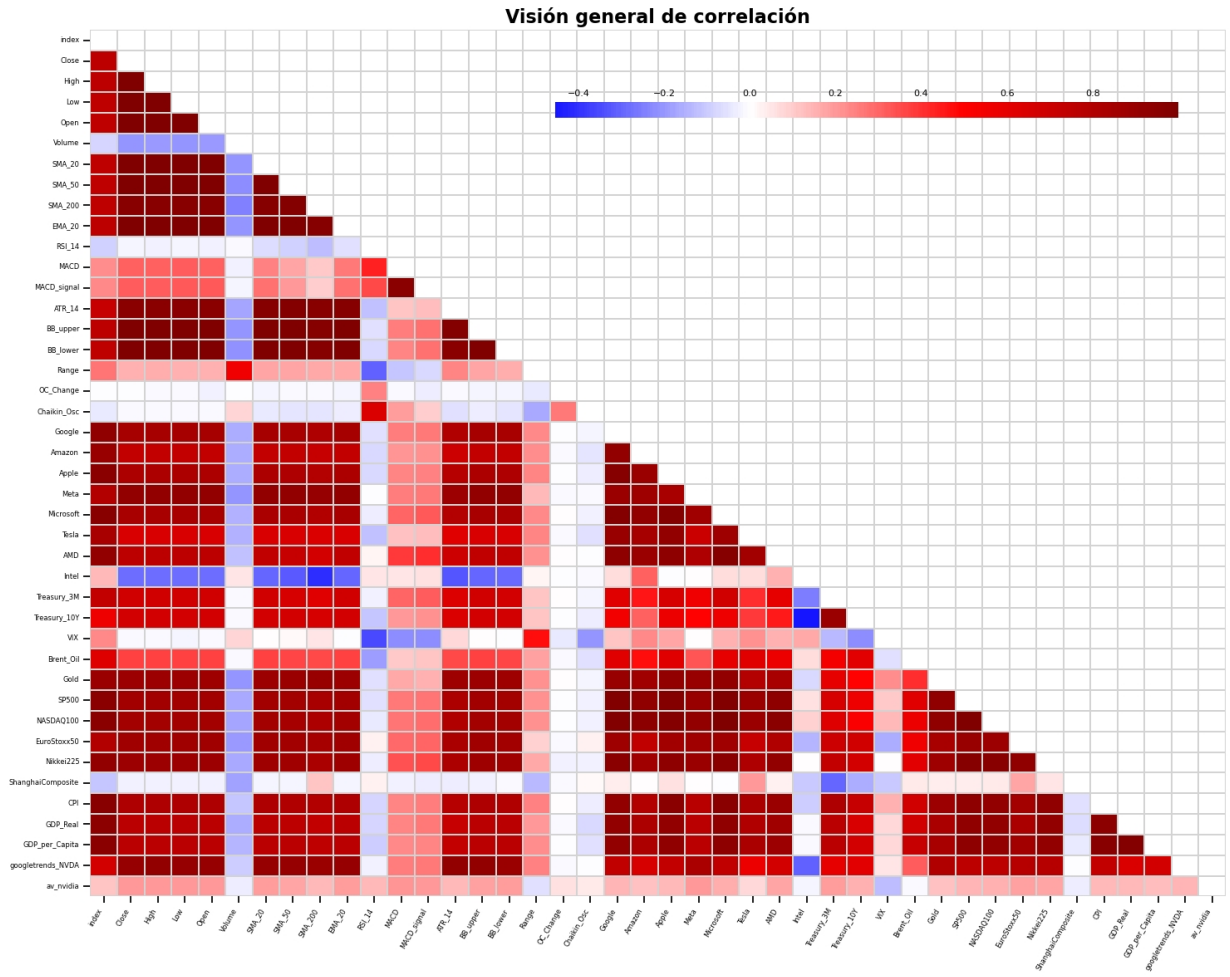


Figura 37 - Correlación de todas las variables (fuente: elaboración propia).

En conclusión, menos 5 variables que muestran muy poca correlación con **Close**, el resto si tienen una correlación destacable.

4.8.7 Normalización y escalado de datos

Para mejorar el rendimiento y la estabilidad de los modelos, se aplican técnicas de normalización y escalado, ya que las diferencias de escala entre variables pueden comprometer la eficiencia del entrenamiento y la calidad de las predicciones. Por ello, se han considerado dos métodos, seleccionando el más adecuado en función del tipo de modelo y su sensibilidad a la magnitud de los datos:

- **Min-Max:**
  Transforma los valores de cada variable para que se encuentren en un rango definido, generalmente entre 0 y 1. Es especialmente útil cuando los datos no siguen una distribución normal y se desea preservar la forma original de la distribución.

- **Estandarización (StandardScaler):**
  Escala los datos en función de su media y desviación estándar, generando una distribución con media cero y varianza unitaria. Es ampliamente utilizado en modelos de aprendizaje profundo y en algoritmos que dependen del cálculo de gradientes, como redes neuronales, debido a que facilita una convergencia más eficiente.

En el caso particular de la variable **googletrends_nvda**, dado que Google Trends impone un límite de cuatro años con granularidad diaria, se dividió el periodo completo en tres intervalos independientes, cada uno normalizado internamente por Google Trends, lo que imposibilita su comparación directa. Para resolver esta limitación, se utilizó un archivo adicional con datos mensuales agregados de 2015 a 2025 como referencia global, a partir del cual se calcularon factores de ajuste mensuales que escalan cada intervalo diario. Posteriormente, se acotaron los valores extremos y se aplicó una normalización Min-Max global, generando una señal continua y coherente entre 0 y 1, adecuada para su uso como variable exógena.

Cada modelo incorpora una estrategia de normalización o escalado únicamente si contribuye a mejorar su rendimiento y evitando transformaciones innecesarias.

4.8.8 Codificación de variables categóricas

En el conjunto de datos utilizado en este proyecto no hay variables categóricas, la única semejante es el análisis de sentimiento. Para incorporarlo a los modelos sin recurrir a codificación categórica, el sentimiento se transforma en un valor continuo entre 0 y 1:

- 0 indica un sentimiento muy negativo.
- 0.5 representa un sentimiento neutro.
- 1 indica un sentimiento muy positivo.

Así, la señal se trata como variable numérica y se integra directamente en el entrenamiento.

Finalmente, en la *Tabla 10*, se muestran todas las variables seleccionadas:

Conjunto de datos	Variables para predecir Close
1 - Datos directos (4)	Open, High, Low, Volume
2 - Indicadores técnicos (13)	SMA_20, SMA_50, SMA_200, EMA_20, RSI_14, MACD, MACD_signal, ATR_14, BB_upper, BB_lower, Range, OC_Change, Chaikin_OSC
3 - Big Tech (8)	Google, Amazon, Apple, Meta, Microsoft, Tesla, AMD, Intel
4 - Índices bursátiles (5)	SP500, NASDAQ100, EuroStoxx50, Nikkei225, ShanghaiComposite
5 - Indicadores económicos (8)	CPI, GDP_Real, GDP_per_Capita, Treasury_3M, Treasury_10Y, VIX, Brent_Oil, Gold
6 - Análisis de sentimiento (2)	googletrends_NVDA, av_nvidia

Tabla 10 - Listado final de variables (por conjunto de datos) después de procesado entre 2015-01-05 y 2025-05-23.

4.9 Herramientas y tecnologías

A continuación, se detalla el conjunto de herramientas y tecnologías utilizadas en el proyecto, explicando brevemente su función y relevancia:

- **Dispositivos y equipos de trabajo.**
  - Portátil: MacBook Pro.
    Un equipo que permite la movilidad y ejecución eficiente de tareas de análisis y desarrollo y, sobre todo, para realizar las presentaciones necesarias.
  - Equipo Fijo: Workstation.
    Una estación de trabajo dedicada con gran capacidad de procesamiento, ideal para entrenar modelos y realizar tareas intensivas en cálculos.

- **Sistemas operativos y software de apoyo.**
  - macOS y Microsoft Windows 11.
    Sistemas operativos que proporcionan la plataforma base para el desarrollo y ejecución de las aplicaciones.
  - WSL2.
    Funcionalidad de Windows que permite ejecutar un entorno Linux completo facilitando el uso de librerías, entornos y scripts nativos de Linux.
  - Microsoft Office (Word, Excel, PowerPoint).
    Herramientas utilizadas para la documentación, análisis de datos y presentación de resultados.
  - Microsoft Project.
    Software de gestión de proyectos que facilita la planificación, seguimiento y control del cronograma del proyecto.
  - Notepad++.
    Un editor de texto ligero utilizado para la edición rápida de scripts.
  - Paint.Net:
    Herramienta de diseño gráfico gratuita.
  - WinMerge.
    Permite comparar y fusionar archivos, útil para la gestión de versiones.
  - PowerBI.
    Herramienta que permite construir paneles interactivos y accesibles para mostrar de forma gráfica información relevante.
  - Docker.
    Plataforma basada en contenedores que permite empaquetar y desplegar la aplicación de simulación de forma consistente, facilitando su ejecución en entornos distintos de forma sencilla.

- **Entorno de desarrollo y herramientas de programación.**
  - Microsoft Visual Studio Code.
    Un potente editor de código que soporta múltiples lenguajes y extensiones, facilitando el desarrollo, depuración y mantenimiento del software.

- Python.
  Lenguaje de programación principal del proyecto, empleado por su versatilidad y amplia comunidad y gran soporte. Se utilizan las siguientes librerías:

Librerías	Descripción
pandas y numpy	Para la manipulación, análisis y procesamiento eficiente de datos.
matplotlib, seaborn y plotly	Herramientas de visualización gráfica que permiten un análisis exploratorio detallado.
kaleido	Permite exportar gráficos a imágenes PNG.
scikit-learn	Proporciona una variedad de algoritmos de aprendizaje automático y herramientas de validación para el modelado predictivo.
statsmodels	Utilizada para implementar modelos estadísticos, como ARIMA, en el análisis de series temporales.
prophet	Librería desarrollada por Facebook para modelar series temporales y facilitar la predicción de tendencias.
pmdarima	Automatiza la implementación y optimización de modelos ARIMA.
pandas-datareader	Integra datos de diversas fuentes en línea, complementando el análisis con datos económicos y financieros.
TensorFlow, Keras y Optuna	Frameworks de aprendizaje profundo, para el desarrollo de modelos de redes neuronales (LSTM y Transformers). Optuna es un optimizador de hiperparámetros.
tabulate	Permite formatear y mostrar datos en forma de tablas ordenadas.
nbformat	Permite leer, escribir y manipular archivos de Jupyter Notebook.
praw	Permite utilizar la API de Reddit, para automatizar la extracción de publicaciones.
XGBoost	Librería de boosting que facilita la construcción de modelos predictivos.
alpha_vantage	Proporciona acceso a datos financieros a través de una API gratuita.
yfinance	Permite la descarga de datos históricos del mercado bursátil desde Yahoo Finance.
requests	Simplifica el envío de peticiones HTTP/HTTPS.
pymongo	Driver que permite interactuar con MongoDB.
argparse	Analiza y obtiene argumentos de línea de comandos de forma fácil y flexible.
PyTorch, Transformers y FinBERT	Framework de aprendizaje profundo utilizado para ejecutar FinBERT: un modelo de lenguaje basado en BERT entrenado con datos financieros.
Flask, WTForms	Flask es un framework para web minimalistas y WTForms se utiliza para la validación, representación y manejo seguro de datos enviados por el usuario.

Tabla 11 – Librerías utilizadas de Python.

- **Gestión y Control de Código.**
  - TortoiseSVN (Subversion).
    Sistema de control de versiones para gestionar el código fuente del proyecto.
  - GitHub.
    Plataforma final para almacenar el proyecto y permitir su visibilidad y uso.

4.10 Evaluación de los modelos

En este apartado se describen las estrategias de evaluación aplicadas a los modelos predictivos, que combinan métricas de regresión y clasificación obtenidas mediante dos pruebas complementarias:

- **Validación estática (SV)** sobre los últimos 60 días (conjunto *test*):
  El modelo dispone de todas las variables exógenas reales del día *t*, por lo tanto, la estimación del precio de cierre es más precisa, aunque menos representativa de un entorno operativo real.

- **Backtesting con *walk-forward* (BT)** [43] en los mismos 60 días:
  En este caso el modelo no conoce las variables exógenas del día a predecir. Para simplificar, se asume que las exógenas de *t* son las de *t-1* salvo $\mathbf{Open_t}$ que se iguala a $\mathbf{Close_{t-1}}$. Esta aproximación, llamada "sin desfase" e ilustrada en la *Figura 38*, no solo incrementa el realismo de la prueba, sino que facilita una futura extensión a predicción intradía, donde se emplearían las variables disponibles en el instante de la inferencia.

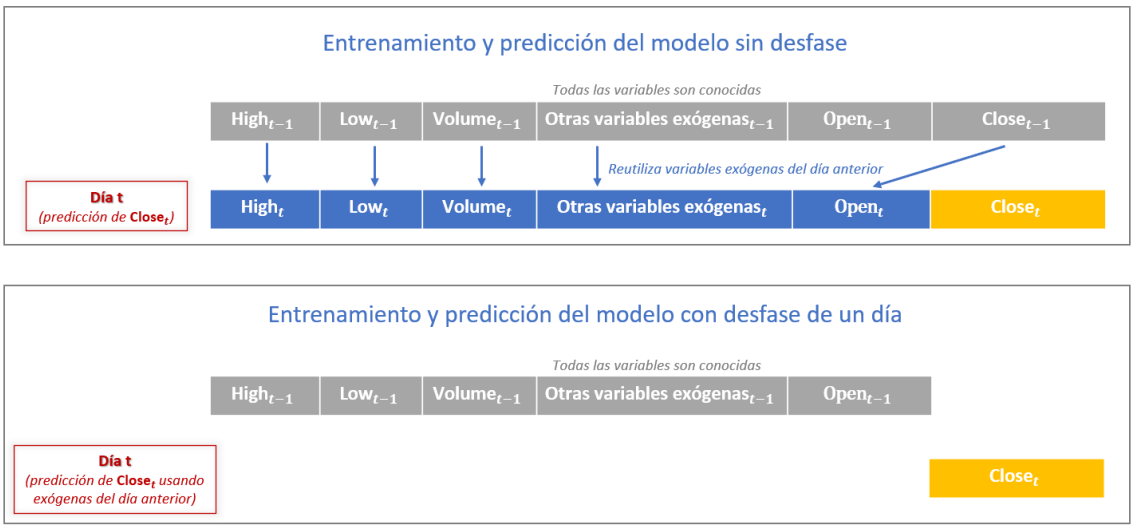


Figura 38 - Entrenamiento y predicción sin de fase y con desfase (fuente: elaboración propia).

Existe la alternativa de entrenar con un desfase de un día: predecir $\mathbf{Close_t}$ empleando todas las exógenas de *t–1*. Sin embargo, el modelo debe aprender simultáneamente la dinámica del precio y la dependencia diferida de las exógenas, lo que incrementa la complejidad y degrada la precisión como se puede observar en las *Figura 49* y *Figura 50* (Anexo 9.1). Por este motivo, dicha variante se ha descartado en favor del esquema sin desfase.

Para mostrar los resultados, se ha creado un panel visual de resultados a modo de plantilla que contiene las métricas de regresión, la gráfica de predicción y valor real y las métricas de clasificación. En la *Figura 39* se muestra un ejemplo y se detalla el código utilizado en los nombres que indica la configuración y conjunto de datos utilizados.

Figura 39 - Panel visual de resultados y significado de su nombre (fuente: elaboración propia).

Debido a la gran cantidad de resultados (~2280), estos paneles visuales de resultados se guardan de forma estructurada como ficheros PNG y CSV con todas las métricas clave en la ruta:

"Resultados/<nombre del modelo>/[SV,BT]_<configuración>_<transformación>_<escalado>_<conjunto de datos>"

Por último, para facilitar la comparación entre modelos se han definido dos índices los cuales se combinan en un índice final agregado ponderando un 55% la clasificación (dirección del precio) y un 45% la regresión (precisión del precio) reflejando la prioridad del proyecto: se considera más relevante anticipar la tendencia que acertar con exactitud absoluta el valor.

- Índice Relativo al Baseline (IRB):
  Utilizando durante el entrenamiento de cada modelo con todas sus posibles combinaciones a modo de comparación contra el predictor base. De modo que la base es 1 y el IRB del modelo/combinación puede ser mayor, igual o menor que 1 indicando si es peor (<1), igual (=1) o mejor (>1). Esto permite una evaluación local e independiente.

- Comparación Global por MinMax (CGMM):
  Se escogen las métricas de regresión y clasificación de cada modelo con los 4 índices IRB más prometedores. Se normalizan todos estos valores mediante escalado Min-Max y se combinan en un índice final agregado de tal modo que se puede comparar entre todos ellos y escoger los mejores.

4.10.1 Métricas de regresión

Para evaluar la precisión en la predicción del precio de cierre de las acciones de NVIDIA se utilizan las siguientes métricas, comparando los modelos desarrollados contra un predictor base (modelo naive), que asume que el precio del día siguiente será igual al del día actual:

- **Error Absoluto Medio (MAE):**
  Evalúa la magnitud promedio de los errores entre las predicciones y los valores reales, sin considerar la dirección del error, y se expresa en la misma unidad de los datos (precio). Es útil para obtener una estimación directa del error promedio.

$$MAE = \frac{1}{n}\sum_{i=1}^{n}\left|y_i - \hat{y}_i\right|$$

  Donde y_i es el valor real del precio de cierre en el día i, $\hat{y}_i$ es el valor predicho por el modelo para el día i y n es el número total de observaciones.

- **Error Cuadrático Medio (MSE):**
  Calcula la media de los errores al cuadrado, dando mayor énfasis a los errores de mayor magnitud. Aunque su valor se expresa en unidades al cuadrado, resulta valioso para identificar la presencia de desviaciones significativas.

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$$

- **Raíz del Error Cuadrático Medio (RMSE):**
  Es la raíz cuadrada del MSE y se expresa en las mismas unidades que los datos originales, permitiendo una interpretación más intuitiva del error global de predicción.

$$RMSE = \sqrt{MSE}$$

- **Coeficiente de Determinación (R^2):**
  Indica la proporción de la variación en el precio de cierre que es explicada por el modelo. Un valor cercano a 1 revela que el modelo captura de forma robusta la variabilidad de los datos, mientras que valores más bajos sugieren un ajuste deficiente.

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2}$$

  Donde $\bar{y}$ representa la media de los valores reales.

4.10.2 Métricas de clasificación

Para complementar la evaluación de modelos, se emplean métricas de clasificación que permiten evaluar la capacidad del modelo para predecir la dirección del movimiento del precio (si subirá o bajará en comparación con el día anterior).

La base de estas métricas es la matriz de confusión, una herramienta que permite comparar de forma sistemática las predicciones del modelo con los valores reales. En esta matriz se registran los aciertos y errores en la clasificación, categorizados en verdaderos positivos, falsos positivos, verdaderos negativos y falsos negativos.

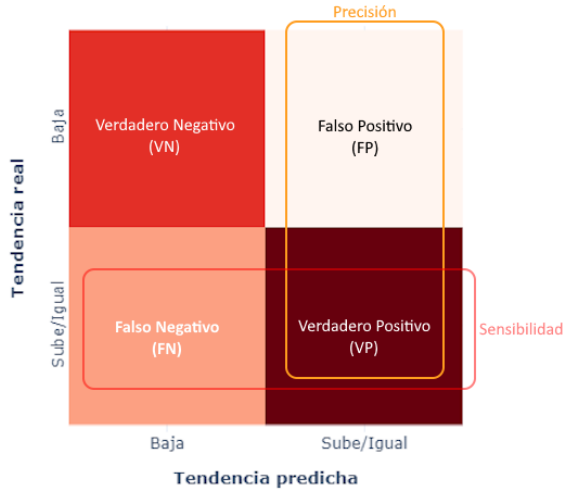


Figura 40 - Descripción gráfica de la matriz de confusión (fuente: elaboración propia).

Donde:

- **VN** son los verdaderos negativos (el modelo predice bajada y efectivamente baja).
- **FP** son los falsos positivos (el modelo predice una subida, pero baja).
- **FN** son los falsos negativos (el modelo predice bajada pero realmente sube).
- **VP** son los verdaderos positivos (por ejemplo, el modelo predice una subida y efectivamente sube)

A partir de esta estructura se derivan métricas que cuantifican tanto la precisión como la capacidad del modelo para detectar correctamente los cambios en la dirección del precio:

- **Exactitud (Accuracy):**
  La exactitud refleja el porcentaje total de predicciones correctas, tanto positivas como negativas, sobre el total de observaciones.

$$Exactitud = \frac{VP + VN}{\text{Total}}$$

  Útil para obtener una visión general del rendimiento del modelo cuando las clases están equilibradas, al medir el porcentaje total de aciertos sobre todas las predicciones.

- **Precisión (Precision):**
  La precisión se define como la proporción de verdaderos positivos entre el total de predicciones positivas realizadas por el modelo.

$$PrecisiQn = \frac{VP}{\text{VP} + \text{FP}}$$

  Indicada cuando el objetivo es minimizar los falsos positivos, relevante en contextos donde clasificar erróneamente un valor como positivo puede tener un alto coste.

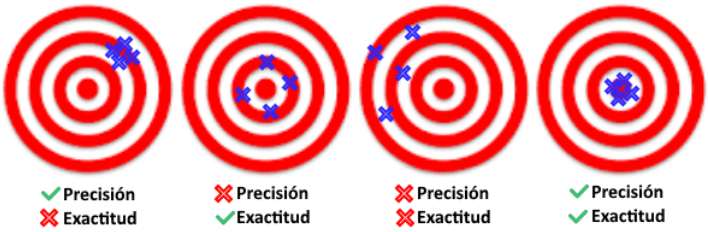


Figura 41 - Diferencias entre Exactitud y Precisión (fuente: elaboración propia).

- **Sensibilidad (Recall):**
  La sensibilidad indica la proporción de verdaderos positivos sobre el total de casos positivos reales.

$$Sensibilidad = \frac{VP}{\text{VP} + \text{FN}}$$

Esta métrica es útil cuando es importante capturar la mayor cantidad de verdaderos positivos posibles.

- **F1-Score:**
  El F1-Score es una medida que combina la precisión y sensibilidad en un solo valor, representando su media armónica.

$$F1Score = 2 \cdot \frac{PrecisiQn \cdot Sensibilidad}{PrecisiQn + Sensibilidad}$$

Esta métrica es especialmente útil cuando se busca un equilibrio entre precisión y sensibilidad, y cuando existe una distribución desigual entre clases.
Un valor alto de F1 indica que el modelo tiene tanto buena precisión como buena capacidad de detección de casos positivos reales.

- **ROC-AUC (Receiver Operating Characteristic - Area Under the Curve):**
  La curva ROC representa gráficamente la relación entre la tasa de verdaderos positivos (VP) y la tasa de falsos positivos (FP) a distintos umbrales de clasificación. El AUC (Área Bajo la Curva) mide la capacidad general del modelo para discriminar entre clases.

$$AUC = \int ROC$$

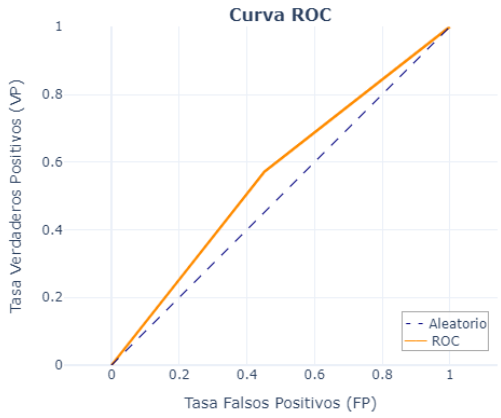


Figura 42 – Ejemplo de curva ROC (fuente: elaboración propia).

El valor del AUC oscila entre 0 y 1, donde 0.5 representa un modelo sin capacidad predictiva (equivalente al azar) y 1.0 indica una clasificación perfecta. Es una métrica robusta frente al desbalance de clases y muy utilizada para evaluar modelos de clasificación binaria.

4.11 Modelos predictivos

En este apartado se presenta la base teórica y el funcionamiento de los modelos predictivos evaluados: desde los modelos que establecen una línea base, pasando por enfoques estadísticos tradicionales, de aprendizaje automático, hasta técnicas de aprendizaje profundo. Para cada modelo se incorpora una ficha técnica que detalla los hiperparámetros explorados y los conjuntos de datos utilizados.

4.11.1 Modelos de línea base

Para establecer una referencia sólida y establecer niveles mínimos de rendimiento, se han implementado dos modelos básicos, uno orientado a evaluar métricas de clasificación y otro de métricas de regresión. A lo largo de este proyecto, el término de predictor base incluye ambos predictores: el aleatorio y el de persistencia.

- **Predictor aleatorio**:
  Se utiliza como referencia para evaluar las métricas de clasificación de la dirección del cambio del precio (baja o sube/igual). Este modelo no incorpora ninguna información sobre la dinámica del mercado, sino que basa sus predicciones únicamente en las frecuencias históricas observadas:

  - **Versión histórica completa**: Calcula, a partir de la totalidad de los datos de NVIDIA disponibles desde 2015, la probabilidad de que el precio baje o suba, basándose en la proporción histórica de días con variación negativa o positiva.

$$P_{Sube}(t+1) = 100 \cdot \frac{N\mathcal{X}mero\ de\ d\mathcal{K}as con\ subida}{Total\ de\ d\mathcal{K}as}$$

  - **Versión de 60 días**: Utiliza únicamente los datos de NVIDIA de los últimos 60 días, obteniendo la distribución de variación de precios reciente para generar la predicción. Esta versión intenta capturar dinámicas de mercado más recientes.

$$P_{Sube}(t+1) = 100 \cdot \frac{xltimos\ 60\ d\mathcal{K}as\ con\ subida}{60}$$

- **Predictor de persistencia:**
  Este modelo se utiliza como línea base en la evaluación de métricas de regresión para predecir el precio de cierre de NVIDIA. Su lógica es extremadamente simple y efectiva en horizontes de predicción muy cortos (un día): asume que la mejor predicción para el valor de mañana es el valor observado hoy.

$$y(t+1) = y(t)$$

  Aunque este modelo podría emplearse también para métricas de clasificación, su rendimiento es considerablemente inferior al predictor aleatorio como se puede observar en la *Figura 48* (Anexo 9.1).

4.11.2 Modelos basados en métodos estadísticos tradicionales

En este subapartado se describen modelos estadísticos clásicos para series temporales, dos modelos que utilizan únicamente el histórico de la variable a predecir (enfoque univariante) y uno que utiliza variables exógenas (enfoque multivariante).

- **ARIMA (AutoRegressive Integrated Moving Average)**
  ARIMA es uno de los modelos estadísticos más extendidos para el análisis y predicción de series temporales. Modela la variable de interés basándose en su propia inercia (dependencia de valores pasados) y en errores de predicción pasados. El modelo se define por tres parámetros (p, d, q):

  - **AR (p):** Componente autorregresivo, que modela la dependencia entre una observación y p observaciones anteriores (retardos).

  - **I (d):** Componente de integración, que indica el número de diferenciaciones necesarias para hacer la serie estacionaria. Una serie estacionaria es una serie temporal de datos ordenados en el tiempo que cumple:
    - Sin tendencia: La serie no muestra un aumento o disminución constante a lo largo del tiempo (la media es constante).
    - Varianza constante: La dispersión de los datos se mantiene estable a lo largo del tiempo.
    - Propiedades invariantes: No depende del instante de tiempo, es decir, sus características estadísticas (como la media, varianza y autocorrelación) se mantienen iguales, lo que permite que los patrones se repitan de forma similar en distintos períodos.

  - **MA (q):** Componente de media móvil, que modela la dependencia entre una observación y q errores residuales de predicciones anteriores.

$$\left(1 - \sum_{i=1}^{p} \boldsymbol{E}_i \cdot L^i\right) \cdot (1 - L)^d \cdot y\,(t) = c + \left(1 + \sum_{j=1}^{q} \boldsymbol{\theta}_j \cdot L^j\right) \cdot \boldsymbol{\varepsilon_t}$$

  Donde:
  - ***L*:** operador de retardo, de modo que $L^i\, y(t) = y(t-i)$.
  - ***p*:** orden del componente autorregresivo (AR), indica el número de retardos incluidos.
    - $\mathbf{E_i}$**:** coeficientes autorregresivos que miden la influencia de los valores pasados en y(t)
  - ***d*:** orden de integración (I), que determina el número de diferencias requeridas para lograr la estacionariedad.

- **q:** orden del componente de media móvil (MA), indica el número de errores pasados que se incluyen.
  - **θ_j:** coeficientes de media móvil que cuantifican el impacto de los errores pasados en y(t).
- **c:** término constante o intercepto.
  - **ε_t:** error o residuo en el tiempo t, considerado como ruido blanco.

Este modelo solo utiliza un dato directo, **Close**, el mismo que intenta predecir.

Modelo ARIMA	
Código fuente (GitHub)	"Notebook/Predictor ARIMA.ipynb"
Valores faltantes	✓ Eliminados ☐ "Forward Fill" ☐ Media
Distribución	✓ Sin transformación ✓ Logaritmo ✓ Retorno log ✓ Yeo-Johnson
Normalización	✓ Sin normalización ☐ Min-Max ☐ Standard
Variables exógenas	✓ No ☐ Directos ☐ I. técnicos ☐ Big Tech ☐ I. bursátiles ☐ I. económicos ☐ Análisis de sentimiento
Validación cruzada	✓ No ☐ Sí (n = 1)
Configuración especial	Solo usa **CPU**. Entrenamiento en backtesting: **cada día**. **auto_arima** con: method=' lbfgs' information_criterion='AIC' m=1 seasonal=False stepwise=False test='kpss' p=[0,3] d=[0,1] q=[0,3]
Número de tests	Combinaciones de hiperparámetros: 128 (4 x 32) Número de tests: 8 (4 x 2) Tiempo de entrenamiento y evaluación: ~5 minuto

Tabla 12 - Ficha del modelo ARIMA.

- **Prophet**
  Prophet es un modelo de pronóstico de series temporales desarrollado por Facebook. Está diseñado para manejar características comunes en series temporales de negocios, como múltiples estacionalidades (diaria, semanal, anual), efectos de días festivos y robustez frente a datos faltantes y atípicos.

$$y(t) = g(t) + s(t) + h(t) + \varepsilon(t)$$

  Donde:
  - ***g(t)***: componente de tendencia (lineal por tramos o logística).
  - ***s(t)***: componente de estacionalidad (modelada con series de Fourier para adaptarse a patrones periódicos).
  - ***h(t)***: componente de efectos de días festivos o eventos específicos.
  - ***ε(t)***: término de error, asumido como ruido blanco Gaussiano.

Este modelo solo utiliza un dato directo, **Close**, el mismo que intenta predecir.

Modelo Prophet	
Código fuente (GitHub)	"Notebook/Predictor Prophet.ipynb"
Valores faltantes	✓ Eliminados ☐ "Forward Fill" ☐ Media
Distribución	✓ Sin transformación ✓ Logaritmo ✓ Retorno log ✓ Yeo-Johnson
Normalización	✓ Sin normalización ☐ Min-Max ☐ Standard
Variables exógenas	✓ No ☐ Directos ☐ I. técnicos ☐ Big Tech ☐ I. bursátiles ☐ I. económicos ☐ Análisis de sentimiento
Validación cruzada	✓ No ☐ Sí (n = 1)
Configuración especial	Solo usa **CPU**. Entrenamiento en backtesting: **cada día**. **Grid search** con: changepoint_prior_scale=[0.1, 0.3, 0.5, 0.8, 1.0] seasonality_prior_scale=[0.1, 0.5, 1.0, 5.0, 10.0] holidays_prior_scale=[0.1, 0.5, 1.0, 5.0, 10.0] seasonality_mode=['additive', 'multiplicative'] n_changepoints=[15, 35, 50] changepoint_range=[0.8, 0.9, 0.95] interval_width=[0.8, 0.9, 0.95]
Número de tests	Combinaciones de hiperparámetros: 27000 (4 x 6750) Número de tests: 8 (4 x 2) Tiempo de entrenamiento y evaluación: ~4 horas (paralelizado x4)

Tabla 13 - Ficha del modelo Prophet.

- **SARIMAX (ARIMA estacional con variables exógenas)**
  El modelo ARIMAX extiende al ARIMA básico añadiendo dos elementos extra:

  - **Variables exógenas (X):** además de la inercia de la serie, permite incorporar una o más series externas X_t que influyen directamente en la variable objetivo. De este modo las predicciones ya no dependen solo de $y(t-i)$ y de errores pasados, sino también de información contextual.

  - **Componente estacional (SARIMAX):** para capturar patrones que se repiten cada s periodos (meses, semanas, etc.) se añade un segundo conjunto AR-I-MA con parámetros $(P\ ,D\ ,Q)_s$, donde:
    - *P* y *Q* son los órdenes AR y MA estacionales.
    - D es el número de diferencias estacionales.

$$\left(1-\sum_{i=1}^{p}\boldsymbol{E}_i\cdot L^i\right)\cdot\left(1-\sum_{k=1}^{P}\Phi_k\cdot L^{ks}\right)\cdot(1-L)^d\cdot\left(1-L^s\right)^D\cdot y\left(t\right)=$$
$$c+\left(1+\sum_{j=1}^{q}\boldsymbol{\theta}_j\cdot L^j\right)\cdot\left(1+\sum_{m=1}^{Q}\boldsymbol{\Theta}_m\cdot L^{ms}\right)\cdot\boldsymbol{\varepsilon_t}+\sum_{n=1}^{r}\beta_n\cdot X_n\left(t\right)$$

  Donde:
  - $\boldsymbol{\Phi_k}$: coeficientes autorregresivos estacionales (para retardos $k\cdot s$).
  - $\boldsymbol{D}$: orden de diferenciación estacional (número de diferencias estacionales).
  - **s**: es la longitud del ciclo (12 para datos mensuales con estacionalidad anual, 7 para diarios con ciclo semanal, etc).
  - $\boldsymbol{\Theta}_m$: coeficientes de media móvil estacionales (para errores en $m\cdot s$).
  - $\boldsymbol{\beta_n}$: coeficiente de regresión para la n-ésima variable exógena que mide el impacto de dicha variable en *y(t)*.
  - $\boldsymbol{X_n(t)}$: el valor de la n-ésima variable exógena en el tiempo t.

Este modelo utiliza las 63 combinaciones posibles de los conjuntos de datos disponibles. Para los tests de backtesting, el modelo se ha entrenado cada 5 días (simulando un periodo semanal), esto acelera 5 veces este test y consigue que sea más realista ya que en producción no se suele entrenar el modelo diariamente. Como se aprecia en la *Figura 51*, el rendimiento cae un ~6% al entrenar cada 5 días y un ~17% si se entrena cada 10 días con respecto al diario.

Modelo SARIMAX	
Código fuente (GitHub)	"Notebook/Predictor SARIMAX.ipynb"
Valores faltantes	✓ Eliminados ☐ "Forward Fill" ☐ Media
Distribución	✓ Sin transformación ✓ Logaritmo ✓ Retorno log ✓ Yeo-Johnson
Normalización	✓ Sin normalización ✓ Min-Max ✓ Standard
Variables exógenas	☐ No ✓ Directos ✓ I. técnicos ✓ Big Tech ✓ I. bursátiles ✓ I. económicos ✓ Análisis de sentimiento
Validación cruzada	✓ No ☐ Sí (n = 1)
Configuración especial	Solo usa **CPU**. Entrenamiento en backtesting: **cada 5 días**. Winsorization: 0.005 **auto_arima** con: method=' lbfgs' information_criterion='AIC' m=5 seasonal=True stepwise=True p=[0,3] d=[0,2] q=[0,3] P=[0,2] D=[0,1] Q=[0,2]
Número de tests	Combinaciones de hiperparámetros: 653184 (4 x 3 x 63 x 864) Número de tests: 1512 (4 x 3 x 63 x 2) Tiempo de entrenamiento y evaluación: ~60 horas

Tabla 14 - Ficha del modelo SARIMAX.

4.11.3 Modelos de aprendizaje automático

En este subapartado se describen dos algoritmos de aprendizaje automático supervisado utilizados en tareas de regresión con series temporales: Random Forest y XGBoost.

A diferencia de los modelos estadísticos clásicos, estos métodos permiten capturar relaciones no lineales, aprovechar la multivarianza de los datos y modelar dependencias complejas en el tiempo.

Random Forest y XGBoost son modelos basados en árboles de decisión, robustos y eficientes, especialmente útiles cuando se dispone de múltiples variables explicativas.

- **Random Forest (Bosque aleatorio)**
  Random Forest es un modelo de conjunto (ensemble) basado en el método de bagging (bootstrap aggregating), que consiste en construir múltiples árboles de regresión entrenados sobre distintas muestras aleatorias del conjunto de datos. Cada árbol se entrena de forma independiente, utilizando además un subconjunto aleatorio de características en cada nodo de decisión, lo que introduce diversidad y mejora la capacidad de generalización del modelo.

  La predicción final del modelo se obtiene como el promedio de las predicciones individuales de todos los árboles:

$$\hat{y} = \frac{1}{B}\sum_{b=1}^{B} T_b(x)$$

  Donde:
  - B: número total de árboles entrenados.
    $T_b(x)$: predicción del árbol b-ésimo para la entrada x.

  Las principales ventajas de Random Forest son su robustez frente al sobreajuste, su capacidad para capturar relaciones no lineales y el hecho de que requiere poca preparación previa de los datos, como transformación o normalización. Este modelo utiliza las 63 combinaciones posibles de los conjuntos de datos disponibles.

Modelo Random Forest	
Código fuente (GitHub)	"Notebook/Predictor Random Forest.ipynb"
Valores faltantes	✓ Eliminados ☐ "Forward Fill" ☐ Media
Distribución	✓ Sin transformación ☐ Logaritmo ☐ Retorno log ☐ Yeo-Johnson
Normalización	✓ Sin normalización ☐ Min-Max ☐ Standard
Variables exógenas	☐ No ✓ Directos ✓ I. técnicos ✓ Big Tech ✓ I. bursátiles ✓ I. económicos ✓ Análisis de sentimiento
Validación cruzada	✓ No ☐ Sí (n = 1)
Configuración especial	Solo usa **CPU**. Entrenamiento en backtesting: **cada día**. **Grid search** con: n_estimators=[50, 100, 150, 200] max_depth=[10, 15, 20, None] min_samples_split=[2, 5, 10] min_samples_leaf=[1, 2, 4] max_features=['sqrt', 'log2', None] Bootstrap=[True, False]
Número de tests	Combinaciones de hiperparámetros: 54432 (1 x 1 x 63 x 864) Número de tests: 126 (1 x 1 x 63 x 2) Tiempo de entrenamiento y evaluación: ~2 horas (paralelizado x3)

Tabla 15 - Ficha del modelo Random Forest.

- **XGBoost (Extreme Gradient Boosting)**
  XGBoost es una implementación optimizada del algoritmo de boosting por gradiente. A diferencia del bagging, el boosting entrena los árboles de forma secuencial: cada nuevo árbol intenta corregir los errores de predicción cometidos por los árboles anteriores. Este enfoque permite construir modelos altamente precisos y eficientes.

  Cada iteración minimiza una función de pérdida regularizada que controla tanto el error de predicción como la complejidad del modelo:

$$L^t = \sum_{i=1}^{n} l\left(y_i,\ \hat{y}_i^{(t-1)} + f_t(x_i)\right) + \Omega(f_t)$$

  Donde:
  - l: función de pérdida.
  - $\hat{y}_i^{(t-1)}$: predicción acumulada hasta la iteración (t-1).

o f_t: nuevo árbol añadido en la iteración t.
$\Omega(f_t)$: término de regularización que penaliza la complejidad del modelo.

XGBoost es especialmente eficaz en contextos donde se busca precisión elevada, regularización automática y compatibilidad con datos faltantes. Como Random Forest, requiere poca preparación previa de los datos. Este modelo utiliza las 63 combinaciones posibles de los conjuntos de datos disponibles.

Modelo XGBoost	
Código fuente (GitHub)	"Notebook/Predictor XGBoost.ipynb"
Valores faltantes	✓ Eliminados ☐ "Forward Fill" ☐ Media
Distribución	✓ Sin transformación ☐ Logaritmo ☐ Retorno log ☐ Yeo-Johnson
Normalización	✓ Sin normalización ☐ Min-Max ☐ Standard
Variables exógenas	☐ No ✓ Directos ✓ I. técnicos ✓ Big Tech ✓ I. bursátiles ✓ I. económicos ✓ Análisis de sentimiento
Validación cruzada	✓ No ☐ Sí (n = 1)
Configuración especial	Puede usar **CPU** o **GPU**. Entrenamiento en backtesting: **cada día**. **Grid search** con: n_estimators=[100, 150, 200] max_depth=[4, 5, 6] learning_rate=[0.05, 0.1, 0.2] subsample=[0.8, 1.0] colsample_bytree=[0.8, 0.9, 1.0] reg_alpha=[0, 0.1] reg_lambda=[1, 1.5]
Número de tests	Combinaciones de hiperparámetros: 40824 (1 x 1 x 63 x 648) Número de tests: 126 (1 x 1 x 63 x 2) Tiempo de entrenamiento y evaluación: ~1 hora

Tabla 16 - Ficha del modelo XGBoost.

4.11.4 Modelos de aprendizaje profundo

Dentro del ámbito del aprendizaje profundo se han considerado dos arquitecturas representativas aplicadas a la predicción de series temporales. Por un lado, LSTM (Long Short-Term Memory), un tipo de red neuronal recurrente capaz de aprender dependencias de largo plazo y manejar secuencias con persistencias o retardos significativos. Por otro, el modelo Transformer, basado en el mecanismo de autoatención, que permite ponderar de manera simultánea la influencia de todas las observaciones de una secuencia y capturar relaciones complejas y de largo alcance de forma más eficiente y paralelizable que los enfoques recurrentes.

- **LSTM**

  LSTM es un tipo especializado de red neuronal recurrente (RNN) diseñado para modelar secuencias de datos con dependencias a largo plazo. A diferencia de las RNN tradicionales, las celdas LSTM integran mecanismos de control que permiten preservar o descartar información a lo largo del tiempo, evitando así los problemas de desvanecimiento o explosión del gradiente durante el entrenamiento.

  Una celda LSTM se compone de tres compuertas principales:

  - Compuerta de olvido $\boldsymbol{f_t}$: decide qué información debe descartarse del estado anterior.
  - Compuerta de entrada $\boldsymbol{i_t}$: determina qué nueva información se añade al estado de la celda.
  - Compuerta de salida $\boldsymbol{o_t}$: regula qué parte del estado interno se propaga hacia la salida.

  En la siguiente figura, se muestra la arquitectura completa de un modelo LSTM con sus ecuaciones fundamentales:

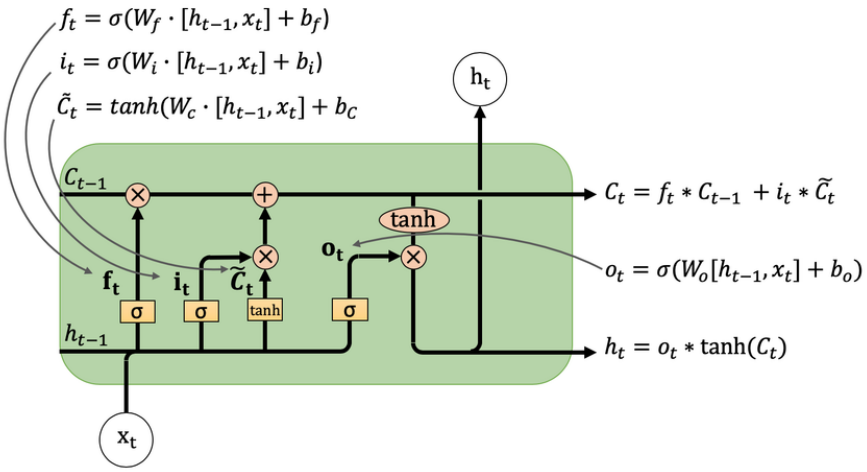


Figura 43 - Arquitectura de una red Long Short-Term Memory Networks (fuente: ResearchGate [44]).

Donde:

- x_t: vector de entrada en tiempo t.
- h_t: salida oculta actual de la celda.
- C_t: estado actualizado de la celda en el tiempo t.
- $\tilde{C}_t$: nuevo candidato de contenido para la celda.

Las redes neuronales son especialmente sensibles a la escala y distribución de los datos, por lo que se requiere un preprocesamiento adecuado. Se han seleccionado dos transformaciones, descartando el uso de retornos logarítmicos. Para el escalado, se ha optado por StandardScaler, dada su mayor robustez frente a valores atípicos y su idoneidad para este tipo de modelos. La arquitectura de la red se ha configurado específicamente para el tamaño del conjunto de datos disponible (≈2.500 registros), evitando arquitecturas complejas como capas bidireccionales, más de dos capas ocultas o un número elevado de neuronas, con el objetivo de reducir el riesgo de sobreajuste. Este modelo utiliza las 63 combinaciones posibles de conjuntos de datos disponibles.

Modelo LSTM	
Código fuente (GitHub)	"Notebook/Predictor LSTM.ipynb"
Valores faltantes	✓ Eliminados ☐ "Forward Fill" ☐ Media
Distribución	☐ Sin transformación ✓ Logaritmo ☐ Retorno log ✓ Yeo-Johnson
Normalización	☐ Sin normalización ☐ Min-Max ✓ Standard
Variables exógenas	☐ No ✓ Directos ✓ I. técnicos ✓ Big Tech ✓ I. bursátiles ✓ I. económicos ✓ Análisis de sentimiento
Validación cruzada	☐ No ✓ Sí (n = 5)
Configuración especial	Puede usar **CPU** o **GPU**. Entrenamiento en backtesting: **cada 5 días**. Winsorization: 0.05 **Optune** con: sequence_length=[10, 15, 20, 25, 30, 35, 40] n_lstm_layers=[1, 2] lstm_units=[96, 128, 160, 192] dropout_rate=[0.05, 0.1, 0.15, 0.20, 0.25, 0.30] batch_size=[64, 96, 128] learning_rate=[0.0001 a 0.01]
Número de tests	Combinaciones de hiperparámetros: 6350400 (2 x 1 x 63 x 50400) Número de tests: 252 (2 x 1 x 63 x 2) Tiempo de entrenamiento y evaluación: ~42 horas

Tabla 17 - Ficha del modelo LSTM.

- **Transformer**

  El Transformer es un tipo de red neuronal introducido en el artículo *"Attention is all you need"* [10], cuyo rasgo distintivo es el uso del mecanismo de autoatención (self-attention). A diferencia de las arquitecturas recurrentes o convolucionales, evalúa de forma simultánea la relevancia de todos los elementos de una secuencia, lo que le permite capturar dependencias tanto de corto como de largo alcance de manera más eficiente y paralelizable.

  En este proyecto, se ha adaptado un modelo Transformer Encoder para la predicción de series temporales multivariantes. Esta arquitectura resulta especialmente adecuada en comparación con modelos recurrentes como LSTM, ya que facilita la identificación de relaciones complejas entre múltiples variables dentro de una ventana temporal sin necesidad de procesarlas secuencialmente.

  La arquitectura aplicada en este proyecto puede resumirse en cuatro componentes principales:

  1. **Proyección de entradas (Input Embedding):** cada paso temporal, representado por un vector de características, se proyecta a un espacio latente de dimensión *d_model*.
  2. **Bloques codificadores (Encoder Blocks):** cada bloque aplica mecanismos de autoatención multi-cabeza y redes feed-forward, junto con conexiones residuales y normalización, para refinar las representaciones.
  3. **Agregación temporal (Pooling):** las secuencias procesadas se condensan en un único vector representativo mediante un promedio global a lo largo de la dimensión temporal.
  4. **Cabezal de regresión (Prediction Head):** el vector final se pasa por una red densa con salida lineal, generando la predicción del precio de cierre.

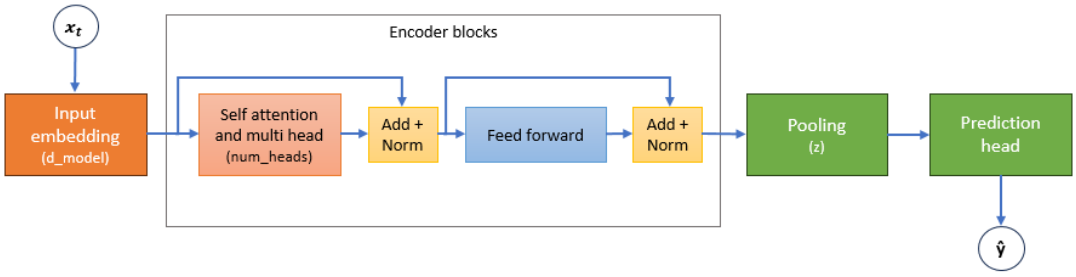


Figura 44 - Arquitectura del Transformer Enconder (fuente: elaboración propia).

Donde:

- x_t: vector de entrada en el instante temporal t, dentro de una secuencia de longitud sequence_length y compuesto por los datos de entrada.
- **d_model**: dimensión latente del espacio donde se proyectan los datos de entrada.

- **num_heads**: número de cabezas de autoatención paralelas. Cada cabeza aprende distintos patrones de dependencia temporal o entre variables.
- **z**: vector resultante tras la agregación global promedio de la secuencia.
- **ŷ**: salida del cabezal de regresión, implementado como red densa con activación lineal, que produce la predicción numérica del precio de cierre.

Al igual que en el caso de LSTM, la sensibilidad del modelo a la escala y distribución de los datos hace imprescindible un preprocesamiento adecuado. Se han aplicado dos transformaciones, descartando el uso de retornos logarítmicos. Para el escalado, se ha optado por StandardScaler, dada su mayor robustez frente a valores atípicos y su idoneidad para modelos de este tipo. La arquitectura se ha configurado considerando el tamaño del conjunto de datos disponible (≈2.500 registros), limitando la profundidad y el número de cabezas de atención con el fin de evitar el sobreajuste. Este modelo también utiliza las 63 combinaciones posibles de conjuntos de datos disponibles.

Modelo Transformer	
Código fuente (GitHub)	"Notebook/Predictor Transformer.ipynb"
Valores faltantes	✓ Eliminados ☐ "Forward Fill" ☐ Media
Distribución	☐ Sin transformación ✓ Logaritmo ☐ Retorno log ✓ Yeo-Johnson
Normalización	☐ Sin normalización ☐ Min-Max ✓ Standard
Variables exógenas	☐ No ✓ Directos ✓ I. técnicos ✓ Big Tech ✓ I. bursátiles ✓ I. económicos ✓ Análisis de sentimiento
Validación cruzada	☐ No ✓ Sí (n = 5)
Configuración especial	Puede usar **CPU** o **GPU**. Entrenamiento en backtesting: **cada 5 días**. Winsorization: 0.05 **Optune** con: sequence_length=[10, 15, 20, 25, 30] d_mode=[64, 96, 128] num_heads=[4, 8] ff_dim=[128, 256] num_transform_blocks=[1, 2] dropout_rate=[0.1, 0.15, 0.20, 0.25, 0.30, 0.35] batch_size=[64, 96, 128] learning_rate_base=[0.0001 a 0.005]
Número de tests	Combinaciones de hiperparámetros: 5443200 (2 x 1 x 63 x 43200) Número de tests: 252 (2 x 1 x 63 x 2) Tiempo de entrenamiento y evaluación: ~10 horas

Tabla 18 - Ficha del modelo Transformer.

4.12 Aplicación de simulación

Como caso de uso práctico, se ha desarrollado una aplicación web que actúa como simulador de inversión bursátil en NVIDIA. Su objetivo es proporcionar apoyo informativo a un posible inversor mediante la combinación de los tres mejores modelos predictivos seleccionados entre más de 2.200 configuraciones evaluadas:

- El mejor en rendimiento global.
- El mejor en regresión.
- El mejor en clasificación.

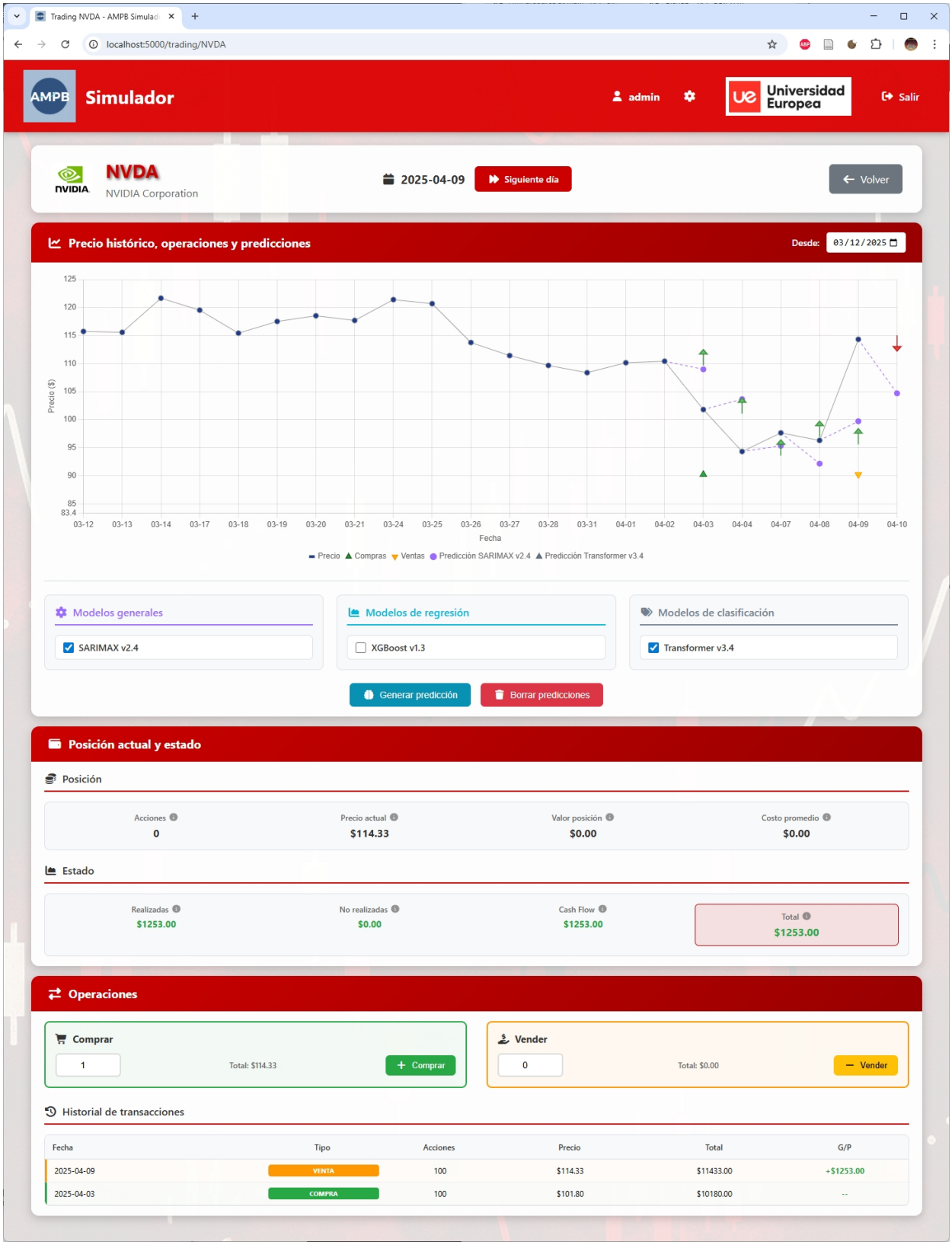
La aplicación presenta las siguientes características técnicas y funcionales:

- **Arquitectura y seguridad**: implementada en Python con Flask, desplegada en contenedores Docker y respaldada por una base de datos MongoDB. Es multiusuario, incorpora gestión de credenciales de forma cifrada, validación completa de formularios y protección frente a inyecciones mediante CSRFProtect [45]. Asimismo, incluye un sistema de roles con administración de usuarios, cambio de contraseñas y soporte SSL.

- **Panel de configuración**: permite visualizar los tickers disponibles, los modelos asociados y los conjuntos de datos empleados. Desde este panel también es posible reiniciar simulaciones, predicciones y transacciones. El usuario administrador dispone además de la capacidad de crear o eliminar cuentas de usuario.

- **Ventana principal de simulación**: muestra el día simulado, la evolución real del precio de NVIDIA y las predicciones generadas por los tres modelos destacados. Los resultados se integran en una gráfica interactiva que incorpora funcionalidades de resaltado al pasar el cursor y ventanas emergentes de información. También simula las funciones de un pequeño broker: posibilita la compra y venta de acciones ficticias, registrando cada transacción en un histórico que calcula automáticamente el número de acciones en cartera, el precio medio de adquisición y las ganancias o pérdidas acumuladas. Una captura de esta ventana de simulación se presenta en la *Captura de pantalla 1*.

La simulación cubre el periodo comprendido entre el 2 de abril de 2025 y el 23 de mayo de 2025, proporcionando un escenario controlado y realista que facilita la evaluación práctica del sistema predictivo desarrollado.



Captura de pantalla 1 - Aplicación de simulación con predicciones y compra/venta de acciones.

Capítulo 5. DISCUSIÓN

En este capítulo se presentan y discuten los principales resultados obtenidos a lo largo del proyecto. Se parte de un análisis descriptivo del conjunto de datos tras su preprocesamiento, seguido de una comparación estructurada del rendimiento de los distintos modelos evaluados. Finalmente, se reflexionará sobre las implicaciones prácticas del trabajo, sus limitaciones y los ajustes realizados respecto a los objetivos iniciales. El propósito es proporcionar una visión crítica y global del sistema desarrollado y del proceso seguido para su construcción.

5.1 Análisis descriptivo

Este apartado presenta las principales características del conjunto de datos utilizado en el proyecto, una vez preprocesado y preparado para el entrenamiento de modelos, de modo que se asegure su calidad y utilidad para la predicción bursátil.

5.1.1 Fuentes y naturaleza de los datos

Los conjuntos de datos fueron construidos a partir de fuentes abiertas y gratuitas, cubriendo el periodo 2015–2025. La fuente principal es la cotización diaria de NVIDIA, complementada con variables técnicas derivadas y factores exógenos agrupados en seis conjuntos de datos:

1. Datos directos (Open, High, Low, Volume).
2. Indicadores técnicos (SMA, EMA, ATR, Bollinger Bands, Range, MACD, RSI, etc.).
3. Otras Big Tech (Google, Amazon, Apple, Meta, Microsoft, Tesla, AMD e Intel).
4. Índices bursátiles internacionales (S&P 500, Nasdaq 500, EuroStoxx 50, etc.).
5. Indicadores económicos (CPI, GDP, petróleo, oro, tipos de interés, etc.),
6. Análisis de sentimiento (GoogleTrends, noticias de Alpha Vantage).

El total asciende a 40 variables predictoras (ver *Tabla 10*) todas de tipo numérico continuo, sin presencia de variables categóricas. La diversidad de fuentes permite modelar tanto la dinámica interna del activo como su entorno financiero, económico y emocional.

Los datos se almacenan y gestionan desde un data warehouse ligero basado en MongoDB, lo que permite una gestión eficiente y centralizada.

5.1.2 Tratamiento de valores faltantes

Durante el análisis inicial, se detectaron valores nulos en 20 de las variables (ver *Figura 10* y *Figura 12*). En el apartado, *Detección de valores faltantes y duplicados,* se detalla en profundidad los pasos realizados. En resumen, se ha utilizado la técnica de propagación hacia delante (forward fill) y reparado algunos errores de ingestión de modo que no hay ningún valor faltante (salvo las medias por motivos obvios).

5.1.3 Distribución y transformación de variables

Las variables financieras presentan distribuciones sesgadas y escalas heterogéneas. Por ello, se contemplan diferentes transformaciones (detalladas en *Distribuciones e histogramas*):

- Logarítmica.
- Retornos logarítmicos.
- Box-Cox.
- Yeo-Johnson.

5.1.4 Valores duplicados y atípicos

No se detectaron registros duplicados, lo cual es coherente al tratarse de series temporales indexadas por fecha.

En cuanto a valores atípicos, para su detección se aplicó el método del rango intercuartílico (IQR). Los valores identificados fueron revisados manualmente (ver *Análisis de valores atípicos*) y considerados válidos en el contexto financiero. Por tanto, no se eliminaron, aunque se deja abierta la posibilidad de que algunos modelos apliquen técnicas de Winsorización para mitigar su impacto.

5.1.5 Análisis de correlación

Se ha realizado una matriz de correlación entre todas las variables predictoras y la variable objetivo (Close). La mayoría muestran correlaciones relevantes, tanto positivas como negativas (ver *Figura 36*).

No obstante, cinco variables presentaron una correlación reducida con Close: RSI_14, OC_Change, Chaikin_Osc, VIX y ShanghaiComposite. A pesar de ello, se han mantenido en el conjunto de datos, ya que modelos no lineales como LSTM o XGBoost podrían extraer relaciones complejas no evidentes en la correlación lineal.

5.1.6 Normalización y escalado

Dado que los modelos empleados tienen diferentes sensibilidades a la escala de los datos, se han aplicado tres estrategias (detalladas en *Normalización y escalado de datos*):

- Sin normalización.
- MinMaxScaler.
- StandardScaler.

Cada modelo selecciona una o varias de estas técnicas que son aplicadas a todo el conjunto de datos elegido.

5.2 Rendimiento de los modelos

Este apartado presenta los resultados obtenidos por los modelos predictivos evaluados, bajo dos esquemas complementarios: una validación estática que representa condiciones ideales, y un backtesting que simula un entorno realista. Se comparan las métricas de regresión y clasificación, con el objetivo final de determinar qué modelo ofrece el mejor rendimiento comparando contra el predictor base con predicciones aleatorias y de persistencia.

5.2.1 Estrategias de evaluación

Los modelos se han evaluado mediante dos esquemas complementarios, definidos en detalle en el apartado *Evaluación de los modelos*:

- Validación estática (SV): utiliza las variables exógenas reales del día objetivo, proporcionando una estimación idealizada del rendimiento.

- Backtesting sin desfase (BT): utiliza las exógenas del día anterior salvo Open, que se iguala al Close previo. Representa un escenario realista de predicción.

Cada prueba ha generado paneles visuales y métricas estructuradas que se almacenan en ficheros específicos por cada modelo (ver *Figura 39*).

5.2.2 Métricas aplicadas

Se han evaluado las predicciones mediante dos bloques de métricas:

- Métricas de regresión:
  Se evalúa la capacidad del modelo para prever el precio de cierre del siguiente día.
  - MAE: error absoluto medio.
  - MSE: error cuadrático medio.
  - RMSE: raíz del error cuadrático medio.
  - R^2: coeficiente de determinación.

- Métricas de clasificación:
  Se evalúa la capacidad del modelo para anticipar correctamente la dirección del movimiento del precio (si sube o baja respecto al día anterior), a través de la matriz de confusión y las siguientes métricas derivadas:
  - Exactitud (Accuracy).
  - Precisión (Precision).
  - Sensibilidad (Recall).
  - F1-Score.
  - AUC-ROC.

5.2.3 Índices comparativos: IRB y CGMM

Para comparar modelos se han utilizado dos índices que combinan métricas de regresión y clasificación con un peso ponderado específico (55% clasificación, 45% regresión), tal y como se detalla en el apartado *Evaluación de los modelos*:

- Índice Relativo al Baseline (IRB):
  Se calcula durante el entrenamiento de cada modelo con todas sus combinaciones para compararse contra el predictor base.

- Comparación Global por MinMax (CGMM):
  Se escogen los cuatro mejores IRB por modelo y se normalizan sus métricas para hacerlas comparables entre sí.

5.2.4 Presentación de resultados

A continuación, se presentan dos tablas resumen (*Tabla 19 y Tabla 20*) con los modelos evaluados y sus resultados agregados según CGMM, mostrando los cuatro mejores candidatos por cada modelo. Se omiten los detalles individuales, que pueden consultarse en los directorios y ficheros correspondientes. Para visualizar los resultados de forma gráfica e intuitiva, también se han generado dos gráficas (*Figura 46 y Figura 47*.) basadas en las dos tablas resumen mostrando la puntuación CGMM dividida en componente de regresión y componente de clasificación.

La *Figura 45* muestra la proporción con la que cada conjunto de datos ha sido utilizado en las configuraciones de mejor rendimiento bajo backtesting.

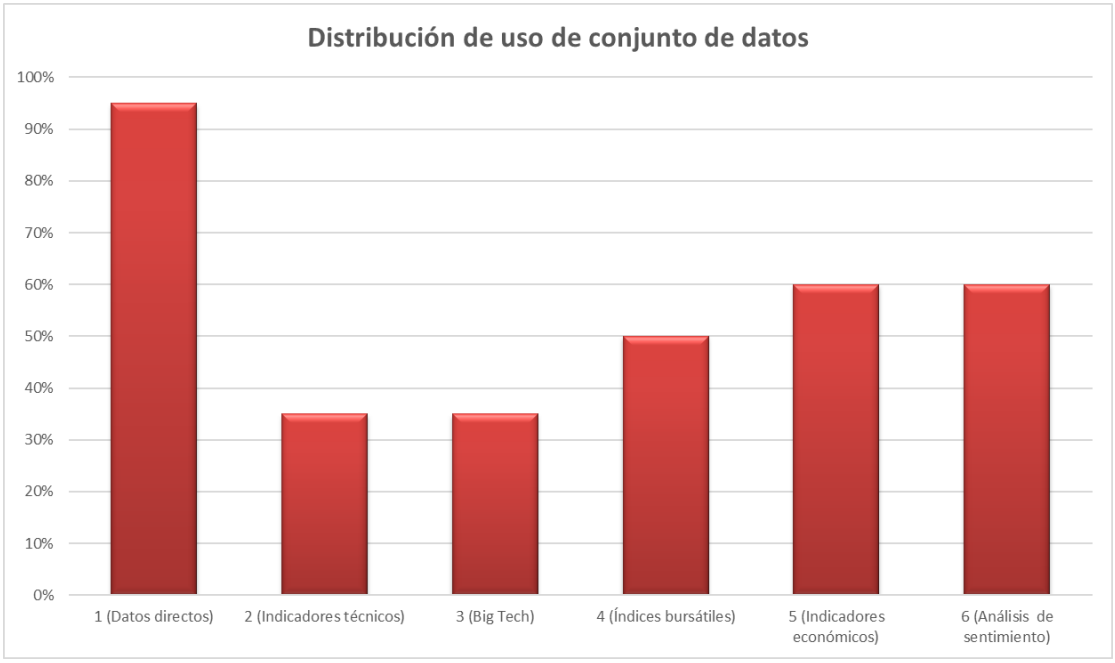


Figura 45 - Distribución de uso de conjuntos de datos exógenos en los modelos con mejor rendimiento (backtesting).

Modelo	Configuración	Métricas de regresión			Métricas de clasificación			Rendimiento CGMM		
		R2	MAE	RMSE	Accuracy	F1-Score	ROC-AUC	Regr	Clas	Total
Base	Persistencia	80,11%	3,29	4,57				0,937	0,526	0,711
	Aleatorio (60 dias)				51,67%	0,55	0,51			
ARIMA v2.3	(3, 1, 2)	-55,00%	11,05	12,78	61,02%	0,61	0,61	0,764	0,610	**0,679**
	(1, 1, 0) (Log)	-35,00%	10,15	11,92	50,58%	0,65	0,50	0,784	0,553	**0,657**
	(1, 1, 0) (YeoJohnson)	-36,00%	10,20	11,96	50,85%	0,65	0,50	0,783	0,554	**0,657**
	(2, 0, 0) (RetLog)	-179,00%	14,37	17,15	50,00%	0,67	0,48	0,667	0,550	**0,603**
Prophet v2.3	(1.0,0.5,0.1,50,0.9,0.8)	-1096,48%	34,14	35,48	55,93%	0,64	0,56	0,115	0,585	0,373
	(0.3,10.0,0.1,50,0.9,0.8) (Log)	-1260,50%	36,25	37,83	50,85%	0,60	0,50	0,038	0,539	0,313
	(0.8,10.0,0.1,50,0.95,0.8) (RetLog)	-1224,63%	26,58	37,33	44,83%	0,38	0,45	0,138	0,428	0,298
	(1.0,5.0,0.1,50,0.9,0.8) (YeoJohnson)	-1341,88%	37,26	38,94	47,46%	0,56	0,47	0,000	0,503	0,277
SARIMAX v2.4	(3, 0, 0)(0, 0, 0, 5) (Log) + 12456	99,19%	0,36	0,93	98,31%	0,98	0,98	1,000	0,983	**0,991**
	(3, 0, 0)(0, 0, 0, 5) (Log) + 1245	99,18%	0,37	0,93	98,31%	0,98	0,98	1,000	0,983	**0,991**
	(3, 0, 0)(0, 0, 0, 5) (Log) + 1256	99,18%	0,37	0,93	98,31%	0,98	0,98	1,000	0,983	**0,991**
	(3, 0, 0)(0, 0, 0, 5) (Log) + 125	99,18%	0,37	0,93	98,31%	0,98	0,98	1,000	0,983	**0,991**
Random Forest v1.3	(50,None,2,1,sqrt) + 14	94,65%	1,84	2,37	81,36%	0,83	0,81	0,973	0,817	0,887
	(50,None,2,1,sqrt) + 1	94,57%	1,61	2,39	72,88%	0,74	0,73	0,975	0,733	0,842
	(50,20,2,1,None) + 1245	91,48%	2,18	2,99	81,36%	0,82	0,81	0,964	0,816	0,882
	(50,None,2,1,sqrt) + 146	93,81%	1,96	2,55	74,58%	0,75	0,75	0,970	0,748	0,848
XGBoost v1.3	(200,6,0.2,1.0,0.9,0.1,1) + 1256	95,62%	1,49	2,15	98,31%	0,98	0,98	0,978	0,983	**0,981**
	(200,6,0.2,1.0,0.9,0,1) + 12	95,97%	1,46	2,06	93,22%	0,93	0,93	0,979	0,933	**0,954**
	(200,6,0.2,1.0,0.9,0,1) + 125	95,68%	1,52	2,13	93,22%	0,93	0,93	0,978	0,933	**0,953**
	(200,6,0.2,1.0,1.0,0,1) + 126	94,95%	1,58	2,31	91,53%	0,92	0,91	0,976	0,916	**0,943**
LSTM v3.4	(30,[192],0.10,64,0.00041) (Log) (Standard) + 234	75,23%	4,17	5,10	74,58%	0,76	0,74	0,923	0,751	0,828
	(20,[192],0.05,128,0.00402) (Log) (Standard) + 123456	57,13%	5,44	6,72	83,05%	0,85	0,83	0,894	0,836	0,862
	(25,[128],0.15,64,0.00082) (YeoJohnson) (Standard) + 12356	71,32%	4,12	5,49	74,58%	0,77	0,74	0,920	0,753	0,828
	(25,[128],0.15,64,0.00082) (YeoJohnson) (Standard) + 126	70,32%	4,02	5,59	74,58%	0,76	0,74	0,919	0,751	0,827
Transformers v3.4	(15,64,4,2,0.30,128,0.00023) (YeoJohnson) (Standard) + 1245	53,79%	5,49	6,97	66,10%	0,66	0,66	0,890	0,659	**0,763**
	(10,128,4,1,0.15,128,0.00075) (YeoJohnson) (Standard) + 26	31,23%	6,70	8,51	57,63%	0,58	0,58	0,861	0,576	**0,704**
	(15,96,4,2,0.25,96,0.00053) (YeoJohnson) (Standard) + 46	23,95%	7,77	8,94	59,32%	0,61	0,59	0,845	0,600	**0,710**
	(15,64,4,2,0.30,128,0.00023) (YeoJohnson) (Standard) + 256	21,24%	7,24	9,10	59,32%	0,61	0,59	0,848	0,600	**0,711**

Tabla 19 - Resultados de los mejores modelos en validación estática.

Modelo	Configuración	Métricas de regresión			Métricas de clasificación			Rendimiento CGMM		
		R2	MAE	RMSE	Accuracy	F1-Score	ROC-AUC	Regr	Clas	Total
Base	Persistencia	80,11%	3,29	4,57				0,980	0,526	0,730
	Aleatorio (60 dias)				51,67%	0,55	0,51			
ARIMA v2.3	(2, 0, 0) (RetLog)	81,38%	3,15	4,43	43,10%	0,46	0,43	0,992	0,440	**0,689**
	(1, 1, 0) (YeoJohnson)	80,38%	3,23	4,54	40,68%	0,44	0,41	0,984	0,419	**0,673**
	(1, 1, 0) (Log)	80,36%	3,23	4,55	40,68%	0,44	0,41	0,983	0,419	**0,673**
	(3, 1, 2)	80,46%	3,32	4,53	40,68%	0,43	0,41	0,981	0,413	**0,669**
Prophet v2.3	(0.8,10.0,0.1,50,0.95,0.8) (RetLog)	71,91%	4,02	5,44	44,83%	0,47	0,45	0,907	0,454	**0,658**
	(1.0,0.5,0.1,50,0.9,0.8)	-20,41%	9,78	11,25	50,85%	0,52	0,51	0,313	0,514	**0,423**
	(1.0,5.0,0.1,50,0.9,0.8) (YeoJohnson)	-85,40%	12,27	13,97	50,85%	0,45	0,51	0,000	0,491	**0,270**
	(0.3,10.0,0.1,50,0.9,0.8) (Log)	-65,73%	11,59	13,20	45,76%	0,43	0,46	0,090	0,448	**0,287**
SARIMAX v2.4	(0, 1, 1)(1, 0, 0, 5) (Log) + 14	80,55%	3,44	4,52	61,02%	0,62	0,61	0,977	0,614	**0,778**
	(1, 0, 0)(0, 0, 1, 5) (Log) (MinMax) + 1456	79,77%	3,57	4,61	61,02%	0,62	0,61	0,968	0,614	**0,773**
	(0, 0, 0)(0, 0, 1, 5) (Log) (MinMax) + 145	79,65%	3,61	4,63	61,02%	0,62	0,61	0,966	0,614	**0,772**
	(0, 0, 3)(0, 0, 0, 5) (Log) (Standard) + 1456	79,33%	3,62	4,66	61,02%	0,62	0,61	0,963	0,614	**0,771**
Random Forest v1.3	(50,None,2,1,sqrt) + 1	80,75%	3,24	4,50	44,07%	0,42	0,44	0,985	0,434	**0,682**
	(50,None,2,1,sqrt) + 126	80,25%	3,31	4,56	42,37%	0,45	0,42	0,980	0,433	**0,679**
	(50,None,2,1,sqrt) + 15	80,57%	3,22	4,52	40,68%	0,44	0,41	0,985	0,419	**0,674**
	(50,None,2,1,sqrt) + 14	80,51%	3,24	4,53	40,68%	0,44	0,41	0,984	0,419	**0,673**
XGBoost v1.3	(200,6,0.2,0.8,1.0,0,1) + 1	80,73%	3,23	4,50	50,85%	0,51	0,51	0,986	0,509	**0,723**
	(200,6,0.2,0.8,1.0,0,1) + 13	81,23%	3,15	4,44	45,76%	0,48	0,46	0,991	0,466	**0,703**
	(200,6,0.2,0.8,1.0,0,1) + 16	82,12%	3,07	4,34	42,37%	0,45	0,42	1,000	0,433	**0,688**
	(200,6,0.2,0.8,1.0,0,1) + 136	81,26%	3,19	4,44	42,37%	0,43	0,42	0,990	0,427	**0,680**
LSTM v3.4	(25,[128],0.15,64,0.00082) (YeoJohnson) (Standard) + 12356	68,02%	4,35	5,80	54,24%	0,58	0,54	0,875	0,556	**0,699**
	(20,[192],0.05,128,0.00402) (Log) (Standard) + 156	57,40%	5,34	6,69	55,93%	0,61	0,56	0,787	0,574	**0,670**
	(20,[192],0.05,128,0.00402) (Log) (Standard) + 123456	70,02%	4,47	5,62	49,15%	0,52	0,49	0,881	0,499	**0,671**
	(25,[128],0.15,64,0.00082) (YeoJohnson) (Standard) + 56	48,05%	6,11	7,39	57,63%	0,62	0,57	0,716	0,589	**0,646**
Transformers v3.4	(10,128,4,1,0.15,128,0.00075) (YeoJohnson) (Standard) + 123456	28,28%	6,85	8,69	67,80%	0,69	0,68	0,605	0,681	**0,647**
	(10,128,8,1,0.15,64,0.00019) (Log) (Standard) + 123456	28,57%	6,47	8,67	66,10%	0,67	0,66	0,620	0,663	**0,644**
	(10,128,4,1,0.15,128,0.00075) (Log) (Standard) + 12456	37,94%	6,08	8,08	57,63%	0,59	0,58	0,673	0,581	**0,622**
	(10,128,4,1,0.15,128,0.00075) (Log) (Standard) + 12345	10,86%	7,46	9,68	62,71%	0,65	0,63	0,514	0,633	**0,579**

Tabla 20 - Resultados de los mejores modelos en backtesting.

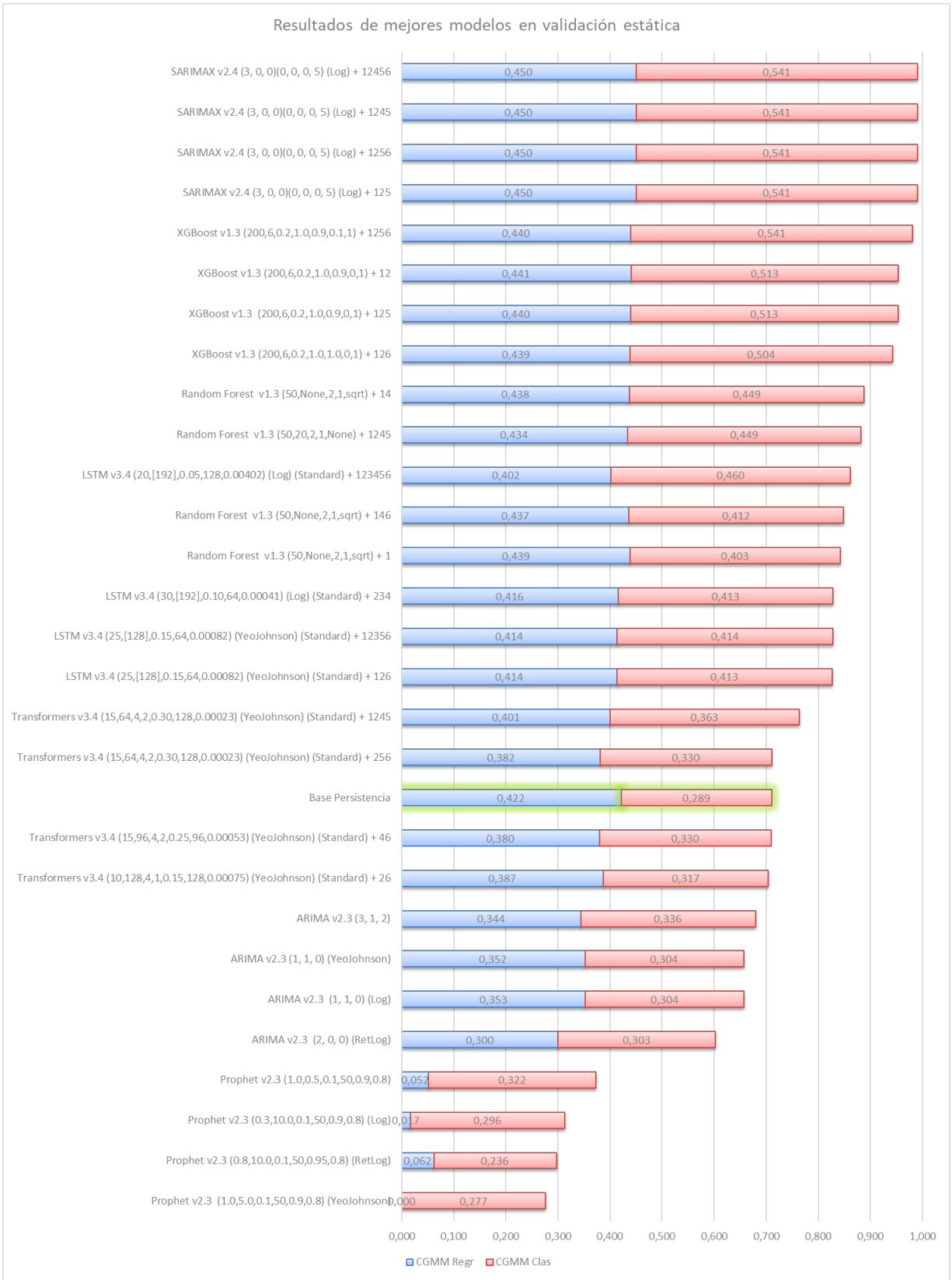


Figura 46 - Resultados de mejores modelos en validación estática.

Figura 47 - Resultados de mejores modelos en backtesting.

5.3 Discusión de los resultados

Este apartado ofrece una reflexión crítica sobre los resultados obtenidos, los retos metodológicos enfrentados y las decisiones clave que han condicionado el rendimiento de los modelos. También se revisan las implicaciones prácticas y las limitaciones tanto del enfoque como del entorno tecnológico.

5.3.1 Interpretación general de los resultados

En condiciones perfectas y no reales (*Figura 46*) de **validación estática**, prácticamente todos los modelos que utilizan variables exógenas se comportan de un modo excelente superando hasta en un 40% al predictor base. El mejor modelo es un SARIMAX que obtiene un R^2 de 0,9919, una exactitud del 98,31% y utiliza los conjuntos de datos 1, 2, 4, 5 y 6 (*Figura 52*). El modelo XGBoost sigue muy de cerca de SARIMAX, después aparecen el resto de modelos que aun superan al predictor base y, finalmente, los modelos que no utilizan variables exógenas como ARIMA y Prophet y algunos de los Transformers, logran peores resultados. Se observa que SARIMAX prefiere utilizar una transformación logarítmica y sin ningún tipo de normalización, sus conjuntos de datos preferidos son el 1, 2 y 5, alternando entre el 4 y el 6 pero sin utilizar el 3.

A continuación, centrando el análisis en las condiciones reales de **backtesting** y como se observa en la *Figura 47*, de nuevo el modelo SARIMAX ofrece el mejor resultado. Sus mejores cuatro versiones superan al predictor base: en regresión quedan prácticamente empatados, pero en clasificación destacan ofreciendo una mejora del 17%. En este caso, el mejor modelo obtiene un R^2 de 0,8055, una exactitud del 61,02% y utiliza los conjuntos de datos 1 y 4 (*Figura 53*). Se observa que SARIMAX prefiere utilizar una transformación logarítmica y, en este caso, hace uso en un modelo de normalización Standard, en dos MinMax y en el cuarto ninguna normalización. Su preferencia por los conjuntos de datos son el 1 y 4, alternando entre el 5 y el 6 pero sin utilizar ni el 2 ni el 3. En estas condiciones, solo los modelos SARIMAX son capaces de superar al predictor base, le siguen de cerca XGBoost, LSTM e incluso ARIMA.

Se da el hecho de que el mejor modelo general no es el mejor en regresión ni en clasificación, estos puestos los ostentan:

- Regresión: XGBoost con los conjuntos de datos 1 y 6 y un R^2 de 0,8212, un 2% mejor que el predictor base *(Figura 54*).

- Clasificación: Transformer con todos los conjuntos de datos y una exactitud del 67,8%, un 30% mejor que el predictor base (*Figura 55*).

Puntos interesantes a destacar:

- Como era de esperar, los modelos sin variables exógenas (solo con el dato Close a predecir) muestran los peores resultados. No obstante, se requiere una explicación a la razón por la que ARIMA y Prophet funcionan mejor en backtesting que en validación estática, la razón es que, en validación estática, estos modelos sin ninguna otra variable

deben predecir los siguientes 60 días por lo que el error se intensifica, sin embargo, en backtesting, el modelo se entrena cada día y tiene acceso siempre al precio de cierre anterior por lo que no tiene que extrapolar más de un día.

- Los modelos basados en redes neuronales (LSTM y Transformer) no consiguen igualar el rendimiento general que se obtiene con SARIMAX, XGBoost, Random Forest e, incluso algunas versiones de ARIMA. Este hecho ha sido investigado durante la ejecución del proyecto y está relacionado con la "maldición de la dimensionalidad"[46] [47] ya descrito por R. Bellman en 1957: debido a que solo se disponen de unos 2500 registros de datos con hasta 40 variables, hay muy pocos registros para que una red neuronal sea capaz de generalizar bien. Por otro lado, se podría pensar que las redes neuronales podrían haber funcionado mejor con conjuntos de datos más reducidos (menos variables) pero en este caso estamos perdiendo información relevante como ha demostrado SARIMAX: los conjuntos de datos 1, 4, 5 y 6 aportan información valiosa.

- Debido al bajo número de registros disponibles, los modelos basados en redes neuronales (LSTM y Transformer) han mostrado un mejor rendimiento en configuraciones más simples. Durante las pruebas, las arquitecturas más complejas tendían a sobre ajustar, lo que impedía una adecuada capacidad de generalización.

Por último, la proporción de los conjuntos de datos exógenos más utilizados en los mejores modelos de backtesting se muestran en la *Figura 45*, donde se puede observar que los más importantes son los datos directos (95%), seguidos de los datos de análisis de sentimiento e indicadores económicos (60%), índices bursátiles (50%) y en último lugar, los indicadores técnicos y las Big Tech (35%).

5.3.2 Implicaciones prácticas

El sistema desarrollado demuestra su viabilidad técnica como herramienta de apoyo a la toma de decisiones de inversión, aportando estimaciones robustas tanto en valor como en dirección del precio.

La aplicación de simulación permite el uso de los mejores modelos predictivos sobre un período histórico de dos meses de una forma controlada y reproducible. Su diseño modular facilita la extensión a otros activos financieros, así como su integración futura con sistemas automatizados de alertas, brokers virtuales o incluso entornos intradías. Dado que el mejor modelo general no es el mejor en la predicción del valor ni de la dirección, la aplicación contempla el uso de tres modelos: el mejor general, el mejor en regresión y el mejor en clasificación. De este modo se dispone de una especie de ensamblaje de modelos que ayuda a tomar una mejor decisión.

Aunque actualmente está planteado como un sistema experimental con fines de investigación y demostrativos, podría escalarse a entornos reales si se asegura la ingesta de datos en tiempo real.

5.3.3 Cambios respecto al planteamiento inicial

Durante el desarrollo del proyecto se han producido varios ajustes relevantes que han condicionado su evolución:

- Cambio en la combinación de conjuntos de datos: inicialmente se iban a combinar de forma acumulativa (de 1 a 6), dando lugar a 6 configuraciones predefinidas. Finalmente, se optó por realizar todas las combinaciones posibles (63), maximizando la capacidad predictiva a costa de un incremento computacional notable. Esta decisión permitió identificar combinaciones óptimas no evidentes.

- Transformaciones con log y Yeo-Johnson: al principio se aplicaban sobre todos los conjuntos de datos, lo que introducía fuga de datos al usar datos futuros para determinar desplazamientos (log) o parámetros lambda (Yeo-Johnson). Posteriormente, se implementó una versión que extraía estos valores exclusivamente del conjunto de entrenamiento y los aplicaba a los de test. Aunque esta solución puede generar errores de transformación en algunos casos por la volatilidad de los datos, se priorizó la integridad metodológica y se descartaron aquellas combinaciones defectuosas.

- Control de errores y estabilidad del pipeline: en versiones iniciales, algunos modelos generaban predicciones infinitas, nulas o numéricamente inestables. Además, ciertas transformaciones y escalados eliminaban un número excesivo de registros del conjunto de entrenamiento o test, lo cual distorsionaba los resultados. Se introdujeron filtros automáticos que abortan ejecuciones no válidas si no se cumplen condiciones mínimas de calidad: sin valores infinitos, límite máximo de predicción basado en el histórico, y pérdida máxima del 10% de datos tras procesado.

- Uso de Winsorización: aunque los valores atípicos detectados eran legítimos en el contexto financiero (ver *Análisis de valores atípicos*), se observó que modelos como SARIMAX, LSTM o Transformers eran muy sensibles a estos extremos. Se implementó una Winsorización controlada con márgenes ajustables por modelo para mejorar la estabilidad sin perder señales clave.

- Sistema de orquestación automática: este sistema permitió lanzar todas las combinaciones de modelos y conjuntos de datos de forma completamente autónoma y reiniciable. Esta herramienta ha sido clave para escalar el número de pruebas, asegurar la trazabilidad de resultados y eliminar errores derivados de la ejecución manual.

- El entorno inicial se había preparado para utilizar siempre que fuera posible la GPU para acelerar el entrenamiento de los modelos, pero debido al reducido número de registros, se demostró que era más rápido utilizar una CPU potente y con muchos núcleos. Esto es debido a que para iniciar los trabajos en una GPU se requieren varios pasos extra (copiar los datos a su memoria) y si luego no hay cientos de miles o millones de registros a procesar, se emplea más tiempo en preparar la ejecución que en la propia ejecución.

5.3.4 Limitaciones del estudio

Entre las principales limitaciones del proyecto destacan:

- Cobertura parcial en variables de sentimiento: los datos de noticias (Alpha Vantage) están disponibles solo desde 2022. Además, Google Trends no ofrece granularidad diaria para periodos largos, lo cual exige descarga manual por tramos y normalización externa (ver *Normalización y escalado de datos*).

- Modelos con alta demanda de recursos: especialmente SARIMAX, cuyo entrenamiento implica 864 combinaciones de hiperparámetros sobre 1.512 configuraciones de datos, requiriendo más de 60 horas de ejecución en CPU. Los modelos de redes neuronales (LSTM y Transformers) también son bastante intensivos, entre 50400 y 43200 combinaciones de parámetros para cada test (252) arrojando entre 20 y 10 horas de entrenamiento que fueron posibles debido a la paralelización, uso de los dos equipos de trabajo y uso de la librería Optuna que optimiza la búsqueda de hiperparámetros.

- Infraestructura computacional limitada: inicialmente se descartó el uso de PySpark al estimarse que el volumen de pruebas era manejable localmente. Sin embargo, la combinación total de variables y transformaciones generó más de 2200 experimentos, que además hubo que repetir varias veces, por lo que un enfoque distribuido podría haber acelerado el proceso.

- Cambio estructural en la empresa NVIDIA: los datos anteriores a 2020 ya no representan el comportamiento actual de la compañía, que ha pasado de ser una firma orientada a videojuegos/minería a liderar el sector de IA y centros de datos. Se estima que el entrenamiento con datos 2020–2025 podría mejorar notablemente la precisión, pero no se aplicó debido al esfuerzo necesario para rehacer todo el procesamiento. Esta mejora queda identificada como una línea de trabajo futura.

Capítulo 6. CONCLUSIONES

El proyecto ha cumplido de manera satisfactoria los objetivos generales y específicos definidos inicialmente, desarrollando un sistema predictivo robusto y automatizado para estimar el precio de cierre diario de las acciones de NVIDIA.

Algunos modelos han logrado superar al predictor base, lo que sugiere que, si bien la teoría del Camino Aleatorio y de los Mercados Eficientes continúan siendo marcos de referencia válidos, los avances recientes en técnicas de inteligencia artificial abren la posibilidad de identificar patrones explotables que cuestionan parcialmente su aplicabilidad estricta.

6.1 Conclusiones del trabajo

A continuación, se detalla el grado de consecución de cada objetivo general y específicos:

- **Preparar los datos y establecer un sistema base como referencia para la evaluación.**
  - **Reunir y procesar datos históricos desde 2015 hasta mayo de 2025:**
    Se han recopilado y normalizado más de 2500 registros diarios con precios de apertura, cierre, volumen e indicadores técnicos de NVIDIA, cotizaciones de Big Tech, índices bursátiles, indicadores económicos y análisis de sentimiento (Google Trends y Alpha Vantage). Los datos se almacenan en un data warehouse ligero en MongoDB, lo que permite un acceso estructurado, centralizado y eficiente.

  - **Aplicar técnicas de preprocesamiento: detección de valores faltantes, tratamiento de valores atípicos, análisis de distribución, escalado y normalización**.
    Se ha realizado tratamiento completo de valores faltantes, análisis de valores atípicos mediante IQR, detección de duplicados y transformaciones (Log, retornos logarítmicos y Yeo-Johnson).
    Se ha asegurado la no contaminación (fuga de datos) en transformaciones, utilizando parámetros calculados únicamente sobre el conjunto de entrenamiento

  - **Implementar un predictor base: basado en uno aleatorio y de persistencia.**
    Ambos predictores han sido implementados como referencia base para comparar el rendimiento de los modelos.
    Sus resultados se han usado para calcular los índices IRB y CGMM.

- **Entrenar y evaluar los modelos de predicción.**
  - **Entrenar y evaluar modelos tradicionales: ARIMA, Prophet y SARIMAX.**

Se han entrenado con múltiples configuraciones, conjuntos de datos y transformaciones. SARIMAX ha mostrado una mejora significativa respecto a ARIMA, destacando en clasificación bajo backtesting.

- **Entrenar y evaluar modelos de aprendizaje automático: Random Forest y XGBoost.**
  Random Forest y XGBoost se han evaluado con cientos de combinaciones de conjuntos de datos e hiperparámetros.

- **Entrenar y evaluar modelos de aprendizaje profundo: LSTM y arquitecturas Transformer.**
  LSTM se ha entrenado con diferentes configuraciones. Durante este proceso se constató su sensibilidad a datos muy volátiles y a la "maldición de la dimensionalidad". Asimismo, se desarrolló un Transformer adaptado a series temporales multivariantes, que mediante autoatención capta dependencias de corto y largo plazo entre múltiples variables.

- **Probar cada modelo con distintas combinaciones de datos.**
  Inicialmente se planteó un enfoque acumulativo (conjuntos 1 a 6), pero se mejoró el planteamiento evaluando las 63 combinaciones posibles.
  Esto permitió identificar configuraciones óptimas no evidentes, a cambio de mayor carga computacional.

- **Validación sobre el conjunto de test y una simulación realista del último mes.**
  Se ha aplicado una doble estrategia de evaluación:
  Validación estática (idealizada, con todas las exógenas reales del día t).
  Backtesting con walk-forward sin desfase, simulando un entorno real de producción extendido sobre dos meses.

- **Comparativa final y desarrollo de aplicación de simulación de predicción diaria.**
  - **Identificar la mejor combinación de modelo y datos.**
    Se han utilizado los índices IRB y CGMM para seleccionar las mejores variantes de cada modelo, ponderando regresión (45%) y clasificación (55%).
    Los resultados están estructurados y visualizados mediante gráficas (*Figura 46* y *Figura 47*), paneles visuales y tablas resumen (*Tabla 19* y *Tabla 20*).

  - **Crear una tabla comparativa con todos los modelos y configuraciones de datos y destacar aquellos que superan el predictor base.**
    Se han generado más de 2200 pruebas cuyos resultados se recogen automáticamente en ficheros estructurados (CSV, PNG).
    Se han creado gráficas donde se muestran los modelos, configuración y conjunto de datos exógenos que superan al predictor base.

- **Desarrollar una aplicación que utilice el mejor modelo y conjunto de datos desplegada con contenedores que simule la predicción diaria del último mes**.
  Se ha implementado una aplicación de simulación en Flask, desplegada con Docker, que simula la predicción diaria sobre un período histórico de dos meses. En lugar de utilizar un modelo único, se ha optado por un ensamblaje de tres modelos complementarios, lo que refuerza la robustez de las predicciones.
  Esta aplicación constituye una prueba de concepto que demuestra la viabilidad técnica y la escalabilidad del sistema desarrollado.

Además, como aporte adicional no previsto inicialmente, se desarrolló un sistema de orquestación automática que permite ejecutar cientos de combinaciones de modelos de forma autónoma, reiniciable y sin errores humanos. Esta automatización ha reducido drásticamente el tiempo de prueba (de casi una hora a menos de un minuto por cada 8 combinaciones) y ha sido clave para garantizar la reproducibilidad experimental y facilitar la escalabilidad del sistema.

6.2 Conclusiones personales

La realización de este proyecto ha sido una experiencia mucho más compleja, exigente y enriquecedora de lo que inicialmente anticipaba. Infravaloré tanto la dificultad técnica como el tiempo necesario para construir un sistema predictivo robusto y evaluarlo en condiciones realistas. A medida que avanzaba, comprendí la magnitud de los retos asociados a la predicción bursátil, no solo por la volatilidad de los mercados, sino también por la gran cantidad de decisiones metodológicas, técnicas y prácticas que he tenido que tomar a lo largo del proyecto, con el apoyo de Samuel y del resto de profesores del máster. Ha sido una grata sorpresa poder aplicar muchos de los conocimientos adquiridos durante el Máster en Big Data, incluso en asignaturas que aún estaban en curso.

Además, este trabajo me ha demostrado lo complejo que resulta adquirir buenos datos, validar correctamente los modelos, prevenir la fuga de datos y llevar a cabo una ingeniería de características eficaz. Del mismo modo, mi conocimiento del ámbito financiero era muy limitado, y este proyecto ha sido una excelente oportunidad para iniciarse en la lógica de los mercados, la interpretación de indicadores económicos y el impacto del sentimiento colectivo en el precio de los activos.

Por todo ello, considero que he aprendido mucho más de lo que esperaba, tanto a nivel técnico como personal. Me siento especialmente satisfecho de haber construido un sistema funcional y escalable desde cero, que espero poder adaptar o ampliar en el futuro. Confío en que esta experiencia me sirva como base para seguir desarrollándome profesionalmente, y aspiro a poder aplicar todo este conocimiento en un entorno laboral real vinculado a la inteligencia artificial.

Capítulo 7. FUTURAS LÍNEAS DE TRABAJO

Este proyecto presenta varias oportunidades de mejora para aumentar la precisión de las predicciones y su aplicación práctica:

- **Selección y optimización de variables**: en lugar de limitarse a los seis conjuntos de datos definidos en este trabajo, podría explorarse el uso directo de las 40 variables disponibles y sus posibles combinaciones. Dado que el número de combinaciones asciende a más de un billón, esta aproximación solo sería viable mediante técnicas de optimización y selección de variables que reduzcan dimensionalidad y descarten las de baja relevancia.

- **Transformaciones individualizadas**: el uso de transformaciones y escalados específicos para cada variable, en lugar de un tratamiento uniforme, permitiría una mejor adaptación de los modelos a la heterogeneidad de los datos financieros.

- **Uso de datos representativos de NVIDIA actual**: restringir el entrenamiento al periodo 2020–2025, cuando NVIDIA consolidó su papel como líder mundial, evitaría el sesgo introducido por contextos históricos poco comparables con la situación presente.

- **Mayor granularidad temporal**: trabajar con datos horarios o de intervalos más cortos permitiría capturar dinámicas intradías y avanzar hacia predicciones aplicables en negociación de alta frecuencia. Esta aproximación incrementaría el número de registros disponibles, mitigando en parte la "maldición de la dimensionalidad" que afecta a los modelos de aprendizaje profundo.

- **Automatización de Google Trends**: implementar un sistema de captura y actualización totalmente automatizada de las series de Google Trends garantizaría un flujo continuo de información y eliminaría la necesidad de pasos manuales.

- **Extensión a otros activos financieros**: replicar la metodología en otras compañías tecnológicas, índices bursátiles o incluso sectores diferentes.

- **Evolución de la aplicación de simulación**: la aplicación está desarrollada en Flask (orientada a entornos de desarrollo) y podría migrar a un entorno productivo basado en Gunicorn y adaptarse para un entorno de predicción en tiempo real.

- **Despliegue en la nube:** el uso de contenedores Docker y notebooks facilita trasladar el sistema completo a un entorno en la nube, mejorando la escalabilidad, la accesibilidad y la capacidad de experimentación colaborativa.

En conjunto, estas líneas de trabajo constituyen la evolución natural que reforzaría el valor científico y práctico del sistema, acercándose a escenarios de predicción bursátil más realistas.

Capítulo 8. REFERENCIAS

[1] Y. LeCun, Y. Bengio, y G. Hinton, «Deep learning», *Nature*, vol. 521, n.º 7553, pp. 436-444, may 2015, doi: 10.1038/nature14539.

[2] E. F. Fama, «Efficient Capital Markets: A Review of Theory and Empirical Work», *The Journal of Finance*, vol. 25, n.º 2, pp. 383-417, 1970, doi: 10.2307/2325486.

[3] «*A Random Walk Down Wall Street*», *Wikipedia*. 19 de enero de 2025. Accedido: 15 de febrero de 2025. [En línea]. Disponible en: https://en.wikipedia.org/w/index.php?title=A_Random_Walk_Down_Wall_Street&oldid=1270398623

[4] W. Brock, J. Lakonishok, y B. LeBARON, «Simple Technical Trading Rules and the Stochastic Properties of Stock Returns», *The Journal of Finance*, vol. 47, n.º 5, pp. 1731-1764, 1992, doi: 10.1111/j.1540-6261.1992.tb04681.x.

[5] «Modelo autorregresivo integrado de media móvil», *Wikipedia, la enciclopedia libre*. 28 de noviembre de 2024. Accedido: 14 de febrero de 2025. [En línea]. Disponible en: https://es.wikipedia.org/w/index.php?title=Modelo_autorregresivo_integrado_de_media_m%C3%B3vil&oldid=163821512

[6] U. D. B, S. D, y A. P, «An Effective Time Series Analysis for Stock Trend Prediction Using ARIMA Model for Nifty Midcap-50», *IJDKP*, vol. 3, n.º 1, pp. 65-78, ene. 2013, doi: 10.5121/ijdkp.2013.3106.

[7] «Red neuronal recurrente», *Wikipedia, la enciclopedia libre*. 5 de diciembre de 2024. Accedido: 15 de febrero de 2025. [En línea]. Disponible en: https://es.wikipedia.org/w/index.php?title=Red_neuronal_recurrente&oldid=163959466

[8] «Memoria larga a corto plazo», *Wikipedia, la enciclopedia libre*. 14 de febrero de 2025. Accedido: 15 de febrero de 2025. [En línea]. Disponible en: https://es.wikipedia.org/w/index.php?title=Memoria_larga_a_corto_plazo&oldid=165411682

[9] Z. Jin, Y. Yang, y Y. Liu, «Stock closing price prediction based on sentiment analysis and LSTM», *Neural Comput & Applic*, vol. 32, n.º 13, pp. 9713-9729, jul. 2020, doi: 10.1007/s00521-019-04504-2.

[10] A. Vaswani *et al.*, «Attention Is All You Need», 2 de agosto de 2023, *arXiv*: arXiv:1706.03762. doi: 10.48550/arXiv.1706.03762.

[11] M. Jin *et al.*, «Time-LLM: Time Series Forecasting by Reprogramming Large Language Models», 29 de enero de 2024, *arXiv*: arXiv:2310.01728. doi: 10.48550/arXiv.2310.01728.

[12] Y. Nie, N. H. Nguyen, P. Sinthong, y J. Kalagnanam, «A Time Series is Worth 64 Words: Long-term Forecasting with Transformers», 5 de marzo de 2023, *arXiv*: arXiv:2211.14730. doi: 10.48550/arXiv.2211.14730.

[13] Y. Hu *et al.*, «FinMamba: Market-Aware Graph Enhanced Multi-Level Mamba for Stock Movement Prediction», 10 de febrero de 2025, *arXiv*: arXiv:2502.06707. doi: 10.48550/arXiv.2502.06707.

[14] R. Ding, K.-D. Luong, E. Rodriguez, A. C. A. L. da Silva, y W. Hsu, «Combining Graph Neural Network and Mamba to Capture Local and Global Tissue Spatial Relationships in Whole Slide Images», 5 de junio de 2024, *arXiv*: arXiv:2406.04377. doi: 10.48550/arXiv.2406.04377.

[15] Y. Wang *et al.*, «TimeXer: Empowering Transformers for Time Series Forecasting with Exogenous Variables», 11 de noviembre de 2024, *arXiv*: arXiv:2402.19072. doi: 10.48550/arXiv.2402.19072.

[16] *thuml/TimeXer*. (6 de abril de 2025). Python. THUML @ Tsinghua University. Accedido: 7 de abril de 2025. [En línea]. Disponible en: https://github.com/thuml/TimeXer

[17] «[2310.03589] TimeGPT-1». Accedido: 4 de abril de 2025. [En línea]. Disponible en: https://arxiv.org/abs/2310.03589

[18] *Nixtla/nixtla*. (4 de abril de 2025). Jupyter Notebook. Nixtla. Accedido: 4 de abril de 2025. [En línea]. Disponible en: https://github.com/Nixtla/nixtla

[19] ynie, *yuqinie98/PatchTST*. (3 de abril de 2025). Python. Accedido: 4 de abril de 2025. [En línea]. Disponible en: https://github.com/yuqinie98/PatchTST

[20] «Robotrader». Accedido: 17 de febrero de 2025. [En línea]. Disponible en: https://blogs.upm.es/robotrader/

[21] «Numerai», Numerai. Accedido: 7 de abril de 2025. [En línea]. Disponible en: https://www.numer.ai/

[22] L. Krauskopf, «Nvidia's $4 trillion milestone caps rise of stock market behemoth», *Reuters*, 9 de julio de 2025. Accedido: 11 de septiembre de 2025. [En línea]. Disponible en: https://www.reuters.com/business/finance/nvidias-4-trillion-milestone-caps-rise-stock-market-behemoth-2025-07-09/

[23] DeepSeek-AI *et al.*, «DeepSeek-R1: Incentivizing Reasoning Capability in LLMs via Reinforcement Learning», 22 de enero de 2025, *arXiv*: arXiv:2501.12948. doi: 10.48550/arXiv.2501.12948.

[24] «Stock Market News, Jan. 27, 2025: Nvidia Stock Sinks in AI Rout Sparked by China's DeepSeek», WSJ. Accedido: 17 de febrero de 2025. [En línea]. Disponible en: https://www.wsj.com/livecoverage/stock-market-today-dow-sp500-nasdaq-live-01-27-2025

[25] R. P. Prieto, «Pac-Man Evolution – MegaStorm Systems». Accedido: 17 de febrero de 2025. [En línea]. Disponible en: https://www.megastormsystems.com/games/pac-man-evolution

[26] «Las 7 magníficas: quiénes son y por qué ganan cada vez más peso en Wall Street», FundsPeople España. Accedido: 8 de abril de 2025. [En línea]. Disponible en: https://fundspeople.com/es/?post_type=glossary&p=201879

[27] «What Is A Data Warehouse? | A Full Guide», MongoDB. Accedido: 1 de julio de 2025. [En línea]. Disponible en: https://www.mongodb.com/resources/basics/cloud-explained/data-warehouse

[28] «Cross-industry standard process for data mining», *Wikipedia*. 26 de mayo de 2025. Accedido: 6 de junio de 2025. [En línea]. Disponible en: https://en.wikipedia.org/w/index.php?title=Cross-industry_standard_process_for_data_mining&oldid=1292359930

[29] «Centro de Computación Avanzada», Universidad Europea. Accedido: 8 de junio de 2025. [En línea]. Disponible en: https://universidadeuropea.com/tu-experiencia/instalaciones/centro-de-computacion-avanzada/

[30] «ProsusAI/finbert · Hugging Face». Accedido: 20 de junio de 2025. [En línea]. Disponible en: https://huggingface.co/ProsusAI/finbert

[31] «RPP-dev/AMPB: TFM - Análisis de modelos predictivos en bolsa». Accedido: 4 de mayo de 2025. [En línea]. Disponible en: https://github.com/RPP-dev/AMPB

[32] «Yahoo Finance - Stock Market Live, Quotes, Business & Finance News», Yahoo Finance. Accedido: 25 de marzo de 2025. [En línea]. Disponible en: https://finance.yahoo.com/

[33] «Free Stock APIs in JSON & Excel | Alpha Vantage». Accedido: 25 de marzo de 2025. [En línea]. Disponible en: https://www.alphavantage.co/

[34] «Stocks - Investing and trading for all». Accedido: 27 de marzo de 2025. [En línea]. Disponible en: https://www.reddit.com/r/stocks/

[35] «Google Trends». Accedido: 27 de marzo de 2025. [En línea]. Disponible en: https://trends.google.com/trends/

[36] «Free Stock Market Data API for Real-Time & Historical Data». Accedido: 7 de abril de 2025. [En línea]. Disponible en: https://marketstack.com/

[37] Finnhub.io, «Finnhub - Free realtime APIs for stock, forex and cryptocurrency.» Accedido: 27 de marzo de 2025. [En línea]. Disponible en: https://finnhub.io/

[38] «Google News», Google News. Accedido: 26 de mayo de 2025. [En línea]. Disponible en: https://news.google.com

[39] «International: Top News And Analysis», CNBC. Accedido: 27 de marzo de 2025. [En línea]. Disponible en: https://www.cnbc.com/world/

[40] «pandas.DataFrame.ffill — pandas 2.2.3 documentation». Accedido: 24 de abril de 2025. [En línea]. Disponible en: https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.ffill.html

[41] «pandas.DataFrame.fillna — pandas 2.2.3 documentation». Accedido: 27 de mayo de 2025. [En línea]. Disponible en: https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.fillna.html

[42] «(PDF) Winsorization for Identifying and Treating Outliers in Business Surveys.», ResearchGate. Accedido: 27 de julio de 2025. [En línea]. Disponible en: https://www.researchgate.net/publication/307632859_Winsorization_for_Identifying_and_Treating_Outliers_in_Business_Surveys

[43] «Walk forward optimization», *Wikipedia*. 19 de mayo de 2025. Accedido: 10 de junio de 2025. [En línea]. Disponible en: https://en.wikipedia.org/w/index.php?title=Walk_forward_optimization&oldid=1291124547

[44] «Complete LSTM architecture with equations showing how information moves... | Download Scientific Diagram», ResearchGate. Accedido: 24 de julio de 2025. [En línea]. Disponible en: https://www.researchgate.net/figure/Complete-LSTM-architecture-with-equations-showing-how-information-moves-through-the-cell_fig2_351291905

[45] «What is CSRF (Cross-site request forgery)? Tutorial & Examples | Web Security Academy». Accedido: 5 de septiembre de 2025. [En línea]. Disponible en: https://portswigger.net

[46] R. Bellman, *Dynamic programming*. Princeton, NJ: Princeton Univ. Pr, 1984.

[47] U. Khurana, H. Samulowitz, y D. Turaga, «Feature Engineering for Predictive Modeling using Reinforcement Learning», 21 de septiembre de 2017, *arXiv*: arXiv:1709.07150. doi: 10.48550/arXiv.1709.07150.

Capítulo 9. ANEXOS

9.1 Figuras complementarias y de apoyo

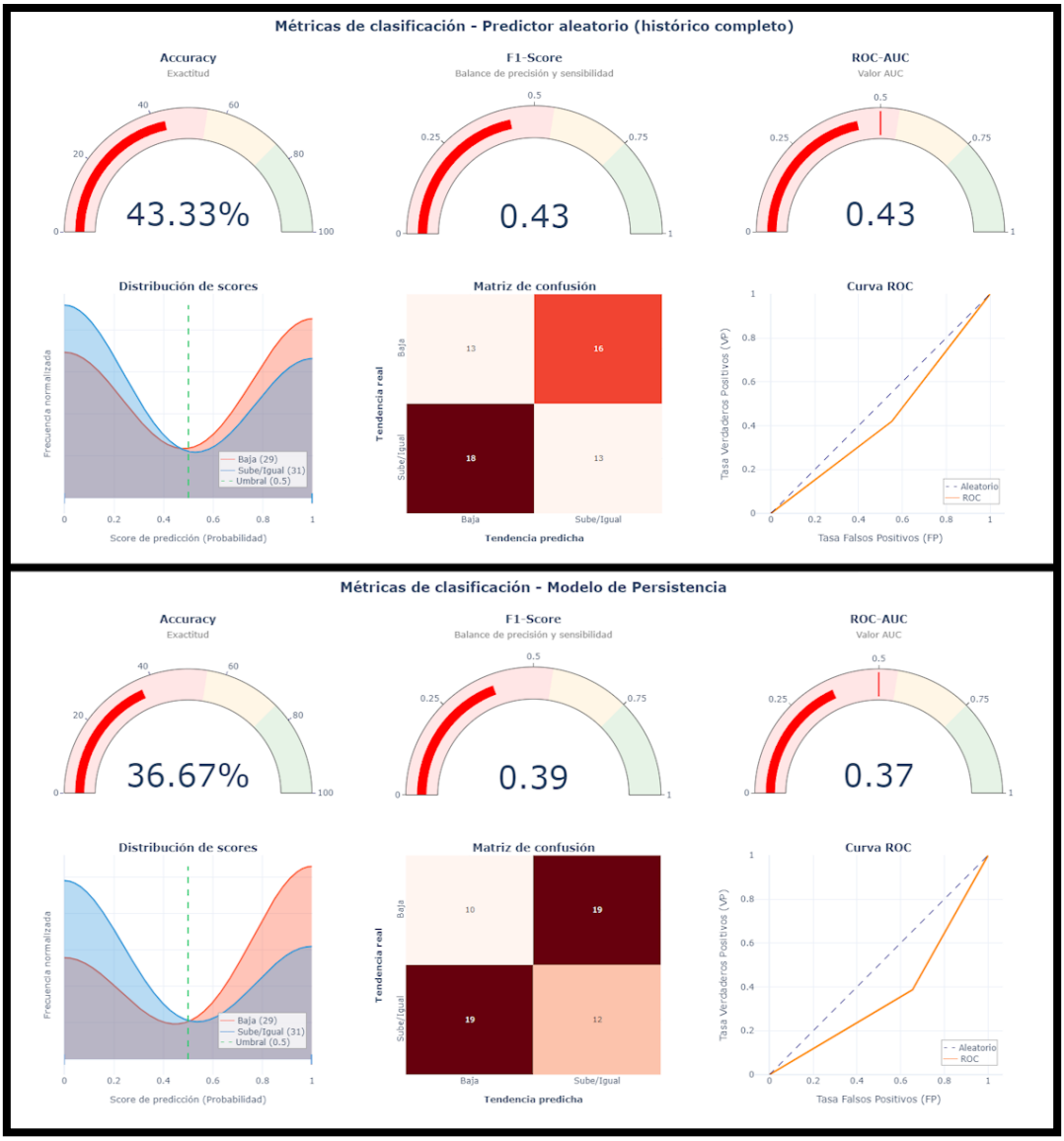


Figura 48 - Métricas de clasificación de predictor aleatorio y de persistencia (simulado).

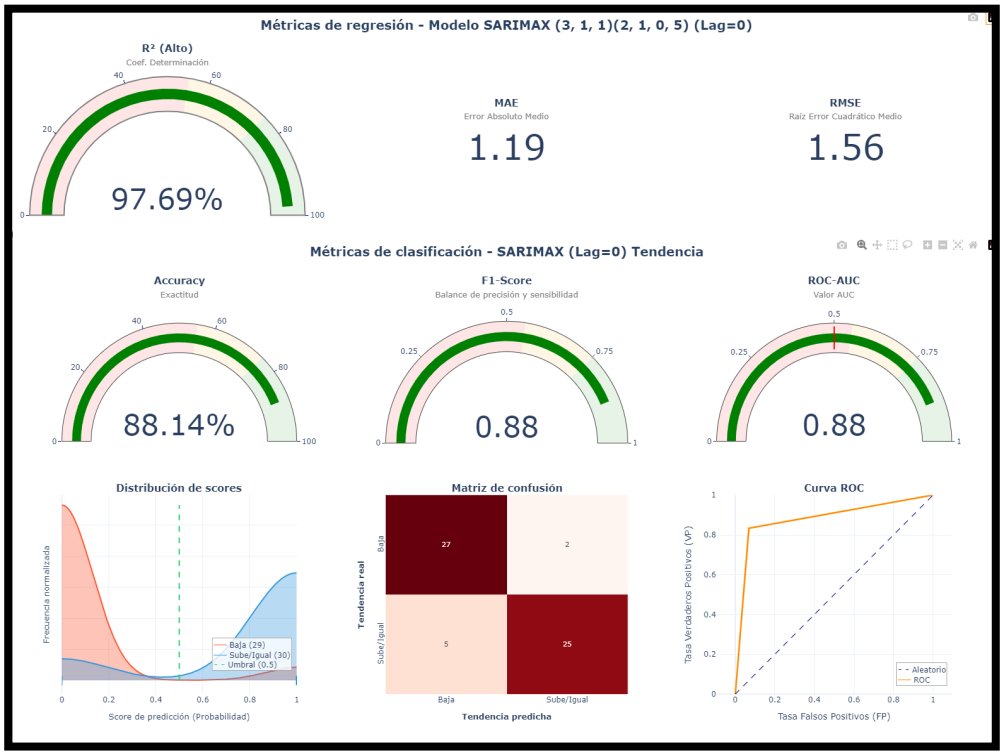


Figura 49 - Métricas de modelo SARIMAX en conjunto de test sin desfase (lag=0).

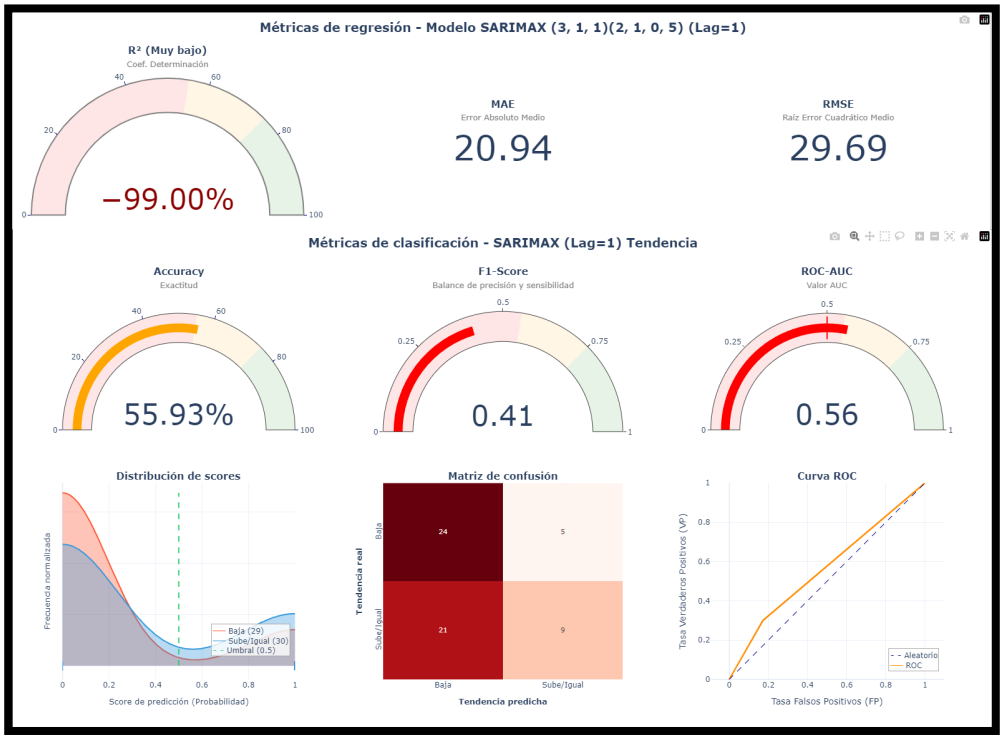


Figura 50 - Métricas de modelo SARIMAX en conjunto de test con desfase (lag=1).

Figura 51 – Métricas de backtesting de modelo SARIMAX con entrenamiento diario, cada 5 días y cada 10 días.

Figura 52 - Panel visual de resultados del mejor modelo en validación estática (CGMM total).

Figura 53 - Panel visual de resultados del mejor modelo en backtesting (CGMM total).

Figura 54 - Panel visual de resultados del mejor modelo en backtesting (CGMM regresión).

Figura 55 - Panel visual de resultados del mejor modelo en backtesting (CGMM clasificación).

9.2 Estructura de código en GitHub

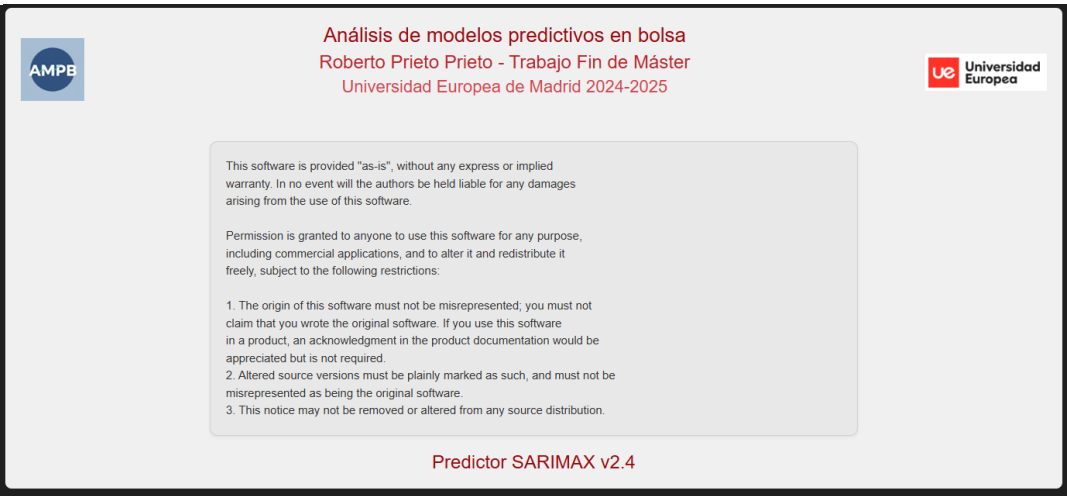
https://github.com/RPP-dev/AMPB

Acceso con token:

git clone https://RPP-dev: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@github.com/RPP-dev/AMPB.git

```
AMPB/
├── AMPBLib/                          # Módulo Python con funciones auxiliares
│
├── App/                              # Aplicación web de simulación
│   ├── Dockerfile.ampb               # Dockerfile para desplegar la aplicación
│   ├── AMPB-Sim.py                   # Script principal de la aplicación Flask
│   ├── AMPB-Sim_requirements.txt     # Dependencias de Python de la aplicación
│   ├── model_preds/                  # Predicciones generadas por los modelos
│   ├── static/                       # Archivos estáticos (CSS, JS, imágenes)
│   ├── templates/                    # Plantillas HTML para la interfaz web
│   ├── ssl_certs/                    # Certificados SSL
│   └── NVDA_2015-01-05_2025-05-23_SA.csv # Dataset con análisis de sentimiento
│
├── MongoDB/                          # Base de datos MongoDB (data warehouse ligero)
│
├── Notebook/                         # Jupyter Notebooks de modelos y preprocesamiento
│   ├── Preprocesamiento de datos.ipynb   # Limpieza y preparación de datos
│   ├── Orquestador.ipynb             # Coordinador de entrenamiento y evaluación
│   ├── Predictor Persistencia.ipynb  # Baseline: persistencia
│   ├── Predictor Aleatorio.ipynb     # Baseline: aleatorio
│   ├── Predictor ARIMA.ipynb         # Modelo ARIMA
│   ├── Predictor SARIMAX.ipynb       # Modelo SARIMAX
│   ├── Predictor Random Forest.ipynb # Modelo Random Forest
│   ├── Predictor XGBoost.ipynb       # Modelo XGBoost
│   ├── Predictor LSTM.ipynb          # Modelo LSTM
│   ├── Predictor Transformer.ipynb   # Modelo Transformer
│   ├── Predictor Prophet.ipynb       # Modelo Prophet
│   ├── Preprocesamiento de datos.csv # Datos preprocesados
│   ├── NVDA_2015-01-05_2025-05-23_SA.csv # Dataset completo con sentimiento
│   └── Requirements.txt              # Dependencias de los modelos
│
├── Resultados/                       # Resultados de todos los modelos (~2.280 pruebas)
│
├── Scripts/                          # Scripts de recogida y análisis de datos
│   ├── Dockerfile.python             # Dockerfile para entorno de recolección de datos
│   ├── GetData_requirements.txt      # Dependencias para obtención de datos
│   ├── GetDataAV.py                  # Datos de Alpha Vantage
│   ├── GetDataAV_SA.py               # Análisis de sentimiento con Alpha Vantage
│   ├── GetDataYF.py                  # Datos de Yahoo Finance
│   ├── GetDataYF_SA.py               # Análisis de sentimiento con Yahoo Finance
│   ├── GetDataRD_SA.py               # Análisis de sentimiento con Reddit
│   ├── GetData_cron                  # Configuración cron de recolección automática
│   ├── SA_Analyzer.py                # Script de análisis de sentimiento
│   ├── SA-GetAnalyzer_requirements.txt   # Dependencias del análisis de sentimiento
│   ├── SA-Analyzer_cron              # Cron para análisis automático
│   └── entrypoint.sh                 # Script de inicio para contenedores
│
├── docker-compose.yml                # Orquestador de contenedores Docker
└── README.md                         # Información general del proyecto
```

9.3 Código completo de un modelo




```python
# Importar librerías
import os
import argparse
os.environ['TF_CPP_MIN_LOG_LEVEL'] = '3'
import pandas as pd
import numpy as np
import pmdarima as pm
from statsmodels.tsa.statespace.sarimax import SARIMAX
from pmdarima.arima import ndiffs, nsdiffs
import time
import warnings

warnings.filterwarnings("ignore")
from statsmodels.tools.sm_exceptions import ConvergenceWarning, ValueWarning
warnings.filterwarnings("ignore", category=ConvergenceWarning)
warnings.filterwarnings("ignore", category=ValueWarning)

from ampblib import AMPBConfig, processData, generateEvaluation, updateNextDayExog, createReport, getExogVars, reverseTransformPredictions,
createModelIdentity

# 0. PARÁMETROS CONFIGURABLES
model_name = "SARIMAX"
model_version = "v2.4"

# Por defecto, permite ejecucion interactiva
default_transformation = "RetLog" # "None", "Log", "RetLog", "YeoJohnson"
default_exog_scaling = "None"   # "None", "Standard", "MinMax"
default_exog_set_id = 1 # 1="Directos", 2="IndicadoresTecnicos", 3="BigTech", 4="IndicesBursatiles", 5="IndicadoresEconomicos", 6="AnalisisSentimiento"

# Estos son fijos e internos, no los exponemos
nombre_archivo = "NVDA_2015-01-05_2025-05-23_SA.csv"
test_size = 60          # Número de días para el conjunto de test
optimize_orders = True   # True para optimizar (p,d,q) con auto_arima, False para usar valores fijos
fixed_order = (0,0,3)    # Orden (p,d,q) si optimizar = False.
fixed_seasonal_order = (0,0,0,5)  # Orden estacional (P,D,Q,s) si optimize_orders = False
winsorization_value = 0.005  # Aplicar winsorización (solo si hay alguna transformacion)
run_backtesting = True   # True para Backtesting con Walk-Forward y False para no realizar este test (utiliza orders optimizados anteriores o fijos)
retrain_interval = 5 # Reentrenar modelo completo cada n días en backtesting. 5 es un valor adecuado, +velocidad a cambio de perder un ~6% de rendimiento.

# Argumentos de línea de comandos
if AMPBConfig.INTERACTIVE:
    transformation = default_transformation
    exog_scaling = default_exog_scaling
    exog_set_id = default_exog_set_id
else:
    parser = argparse.ArgumentParser(description='Ejecuta modelo SARIMAX.')
    parser.add_argument('--transformation', type=str, default=default_transformation, choices=['None', 'Log', 'RetLog', 'YeoJohnson'])
    parser.add_argument('--exog_scaling', type=str, default=default_exog_scaling, choices=['None', 'Standard', 'MinMax'])
    #parser.add_argument('--exog_set_id', type=int, default=default_exog_set_id, choices=range(1, 7))
    def valid_exog(x):
        if not all(c in '123456' for c in str(x)):
            raise argparse.ArgumentTypeError(f"Solo dígitos 1-6: {x}")
        return x
    parser.add_argument('--exog_set_id', type=valid_exog, default=default_exog_set_id)
    args = parser.parse_args()
    transformation = args.transformation
    exog_scaling = args.exog_scaling
    exog_set_id = args.exog_set_id

AMPBConfig.printHeader(title=f"Predictor {model_name} {model_version}", testsize=test_size,
                       optimize=optimize_orders, backtesting=run_backtesting, transform=transformation,
                       exogscaling=exog_scaling, exogsetid=exog_set_id)

# 1. CARGA Y PREPARACIÓN DE DATOS
datos = pd.read_csv(nombre_archivo)

# Seleccionar columnas relevantes
mandatory_vars = ['Date','Close', 'Trend']

# Obtenemos lista de exogenas
exog_vars = getExogVars(exog_set_id)
df = datos[mandatory_vars + exog_vars]
```

```
# Convertir fechas a datetime
df['Date'] = pd.to_datetime(df['Date'])

# Verificar y mostrar estadísticas básicas de los datos
print(f"Datos en crudo cargados: {len(df)} registros de {df['Date'].min()} a {df['Date'].max()}.")

# Detectar valores faltantes
missing = df.isna().sum()
missing = missing[missing > 0]

if not missing.empty:
    total_before = len(df)
    print("Valores faltantes por columna (se borrarán estas filas):")
    for col, cnt in missing.items():
        print(f"  • {col}: {cnt} valores faltantes")
    df = df.dropna().reset_index(drop=True)
    total_after = len(df)
    removed = total_before - total_after
    print(f"\nFilas borradas: {removed}")
else:
    print("No se encontraron valores faltantes.")

# Poner Date como indice
df.set_index('Date', inplace=True)
#df = df.asfreq('B', method='pad')  # Se asume que se trata de datos bursátiles (días hábiles)

print(f"\nDatos cargados: {AMPBConfig.COLOR_VALUE}{len(df)}{AMPBConfig.COLOR_RESET} registros. Variables exógenas seleccionadas:
{AMPBConfig.COLOR_VALUE}{len(exog_vars)}{AMPBConfig.COLOR_RESET}\n{exog_vars}")

# 2. DIVISIÓN ENTRE ENTRENAMIENTO Y TEST

# 2A. Se utiliza el 90% de los datos para entrenamiento y el 10% para test
#split_index = int(len(df) * 0.90)
#df_train = df.iloc[:split_index].copy()
#df_test = df.iloc[split_index:].copy()

# 2B. Separamos datos de entrenamiento y de test por fecha
#split_date = pd.Timestamp('2024-12-01')
#df_train = df.loc[:split_date].copy()
#df_test = df.loc[split_date:].copy()

# 2C. Separamos por numero de dias.
df_train = df.iloc[:-test_size]
df_test = df.iloc[-test_size:]

# Guardar valores originales (inmutables para referencia futura)
y_train_original = df_train['Close'].copy()
y_test_original = df_test['Close'].copy()
X_train_original = df_train[exog_vars].copy()
X_test_original = df_test[exog_vars].copy()

# Variables de trabajo (se transformarán/escalarán según configuración)
y_train = df_train['Close']
y_test  = df_test['Close']
X_train = df_train[exog_vars].copy()
X_test = df_test[exog_vars].copy()

print(f"\nDatos divididos:")
print(f"  Entrenamiento: {AMPBConfig.COLOR_VALUE}{len(y_train)}{AMPBConfig.COLOR_RESET} filas (hasta {y_train.index[-1].date()})")
print(f"  Test:          {AMPBConfig.COLOR_VALUE}{len(y_test)}{AMPBConfig.COLOR_RESET} filas (desde {y_test.index[0].date()})")


# 3. PROCESAR DATOS: TRANSFORMACIONES, ESCALADO, ALINEACION Y ANALISIS DE CALIDAD
# Bajo determinadas circunstancias, puede abortar la ejecucion.
processing_results = processData(
    y_train, y_test, X_train, X_test,
    y_train_original, y_test_original, X_train_original, X_test_original,
    df_test, exog_vars, transformation, exog_scaling,
    winsorization_value=winsorization_value,
    analyze=False               # True para ejecutar análisis de calidad
)
params_close = processing_results['params_close']
params_exog  = processing_results['params_exog']
y_scaler = processing_results['y_scaler']
exog_scaler = processing_results['exog_scaler']
df_test_aligned = processing_results['df_test_aligned']
prediction_max_limit = processing_results['prediction_max_limit']
quality_results = processing_results['quality_results']

# 4. ENTRENAMIENTO SARIMAX (SOLO EN TRAIN DATA)
# 4A. OPTIMIZACIÓN DE PARÁMETROS CON AUTO_ARIMA
if optimize_orders:
    # Definición de parámetros dinámicos según la transformación
    if transformation == "RetLog":
        d_values = [0]
        D_values = [0]
        print(f"Transformación 'RetLog': forzando d=0, D=0 (serie ya diferenciada)")
    else:
        # Para otras transformaciones, forzar exploración explícita de d y D
        suggested_d = ndiffs(y_train, test='kpss')
        suggested_D = nsdiffs(y_train, test='ch', m=5)  # Test de estacionariedad estacional
        max_d = min(2, suggested_d)  # Máximo: sugerido + 1, pero no más de 2
        max_D = min(1, suggested_D)  # Máximo: sugerido + 1, pero no más de 1
        d_values = list(range(0, max_d + 1))  # [0, 1, 2] según max_d
        D_values = list(range(0, max_D + 1))  # [0, 1] según max_D
        print(f"Transformación '{transformation}': ndiffs sugiere d={suggested_d}, nsdiffs sugiere D={suggested_D}")
        print(f"Forzando auto_arima a explorar d={d_values}, D={D_values}")

    print("\nBuscando el mejor orden para SARIMAX...")

    # Búsqueda manual por combinaciones de d y D para asegurar exploración completa
    best_aic = np.inf
    best_model = None

    for d_val in d_values:
        for D_val in D_values:
            print(f"\n--- Explorando d={d_val}, D={D_val} ---")
            try:
```

```
                temp_model = pm.auto_arima(y_train,
                                           X=X_train,                    # Variables exógenas
                                           d=d_val,                      # Valor fijo para d
                                           D=D_val,                      # Valor fijo para D
                                           method='lbfgs',               # Usa el optimizador L-BFGS (rápido y eficiente)
                                           start_p=0, max_p=3,
                                           start_q=0, max_q=3,
                                           start_P=0, max_P=2,
                                           start_Q=0, max_Q=2,
                                           m=5,                          # Frecuencia estacional (5 para datos bursátiles semanales)
                                           seasonal=True,
                                           stepwise=True,                # Búsqueda inteligente
                                           suppress_warnings=True,       # Suprimir warnings de convergencia, etc.
                                           error_action='ignore',        # Ignorar órdenes que fallen
                                           trace=False,
                                           information_criterion='aic',  # Criterio para seleccionar el mejor modelo.
                                           n_jobs=-1                     # Paraleliza la ejecución
                                          )

                if temp_model.aic() < best_aic:
                    best_aic = temp_model.aic()
                    best_model = temp_model
                    print(f"Nuevo mejor modelo encontrado: {temp_model.order}{temp_model.seasonal_order} con AIC={best_aic:.3f}")

            except Exception as e:
                print(f"Error explorando d={d_val}, D={D_val}: {str(e)[:100]}...")
                continue

    auto_model = best_model

    best_order = auto_model.order
    best_order_seasonal = auto_model.seasonal_order
    print(f"\nAuto-ARIMA encontró órdenes óptimos: {best_order} y {best_order_seasonal}")

else:
    # Usar órdenes fijos
    best_order = fixed_order
    best_order_seasonal = fixed_seasonal_order
    print(f"Utilizando órdenes fijos: {best_order} y {best_order_seasonal}")

# 4B. AJUSTE DE SARIMAX
print("Ajustando modelo...")
best_model = SARIMAX(y_train, exog=X_train, order=best_order, seasonal_order=best_order_seasonal,
                     enforce_stationarity=False, enforce_invertibility=False).fit(disp=False, method='lbfgs')

# Información del modelo
model_title, model_hash = createModelIdentity(model_name, model_version, f"{best_order}{best_order_seasonal}", transformation, exog_scaling, exog_set_id)
print(f"\n{AMPBConfig.COLOR_INFO}Modelo {model_name}{model_version}:{AMPBConfig.COLOR_RESET}")
print(f" Order (p,d,q): {best_order}")
print(f" Seasonal Order (P,D,Q,s): {best_order_seasonal}")
print(f" Título: '{model_title}' | HashID: {model_hash}\n")

# 5A. PREDICCIÓN Y EVALUACIÓN EN EL CONJUNTO DE TEST (VALIDACIÓN ESTÁTICA)
print(f"{AMPBConfig.COLOR_INFO}Validación Estática{AMPBConfig.COLOR_RESET}")

# Predicción del período de test completo
forecast_scaled_transformed = best_model.get_forecast(steps=len(y_test), exog=X_test).predicted_mean
forecast_scaled_transformed.index = y_test.index

# Aplicar pipeline de des-transformación
forecast_original = reverseTransformPredictions(
    forecast_scaled_transformed,
    y_train_original.iloc[-1],
    y_scaler,
    transformation, params_close, prediction_max_limit)

# Predicción día siguiente
X_next_day = updateNextDayExog(
    X_test,
    feature_original_close=y_test_original.iloc[-1],
    transformation=transformation,
    params_exog=params_exog,
    exog_scaler=exog_scaler,
    prev_open_original = (
        X_test_original['Open'].iloc[-1]
        if transformation == "RetLog" and 'Open' in X_test_original.columns
        else None
    )
)

next_day_date = y_test.index[-1] + pd.tseries.offsets.BDay(1)
X_next_day.index = [next_day_date]
next_forecast = best_model.get_forecast(steps=1, exog=X_next_day).predicted_mean
next_day_forecast_original = reverseTransformPredictions(
    next_forecast,
    y_test_original.iloc[-1], # Para la referencia del día siguiente, usar el último valor real del test
    y_scaler,
    transformation, params_close, prediction_max_limit).iloc[0]

# 5B. EVALUACIÓN Y GRÁFICAS
# Evaluación de validación estática
sv_r2, sv_mae, sv_rmse, sv_accuracy, sv_f1_score, sv_roc_auc = generateEvaluation(
    y_test_original, forecast_original, df_test_aligned, model_title, model_hash, next_day_date, next_day_forecast_original, "Static Validation")

# Guardar reporte
createReport(model_name, "SV", f"{transformation}_{exog_scaling}_{exog_set_id}", model_title, model_hash, sv_r2, sv_mae, sv_rmse, sv_accuracy,
sv_f1_score, sv_roc_auc)

# 6. EVALUACIÓN CON CROSS VALIDATION (VALIDACIÓN CRUZADA)
# Opcional, no disponible en este modelo.

# 7A. MODO DE BACKTESTING: PREDICCIÓN DÍA A DÍA (VALIDACIÓN BACKTESTING)
if run_backtesting:
    model_title_backtest = f'{model_title} (Backtesting [{retrain_interval}d])'
    print(f"\n{AMPBConfig.COLOR_INFO}Modo Backtesting con Walk-Forward (Retrain cada {retrain_interval} días){AMPBConfig.COLOR_RESET}")

    # Inicializar historiales
    history_y = y_train.copy()              # Datos transformados/escalados para el modelo
```

```
    history_X = X_train.copy()             # Datos transformados/escalados para el modelo
    history_y_original = y_train_original.copy()  # Valores originales para referencias
    history_X_original = X_train_original.copy()

    predictions_original_bt = []
    model_bt = None  # Modelo que se reutilizará entre reentrenamientos

    bt_start = time.time()
    for t in range(len(y_test)):
        print(f" Backtesting: {t+1}/{len(y_test)}", end='')

        # Reentrenar el modelo cuando sea necesario
        if t % retrain_interval == 0:
            print(f" [Reentrenando...]", end='')
            model_bt = SARIMAX(history_y, exog=history_X, order=best_order,
                               seasonal_order=best_order_seasonal,
                               enforce_stationarity=False, enforce_invertibility=False).fit(disp=False, maxiter=50)
            print(f" [✓]", end='')

        print(f"")  # Nueva línea

        # Preparar exógenas propagadas
        X_current_propagated = updateNextDayExog(
            history_X,
            feature_original_close=history_y_original.iloc[-1],        # Close_{t-1} original
            transformation=transformation,
            params_exog=params_exog,
            exog_scaler=exog_scaler,
            prev_open_original = (
                history_X_original['Open'].iloc[-1]
                if transformation == "RetLog" and 'Open' in X_test_original.columns
                else None
            )
        )

        # Predecir 1 paso adelante
        forecast_step_scaled_transformed = model_bt.get_forecast(steps=1, exog=X_current_propagated).predicted_mean

        # Des-transformar usando pipeline centralizado
        reference_val = history_y_original.iloc[-1]
        forecast_step_original = reverseTransformPredictions(
            forecast_step_scaled_transformed,
            reference_val,
            y_scaler,
            transformation,
            params_close,
            prediction_max_limit
        ).iloc[0]

        # Guardar predicción
        predictions_original_bt.append(forecast_step_original)

        # Actualizar historiales con datos reales del día t
        history_y = pd.concat([history_y, y_test.iloc[t:t+1]])
        history_X = pd.concat([history_X, X_test.iloc[t:t+1]])
        history_y_original = pd.concat([history_y_original, pd.Series([y_test_original.iloc[t]], index=[y_test_original.index[t]])])
        history_X_original = pd.concat([history_X_original, X_test_original.iloc[t:t+1]])

    # Estadísticas de reentrenamiento
    total_retrains = (len(y_test) + retrain_interval - 1) // retrain_interval
    print(f" Backtesting completado en {time.time() - bt_start:.1f}s")
    print(f" Reentrenamientos realizados: {total_retrains} (cada {retrain_interval} días)\n")

    # Crear Serie con predicciones del backtesting
    forecast_backtest_original = pd.Series(predictions_original_bt, index=y_test_original.index)

# 7B. MODO DE BACKTESTING: PREDICCIÓN SIGUIENTE DÍA
if run_backtesting:
    X_next_day_bt = updateNextDayExog(
        history_X,
        feature_original_close=history_y_original.iloc[-1],
        transformation=transformation,
        params_exog=params_exog,
        exog_scaler=exog_scaler,
        prev_open_original = (
                history_X_original['Open'].iloc[-1]
                if transformation == "RetLog" and 'Open' in X_test_original.columns
                else None
            )
    )
    X_next_day_bt.index = [next_day_date]
    # Predecir y des-transformar usando pipeline centralizado
    next_forecast_scaled_transformed_bt = model_bt.get_forecast(steps=1, exog=X_next_day_bt).predicted_mean
    next_day_forecast_val_bt_original = reverseTransformPredictions(
        next_forecast_scaled_transformed_bt,
        history_y_original.iloc[-1], # Predicción para el día siguiente usando el último modelo del backtesting
        y_scaler,
        transformation,
        params_close,
        prediction_max_limit
    ).iloc[0]

# 7C. MODO DE BACKTESTING: EVALUACIÓN Y GRÁFICAS
if run_backtesting:
    # Evaluación del backtesting
    bt_r2, bt_mae, bt_rmse, bt_accuracy, bt_f1_score, bt_roc_auc = generateEvaluation(
        y_test_original,
        forecast_backtest_original,
        df_test_aligned,
        model_title_backtest,
        model_hash, next_day_date, next_day_forecast_val_bt_original,  "Backtesting")

    # Guardar informe del backtesting
    createReport(model_name, "BT", f"{transformation}_{exog_scaling}_{exog_set_id}", model_title, model_hash, bt_r2, bt_mae, bt_rmse, bt_accuracy,
bt_f1_score, bt_roc_auc)
```

9.4 Historial de cambios en repositorio Subversion

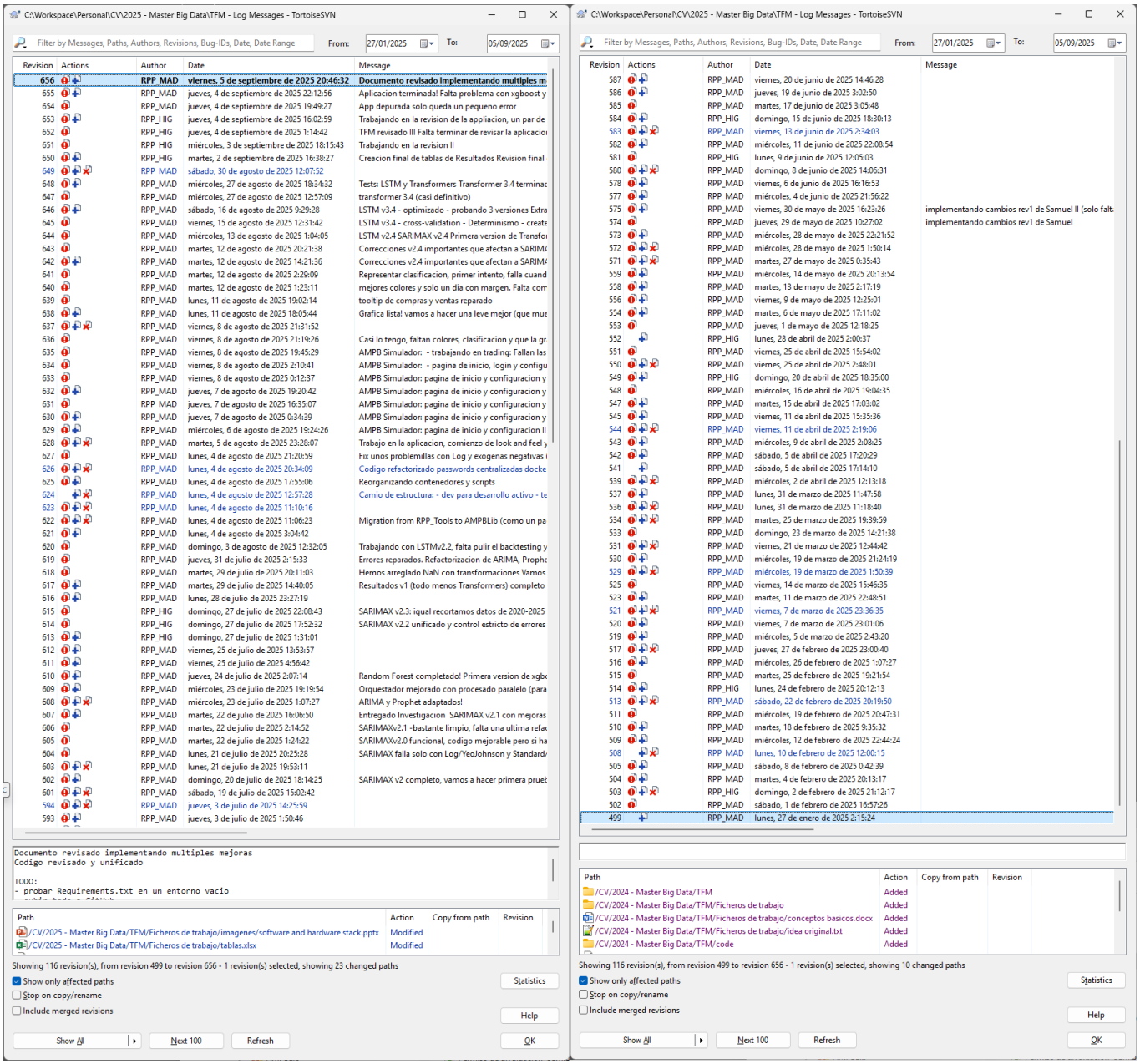


Figura 56 - Historial de cambios en repositorio Subversion.

Meses de actividad: 8

Envió de cambios totales: 116

Media por mes: 14 (mínimo 1 y máximo 30)

Número total de ficheros cambiados: 15657

[PÁGINA INTENCIONADAMENTE EN BLANCO]