

Aprendizaje Automático aplicado al Trading Algorítmico

Máster en Big Data y Business Analytics

Trabajo de Fin de Máster

(30 de noviembre de 2021 - 30 de noviembre de 2023)



Autor: Gabriel Esteban Manzanares

Tutor: Juan Manuel Moreno Lamparero

Abstract

Este estudio se enfoca en el desarrollo de un algoritmo de trading con machine learning para el índice NASDAQ Composite, con el objetivo de batir tanto en rentabilidad como en riesgo a su índice de referencia. Para ello, se implementa una arquitectura que extrae datos de diversas fuentes y utiliza una solución de almacenamiento persistente en SQL. Las dinámicas que rigen los mercados financieros son expuestas en un análisis exploratorio de datos, para luego aplicar diversas técnicas de preprocesamiento a series temporales financieras.

En la fase de entrenamiento y evaluación de modelos, se testea la capacidad predictiva para este tipo de problemas de algoritmos como Extreme Gradient Boosting (XGBoost), Support Vector Machine, Random Forest y redes neuronales recurrentes como Long Short-Term Memory. El mejor de estos algoritmos es combinado con el algoritmo K-Nearest Neighbors (KNN), en una innovadora técnica basada en la unión de clasificación supervisada y no supervisada. Una vez realizadas las predicciones, se exploran ensambles entre los diferentes algoritmos para mejorar la capacidad predictiva y reducir el riesgo.

Finalmente, se logra construir un modelo ensamblando los algoritmos K-Nearest Neighbors, Extreme Gradient Boosting y Random Forest, generando una rentabilidad del 155.60% entre 2021 y 2023, en un período en el que el NASDAQ Composite cayó un -6.41%. Estos resultados se obtienen mediante técnicas de preprocesamiento como Wavelet Transform, Normalización y Stacked Autoencoders, cuya contribución a la capacidad predictiva se compara y expone en diferentes algoritmos.

Índice

1. INTRODUCCIÓN.....	5
2. ANTECEDENTES.....	6
2.1 Activos	7
2.2 Algoritmos.....	7
2.3 Preprocesamiento.....	8
2.4 Análisis.....	8
2.5 Aprendizaje por refuerzo.....	9
3. HIPÓTESIS.....	9
4. OBJETIVOS.....	11
5. METODOLOGÍA Y DESARROLLO.....	12
5.1 Extracción de datos.....	13
5.2 Transformación e integración de datos.....	16
5.3 Análisis exploratorio de datos.....	18
<i>5.3.1 Resumen estadístico inicial.....</i>	19
<i>5.3.2 Valores atípicos.....</i>	20
<i>5.3.3 Normalidad.....</i>	21
<i>5.3.4 Correlación.....</i>	22
<i>5.3.5 Heterocedasticidad.....</i>	24
<i>5.3.6 Interdependencia temporal.....</i>	25
<i>5.3.7 Separabilidad lineal.....</i>	26
<i>5.3.8 Estacionalidad.....</i>	27
<i>5.3.9 Métricas de riesgo y rentabilidad.....</i>	28
<i>5.3.10 Conclusiones del EDA.....</i>	30
5.4 Ingeniería de características.....	31

<i>5.4.1 División de los datos</i>	32
<i>5.4.2 Wavelet transform</i>	33
<i>5.4.3 Normalización</i>	35
<i>5.4.4 Stacked Autoencoders</i>	36
<i>5.4.5 Procesamiento y exportación</i>	37
5.5 Entrenamiento y evaluación de modelos	37
<i>5.5.1 Regresión logística</i>	38
<i>5.5.2 Random Forest</i>	39
<i>5.5.3 Extreme Gradient Boosting Classifier (XGBoost)</i>	40
<i>5.5.4 Support Vector Machine</i>	40
<i>5.5.5 Long Short-Term Memory</i>	42
<i>5.5.6 K-nearest neighbors (KNN)</i>	43
6. RESULTADOS	45
7. DISCUSIÓN	49
8. CONCLUSIÓN	52
9. BIBLIOGRAFÍA	55
10. APÉNDICE	58

1. Introducción

En un mundo en el que las pirámides demográficas invertidas y la elevada deuda pública amenazan con quebrar los sistemas de pensiones de reparto de las principales economías desarrolladas, la gestión efectiva de patrimonios y la rentabilización de los ahorros emergen como imperativos esenciales a la hora de asegurar un futuro financiero sostenible. España, según la Oficina Europea de Estadística (Eurostat), se enfrenta a una acelerada disminución de trabajadores por pensionista, pasando de 3.3 en 2022 a 1.7 en 2050 (Esteban, 2023). Las limitaciones fiscales del Estado para abordar esta problemática se vuelven evidentes, con una deuda sobre el PIB que alcanza el 113.1% en junio de 2023, según el Banco de España.

Ante esta situación, los mercados financieros se postulan como una herramienta alternativa de generación de riqueza a largo plazo para hacer frente a este problema. Aparte de su papel como proveedores de liquidez de los inversores del mercado primario y de tratar de ofrecer una valoración justa de los activos financieros, los mercados secundarios de renta variable pueden suponer una alternativa a los planes públicos de pensiones, no solo para los grandes patrimonios, sino también para los inversores minoristas.

Sin embargo, para que esta alternativa sea viable, Sheta, Ahmed y Faris (2015) destacan la necesidad de enfoques estables, robustos y adaptables que proporcionen modelos capaces de predecir con precisión los índices bursátiles, dada la complejidad de unos mercados influenciados por factores políticos, financieros y eventos de reserva. Hasta la fecha, el análisis técnico era una de las principales herramientas con las que contaba el inversor para apoyarse en su toma de decisiones. No obstante, este se ha vuelto simplemente obsoleto en la coyuntura actual. Nayak, Misra y Behera (2014) señalan que, con el avance de la tecnología de la Inteligencia Artificial y los mercados interrelacionados, el análisis técnico basado únicamente en modelos de pronóstico ya no es suficiente para afrontar los desafíos del mercado global actual.

Este es uno de los motivos por los que el sector de la tecnología financiera, conocido como FinTech, ha experimentado un crecimiento significativo en los últimos años, revolucionando la forma en que se abordan diversas cuestiones asociadas a esta industria, tales como la gestión de carteras, la gestión de riesgo, la inversión o la detección de fraudes. Li, Zheng y

Zheng (2019) resaltan la demanda tanto de inversores institucionales como individuales para explorar algoritmos de trading autónomos basados en Inteligencia Artificial que se adapten al mercado de manera dinámica.

En este contexto, el trading algorítmico, que a menudo se considera sinónimo de *quantitative trading*, surge como una técnica que utiliza computadoras y reglas matemáticas para realizar decisiones de trading automáticas. Este enfoque proporciona la ventaja de suprimir la complicada y subjetiva tarea de definir reglas explícitas para operar mercados financieros específicos. Además, ha demostrado ser de utilidad para el conjunto de inversores, ya que ha mejorado sustancialmente la liquidez del mercado (Sazu, 2022).

Combinando este enfoque con los últimos avances tecnológicos, este trabajo se propone desarrollar una estrategia de trading algorítmico que, haciendo uso de la Inteligencia Artificial, consiga generar una rentabilidad consistente sobre el capital invertido a medio y largo plazo. Esta propuesta busca cubrir la necesidad de gestionar y rentabilizar eficientemente el capital en un entorno financiero dinámico y desafiante. A lo largo de esta investigación, se abordarán los desafíos y oportunidades que presenta este enfoque, considerando la eficiencia del mercado¹, la predicción en mercados globales y la evolución del análisis técnico en medio de la revolución industrial que ha supuesto la Inteligencia Artificial. Por tanto, se busca contribuir al creciente cuerpo de conocimientos en el ámbito del trading algorítmico y ofrecer perspectivas valiosas para inversores, gestores de fondos y otros actores del mercado financiero.

2. Antecedentes

La literatura existente sobre el uso de machine learning en trading algorítmico se caracteriza por su heterogeneidad, abordando diversas estrategias y desafíos desde enfoques muy distintos. A modo introductorio, Hansen (2020) destaca la creciente presencia de modelos de machine learning en el trading algorítmico e investiga cómo los quants² gestionan la complejidad de estos modelos. A través de entrevistas, muestra que los quants aplican la

¹ Hipótesis del Mercado Eficiente, la cual será comentada en las siguientes secciones.

² Un "quant" (abreviatura de "quantitative analyst" o analista cuantitativo en español) es un profesional especializado en el análisis cuantitativo y la aplicación de modelos matemáticos y estadísticos para entender y tomar decisiones en los mercados financieros.

navaja de Ockham³ como una herramienta heurística para evitar el exceso de complejidad en los modelos y garantizar un nivel de control e interpretabilidad. Este enfoque revela la transformación del papel humano en finanzas impulsadas por datos y modelos.

2.1 Activos

La variedad de activos en los que se han aplicado los modelos de machine learning también es amplia, yendo desde las acciones hasta los índices bursátiles, pasando por las materias primas y las criptomonedas. El auge de estas últimas durante los últimos años ha provocado el florecimiento de la experimentación en este campo. Cabe destacar el trabajo de Sebastião y Godinho (2021), en el que se examina el éxito de un enfoque de ensamblaje de modelos de machine learning en condiciones de mercado adversas para incrementar la predictibilidad. Otros trabajos, como los de Koker y Koutmos (2020) y Madan, Saluja y Zhao (2015) exploran la precisión y rentabilidad ajustada al riesgo de estrategias de trading con criptomonedas basadas en algoritmos de machine learning. En la conclusión de este último, se propone un método de preprocesamiento para trabajos futuros basado en clasificación no supervisada que ha servido de inspiración en uno de los algoritmos empleados en este trabajo.

2.2 Algoritmos

En lo que respecta a tipos de algoritmos, las redes neuronales recurrentes como Long Short-Term Memory (LSTM) son uno de los más utilizados debido a su capacidad para almacenar, olvidar y leer información del estado a largo plazo de la dinámica subyacente (Chen, Chen y Fushimi 2017). Estos autores se centran en la aplicación de redes LSTM para predecir el precio de acciones y desarrollar estrategias de trading algorítmico. Los resultados indican que la estrategia basada en las predicciones de LSTM supera a otras estrategias de trading. Por su parte, Bao, Yue y Rao (2017) presentan un marco de trabajo de deep learning que combina transformadas wavelet, stacked autoencoders y LSTM para la predicción de precios de acciones, superando a modelos similares en capacidad predictiva y rentabilidad.

Support Vector Machine (SVM) es otro de los algoritmos más recurrentes en la literatura y que mejores resultados ha demostrado. Sheta, Ahmed y Faris (2015) comparan el rendimiento

³ La navaja de Ockham es un principio filosófico que establece que, en caso de que haya varias explicaciones posibles para un fenómeno, la explicación más simple es la más probable de ser correcta.

de modelos de regresión, redes neuronales artificiales y SVM para predecir el precio del S&P500, siendo este último el que mejor desempeño tuvo. En Pai y Wei (2007) se emplea un modelo SVM para predecir direcciones de movimiento de futuros de índices bursátiles, destacando la influencia del preprocesamiento de datos en la precisión de la predicción. También se han realizado trabajos de otros algoritmos como K-Means (Xiong, Zhang, 2016) o Random Subspace Classifier Network (Zhora, 2005), pero estos son una minoría.

2.3 Preprocesamiento

El preprocesamiento de los datos antes de entrenar los modelos es uno de los tópicos que más protagonismo han recibido en la literatura debido a su importancia en la mejora de la capacidad predictiva de los modelos. En Liang et al. (2019) se introduce el uso de una variante de la transformada wavelet llamada Multioptimal Combination Wavelet Transform (MOCWT) para el preprocesamiento de datos en la predicción de precios de acciones, mostrando mejoras en la precisión predictiva. Kumar y Kalia (2012) proponen una representación simbólica de datos bursátiles y destaca la importancia del preprocesamiento en la minería de datos financieros. En el trabajo de Vincent et al. (2010) se explora una estrategia basada en una variación de los stacked autoencoders denominada Stacked Denoising Autoencoders (SDAE) que disminuye la cantidad de ruido en las variables del modelo, demostrando un incremento en el rendimiento. Otros autores evalúan el impacto de diferentes métodos de normalización en modelos de redes neuronales artificiales para la predicción de índices bursátiles (Nayak, Misra & Behera, 2014).

2.4 Análisis

En cuanto a los análisis exploratorios de datos (EDA) de series temporales financieras, la literatura es más bien reducida. Uno de los trabajos a destacar es el de Al-Ameer y Fouad (2021), el cual se enfoca en la exploración de datos y técnicas de inteligencia artificial para el trading de valores y criptomonedas. Los resultados generales sugieren que debería haber un mayor enfoque en el Análisis Exploratorio de Datos (EDA) en la investigación de modelado de aprendizaje automático para aplicaciones de predicción del mercado de valores.

2.5 Aprendizaje por refuerzo

Por último, cabe destacar uno de los tópicos más recurrentes en el estado del arte: el aprendizaje por refuerzo o *reinforcement learning*. Trabajos como el de Li, Zheng & Zheng (2019) o el de Sazu (2022) exploran este campo de la inteligencia artificial cuya implementación en la industria financiera está aún en un estado muy prematuro.

3. Hipótesis

El activo escogido para el desarrollo de la estrategia de trading algorítmico con machine learning es el NASDAQ Composite, un índice bursátil que incluye acciones de todas las compañías que cotizan en el mercado de valores NASDAQ. Este índice abarca una amplia gama de empresas, especialmente aquellas en el sector tecnológico, pero también de otros sectores como salud y servicios financieros.

El problema a resolver será un problema de clasificación supervisada que cada día intentará predecir el signo de la diferencia porcentual entre el precio de apertura y el precio de cierre del NASDAQ al día siguiente. Si la señal es positiva, el precio de apertura será inferior al de cierre. En este caso, se realizaría una compra al inicio de la sesión y una venta al final de la sesión en el mismo día. Si es negativa, el precio de apertura será mayor que el de cierre, por lo que al día siguiente no se operará. Este problema puede definirse de la siguiente manera:

$$S_t \Rightarrow \begin{cases} 1 & \text{si } y_{t(o)+1} < y_{t(c)+1} \\ 0 & \text{si } y_{t(o)+1} > y_{t(c)+1} \end{cases}$$

Donde:

- S_t es la señal de compra en el tiempo t
- $y_{t(o)+1}$ es el precio de apertura en el tiempo $t + 1$
- $y_{t(c)+1}$ es el precio de cierre en el tiempo $t + 1$

En la gran mayoría de trabajos de clasificación supervisada de la literatura, la variable objetivo es el signo de la diferencia entre el precio de cierre de hoy y el precio de cierre de mañana. Esto puede servir como marco teórico, pero en la práctica es poco útil. El principal motivo es que comprar un activo al precio de cierre una vez que el mercado está cerrado es imposible. Se puede comprar el activo al precio de apertura al día siguiente, pero pudiera darse el caso de que el precio de apertura fuese superior al de cierre y perder dinero aunque el algoritmo hubiera predicho correctamente una subida entre dos precios de cierre consecutivos. Una posible solución es programar un *bot* que opere solo si el algoritmo ha dado una señal de entrada y, además, el precio de apertura fuese igual o inferior al precio de cierre del día anterior, pero esto limitaría mucho los días de operativa y por lo tanto la rentabilidad de la estrategia. Por estos motivos, la elección de la variable objetivo de este trabajo busca crear un algoritmo que sea perfectamente factible y que se pueda poner en producción en la realidad.

La hipótesis principal del trabajo es que la negociación diaria del NASDAQ Composite, al ser un índice representativo de la economía americana que aglutina decenas de empresas, se ve influenciada por factores macroeconómicos, indicadores técnicos, materias primas e índices americanos y de otras partes del mundo. La selección adecuada de características en modelos predictivos es esencial para la precisión y fiabilidad (Al-Ameer, Fouad, 2021). Además, como señalan Xiong y Zhang (2016), un único indicador técnico no puede determinar la tendencia de un activo de manera precisa, sino más bien la combinación de varios indicadores. Por estos motivos, se ha realizado una selección de varias variables que tengan una relación o impacto en la rentabilidad diaria del NASDAQ. La explicación de cada una de ellas viene detallada de la **Tabla 1**, pero aquí se ofrece un resumen de los cuatro grupos:

1. Variables de cotización diaria: precios diarios del NASDAQ, volumen de negociación y rentabilidad intradía.
2. Indicadores técnicos: medias móviles y otros indicadores, incluyendo variables de creación propia para la detección de soportes y resistencias.
3. Variables macroeconómicas: indicadores económicos que sirven como termómetro de la economía y que son tenidos en cuenta por el conjunto de inversores para su toma de decisiones. Destacan algunos conocidos indicadores adelantados como la curva de rendimiento, las expectativas del consumidor o la producción industrial.

4. Otros índices bursátiles y materias primas: índices americanos, europeos y asiáticos cuyo comportamiento está correlacionado con el del NASDAQ. También se incluyen contratos de futuro cotizados en COMEX⁴ y NYMEX⁵ de dos materias primas esenciales: oro y petróleo.

En este punto cabe mencionar que esta hipótesis asume la naturaleza determinista de los mercados financieros, por lo que rechaza la Hipótesis del Mercado Eficiente. La idea subyacente en el planteamiento de este trabajo es que existen patrones o reglas implícitas que gobiernan el comportamiento del mercado y que, si se identifican correctamente, permitirían hacer predicciones precisas sobre los precios.

La Hipótesis del Mercado Eficiente plantea que los precios de acciones se mueven de manera estocástica y son impredecibles a largo plazo (Al-Ameer, Fouad, 2021). En el caso de ser cierto, eso implicaría que es difícil y poco probable obtener ganancias consistentes al prever los movimientos de precios (Nayak, Misra & Behera, 2014). Como se detalla en la siguiente sección, uno de los objetivos de este trabajo es demostrar que el comportamiento de los mercados financieros no es completamente estocástico.

4. Objetivos

Como se ha comentado en la introducción, el principal objetivo de este trabajo es crear una estrategia de trading algorítmico que sea rentable a largo plazo. Sin embargo, existen una serie de objetivos secundarios que serán detallados a continuación:

1. Desarrollar un algoritmo que sea capaz de batir en rentabilidad a una estrategia *buy&hold*⁶ en el NASDAQ durante el mismo período de tiempo, lo que se conoce como *alpha investing*⁷. Este exceso de rentabilidad deberá ser razonable en términos de riesgo.

⁴ El Commodity Exchange, Inc. conocido como 'COMEX' es la principal bolsa de comercio de futuros de metales, tales como oro, plata, cobre y aluminio del mundo.

⁵ La New York Mercantile Exchange (NYMEX) es una bolsa de materias primas, con sede en la ciudad de Nueva York.

⁶ Una estrategia *buy&hold* consiste en adquirir un activo al principio de un determinado período de tiempo y mantenerlo en cartera sin venderlo hasta el final de ese período.

⁷ El objetivo del alpha investing es generar rendimientos que superen a los del mercado. Los gestores de fondos y los inversores buscan identificar estrategias, activos o combinaciones que ofrezcan rendimientos superiores a los del índice de referencia.

2. Crear y comparar diferentes algoritmos de clasificación supervisada y sus versiones combinadas (ensambles) para evaluar cuales son los mejores para este tipo de problemas.
3. Demostrar el impacto en la capacidad predictiva de los modelos de los diferentes pasos aplicados en el preprocesamiento de datos.
4. Probar la utilidad de la combinación de modelos de clasificación no supervisada y supervisada en problemas de predicción en mercados financieros, lo cual ha sido propuesto por Madan, Saluja y Zhao (2015) pero no llevado a cabo.
5. Entender las dinámicas que rigen los mercados financieros en el EDA.
6. Testear la utilidad de las variables creadas mediante un algoritmo de detección de soportes y resistencias.

En la siguiente sección se detalla el proceso llevado a cabo para la consecución de estos objetivos.

5. Metodología y desarrollo

El lenguaje de programación escogido para el desarrollo del proyecto es Python 3.10.12. El entorno de desarrollo en el que se ejecuta es Google Colab, una plataforma en la nube basada en los Notebooks de Jupyter. Los dos motivos principales por los que se ha escogido este entorno de desarrollo son, por un lado, la accesibilidad desde cualquier terminal al estar hospedado en la nube, y por otro lado, la posibilidad ejecutar el código en GPU⁸ y TPU⁹ para acelerar el entrenamiento de ciertos algoritmos. Estos procesadores no están disponibles en la versión gratuita de Google Colab, por lo que se ha realizado la compra de 100 unidades informáticas para emplearlas en el entrenamiento de modelos en los servidores de Google.

La base de datos persistente utilizada a lo largo de todo el proyecto es un servidor MySQL gratuito en la nube ofrecido por la página web *db4free.net*. La interfaz empleada para gestionar esta base de datos SQL es phpMyAdmin, una herramienta de software libre escrita

⁸ GPU (Unidad de Procesamiento Gráfico): Una GPU es un procesador especializado diseñado para manejar tareas relacionadas con gráficos y cálculos intensivos. En el contexto del aprendizaje automático, las GPUs son ampliamente utilizadas para acelerar el procesamiento de modelos de inteligencia artificial, ya que pueden realizar cálculos en paralelo, lo que las hace especialmente eficientes para tareas de entrenamiento.

⁹ TPU (Unidad de Procesamiento Tensorial): Una TPU es un tipo específico de procesador diseñado por Google para manejar operaciones tensoriales, que son fundamentales en el aprendizaje profundo. Las TPUs están optimizadas para acelerar el entrenamiento y la inferencia de modelos de aprendizaje automático.

en PHP que proporciona una interfaz gráfica basada en web para administrar y gestionar bases de datos MySQL. Los motivos por los cuales es necesario emplear una base de datos persistente para el desarrollo del estudio son principalmente tres. El primero es la necesidad de contar con *checkpoints* o puntos de guardado al final de cada fase, ya que de lo contrario sería necesario ejecutar todas las celdas de código cada vez que se vuelva a conectar al entorno de ejecución. El segundo es disponer de la capacidad de guardar los resultados cuando sea necesario cambiar el entorno de ejecución para conectarse a la GPU o TPU, ya que al hacer esto se pierden todas las variables locales. Por último, tener una base de datos persistente permite guardar los resultados de las predicciones y los conjuntos de datos una vez el proyecto haya finalizado. Para conectar con la base de datos y exportar o importar la información se hará uso de las librerías de Python *mysql.connector* y *sqlalchemy*, a las que se dará como input las credenciales de la base de datos.

La arquitectura implementada se puede observar en la **Figura 1**. El trabajo consta de 6 fases principales que se describen detalladamente a lo largo de esta sección. La extracción de datos, en la que se obtendrán los datos en bruto; la transformación e integración, donde se limpiarán los datos; el análisis exploratorio para estudiar la naturaleza y la relación entre las variables; la ingeniería de características en la que tendrá lugar el preprocesamiento de los datos; el entrenamiento y evaluación de los modelos y finalmente la medición de los resultados.

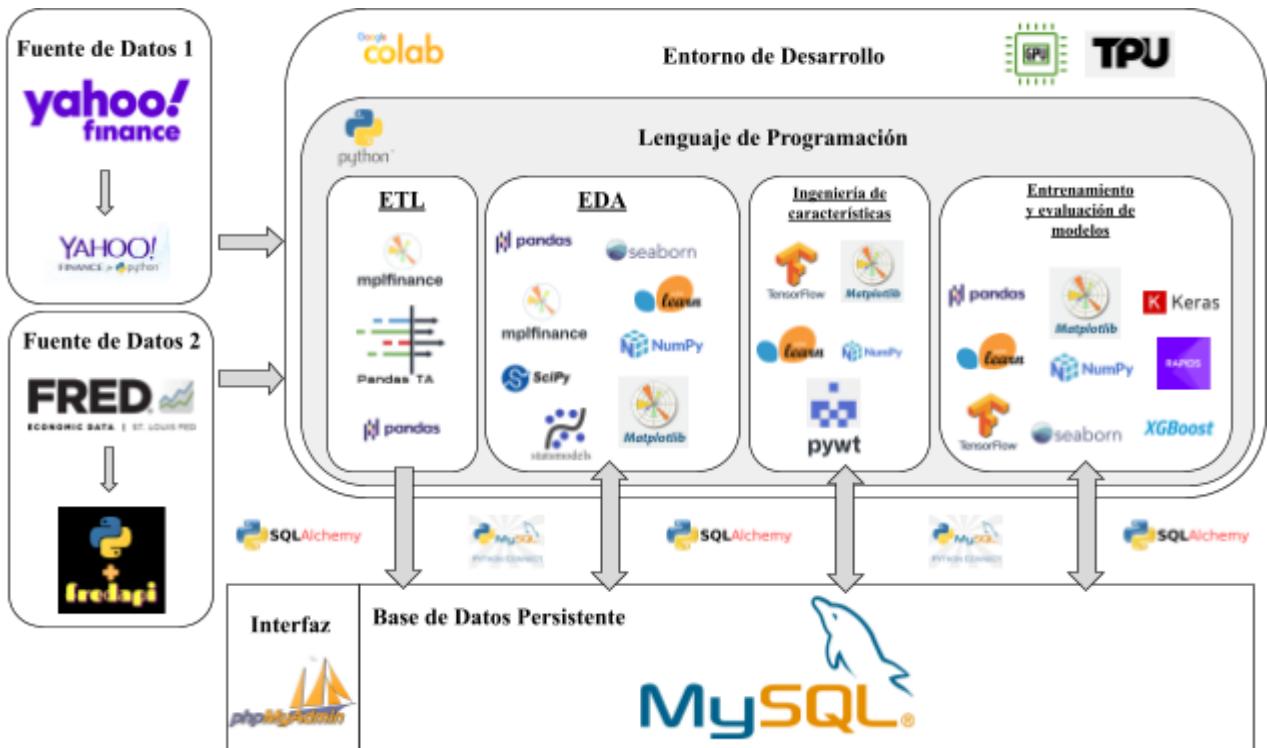


Figura 1: arquitectura del proyecto

5.1 Extracción de datos

Como se ha explicado anteriormente, las variables independientes se pueden clasificar en cuatro grupos:

1. Los precios de apertura, cierre, máximos y mínimos, conocidas como variables OHLC (del inglés Open, High, Low y Close).
2. Indicadores técnicos.
3. Variables macroeconómicas, que a su vez podrían ser subdivididas en variables macroeconómicas mensuales y diarias.
4. Cotizaciones de otros activos, incluyendo índices y futuros de *commodities* como el oro o el barril de petróleo.

La obtención del primer grupo se ha realizado mediante la biblioteca *yfinance* para obtener los datos de cotización directamente desde el portal de Yahoo!Finance. Se introdujo el ticker del NASDAQ Composite (^IXIC) y se seleccionaron los datos entre las fechas 30/09/2012 y 30/06/2023. El motivo de la selección de estas fechas era contar con 10 años de datos,

teniendo en cuenta que las 200 primeras sesiones iban a ser eliminadas al no estar disponibles para las medias móviles de 200 sesiones.

En cuanto a los indicadores técnicos, existen APIs que los ofrecen pero muchas son de pago y exigen la creación de cuentas. Como se ha visto en la hipótesis, los indicadores técnicos no son más que simples fórmulas matemáticas que se pueden obtener a partir de los datos de cotización, por lo que se ha optado por calcularlos mediante la librería *pandas_ta* y dando como input las variables OHLC y volumen obtenidas de Yahoo!Finance.

A los indicadores técnicos obtenidos se han añadido dos variables binarias para representar niveles de soporte y de resistencia. En análisis técnico, los soportes y resistencias son niveles de precios difíciles de superar para el precio, ya sea por motivos psicológicos u otros factores. Estas variables tomarán valor 1 si el precio de cierre está en un nivel de soporte o resistencia respectivamente, y 0 si no es así. El criterio escogido para determinar un nivel de soporte o resistencia es que las 4 velas consecutivas (este número se ha obtenido mediante prueba y error en un gráfico) inmediatamente anteriores y posteriores cuenten con mínimos decrecientes y máximo crecientes respectivamente en el caso de los soportes, y máximos crecientes y mínimos decrecientes respectivamente en el caso de las resistencias.

Los soportes y resistencias se representan mejor por áreas que por líneas, debido a que en estas zonas los compradores y vendedores entran en conflicto hasta que uno de los dos grupos cede ante el otro, por lo que no se tratan de líneas perfectamente definidas. Para convertirlos en áreas se ha añadido un margen de un 1% a cada lado de las líneas en el eje Y.

Además de esto, dado que la vigencia de los soportes y resistencias no es infinita y cuanto mayor sea su proximidad, mayor validez tienen, se ha limitado su longitud en el eje X a un año (aproximadamente 250 sesiones). Para evitar que los modelos cuenten con información privilegiada a la hora de entrenarse, este período de 250 sesiones se ha colocado siempre de manera posterior a la formación del soporte o resistencia. De esta manera, los modelos se entrenarán con la misma información con la que contarán en la realidad cuando se pongan en producción.

Para la obtención de las variables macroeconómicas se ha empleado la API de la FRED, un servicio web que permite a los desarrolladores escribir programas y construir aplicaciones

que recuperan datos económicos de los sitios web de FRED¹⁰ y ALFRED, alojados por la División de Investigación Económica del Banco de la Reserva Federal de St. Louis. Para su importación ha sido necesaria la creación de una cuenta y el uso de una *api key*, además de la librería de Python *fredapi*. De las variables obtenidas, el desempleo, la inflación, las expectativas del consumidor y la producción industrial son mensuales. La diferencia de la rentabilidad del bono a 10 y 2 años, la volatilidad, los tipos de interés del bono a 5 años, los tipos de interés de la Reserva Federal y la prima de riesgo de los bonos basura son variables diarias, por lo que el número de datos es mucho mayor.

Respecto al último grupo, los índices americanos S&P500, Wilshire 5000 y Dow Jones Industrial Average, el índice japonés Nikkei 225, el europeo Euro STOXX 50, el índice hongkonés Hang Seng Index y el chino Shanghai Stock Exchange, el londinense FTSE 100 y los futuros del oro y del barril de crudo han sido obtenidos de Yahoo!Finance mediante la librería *yfinance*. Nótese que la diferencia horaria no será un problema en la puesta en producción de los modelos, ya que los índices americanos son los últimos en cerrar por su zona horaria. En cuanto al NYMEX y COMEX, los mercados donde se negocian los futuros del petróleo y oro respectivamente, operan casi las 24 horas del día, pero su cierre siempre se produce antes de la apertura al día siguiente del NASDAQ 100, cuyo horario de negociación es de 9.30h a 16.00h (EST).

5.2 Transformación e integración de datos

En esta segunda fase del proyecto se han limpiado y transformado los datos en bruto y se han integrado los cuatro conjuntos de datos de la fase anterior en un único conjunto. Las variables “Dividends” y “Stock Splits” importadas automáticamente de Yahoo!Finance han sido eliminadas ya que son constantes sin ningún valor explicativo para la variable objetivo.

Para la integración se han convertido los índices de los cuatro conjuntos de datos o *dataframes* a formato fecha para evitar problemas de incompatibilidad al unirlos por el índice. El mayor reto en esta parte ha sido la diferencia en el número de filas entre cada uno de los conjuntos de datos, ya que las variables de cotización tenían 2703 registros, las

¹⁰ FRED significa Federal Reserve Economic Data. FRED contiene series temporales económicas de los Estados Unidos actualizadas con frecuencia a niveles macroeconómicos y regionales, abarcando períodos anuales, trimestrales, mensuales, semanales y diarios. FRED recopila datos económicos de diversas fuentes, la mayoría de las cuales son agencias gubernamentales de los Estados Unidos.

macroeconómicas diarias 2805 y las macroeconómicas mensuales tan solo 130. La primera unión se ha hecho eliminando las filas que no tuvieran índices coincidentes. La segunda unión se ha hecho rellenando con valores nulos todos aquellos registros para los cuales no exista un índice coincidente, evitando así acabar con un conjunto de únicamente 130 filas.

En cuanto a los valores nulos del conjunto integrado resultante, obviamente las variables macroeconómicas mensuales son las que mayor número contenían, con un total de 2619 valores nulos. Para solventarlo se ha empleado el método *forward fill*, que ha llenado todos los días de cada mes con el valor inicial de cada mes, ya que el valor que actualiza la Reserva Federal cada mes es el que queda vigente para el resto del mes y el que tienen en cuenta el conjunto de inversores a la hora de operar.

Se han eliminado las primeras 200 sesiones porque la mayoría de indicadores técnicos no cuentan con datos en las primeras filas debido a las fórmulas con las que han sido calculados. También se han eliminado todas las filas que contienen valores nulos para la variable de precio de cierre, ya que más adelante la variable objetivo será derivada de esta variable.

Para evitar perder más información y quedar con un conjunto de datos insuficiente, se ha decidido llenar los valores nulos que aún quedaban tras estas modificaciones en vez de eliminarlos. La interpolación lineal ha sido el método escogido para llenar los valores nulos dada su efectividad en series temporales frente a otros métodos como KNN.

La interpolación lineal implica la creación de una línea recta entre dos puntos conocidos y estimar valores intermedios a lo largo de esa línea. La fórmula básica para la interpolación lineal entre dos puntos $(x_1, f(x_1))$ y $(x_2, f(x_2))$ para un punto intermedio x se puede expresar como:

$$f(x|x_1; x_2) = f(x_1) + \frac{f(x_2) - f(x_1)}{(x_2 - x_1)}(x - x_1)$$

Donde:

- x es el valor temporal del valor nulo en el índice

- x_1 es el valor temporal del valor no-nulo inmediatamente anterior en el índice
- x_2 es el valor temporal del valor no-nulo inmediatamente posterior en el índice
- $f(x_1)$ es el valor no-nulo inmediatamente anterior
- $f(x_2)$ es el valor no-nulo inmediatamente posterior
- $f(x|x_1; x_2)$ es el valor que sustituirá al valor nulo.

Una vez limpiados los datos, integrados en un único conjunto y eliminados todos los valores nulos, el *dataframe* resultante consta de 46 variables excluyendo el índice de fechas y 2504 registros.

5.3 Análisis exploratorio de datos

Antes de adentrarse en el preprocesamiento de datos y la elección de algoritmos, es imperativo comprender a fondo la naturaleza de las variables involucradas y las relaciones entre ellas. Este paso no solo ofrecerá una estrategia para mejorar la capacidad predictiva de los modelos, sino que también contribuirá significativamente al crecimiento del conocimiento en el ámbito de la macroeconomía financiera.

En consonancia con esta necesidad de comprensión profunda, los resultados generales de investigaciones recientes enfatizan la importancia de centrarse más en el Análisis Exploratorio de Datos (EDA), un paso que suele ser omitido en la mayor parte de los trabajos que conforman la literatura de este campo. Según Al-Ameer y Fouad (2021), por ejemplo, existen claros indicios de que deberíamos otorgar una mayor atención al EDA en la investigación de modelos de aprendizaje automático aplicados a la predicción del mercado de valores.

Algunos análisis de datos llevados a cabo demuestran que la clasificación en la predicción del mercado de valores, al utilizar indicadores técnicos como características, es compleja y requiere modelos capaces de abordar tal complejidad (Al-Ameer, Fouad, 2021). Se destacan propiedades fundamentales para tales modelos, incluyendo la no linealidad y la capacidad de estimar distribuciones estadísticas complejas. En este contexto, la necesidad de un análisis exhaustivo de datos se vuelve aún más evidente.

Otros trabajos apuntan a que el campo de investigación en el mercado de valores se ha perfilado como dinámico, no lineal, complicado, no paramétrico y caótico por naturaleza (Sheta, Ahmed & Faris, 2015). Prever su comportamiento no es una tarea trivial y depende del descubrimiento de fuertes regularidades empíricas en las observaciones del sistema. Estas regularidades, como señalan Nayak, Misra & Behera (2014), a menudo se ven oscurecidas por el ruido, y las series temporales financieras suelen exhibir un comportamiento no lineal y no estacionario.

En las siguientes secciones se describirán las distintas partes que componen el Análisis Exploratorio de Datos. Cada una de estas etapas busca extraer conclusiones significativas sobre las características de los datos, proporcionando así la base necesaria para tomar decisiones fundamentadas durante el preprocesamiento de los datos y la elección de algoritmos.

5.3.1 Resumen estadístico inicial

Al principio se ha llevado a cabo una exploración inicial sobre los datos para conocer sus características más generales. Es aquí donde se ha creado una variable con la rentabilidad porcentual de cada día:

$$\frac{\text{Precio de Cierre}_t - \text{Precio de Apertura}_t}{\text{Precio de Apertura}_t}$$

Donde t es una determinada sesión, y la variable objetivo, que toma valor 1 si la diferencia porcentual entre el precio de apertura del día siguiente y el precio de cierre del día siguiente va a ser positiva, y 0 si no es así.

Se ha escogido crear una variable de rentabilidad porcentual y no una variable de diferencia absoluta ya que estas diferencias van en aumento junto con el aumento del valor del NASDAQ, por lo que si queremos comparar diferentes períodos de manera equitativa, esta es una solución.

Del resumen estadístico obtenido con el método *describe* se extraen algunos datos llamativos, como que el futuro del barril de petróleo ha llegado a estar en negativo, hecho que se produjo

en 2020 durante la pandemia. Un 17.30% de las veces el precio del NASDAQ ha cerrado en una zona de soporte, mientras que un 17.62% lo ha hecho en una zona de resistencia.

La rentabilidad diaria media del NASDAQ es de un 0.03%. La mayoría de las veces ha estado entre un -0.44% (cuartil 1) y un 0.57% (cuartil 3), aunque ha alcanzado un máximo de 7.04% y un mínimo de -6.60%. Un 54.29% de las veces el precio ha cerrado por encima de su apertura diaria, mientras que un 45.71% ha cerrado en negativo.

Atendiendo a la definición de valor atípico basada en el rango intercuartílico, la distancia entre los máximos y mínimos y los cuartiles sugieren la existencia de valores atípicos en muchas de las variables. Esto será discutido en la siguiente sección.

5.3.2 Valores atípicos

Para corroborar la existencia de los posibles valores atípicos observados en el resumen estadístico, se ha graficado una matriz de Box Plots de cada una de las variables *float* del conjunto de datos, la cual se puede observar en la **Figura 2** en el apéndice.

La abundancia de valores atípicos ha sido confirmada tanto en esta figura como en su contabilización variable por variable empleando la definición matemática de valor atípico basada en el rango intercuartílico. Se ha tomado la decisión de no eliminar los valores atípicos por los siguientes motivos:

- La eliminación de estos valores atípicos supondría una pérdida de información demasiado grande, ya que habría que eliminar cientos de registros.
- Estos valores atípicos no parecen deberse a errores de medición.
- Los valores atípicos pueden contener información valiosa sobre eventos inusuales o eventos extremos en los datos financieros. Estos eventos pueden ser de interés, ya que podrían indicar cambios significativos en el mercado, noticias importantes o eventos inesperados que pueden tener un impacto en nuestra estrategia de inversión.
- Los valores atípicos son parte de la naturaleza de los datos financieros. Los mercados financieros son volátiles y están sujetos a cambios rápidos e inesperados. Eliminar todos los valores atípicos podría conducir a una representación irrealista de los datos y ocultar la verdadera naturaleza del mercado.

- En algunos casos, los modelos financieros pueden beneficiarse de la inclusión de valores atípicos. Los modelos robustos pueden ser capaces de manejar valores atípicos sin que esto afecte significativamente sus predicciones. Además, los valores atípicos pueden ayudar a detectar cambios estructurales en los datos.
- La eliminación de valores atípicos puede distorsionar las métricas y los indicadores que usaremos para los modelos.

Como consecuencia de esta decisión, será necesario emplear un método de normalización que sea robusto frente a valores atípicos. También se tratará de mitigar parcialmente los efectos de los valores atípicos, sin proceder a su eliminación directa, mediante técnicas de reducción de ruido en ingeniería de características.

5.3.3 Normalidad

De acuerdo con Al-Ameer y Fouad (2021), es importante entender la distribución estadística de las clases y de las características antes de construir los modelos, de lo contrario estos pueden acabar teniendo un desempeño adverso.

Para comprobar si las características siguen una distribución normal, se ha graficado su distribución indicando su curtosis y asimetría en la leyenda (**Figuras 3.1 y 3.2**).

Teóricamente, una distribución normal estándar debería ser mesocúrtica (curtosis igual a 3) y simétrica (asimetría igual a 0), por lo que obtener estas características puede ayudar a dilucidar su distribución. A excepción de algunas pocas variables que parecen aproximarse a estos valores, la mayoría no muestran una distribución normal a simple vista.

También se ha graficado un Q-Q plot para testear visualmente la hipótesis nula de similitud con una distribución normal (**Figura 4**), y finalmente se ha realizado la prueba de Shapiro-Wilk (**Tablas 2.1 y 2.2**). La prueba de Shapiro-Wilk es una de las pruebas más poderosas para verificar la normalidad de una muestra. Es ampliamente recomendada debido a su mayor poder para detectar desviaciones de la normalidad en comparación con otros tests de normalidad. La hipótesis nula (H_0) es que los datos se distribuyen normalmente. La hipótesis alternativa (H_1) es que los datos no se distribuyen normalmente. La prueba calcula una estadística de prueba (W) y un valor p , el cual determina el rechazo de la hipótesis nula para un nivel de significancia escogido, en este caso 0.05.

Tanto de las desviaciones de puntos de la diagonal en los Q-Q plots como de los p-values, cuyo valor no supera el umbral de significancia establecido en ninguna de las variables, se puede inferir que ninguna de las variables del conjunto de datos sigue una distribución normal. Habiendo confirmado esto, será necesario emplear algoritmos robustos a la no-normalidad y no aplicar métodos de normalización en la fase de preprocesamiento que asuman la distribución normal de las variables.

5.3.4 Correlación

Medir la correlación es un paso útil para explorar la relación entre las variables y la variable objetivo, pero no debe considerarse como la única medida de capacidad predictiva, ya que la correlación no necesariamente implica causalidad.

En esta sección se ha impreso un mapa de calor para mostrar una matriz de correlación con todas las variables del conjunto de datos (**Figura 5**). La fórmula empleada es el coeficiente de correlación de Pearson:

$$r_{xy} = \frac{\sum_{i=1}^n (x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^n (x_i - \bar{x})^2} \sqrt{\sum_{i=1}^n (y_i - \bar{y})^2}}$$

Donde:

- n es el tamaño de la muestra.
- x_i, y_i son puntos muestrales individuales indexados con i de cada una de las dos variables.
- \bar{x}, \bar{y} denotan las medias muestrales.

También se han obtenido los 15 pares de variables más positiva y negativamente correlacionados (**Tablas 3 y 4**). En la **Figura 6** hay una matriz con gráficos de dispersión temporal. De estas tablas y de la figura anterior se pueden extraer las siguientes conclusiones:

- En general, y como era de esperar, las variables OHLC y las medias móviles guardan una alta correlación entre sí.
- A excepción del Hang Seng Index, los demás índices están altamente correlacionados con las variables OHLC del NASDAQ.

- El futuro del oro muestra una fuerte correlación positiva con el NASDAQ, mientras que el futuro del barril de petróleo muestra una correlación muy tenue.
- 10y-2y yield curve y federal interest rate tiene una fuerte correlación negativa, ya que cuanto más suben los tipos de interés del banco central, mayor es la rentabilidad del bono a 2 años frente al bono a 10 años. Esto se debe a las expectativas del mercado sobre las tasas de interés futuras. Cuando se espera que las tasas suban, la curva de rendimiento puede invertirse, y esto afecta especialmente a los rendimientos de los bonos a corto y largo plazo.
- Un factor llamativo es que tanto la volatilidad como la prima de riesgo del bono basura son medidas del miedo que tiene el mercado en un determinado momento. Sin embargo, no hay una correlación muy fuerte entre ambas variables.
- Tanto los tipos de interés del banco central como el tipo de interés del bono a 5 años muestran una correlación positiva con la inflación, y negativa con el desempleo. Esto se debe a que los bancos centrales suelen subir los tipos de interés para contrarrestar las subidas de la inflación, mientras que los suelen bajar como respuesta a datos negativos en el mercado de trabajo.
- La producción industrial y el desempleo muestran la correlación más negativa de toda la matriz. Esto es obvio, ya que a menor producción del tejido industrial, mayor será el desempleo.
- Las expectativas del consumidor están negativamente correlacionadas con el oro, volumen, NASDAQ, S&P500 e inflación. Esta última puede deberse a que un entorno inflacionario es percibido como perjuicial para la economía. El resto pueden explicarse debido a la naturaleza de indicador adelantado de las expectativas del consumidor, al igual que ocurre con el indicador 10y-2y yield curve.
- Los precios de cierre del NASDAQ y del S&P500 muestran una correlación positiva casi perfecta en el gráfico de dispersión. El valor de ambos índices ha ido en aumento desde 2013 hasta 2023.
- Como se puede observar, el volumen de negociación se ha ido incrementando con los años, mientras que las expectativas del consumidor han ido decreciendo desde 2018 hasta 2022. Las expectativas del consumidor suelen adelantarse a cambios de tendencia en el NASDAQ. Cuando este indicador está bajando el volumen de negociación sigue en aumento debido a que aún no ha llegado al techo. Cuando el precio empieza a caer y el volumen de negociación baja las expectativas del

consumidor suben augurando que el final de la recesión tendrá lugar en el futuro cercano.

- El DJIA ha ido en aumento desde 2014 a 2022, mientras que la diferencia entre la rentabilidad del bono a 10 y 2 años ha estado en negativo durante todo el 2022, augurando la proximidad de una posible recesión económica.
- La bolsa londinense no muestra un crecimiento tan claro como las americanas. Por otro lado, el spread de los bonos basura alcanzará máximos en 2020, señalando el pico de pánico alcanzado durante la pandemia. De la correlación negativa se puede deducir que la bolsa londinense no responde bien en situaciones de estrés financiero.

Por último, se han ordenado de mayor a menor correlación positiva en un gráfico de barras todas las variables contra la variable de rentabilidad intradía, la cual se ha usado como proxy de la variable objetivo binaria (**Figura 7**). A excepción de algunos indicadores técnicos, no parecen existir correlaciones significativas, lo que será una dificultad para los modelos a la hora de predecir.

5.3.5 Heterocedasticidad

Como análisis complementario se ha testeado la varianza de los errores de un modelo de regresión. La variable objetivo para la regresión es la rentabilidad intradía. El objetivo de este proyecto es resolver un problema de clasificación, pero como muestra para futuros trabajos sobre la materia se ha comprobado la complejidad y dificultad que supondría realizar un modelo de regresión en vez de uno de clasificación sobre estos datos.

La prueba de Breusch-Pagan es una prueba estadística utilizada para evaluar la presencia de heterocedasticidad en un modelo de regresión. La heterocedasticidad se refiere a la variabilidad no constante de los errores en un modelo de regresión, lo que significa que la varianza de los errores no es constante a lo largo de los valores de las variables independientes. Esto puede ser problemático ya que viola una de las suposiciones clave de muchos modelos de regresión lineal, que asume una varianza constante de los errores (homocedasticidad).

Los dos estadísticos de la prueba son inferiores al nivel de significación de 5%, por lo que se rechaza la homocedasticidad como hipótesis nula. Esto es una muestra más de la compleja naturaleza de los datos que estamos tratando y la dificultad de su predicción tanto en

algoritmos de clasificación (demostrado en el resto de apartados) como en algoritmos de regresión.

5.3.6 Interdependencia temporal

Para medir la interdependencia temporal de la variable objetivo se ha utilizado el siguiente gráfico para mostrar la autocorrelación del *lag* 0 al *lag* 30:

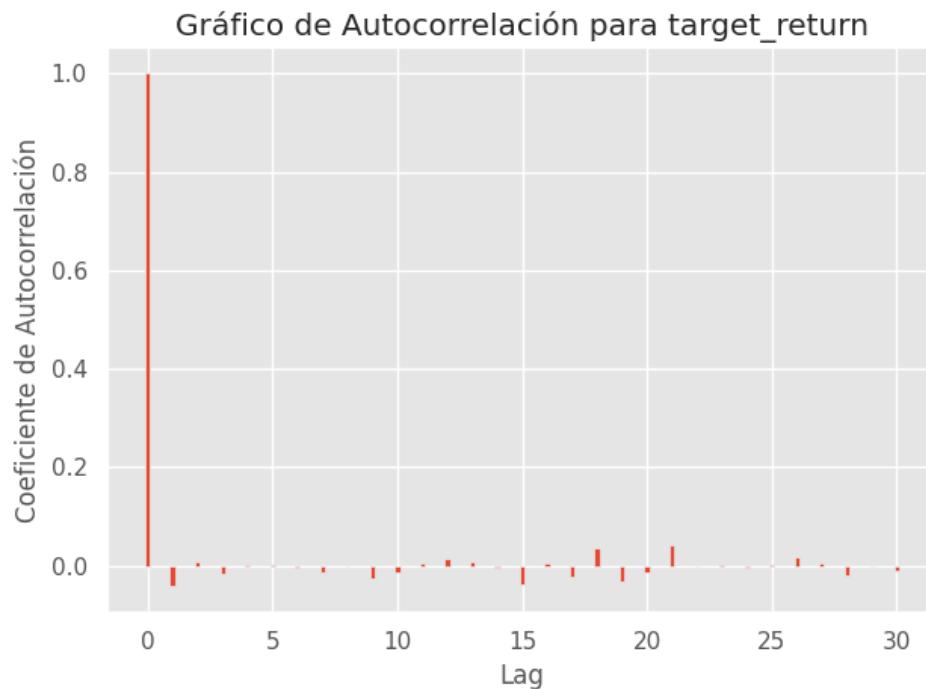


Figura 8: gráfico de autocorrelación de la variable objetivo

La autocorrelación es una medida estadística que evalúa la relación lineal entre una serie de datos y una versión retrasada de sí misma. Conceptualmente su objetivo es comprobar si el presente y futuro dependen del pasado. La fórmula empleada es la siguiente:

$$R(k) = \frac{E[(X_i - \mu)(X_{i-k} - \mu)]}{\sigma^2}$$

Donde:

- $R(k)$ es la autocorrelación en el rezago k .
- E es el valor esperado.
- x_i es el valor de la serie temporal en el tiempo i .

- x_{i-k} es el valor de la serie temporal en el tiempo $i - k$.
- μ es la media de la serie temporal.
- σ^2 es la varianza.

No parece haber signos de una autocorrelación fuerte en el gráfico de barras. Para asegurarse de esto se ha realizado la Prueba de Ljung-Box. Esta prueba se utiliza para evaluar si hay autocorrelación significativa en diferentes rezagos (*lags*) de una serie temporal. La hipótesis nula establece que los datos se distribuyen de forma independiente, es decir, las correlaciones en la población de la que se toma la muestra son 0, de modo que cualquier correlación observada en los datos es el resultado de la aleatoriedad del proceso de muestreo. La interpretación se basa en el p-valor. Si el p-valor es menor que un umbral de significación escogido (en este caso 0.05), se puede rechazar la hipótesis nula y por lo tanto concluir que hay evidencia de autocorrelación en la serie temporal.

En el *lag 1* el p-value es 0.039, mientras que en el resto de *lags* el p-value es superior a 0.05 (**Tabla 5**). Esto prueba que existe autocorrelación de primer orden (*lag 1*) y que por tanto sería interesante entrenar un modelo capaz de captar patrones intertemporales en los datos.

5.3.7 Separabilidad lineal

Dado que el problema a resolver es de clasificación, se ha comprobado si los datos son linealmente separables mediante la evaluación de una máquina de soporte vectorial de margen duro o *Hard Margin Support Vector Machine* (Al-Ameer, Fouad, 2021).

El término "márgen duro" o "*Hard Margin*" se refiere a la naturaleza estricta de esta SVM, que busca encontrar un hiperplano de separación que separe perfectamente las muestras de dos clases en un espacio de características. En otras palabras, busca un hiperplano que tenga un margen máximo (la distancia entre el hiperplano y el punto de datos más cercano) y que no permita ningún punto de datos en el margen.

La capacidad de la SVM para encontrar un hiperplano de separación perfecta o casi perfecta puede indicar si las clases (en este caso el signo de la rentabilidad del día siguiente) son fácilmente distinguibles en los datos. La Hard Margin SVM asume que los datos son linealmente separables, por lo que unos resultados pobres en la evaluación indicarían que los datos no son linealmente separables y que una Soft Margin SVM o métodos más flexibles de

aprendizaje automático, como las redes neuronales, pueden ser más apropiados para resolver este problema de clasificación.

El kernel empleado en la Hard Margin SVM es lineal. El hiperparámetro de penalización C se ha establecido en $1e10$, un valor próximo al infinito, para forzar la clasificación de margen duro. Si la precisión del modelo es 100%, se podría afirmar que los datos son linealmente separables.

Tras la evaluación del modelo, la precisión arrojada no solo es de un 49%, sino que además el modelo no llega a converger ni con un millón de iteraciones. De estos resultados podemos concluir que el SVM lineal de margen duro no aporta valor predictivo y que las clases no son perfectamente separables, siendo este un problema de clasificación más complejo y que requerirá de algoritmos más sofisticados o flexibles para ser resuelto.

5.3.8 Estacionalidad

En esta sección se han graficado las series temporales para observar a simple vista si hay indicios de estacionalidad en alguna de las variables (**Figuras 8 y 9**). Dado que el exceso de ruido en muchas de las variables dificulta determinar si hay patrones cíclicos que se repiten con cierta periodicidad, también se ha hecho la Prueba de Dickey-Fuller Aumentada para testear la estacionalidad (**Tablas 6.1 y 6.2**).

Esta prueba se basa en la idea de que una serie de tiempo no estacionaria se puede transformar en una serie estacionaria a través de diferenciación. La diferenciación implica calcular la diferencia entre valores sucesivos en la serie temporal. La prueba ADF compara una serie original con su versión diferenciada para evaluar si la diferencia es estadísticamente significativa. La hipótesis nula (H_0) de la prueba es que la serie de tiempo tiene raíces unitarias, lo que indica no estacionariedad. La hipótesis alternativa (H_1) es que la serie de tiempo es estacionaria. Si el p-value es menor o igual a un nivel de significación predefinido de 0.05, se rechazará la hipótesis nula y se concluirá que la serie es estacionaria.

De acuerdo a los resultados de la prueba, abundan tanto las series estacionarias como no estacionarias. La variable objetivo no es analizada debido a su naturaleza de variable binaria,

pero utilizando la variable de la rentabilidad intradía como *proxy* se puede inferir que las rentabilidades diarias del NASDAQ son estacionarias.

La **Figura 10** contiene un box plot mensual para observar más de cerca esta variable. Los boxplots mensuales son útiles para visualizar la distribución de los datos en diferentes meses, lo que puede ayudar a detectar patrones estacionales. El número de outliers es similar en todos los meses. Existen ligeros cambios en la mediana y en la distribución de los cuartiles. Visualmente la estacionalidad no es muy pronunciada, pero con la prueba ADF se puede confirmar la estacionalidad de la rentabilidad intradía.

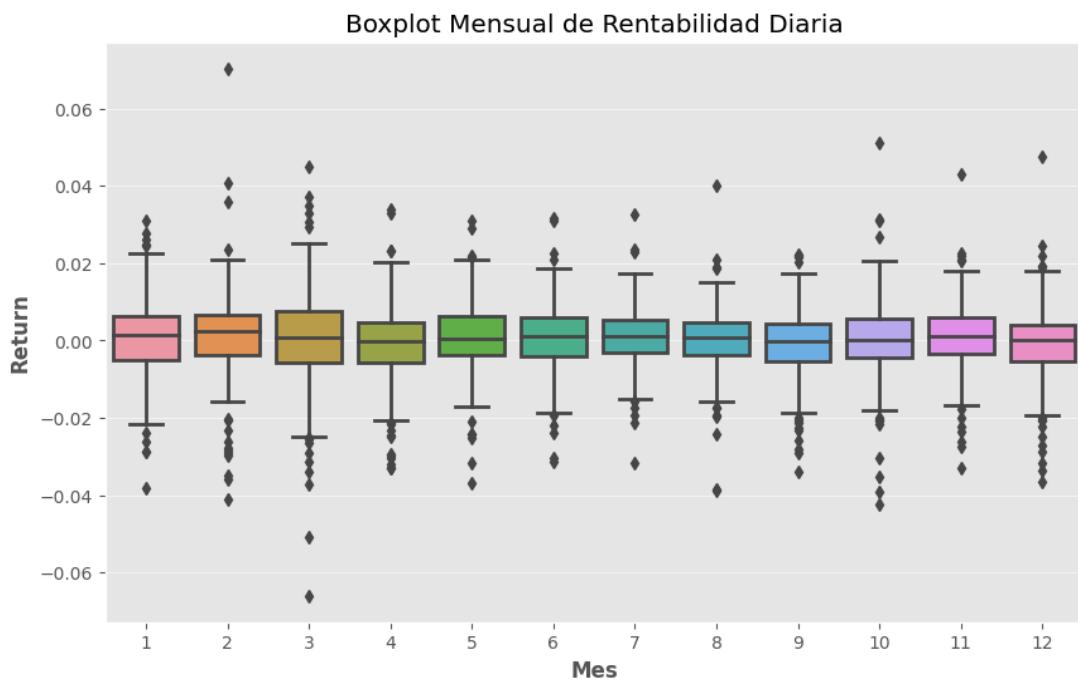


Figura 10: Box Plot mensual de la rentabilidad intradía.

5.3.9 Métricas de riesgo y rentabilidad

En la última sección del EDA se calculan dos métricas de riesgo y dos métricas de rentabilidad ajustada al riesgo de una estrategia que hubiera consistido en comprar y mantener (*Buy&Hold*) el NASDAQ 100 durante los 10 años de datos (desde el 19/7/2013 hasta el 28/6/2023):

- Value at Risk (VaR): es una medida estadística utilizada para estimar las pérdidas potenciales en una inversión o cartera de inversiones en un período de tiempo y con un cierto nivel de confianza.

$$\text{VaR}_\alpha(X) = -\inf\{x \in \mathbb{R} : F_X(x) > \alpha\} = F_Y^{-1}(1 - \alpha)$$

A un nivel de confianza del 95%, el VaR es de -2.14%. Esto significa que existe un 95% de probabilidad de que las pérdidas no superen el -2.14% de la inversión en un día.

- Maximum Drawdown: es una medida que cuantifica la mayor pérdida desde el pico más alto de un valor hasta el punto más bajo antes de comenzar a recuperarse.

$$\text{MDD}(T) = \max_{\tau \in (0, T)} D(\tau) = \max_{\tau \in (0, T)} \left[\max_{t \in (0, \tau)} X(t) - X(\tau) \right]$$

Si se hubiera operado con una estrategia Buy&Hold durante los 10 años la mayor caída hubiera sido de un -36.40%.

- Rendimiento Anual: es el porcentaje de ganancia o pérdida experimentado por una inversión o cartera en un año.

$$RA = (\prod_{i=1}^n (1 + R_i))^{\frac{1}{T}} - 1$$

La rentabilidad anual del NASDAQ desde 2013 hasta 2023 ha sido del 12.87%.

- Sharpe Ratio¹¹: es una medida que evalúa el rendimiento de una inversión en relación con su riesgo. Compara el rendimiento adicional obtenido más allá de una inversión sin riesgo con la volatilidad de la inversión.

$$S = \frac{\mathbb{E}[R - R_f]}{\sigma}$$

El Sharpe Ratio del NASDAQ ha sido 6.25.

- Sortino Ratio: es una métrica similar al Sharpe Ratio, pero se enfoca en la volatilidad negativa en lugar de la volatilidad total. Mide el rendimiento de una inversión en

¹¹ Como rentabilidad del activo libre de riesgo se ha escogido la rentabilidad del bono americano a 10 años (US10YT) el 13 de octubre de 2023: 4.62%.

relación con su riesgo negativo, lo que lo hace especialmente relevante para inversiones donde la reducción del riesgo a la baja es crítica.

$$S = \frac{R - T}{DR} \quad DR = \sqrt{\int_{-\infty}^T (T - r)^2 f(r) dr}$$

El Sortino Ratio del NASDAQ ha sido 7.80

Por último, para dar una idea de los máximos y mínimos alcanzados durante estos 10 años, se ofrecen los siguientes datos:

- La mayor bajada se produjo el 16 de marzo de 2020, en plena pandemia, con una caída del 6.6%. La segunda, también en marzo de 2020, fue del 5.08%. El resto de pérdidas son algo inferiores y están repartidas de manera heterogénea en varios años.
- En cuanto a la mayor ganancia diaria, esta se produjo el 24 de febrero de 2022, con una subida del 7.03%. El resto de ganancias también están repartidas de manera heterogénea, y son algo inferiores.

5.3.10 Conclusiones del EDA

El Análisis Exploratorio de Datos revela varias características clave sobre el conjunto de datos que requieren una consideración cuidadosa. A continuación, se resumen las principales conclusiones extraídas del EDA:

1. Valores Atípicos y Ruido: El análisis de los datos muestra la presencia de valores atípicos en varias de las variables. Estos valores atípicos pueden introducir ruido en el análisis, por lo que será necesario usar métodos de normalización robustos a los valores atípicos y técnicas de reducción de ruido en el preprocesamiento de datos.
2. No-Normalidad de las Variables: La mayoría de las variables analizadas no siguen una distribución normal. Esto se evidencia a través de pruebas de normalidad como el test de Shapiro-Wilk. La falta de normalidad puede afectar la aplicación de algunas técnicas estadísticas que asumen una distribución normal en los datos.
3. Baja Correlación con la Variable Dependiente: Se observa una correlación baja entre las variables independientes y la variable dependiente. Esto sugiere que las variables

independientes no están fuertemente relacionadas con la variable objetivo y pueden suponer un desafío a la hora de construir un modelo predictivo preciso.

4. Presencia de Heterocedasticidad: El análisis de las residuales sugiere la presencia de heterocedasticidad en los datos. La varianza de los errores no es constante en todo el rango de valores de la variable dependiente, lo que plantea problemas para los modelos de regresión.
5. Autocorrelación en la Variable Objetivo: Se ha detectado la presencia de autocorrelación en la variable objetivo en la diferencia de primer orden. Esto indica que los valores sucesivos de la variable objetivo están correlacionados.
6. No Linealmente Separable: Los datos no son linealmente separables, lo que significa que no se puede trazar un límite de decisión lineal claro para clasificar los datos. Esto implica que será necesario emplear algoritmos de clasificación más sofisticados que sean robustos a datos no linealmente separables.
7. Estacionalidad: Se ha encontrado evidencia tanto de no estacionalidad como de estacionalidad en los datos. La presencia de estacionalidad en la variable 'return' como proxy de la variable objetivo sugiere la estacionalidad de la variable objetivo. La falta de estacionalidad en otras variables indica que el patrón de los datos puede variar con el tiempo.

En resumen, los datos presentan una serie de desafíos y complejidades que deben abordarse de manera adecuada. Dadas las características del conjunto de datos, se propone utilizar el método de normalización "Robust Scaler" para manejar los valores atípicos. En cuanto a los algoritmos de clasificación, su elección en base a estos hallazgos se justificará en las siguientes secciones.

5.4 Ingeniería de características

El preprocesamiento es una parte esencial de los proyectos de ciencia de datos. Los datos en el mundo real suelen ser sucios y ruidosos. Esto resulta en datos de baja calidad, lo que a su vez conduce a resultados deficientes (Kumar, Kalia, 2012).

El preprocesamiento de datos de este proyecto está inspirado por el trabajo realizado por Bao, Yue y Rao (2017), pero con algunas variaciones: el primer paso ha sido la división del

conjunto de datos en train, dev y test para evitar el data leakage, se ha añadido un paso de normalización de los datos entre la transformada wavelet y los stacked autoencoders y, además, se ha probado la efectividad de este preprocesamiento en otros algoritmos distintos aparte de LSTM. Las decisiones llevadas a cabo en este preprocesamiento se han fundamentado en los resultados obtenidos en el EDA, los cuales serán comentados en la sección de resultados. A continuación se describen las cinco fases en las que se ha dividido el preprocesamiento:

5.4.1 División de los datos

Antes de aplicar ninguna técnica de preprocesamiento, se han dividido los datos en entrenamiento (*train*, 70%), validación (*dev*, 10%) y prueba (*test*, 20%) para evitar cualquier tipo de contaminación en los datos o *data leakage*.

El *data leakage* es un fenómeno en el que se produce una fuga de información del conjunto de prueba en el conjunto de entrenamiento, lo que puede conducir a una evaluación incorrecta del rendimiento del modelo. En otras palabras, se produce cuando la información del conjunto de datos de entrenamiento incluye accidental o intencionalmente detalles sobre la variable objetivo que no estarían disponibles en un entorno de producción. Esto llevaría a una evaluación demasiado optimista en comparación a los resultados que el modelo obtendrá en la realidad, ya que el modelo contó con información privilegiada a la hora de entrenarse.

El conjunto de entrenamiento servirá para entrenar los modelos. El conjunto de validación se empleará para la búsqueda de hiperparámetros óptimos con técnicas de validación cruzada. Esto ahorrará tiempo y recursos computacionales, ya que el conjunto de validación es más pequeño que el de entrenamiento y al mismo tiempo nos servirá para generalizar en este último conjunto. Este conjunto también servirá como conjunto de validación en el entrenamiento de las redes neuronales LSTM. Por último, el conjunto de prueba servirá para testear la precisión de los modelos. En este punto también se divide la variable objetivo y del resto de variables.

La división viene ilustrada en la **Figura 11**. El período de entrenamiento va desde 19/7/2013 a 2/7/2020, el de validación de 6/7/2020 a 30/6/2021 y el de prueba de 1/7/2021 a 28/6/2023. Como se puede observar, la crisis de marzo de 2020 cae dentro del conjunto de

entrenamiento, lo cual puede ser beneficioso para los modelos al estar entrenados en períodos de alta volatilidad y caídas.



Figura 11: división de los datos en train, dev y test ilustrada en la variable de precio de cierre.

5.4.2 Wavelet transform

Existen dos tipos de información en el conjunto de datos: señal y ruido. La señal se asemeja a la información útil y generalizable en los datos, mientras que el ruido representa las fluctuaciones aleatorias o patrones incidentales que no contribuyen significativamente a la capacidad del modelo para realizar predicciones en nuevos datos.

En el ámbito financiero, la mayoría de los datos presentan un alto nivel de ruido e inestabilidad. La presencia de este alto nivel de ruido en los datos financieros originales puede llevar a que las redes neuronales profundas entrenadas con dichos datos no logren predecir de manera precisa el precio de los activos (Liang et al., 2019).

A diferencia de Nayak, Misra & Behera (2014), donde se emplea una media móvil simple de cinco días, en este trabajo se ha empleado la transformada wavelet como método de suavización y reducción de ruido. El motivo de esta elección es el retraso que tienen las medias móviles en relación con los datos reales, el cual no se produce con la transformada wavelet.

La transformada wavelet es una técnica utilizada en el procesamiento de señales que permite descomponer una señal o conjunto de datos en componentes que representan diferentes niveles de detalle. Cuando se aplica la transformada wavelet a los datos, se obtienen coeficientes wavelet que representan la contribución de diferentes frecuencias o escalas en la señal original. Estos coeficientes se dividen en aproximación (componentes de baja frecuencia) y detalle (componentes de alta frecuencia). La aproximación captura las características generales y suavizadas de la señal, mientras que los detalles resaltan las variaciones finas y cambios rápidos. Para aplicar esta técnica se ha importado la librería *pywt*, que realiza la transformada wavelet discreta, cuya fórmula es la siguiente:

$$DWT[f](a, b) = \langle f, \psi_{a,b} \rangle$$

Donde:

- f es la señal de entrada, en este caso las variables con ruido.
- $\psi_{a,b}$ es la ondícula madre escalada y desplazada.
- a es el factor de escala.
- b es el factor de traslación.
- $\langle f, \psi_{a,b} \rangle$ representa el producto interno entre la señal f y la ondícula madre escalada y desplazada.

Las series temporales se han descompuesto en ondículas y se han eliminado los 3 componentes de mayor frecuencia para suavizar la señal, eliminando así el ruido (ondículas de alta frecuencia). La elección de eliminar los 3 componentes de mayor frecuencia se ha realizado mediante un proceso iterativo de prueba y error en la variable de precio de cierre durante el entrenamiento. Esta elección se ha generalizado al resto de variables y al conjunto de prueba para garantizar consistencia con el entrenamiento. Este ajuste se ha llevado a cabo solo en el conjunto de entrenamiento para evitar *data leakage*.

En cuanto a la elección del tipo de transformada wavelet, se optó por la transformada wavelet de Daubechies (*sym5*) en lugar de la transformada wavelet de Haar, utilizada en Liang et al. (2019). La razón principal es que las wavelets de Daubechies, como '*sym5*', suelen ser más apropiadas debido a su capacidad para representar y analizar eficientemente las fluctuaciones de alta frecuencia, reducir el ruido y preservar las tendencias en los datos. Esto se comprobó

mediante la visualización de la Figura 2, en la que se observó que la reconstrucción de la señal se ajustaba mejor a los datos originales que la función de Haar.

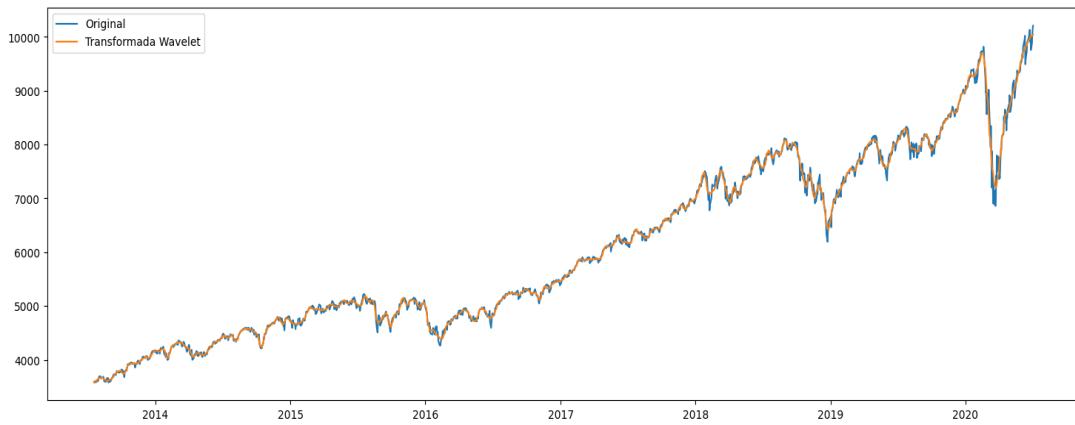


Figura 12: variable de precio de cierre en azul y su señal recomuesta por la transformada wavelet superpuesta en naranja.

5.4.3 Normalización

Debido a la diferencia de magnitudes de las variables de nuestro conjunto de datos, ha sido necesario realizar un escalado de los datos mediante normalización. La normalización es una transformación que reduce la escala de los datos (Kumar, Kalia, 2012). Puede utilizarse para ajustar los datos al mismo rango de valores para cada característica de entrada con el fin de minimizar el sesgo dentro de la red neural de una característica a otra. La normalización de datos también puede acelerar el tiempo de entrenamiento al iniciar el proceso de entrenamiento para cada característica dentro del mismo rango (Nayak, Misra & Behera, 2014). Por estos motivos se ha decidido implementar este paso antes de la transformación con stacked autoencoders, ya que estos son redes neuronales.

Existen varios métodos de normalización. La elección de usar Robust Scaler se ha fundamentado en el hallazgo de outliers en el EDA. Ante la decisión de no eliminarlos, ha sido necesario el uso de un método de normalización que fuera robusto frente a valores atípicos. Este es el caso de Robust Scaler, ya que utiliza los cuartiles en lugar de la media y la desviación estándar, empleadas por otros métodos de normalización. Los cuartiles dividen los datos en cuatro partes iguales, lo que lo hace menos sensible a los valores extremos que pueden afectar significativamente a la media y la desviación estándar. La fórmula es la siguiente:

$$X_{scaled} = \frac{X - Q_1(X)}{Q_3(X) - Q_1(X)}$$

Donde:

- X es el valor original de la variable.
- $Q_1(X)$ es el primer cuartil de la variable (percentil 25).
- $Q_3(X)$ es el tercer cuartil de la variable (percentil 75).

Robust Scaler fue ajustado a los datos de entrenamiento y con ese ajuste se realizó la transformación de los datos de validación y prueba. Si se hubiera realizado sin la previa división, el cálculo de los cuartiles se hubiera visto contaminado por los datos de prueba.

5.4.4 Stacked Autoencoders

Como última técnica de preprocessamiento, se han creado unos codificadores apilados o Stacked Autoencoders. Su inclusión en el preprocessamiento viene motivada por los buenos resultados obtenidos en los trabajos de Bao, Yue y Rao (2017) y Li, Zheng y Zheng (2019).

Los Stacked Autoencoders son una forma especial de redes neuronales artificiales utilizadas para la reducción de dimensionalidad en las variables de un modelo a través de la extracción de sus características profundas. Primero hay una fase de codificación donde la red se entrena para aprender una codificación eficiente de los datos de entrada y reducir su dimensionalidad. Despues tiene lugar una fase de decodificación que intenta reconstruir los datos originales a partir de la representación codificada. El objetivo es que los autoencoders aprendan automáticamente características relevantes de los datos, extrayendo información valiosa y eliminando el ruido. Esto reduce la dimensionalidad al mismo tiempo que se mantiene la información esencial de las variables, mejorando la generalización de los modelos.

Tras un proceso iterativo de prueba y error en el que se monitoreó una función de pérdida basada en *mean squared error (mse)*, se determinó que la estructura óptima de la red eran 5 autoencoders con una capa de entrada, una capa profunda y una capa de salida. La capa profunda cuenta con 150 neuronas y, al igual que la capa de salida, la función de activación es *Exponential Linear Unit (ELU)*.

Una vez definidos y entrenados, los Autoencoders se han apilado en un modelo secuencial de Keras, convirtiéndose en Stacked Autoencoders. Después del apilamiento ya están listos para recibir como entrada las variables originales y producir como salida una representación de sus características profundas. Una vez más, el entrenamiento de las redes se ha realizado únicamente con el conjunto de entrenamiento para evitar *data leakage*.

5.4.5 Procesamiento y exportación

Finalmente, se han consolidado todos los pasos de manera secuencial en una única función. Los pasos descritos anteriormente ya habían sido encapsulados en funciones, dado que la complejidad del preprocesamiento requería una modularización del código. La mayoría de transformaciones han sido ajustadas al conjunto de entrenamiento y luego aplicadas a los otros conjuntos. Al concluir el preprocesamiento, se han exportado tanto conjuntos procesados como no procesados a la base de datos persistente. Esta acción se llevó a cabo con el propósito de demostrar la validez del preprocesamiento realizado.

5.5 Entrenamiento y evaluación de modelos

La metodología llevada a cabo en los cuatro primeros algoritmos es la misma. Primero, se ha realizado una validación cruzada con *5 folds* con GridSearchCV en el conjunto de validación para obtener los hiperparámetros óptimos para cada algoritmo. Estos hiperparámetros no han sido los definitivos, simplemente han sido un punto de partida desde los cuales se ha realizado una aproximación a los hiperparámetros realmente óptimos a través de un proceso iterativo de prueba y error en el conjunto de entrenamiento.

Posteriormente, se ha entrenado cada algoritmo con un conjunto de datos de cada paso del preprocesamiento: un conjunto procesado con stacked autoencoders, normalización y wavelet transform, un conjunto procesado solo con normalización y stacked autoencoders, un conjunto procesado con wavelet transform y, finalmente, un conjunto sin procesar. El objetivo de esto es evaluar el valor añadido que ha aportado cada paso del preprocesamiento a la capacidad predictiva de los modelos. El parámetro Random State se ha fijado en 42 para garantizar la reproducibilidad de los resultados.

Una vez entrenados todos los conjuntos, se ha seleccionado el mejor en base a su precisión, una métrica que mide la proporción de predicciones correctas sobre el total de predicciones:

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN}$$

Donde:

- TP son verdaderos positivos (número de observaciones correctamente clasificadas como positivas).
- TN son verdaderos negativos (número de observaciones correctamente clasificadas como negativas).
- FP son falsos positivos (número de observaciones incorrectamente clasificadas como positivas).
- FN son falsos negativos (número de observaciones incorrectamente clasificadas como negativas).

También se ha impreso una matriz de confusión en porcentajes para obtener una métrica que, teniendo en cuenta que nuestra estrategia sólo operará posiciones en largo, es determinante a la hora de minimizar el riesgo: el porcentaje de veces que el modelo predice 1 (señal de entrada) y la realidad acaba siendo 1 (el precio al día siguiente cierra por encima de su nivel de apertura).

$$Positive\ Accuracy = \frac{TP}{TP+FP}$$

A continuación se describen los algoritmos empleados para la resolución del problema:

5.5.1 Regresión logística

El modelo de regresión logística es un algoritmo de aprendizaje supervisado utilizado para la clasificación binaria. De acuerdo con los hallazgos del EDA, este modelo no debería tener un desempeño superior a los otros modelos, debido a que asume que los datos son linealmente separables, es sensible a valores atípicos y no es capaz de captar la autocorrelación o estacionalidad en los datos porque se basa en la asunción de la independencia de las observaciones. No obstante, se ha incluido para compararlo con el resto de modelos. Su fórmula es la siguiente:

$$P(Y = 1) = \frac{1}{1+e^{-(WX+b)}}$$

Donde:

- $P(Y = 1)$ es la probabilidad de pertenecer a la clase positiva.
- W es el vector de coeficientes.
- X es el vector de variables independientes.
- b es el término constante.

La implementación de la regresión logística se ha llevado a cabo con Scikit-Learn. Se ha utilizado regularización L1, que ayuda a prevenir el sobreajuste al penalizar los coeficientes menos importantes y, potencialmente, llevándolos a cero. El algoritmo de optimización utilizado para la regularización es 'saga' , el cual es adecuado para problemas con regularización L1.

5.5.2 Random Forest

El Random Forest es un algoritmo de aprendizaje supervisado apto para problemas de clasificación. Su operativa se basa en la construcción de múltiples árboles de decisión durante el entrenamiento, cada uno entrenado con una muestra aleatoria del conjunto de datos y utilizando un subconjunto aleatorio de características en cada división del árbol. Esto introduce diversidad en los árboles individuales. Durante la predicción, cada árbol en el bosque emite su propia predicción, realizando una votación entre los árboles para determinar la clase final. La clase más votada se convierte en la predicción del bosque.

El Random Forest es capaz de manejar conjuntos de datos no linealmente separables y es robusto frente a valores atípicos porque emplea árboles de decisión no paramétricos (Madan, Saluja & Zhao, 2015). No es especialmente eficaz para datos autocorrelacionados y estacionales, pero no asume la distribución normal de las variables, por lo que puede ser un buen algoritmo para este problema de clasificación.

Tras la calibración de los hiperparámetros, Random Forest ha sido ajustado con 700 árboles, usando el criterio Gini para medir la calidad de las divisiones, limitando la profundidad de los árboles a 5, estableciendo parámetros mínimos para las divisiones y hojas, sin límite en el número máximo de características, con un máximo de 100 nodos hoja y limitando el número de muestras extraídas a 850 durante el entrenamiento de cada árbol. La biblioteca empleada ha sido Scikit Learn.

5.5.3 Extreme Gradient Boosting Classifier (XGBoost)

XGBoost Classifier es una implementación de Gradient Boosting específicamente diseñada para árboles de decisión. La operativa de XGBoost implica la construcción secuencial de árboles de decisión, donde cada árbol se enfoca en corregir los errores del modelo anterior. La salida final es una combinación ponderada de las predicciones de todos los árboles transformada por una función logística (para esto el hiperparámetro *objective* es “*binary:logistic*”). La fórmula es la misma que en regresión logística, pero la función $F(x) = (WX + b)$ es sustituida por la suma ponderada de las predicciones de todos los árboles. Además, XGBoost utiliza el descenso de gradiente para optimizar esta función de pérdida y ajustar los pesos de los árboles.

Este algoritmo no es muy recurrente en la literatura, pero es una técnica de aprendizaje automático extremadamente flexible y poderosa. No está específicamente diseñado para manejar series temporales con autocorrelación y estacionalidad. No obstante, al igual que otros métodos de ensamblaje, puede manejar conjuntos de datos no linealmente separables y es relativamente robusto ante distribuciones no normales y valores atípicos, por lo que puede tratarse de una buena elección para esta predicción.

Tras el ajuste paramétrico, el modelo ha sido entrenado en la biblioteca *xgboost* con 110 árboles, una tasa de aprendizaje de 0.1 para controlar la contribución de cada árbol al modelo general, una profundidad máxima de 2 para evitar el sobreajuste y un 80% como fracción de observaciones y características para entrenar cada árbol. Además, se ha añadido un control de penalización L1 de 0.001 y L2 de 1 para los valores extremos de los pesos.

5.5.4 Support Vector Machine

Support Vector Machine (SVM) es un algoritmo de aprendizaje supervisado que busca encontrar el hiperplano óptimo que maximiza el margen entre clases en un espacio de características. El hiperplano escogido es aquel hiperplano $wx - b = 0$ que maximiza la distancia entre los puntos más cercanos de las clases diferentes, conocidos como vectores de soporte. Aquí, w es el vector de peso, x es la muestra de entrada y b es el sesgo. La clase de una nueva muestra x_1 se determina calculando $wx_1 - b$. Si el resultado es positivo, la muestra se clasifica como una clase, y si es negativo, se clasifica como la otra clase.

Al igual que los dos últimos algoritmos, SVM no capta directamente autocorrelación ni estacionalidad en los datos, pero es robusto frente a valores atípicos y distribuciones no normales. En cuanto a la separabilidad lineal, el kernel radial (RBF) o el polinómico suele ser más apropiado para detectar relaciones no lineales, pero en la predicción el que mejor resultados ha dado es el kernel lineal. Según Sheta, Ahmed y Faris (2015), en general, las SVM tienen muchas ventajas sobre enfoques clásicos de clasificación como redes neuronales artificiales, árboles de decisión y otros. Estas ventajas incluyen un buen rendimiento en espacios de alta dimensión y el hecho de que son menos propensas a caer en el problema de los mínimos locales.

Debido a la exigente demanda computacional que ha supuesto el entrenamiento de este modelo en particular, concretamente en los conjuntos no procesados, y también con la intención de entrenar las redes neuronales en TPU, se emplearon 100 unidades informáticas en Google Colab para poder entrenar los modelos en GPU y TPU. El objetivo también es demostrar la posible escalabilidad del proyecto para algoritmos más exigentes o para conjuntos de datos más grandes. El entorno de ejecución fue cambiado de CPU con RAM de 13GB a GPU V100 con RAM de 54.8GB. Scikit-learn incluye implementaciones de SVM, pero no es una biblioteca específicamente diseñada para acelerar el entrenamiento de modelos en GPU. Para sacar el máximo rendimiento a la GPU y acelerar significativamente el entrenamiento de SVM lineales, se ha considerado utilizar cuML (Rapids.ai) como biblioteca específica de SVM acelerada por GPU.

Como se ha mencionado, el kernel empleado es lineal. El parámetro de regularización, que controla la compensación entre la clasificación correcta de muestras de entrenamiento y el margen, se ha fijado en 0.15. Esto se ha establecido así para forzar un margen blando o *soft margin* distinto del *hard margin* implementado en el EDA, ya que es más adecuado para clasificar clases que no se pueden separar perfectamente.. La función de pérdida utilizada durante el entrenamiento es *squared hinge* y se ha aplicado regularización L1 para obtener un modelo más interpretable al permitir que características irrelevantes tengan coeficientes exactamente cero.

5.5.5 Long Short-Term Memory

Las redes neuronales Long Short-Term Memory (LSTM) son un tipo de red neuronal recurrente (RNN) diseñada para manejar el problema de las dependencias a largo plazo en las secuencias de datos. Las LSTM destacan por su capacidad única para almacenar, olvidar y leer información de los estados a largo plazo en la dinámica subyacente de los datos.

Desarrolladas por Hochreiter y Schmidhuber en 1997 como una evolución de las RNN, las LSTM han demostrado su eficacia en la resolución de problemas que implican datos secuenciales (Chen, Chen & Fushimi, 2017). En el contexto de clasificación, las LSTM pueden procesar secuencias de datos y aprender patrones temporales complejos. A diferencia de las RNN tradicionales, las LSTM están diseñadas para mitigar el problema de la desaparición o explosión del gradiente (Li, Zheng & Zheng, 2019), lo que las hace más efectivas en la captura de dependencias a largo plazo como es el caso de las series temporales financieras (Chen, Chen & Fushimi, 2017).

La operativa de una LSTM para clasificación implica la alimentación de secuencias de datos en la red. Cada unidad LSTM tiene una memoria interna y puertas (input, forget, output) que regulan el flujo de información. La red aprende adaptativamente qué información retener y cuál desechar a lo largo del tiempo. Cada una de estas puertas puede expresarse de la siguiente manera:

1. Puerta de olvido (Forget Gate):

$$f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f)$$

2. Puerta de entrada (Input Gate):

$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i)$$

$$\bar{C}_t = \tanh(W_c \cdot [h_{t-1}, x_t] + b_c)$$

3. Actualización de la celda (Cell Update):

$$C_t = f_t \cdot C_{t-1} + i_t \cdot \bar{C}_t$$

4. Puerta de salida (Output Gate):

$$o_t = \sigma(W_o \cdot [h_{t-1}, x_t] + b_o)$$

5. Salida de la celda (Hidden State Output):

$$h_t = o_t \cdot \tanh(C_t)$$

Donde:

- x_t es la entrada en el tiempo t .
- h_t es el estado oculto en el tiempo t .
- C_t es el estado de la celda en el tiempo t .
- W_f, W_i, W_c, W_o son matrices de pesos para las diferentes puertas
- b_f, b_i, b_c, b_o son los términos constantes correspondientes.
- σ es la función sigmoide.
- \tanh es la función tangente hiperbólica.

La LSTM ha sido entrenada en la TPU de los servidores de Google Colab, con 37.8 GB de RAM para acelerar el entrenamiento. Tras un proceso de ajuste, se ha concluido que el número óptimo de días es 30, por lo que la matriz con la que alimentaremos las redes son tridimensionales: 1722 secuencias de 30 días de longitud y 47 características cada una. La biblioteca empleada es Tensorflow. El modelo consta de una capa de entrada, tres capas profundas LSTM con 200 neuronas cada una y una capa de salida con función de activación sigmoidal para ajustarse al problema de clasificación binario. Entre las capas se han creado capas de *Dropout* con un ratio de 30% para evitar sobreajuste. Las capas se han ensamblado dentro de un modelo secuencial de Keras. El optimizador es Adam Optimizer, la función de pérdida es *binary crossentropy* y la métrica es la precisión. Se ha añadido el elemento early stopping con valor mínimo 0.001 y paciencia 50 epochs para evitar el sobreajuste y ahorrar recursos computacionales. El número de epochs es 1000, el tamaño de los batches es 130 y los conjunto de validación es *dev*.

5.5.6 K-nearest neighbors (KNN)

Como se ha mencionado anteriormente, en este proyecto se ha implementado una novedosa combinación entre el algoritmo de clasificación no supervisada K-Nearest Neighbors (KNN) y el mejor de los algoritmos supervisados anteriores. La idea fue propuesta

pero no desarrollada por Madan, Saluja y Zhao (2015). En su trabajo, proponían examinar patrones en subconjuntos de los datos de precios, dividiendo los datos en combinaciones secuenciales vectores de 100 elementos para luego clasificarlos con k-means y entrenar modelos solo con datos de la misma clase.

En este trabajo la metodología es algo diferente. Primero, se ha obtenido el número óptimo de clústers con el método de silhouette. Como se puede observar en la **Figura 13**, el número óptimo de grupos es 3.

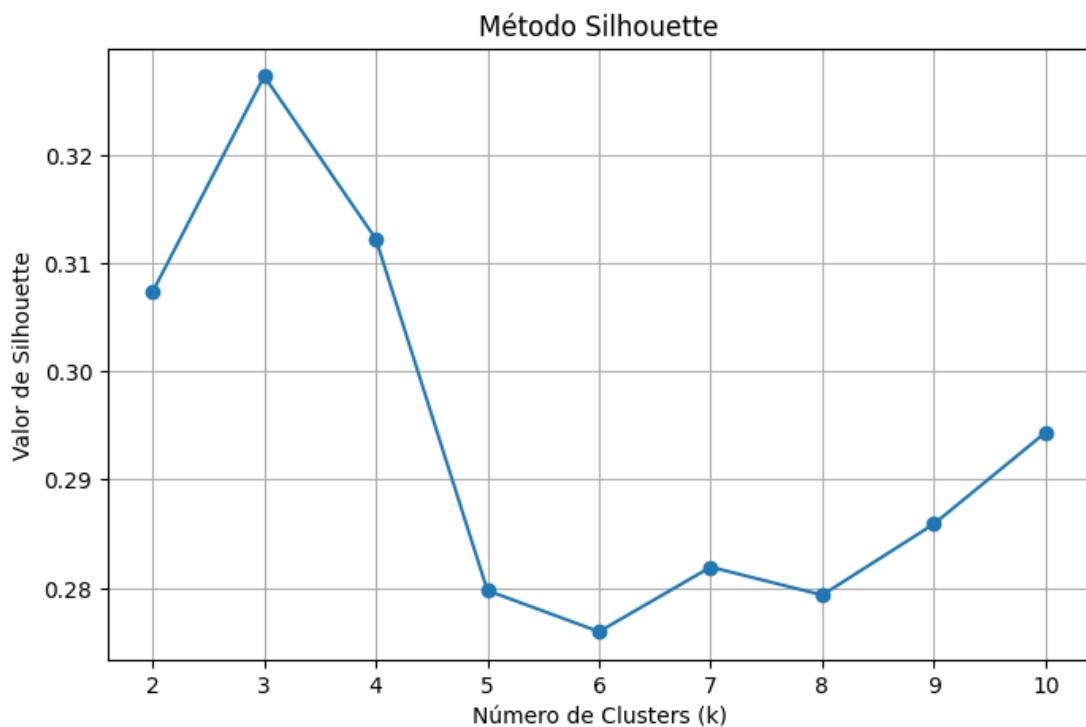


Figura 13: Gráfico del método silhouette.

Posteriormente, se han clasificado los datos de entrenamiento sin las etiquetas para igualar las condiciones a los datos de prueba. El mejor de los algoritmos anteriores se ha entrenado en cada grupo, ajustándose a días de cotización que son similares entre sí.

Finalmente, se ha ido clasificando cada día del conjunto de prueba en base a los grupos detectados en el conjunto de entrenamiento. Para cada grupo se ha empleado el algoritmo específicamente entrenado para ese conjunto de datos. Los objetivos de esta estrategia son dos. Por un lado, mejorar la capacidad predictiva entrenando un modelo en datos que son similares entre sí, incrementando la capacidad de generalización que tendrá el modelo en

datos similares. Por otro lado, reducir el número de datos que requiere un modelo para entrenarse para ahorrar recursos computacionales, ya que se pasará de entrenar un modelo con todo el conjunto de datos a entrenarlo con un subgrupo dentro del conjunto. Esto puede ser especialmente útil en conjuntos de datos muy grandes, con miles de registros. En este caso, todos los días del conjunto de prueba se han clasificado dentro del mismo grupo, por lo que solo ha hecho falta entrenar un modelo. El conjunto de datos de entrenamiento se ha reducido de 1752 filas a tan solo 680, haciendo el entrenamiento mucho menos costoso.

6. Resultados

La primera evaluación realizada es sobre el impacto de cada técnica de preprocesamiento en la capacidad predictiva de los algoritmos. Todas las evaluaciones de esta sección se han realizado entre el 01/07/2021 y el 28/06/2023. Estos resultados vienen descritos en la **Tabla 7**, en la cual se ha empleado la fórmula de precisión *accuracy*, definida en el apartado 5.5. La nomenclatura se describe en el pie de la figura.

	Sin Procesar	WT	WT + NORM	WT + NORM + SAE
LR	52,10%	52,10%	58,88%	59,48%
RF	48,70%	60,28%	60,28%	59,68%
XGB	49,50%	55,89%	55,89%	61,68%
SVM	52,10%	54,49%	60,08%	59,68%
LSTM	51,59%	51,59%	55,84%	55,20%

Tabla 7: Precisión después de cada paso en la fase de preprocesamiento. Modelos: regresión logística (LR), Random Forest (RF), XGBoost (XGB), Support Vector Machine (SVM), Long Short-Term Memory (LSTM). Técnicas: wavelet transform (WT), wavelet transform y normalización (WT + NORM), wavelet transform, normalización y stacked autoencoders (WT + NORM+SAE).

La mejor de las versiones de cada modelo ha sido seleccionada para la segunda parte de la evaluación, la cual consiste en tres aspectos: precisión positiva, rentabilidad y riesgo. La **Tabla 8** muestra los resultados de la precisión positiva, definida en el apartado 5.5 como *positive accuracy* (ver matrices de confusión en la **Figura 14**). “Sin predicción” simboliza los resultados de una estrategia que consistiera en comprar todos los días al precio de apertura y vender al precio de cierre, no realizando por tanto ninguna predicción. Del modelo

combinado de KNN y XGBoost, el cual fue seleccionado para la combinación debido a que fue el que mejores resultados obtuvo, cabe destacar que la rentabilidad fue menor en los modelos entrenados con conjuntos de datos pertenecientes a otros grupos según la clasificación de KNN (55,69% y 58,68%). Esto puede explicarse debido a su inferior similitud:

MODELOS	PRECISIÓN	PRECISIÓN POSITIVA
Sin Predicción	52,10%	52,10%
LR	59,48%	60,98%
RF	60,28%	59,17%
XGB	61,68%	63,02%
SVM	60,08%	63,80%
LSTM	55,84%	55,10%
KNN-XGB	61,68%	61,02%

Tabla 8: Precisiones individuales de los modelos: regresión logística (LR), Random Forest (RF), XGBoost (XGB), Support Vector Machine (SVM), Long Short-Term Memory (LSTM), Combinación de KNN y XGB (KNN-XGB)

Para mejorar la capacidad predictiva, se ha decidido emplear el método descrito en Sebastião y Godinho (2021), el cual consiste en la combinación de los modelos con mejores resultados para aumentar la precisión. Los ensamblajes son utilizados para reducir el sesgo y la varianza en las predicciones y suelen producir resultados más robustos y precisos que los modelos individuales. De esta manera, en cada combinación o ensamblaje solo se realizaría una orden de compra para el día siguiente si todos los algoritmos están de acuerdo en que el precio va a subir al día siguiente. Los resultados se pueden observar en la **Tabla 9**. LSTM ha sido omitido en los ensamblajes debido a su baja capacidad predictiva:

ENSAMBLAJES	PRECISIÓN	PRECISIÓN POSITIVA
KNN-XGB+XGB	62,67%	65,16%
KNN-XGB+XGB+RF	63,07%	66,38%
KNN-XGB+XGB+RF+SVM	60,68%	68,39%
KNN-XGB+XGB+RF+SVM+LR	59,88%	68,52%

Tabla 9: Precisiones de ensamblajes de modelos. La nomenclatura es la misma que en la Figura 10. “+” simboliza combinación.

Para ofrecer otro punto de vista sobre los resultados se ha calculado la rentabilidad de cada modelo. El motivo de esto es que la precisión positiva por sí sola no es suficiente para evaluar los resultados, ya que un modelo puede tener una precisión positiva alta pero operar muy pocos días (como es el caso de los ensamblajes de muchos modelos), lo que no necesariamente significa mayor rentabilidad. La **Tabla 10** describe la rentabilidad acumulativa desde el inicio de cada año hasta el final de cada año. La fórmula se define de la siguiente manera:

$$R = \prod_{i=1}^N (1 + r_i) - 1$$

Donde:

- N es el número de días en los que se opera
- r_i es el rendimiento diario en el día i
- Π denota el producto acumulativo

La fila “100% acierto” muestra los resultados de un modelo que hubiera tenido una precisión del 100%. “Buy&Hold” muestra los resultados de una estrategia que consiste en comprar el NASDAQ al principio de un período y venderlo al final de cada período.

	2021	2022	2023
LR	13,72%	29,13%	32,75%
RF	19,74%	47,72%	28,44%
XGB	14,21%	52,53%	31,80%
SVM	20,09%	46,16%	30,44%
LSTM	9,27%	11,45%	30,18%
KNN-XGB	14,92%	50,76%	34,97%
KNN-XGB+XGB	14,21%	62,21%	32,72%
KNN-XGB+XGB+RF	16,13%	65,84%	32,72%
KNN-XGB+XGB+RF+SVM	15,82%	52,09%	25,83%
KNN-XGB+XGB+RF+SVM+LR	12,33%	52,09%	25,14%
Sin Predicción	1,43%	-17,67%	28,44%
100% Acierto	52,85%	402,92%	87,57%
Buy&Hold	7,73%	-33,10%	29,86%

Tabla 10: Rentabilidad acumulada anual de cada modelo

La **Tabla 11** muestra la rentabilidad acumulada de 2021, de 2021 y 2022 y finalmente la rentabilidad acumulada de todo el período: 2021, 2022 y 2023.

	2021	2022	2023
LR	13,72%	46,85%	94,94%
RF	19,74%	76,88%	127,19%
XGB	14,21%	74,21%	129,61%
SVM	20,09%	75,53%	128,96%
LSTM	9,27%	21,78%	58,53%
KNN-XGB	14,92%	73,26%	133,85%
KNN-XGB+XGB	14,21%	85,27%	145,88%
KNN-XGB+XGB+RF	16,13%	92,59%	155,60%
KNN-XGB+XGB+RF+SVM	15,82%	76,15%	121,64%
KNN-XGB+XGB+RF+SVM+LR	12,33%	70,84%	113,78%
Sin Predicción	1,43%	-16,49%	7,26%
100% Acierto	52,85%	668,71%	1341,83%
Buy&Hold	7,73%	-27,93%	-6,41%

Tabla 11: Rentabilidad acumulada de todo el período.

Finalmente, para visualizar el riesgo de cada uno de los modelos, se calculan tres métricas de riesgo en la **Tabla 12**: VaR, Maximum Drawdown y Sharpe Ratio. Las tres fórmulas han sido previamente definidas en la sección 5.3.9:

	VaR al 5%	Maximum Drawdown	Sharpe Ratio
LR	-1,67%	-12,56%	3,06
RF	-1,66%	-12,44%	3,01
XGB	-1,58%	-4,91%	2,94
SVM	-1,37%	-3,96%	2,94
LSTM	-1,91%	-12,51%	3,00
KNN-XGB	-1,59%	-8,73%	2,80
KNN-XGB+XGB	-1,36%	-3,60%	2,79
KNN-XGB+XGB+RF	-1,34%	-3,60%	2,82
KNN-XGB+XGB+RF+SVM	-1,34%	-3,60%	2,84
KNN-XGB+XGB+RF+SVM+LR	-1,36%	-3,60%	2,83
Sin Predicción	-2,31%	-26,08%	0,36
100% Acierto	0,04%	0,00%	3,23
Buy&Hold	-2,60%	-36,40%	-0,97

Tabla 12: Métricas de riesgo para cada modelo

7. Discusión

De los resultados mostrados en la **Tabla 7** se puede concluir que el preprocesamiento en este proyecto ha aportado un gran valor añadido a la capacidad predictiva de los modelos. En la regresión logística la transformada wavelet parece no añadir ningún valor, pero la normalización y los stacked autoencoders incrementan la precisión en más de un 7%.

En cuanto a los modelos de árboles, tanto en Random Forest como en XGBoost, la normalización no ha mejorado los resultados. Esto puede deberse a que los modelos de árboles invariantes a la escala de las variables predictoras. No obstante, para los dos la aplicación de la transformada wavelet supone una mejora considerable, con un incremento en la precisión en el caso de Random Forest de un impresionante 11,58%. Random Forest es perjudicado por los stacked autoencoders, con un decrecimiento en la precisión inferior al 1%, mientras que XGBoost se ve ampliamente beneficiado, con un incremento de casi un 6%, hasta convertirse en el modelo más preciso con un 61,68%.

El algoritmo Support Vector Machine también se ve ligeramente perjudicado tras la aplicación de stacked autoencoders. Sin embargo, la transformada wavelet y, sobre todo, la normalización, aportan un gran valor añadido, hasta alcanzar una precisión del 60,08%.

Por otro lado, el algoritmo con peores resultados con diferencia son las redes neuronales recurrentes LSTM, cuya precisión en la mejor de las versiones es de un 55,84%. El proceso de ajuste de hiperparámetros fue laborioso y muy extenso hasta lograr dicha predicción. En ninguna de las iteraciones el modelo consiguió superar el 60% en precisión, y las técnicas de preprocesamiento añaden tan poco valor que cabe preguntarse si dichas diferencias son producto de la aleatoriedad. En las **Figuras 17 y 18** se observa que el modelo padece de sobreajuste. Se intentaron aplicar técnicas de prevención de sobreajuste más agresivas que el *dropout rate*, como las regularizaciones L1 y L2, pero estos ajustes empeoraron la capacidad predictiva del modelo. La explicación más coherente para este fracaso es el tamaño del conjunto de datos. Por un lado, los 1722 registros que sirvieron de entrenamiento parecen ser insuficientes. Las redes neuronales, y especialmente las LSTM, a menudo requieren grandes cantidades de datos para aprender patrones complejos. Por otro lado, quizás las 47 variables predictoras suponen un problema de excesiva dimensionalidad, dificultando el aprendizaje de la red de patrones significativos. Para futuros trabajos, sería interesante experimentar con este algoritmo en un conjunto de datos con muchos más registros, con datos de cotización por horas o minutos, por ejemplo, y reduciendo el número de variables predictoras para disminuir la dimensionalidad.

En general, la transformada wavelet ha sido el paso del preprocesamiento que más valor añadido ha generado para la capacidad predictiva de los modelos, con un incremento agregado de 20,36% en las precisiones, seguido de la normalización con un 16,62%, y por último los stacked autoencoders con 4,75%. A pesar de esta aparente poca efectividad, cabe destacar que los stacked autoencoders han supuesto un incremento de 5,79% en el mejor de los modelos.

La novedosa combinación entre el algoritmo de clasificación no supervisada K-Nearest Neighbors y XGBoost ha resultado ser un éxito, ya que ha alcanzado la misma precisión que XGBoost pero con un conjunto de datos de los 680 registros más similares, en vez de con los 1752 registros de todo el conjunto de datos. Llevado a gran escala, esta combinación puede suponer un ahorro mayúsculo en recursos computacionales. Además, las predicciones no son

exactamente las mismas que XGBoost, por lo que la combinación de ambos ha logrado incrementar tanto la precisión como la precisión positiva, alcanzando un 62,67% y un 65,16% respectivamente.

En cuanto a las variables de propia creación para detectar soportes y resistencias, en Random Forest no han adquirido mucha importancia (**Figura 16**). No obstante, la variable de soportes si ha sido tenida en cuenta en el mejor modelo, XGBoost, con una importancia similar al resto de variables tenidas en cuenta (**Figura 15**). Estos resultados son esperanzadores y pueden suponer una nueva vía de investigación en futuros trabajos sobre trading algorítmico.

La mejor marca en la precisión de predicciones positivas ha sido obtenida por Support Vector Machine, donde un 63,80% de las veces que el modelo predijo una entrada el NASDAQ acabó subiendo al día siguiente. XGBoost está ligeramente por detrás, con un 63,02%, y el peor vuelve a ser LSTM. Si se hubiera comprado y vendido todos los días durante los 2 años, el porcentaje de acierto hubiera sido de un 52,10%.

Respecto a la combinación de algoritmos, se ha logrado demostrar su efectividad, siendo la combinación de K-Nearest Neighbors y XGBoost, XGBoost y Random Forest la mayor precisión alcanzada en este trabajo, un 63,07%. En cuanto a la predicción positiva, los resultados pueden ser engañosos ya que obviamente al añadir más algoritmos al ensamblaje se aumentará la precisión, pero ello no quiere decir que haya un incremento en la rentabilidad.

Analizando la rentabilidad, el mejor modelo en la segunda mitad de 2021 fue Support Vector Machine, con una rentabilidad acumulada del 20,09% en 6 meses, mientras una estrategia *buy&hold* generó tan solo un 7,73%, y una operativa diaria sin predicción un 1,43%. En 2022, mientras la estrategia *buy&hold* generaba pérdidas de hasta un -33,10%, la combinación KNN-XGBoost, XGBoost y Random Forest alcanzaron una sorprendente rentabilidad en un año de 65,84%. La estrategia sin predicción hubiera perdido un -17,67% ese año. Finalmente, en el primer semestre de 2023 el mejor algoritmo fue KNN-XGBoost, con una rentabilidad del 34,97% en unos meses en el que el NASDAQ alcanzó una rentabilidad del 29,86%. Cabe destacar que, pese a sus malos resultados, LSTM obtuvo una rentabilidad en 2023 de 30,18%, batiendo también al mercado.

Dada la heterogeneidad de los resultados, conviene comparar la rentabilidad total acumulada de todos los algoritmos para escoger al mejor en términos de rentabilidad. Tras dos años de operativa, el pódium es ocupado en primer lugar por el ensamblaje KNN-XGBoost, XGBoost y Random Forest, con una impresionante rentabilidad total acumulada de 155,60%. En segundo lugar está el ensamblaje KNN-XGBoost y XGBoost, con una rentabilidad del 145,88%. Finalmente, en tercer lugar están los algoritmos KNN-XGBoost, con un 133,85%. El peor algoritmo fue LSTM, con un 58,53%, pero aún así logró batir al mercado, que tras esos dos años cayó un -6,41%, y a la estrategia sin predicción, que tan solo generó un 7,26%.

Para analizar si este exceso de rentabilidad ha supuesto un incremento en el riesgo, hay que fijarse en las métricas Value at Risk (VaR), Maximum Drawdown y Sharpe Ratio. De las dos primeras se infiere que los modelos de ensamblaje fueron, con creces, los más seguros, ya que las máximas caídas que experimentaron fueron de tan solo un -3,60%, mientras que la mayor caída de la estrategia *buy&hold* fue de -36,40%. Al 5% de confianza, sus pérdidas estuvieron entre -1,34% y -1,36%. Los resultados del Sharpe Ratio difieren ligeramente, donde la regresión logística es el mejor modelo (3.06). Esto puede deberse a que el Sharpe Ratio mide la rentabilidad ajustada a la volatilidad, y la volatilidad puede presentarse tanto en dirección positiva como negativa. En este contexto, una estrategia con una alta volatilidad, pero sin caídas pronunciadas, puede ser considerada favorable, reflejando un perfil de riesgo-recompensa atractivo.

8. Conclusión

En conclusión, este trabajo ha alcanzado exitosamente los objetivos propuestos, tal y como han demostrado sus resultados. Se ha logrado desarrollar un algoritmo de trading con machine learning capaz de superar, no sólo en rentabilidad sino también en riesgo, a una estrategia *buy&hold* en el NASDAQ Composite, batiendo así su índice de referencia en 2 años consecutivos. La rentabilidad acumulada obtenida por la combinación de los algoritmos K-Nearest Neighbors y XGBoost, XGBoost y Random Forest fue de un 155,60% en un período en el que el NASDAQ Composite cayó un -6,41%. Además, en términos de riesgo, la mayor caída experimentada por nuestro algoritmo fue de un -3,60%, siendo la del NASDAQ 10 veces superior, con un -36,40%. Estos resultados se añaden a una larga lista de trabajos

que sugieren la naturaleza determinista de los mercados financieros, rechazando al menos parcialmente la Hipótesis del Mercado Eficiente.

Asimismo, se han creado y comparado diferentes algoritmos de clasificación supervisada y sus versiones combinadas, una idea desarrollada anteriormente por Sebastião y Godinho (2021), para evaluar cuáles son los más adecuados para resolver este tipo de problemas. Los resultados han demostrado los beneficios de esta estrategia en términos de precisión, rentabilidad y riesgo, hasta el punto de producir el modelo con mejores resultados del proyecto.

También se ha llevado a cabo la idea propuesta pero no desarrollada por Madan, Saluja y Zhao (2015), que sugería la combinación de un algoritmo de clasificación no supervisada como preprocesamiento para entrenar un modelo de clasificación supervisada con datos similares entre sí. El éxito de esta estrategia se evidencia al observar que los tres modelos con la mayor rentabilidad lograda incorporan esta combinación.

La efectividad de la Wavelet Transform, los Stacked Autoencoders y la normalización como técnicas de preprocesamiento de series temporales financieras ha sido demostrada en los incrementos de la capacidad predictiva de los modelos. Algunos de estos beneficios ya han sido demostrados en redes neuronales como Long Short-Term Memory (Bao, Yue & Rao, 2017) y en Support Vector Machine (Vincent et al., 2010), pero en este trabajo se demuestra su efectividad para algoritmos como Random Forest, Regresión Logística y XGBoost, un algoritmo poco recurrente en la literatura.

Este último algoritmo da indicios de la validez de las novedosas variables para detección de soportes y resistencias creadas en este trabajo, ya que una de ellas fue tenida en cuenta en los cálculos del algoritmo que mejores resultados individuales ha dado. Este hallazgo abre una posible vía de investigación nueva para futuros trabajos.

En resumen, este trabajo trasciende las meras evaluaciones de algoritmos y se adentra en cuestiones fundamentales sobre la predictibilidad en los mercados financieros. Explora enfoques innovadores y demuestra la utilidad del machine learning como una herramienta eficaz para la toma de decisiones de inversión. Los resultados finales consolidan la aportación práctica de este estudio al campo del análisis financiero y la predicción de mercados, cuya

dificultad ha quedado reflejada en el análisis exploratorio de datos al confirmarse la no linealidad, no normalidad, ruido, interdependencia y estacionalidad presentes en este tipo de problemas.

9. Bibliografía

Esteban, J. 2023, *España pasará de 3,3 a 1,7 trabajadores por pensionista: la dependencia se duplicará en 2050*, eleconomista.es.

<https://www.eleconomista.es/economia/noticias/12237341/04/23/de-33-a-17-trabajadores-por-pensionista-la-dependencia-se-duplicara-en-2050.html>

Hansen, K.B. 2020, "The virtue of simplicity: On machine learning models in algorithmic trading", *Big Data & Society*, vol. 7, no. 1, pp. 2053951720926558.

<https://doi.org/10.1177/2053951720926558>

Chen, G., Chen, Y. & Fushimi, T. 2017, "Application of deep learning to algorithmic trading", *Tech.Rep, Tech.Rep.*, . <https://cs229.stanford.edu/proj2017/final-reports/5241098.pdf>

Li, Y., Zheng, W. & Zheng, Z. 2019, "Deep robust reinforcement learning for practical algorithmic trading", *IEEE Access*, vol. 7, pp. 108014-108022.

<https://ieeexplore.ieee.org/abstract/document/8786132>

Sheta, A.F., Ahmed, S.E.M. & Faris, H. 2015, "A comparison between regression, artificial neural networks and support vector machines for predicting stock market index", *Soft Computing*, vol. 7, no. 8, pp. 2.

<https://citeseerx.ist.psu.edu/document?repid=rep1&type=pdf&doi=59239d5c6c3ef17c0fb860d3809c1d2d3988bd5f>

Vincent, P., Larochelle, H., Lajoie, I., Bengio, Y., Manzagol, P. & Bottou, L. 2010, "Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion.", *Journal of machine learning research*, vol. 11, no. 12.

<https://www.jmlr.org/papers/volume11/vincent10a/vincent10a.pdf?ref=https://githubhelp.com>

Sazu, M.H. 2022, "How machine learning can drive high frequency algorithmic trading for technology stocks", *International Journal of Data Science and Advanced Analytics*, vol. 4, no. 4, pp. 84-93. <http://ijdsaa.com/index.php/welcome/article/view/97>

Madan, I., Saluja, S. & Zhao, A. 2015, "Automated bitcoin trading via machine learning algorithms", URL: <http://cs229.stanford.edu/proj2014/Isaac%20Madan>, vol. 20.

Sebastião, H. & Godinho, P. 2021, "Forecasting and trading cryptocurrencies with machine learning under changing market conditions", *Financial Innovation*, vol. 7, no. 1, pp. 1-30. <https://jfin-swufe.springeropen.com/articles/10.1186/s40854-020-00217-x>

Koker, T.E. & Koutmos, D. 2020, "Cryptocurrency trading using machine learning", *Journal of Risk and Financial Management*, vol. 13, no. 8, pp. 178. <https://www.mdpi.com/1911-8074/13/8/178>

Al-Ameer, A. & Fouad, A. 2021, "A methodology for securities and cryptocurrency trading using exploratory data analysis and artificial intelligence", *2021 1st International Conference on Artificial Intelligence and Data Analytics (CAIDA)IEEE*, , pp. 54. <https://ieeexplore.ieee.org/abstract/document/9425223/>

Xiong, Z. & Zhang, Z. 2016, "A data preprocessing method applied to cluster analysis on stock data by kmeans", *2016 International Conference on Intelligent Control and Computer Application (ICCA 2016)*Atlantis Press, , pp. 142. <https://www.atlantis-press.com/proceedings/icca-16/25847690>

Liang, X., Ge, Z., Sun, L., He, M. & Chen, H. 2019, "LSTM with wavelet transform based data preprocessing for stock price prediction", *Mathematical Problems in Engineering*, vol. 2019. <https://www.hindawi.com/journals/mpe/2019/1340174/>

Kumar, M. & Kalia, A. 2012, "Preprocessing and symbolic representation of stock data", *2012 Second International Conference on Advanced Computing & Communication TechnologiesIEEE*, , pp. 83. <https://ieeexplore.ieee.org/abstract/document/6168338/>

Nayak, S.C., Misra, B.B. & Behera, H.S. 2014, "Impact of data normalization on stock index forecasting", *International Journal of Computer Information Systems and Industrial Management Applications*, vol. 6, no. 2014, pp. 257-269. http://www.mirlabs.org/ijcisim/regular_papers_2014/IJCISIM_24.pdf

Pai, P. & Wei, W. 2007, "Predicting movement directions of stock index futures by support vector models with data preprocessing", *2007 IEEE International Conference on Industrial Engineering and Engineering Management*IEEE, , pp. 169.

<https://ieeexplore.ieee.org/abstract/document/4419173/>

Zhora, D.V. 2005, "Data preprocessing for stock market forecasting using random subspace classifier network", *Proceedings. 2005 IEEE International Joint Conference on Neural Networks, 2005*.IEEE, , pp. 2549. <https://ieeexplore.ieee.org/abstract/document/1556304/>

Bao, W., Yue, J. & Rao, Y. 2017, "A deep learning framework for financial time series using stacked autoencoders and long-short term memory", *PloS one*, vol. 12, no. 7, pp. e0180944.

<https://journals.plos.org/plosone/article?id=10.1371/journal.pone.0180944>

10. Apéndice

	CARACTERÍSTICAS	EXPLICACIÓN
COTIZACIÓN DIARIA	OHLC	Precios de apertura, máximo, mínimo y cierre del NASDAQ en un día
	Volumen	Volumen de negociación del NASDAQ en un día
	Rentabilidad intradía	Variación porcentual entre el precio de apertura y el de cierre de un mismo día
INDICADORES TÉCNICOS	SMA (50, 200)	Media móvil simple de 50 y 200 períodos
	EMA (9, 50, 200)	Media móvil exponencial de 9, 50 y 200 períodos
	WMA (9, 50, 200)	Media móvil ponderada de 9, 50 y 200 períodos
	RSI (14)	Índice de fuerza relativa. Mide la magnitud de las ganancias frente a las pérdidas
	MACD	Media móvil de convergencia y divergencia entre las medias móviles de 12 y 26 períodos
	Estocástico	Indicador que compara el precio de cierre del NASDAQ con su rango de precios en 14 períodos
	Momentum	Indicador que mide la velocidad del cambio de precios en 10 períodos
	Bandas de Bollinger	Media móvil y dos bandas representando las desviaciones estándar
	Soportes	Niveles de precios en los que se espera que el precio encuentre un freno a la baja
	Resistencias	Niveles de precios en los que se espera que el precio encuentre un freno al alza
VARIABLES MACROECONÓMICAS	10y-2y yield curve	Diferencia entre los rendimientos de los bonos del Tesoro a 10 y 2 años
	Volatilidad (VIX)	Mide la expectativa del mercado sobre la volatilidad a corto plazo transmitida por los precios de las opciones de índices bursátiles (CBOE)
	5y interest rate	Rentabilidad del bono del Tesoro a 5 años cotizado en el mercado secundario
	federal interest rate	Tasa de interés establecida por la Reserva Federal
	junk bonds spread	Diferencial entre los rendimientos de bonos basura y bonos del Tesoro
	Desempleo	Tasa que representa la proporción de la fuerza laboral desempleada
	Inflación	Tasa de cambio de precios de bienes y servicios en la economía estadounidense.
	Expectativas del consumidor	Percepciones y previsiones de los consumidores sobre la economía y su situación financiera (OCDE)
	Producción industrial	Producción real de todos los establecimientos relevantes ubicados en los Estados Unidos
ÍNDICES Y FUTUROS	S&P500	Índice que refleja el desempeño de las 500 mayores empresas en el mercado estadounidense
	Wilshire 5000	Índice de mercado que incluye prácticamente todas las acciones públicas de Estados Unidos
	DJIA	Índice que refleja el desempeño de 30 grandes empresas estadounidenses
	Nikkei 225	Índice de la Bolsa de Tokio que refleja el desempeño de 225 grandes empresas japonesas
	Euro STOXX 50	Índice que refleja el desempeño de las 50 principales empresas en la Eurozona
	Hang Seng Index	Índice que refleja el desempeño de las empresas más grandes que cotizan en la Bolsa de Hong Kong
	Shanghai Stock Exchange	Bolsa de valores de Shanghái
	FTSE 100	Índice que refleja el desempeño de las 100 empresas más grandes en la Bolsa de Londres
	Gold futures	Contratos de futuros de oro
	Crude oil futures	Contratos de futuros de petróleo crudo

Tabla 1: lista de las variables independientes

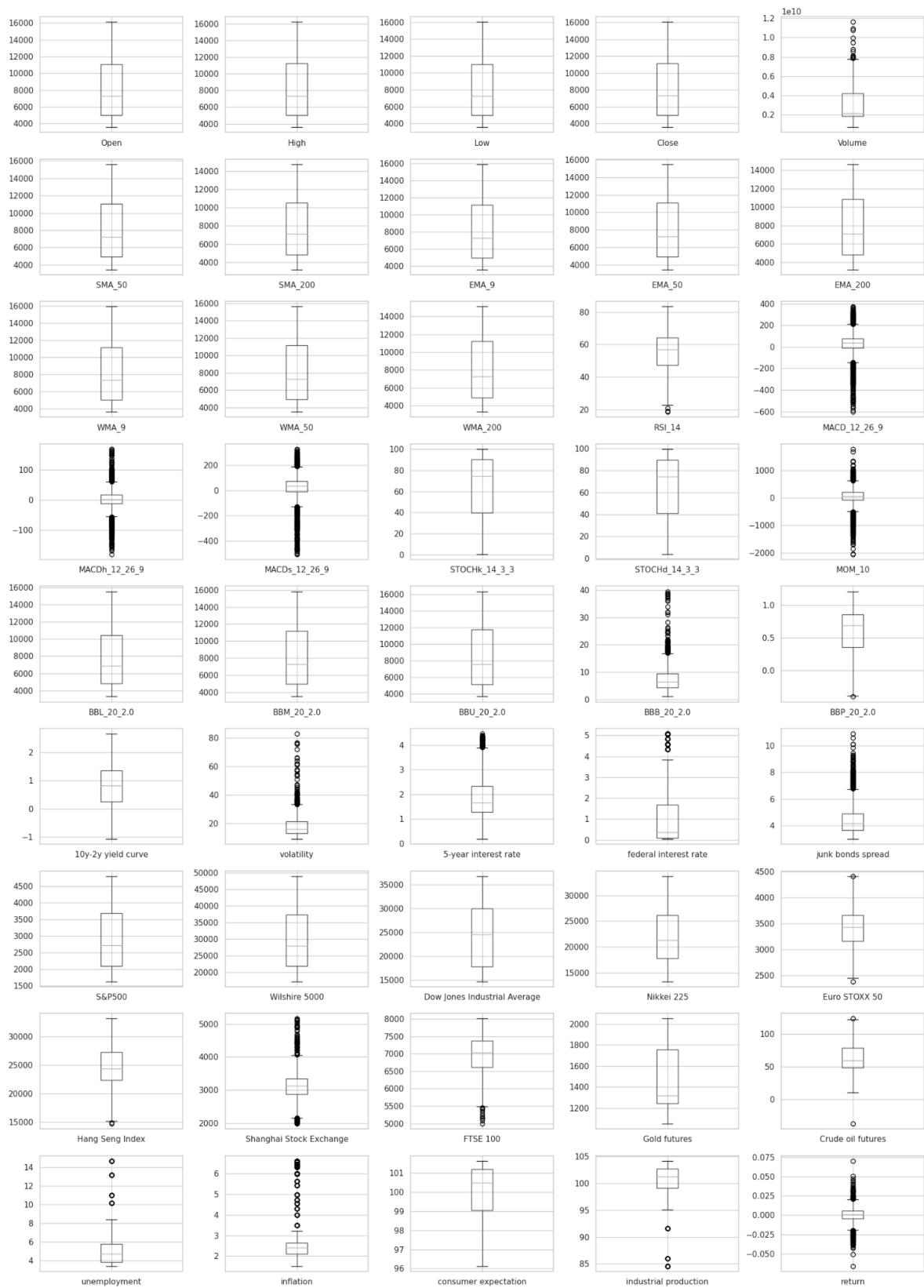


Figura 2: matriz de box plots de todas las variables float para visualizar los outliers

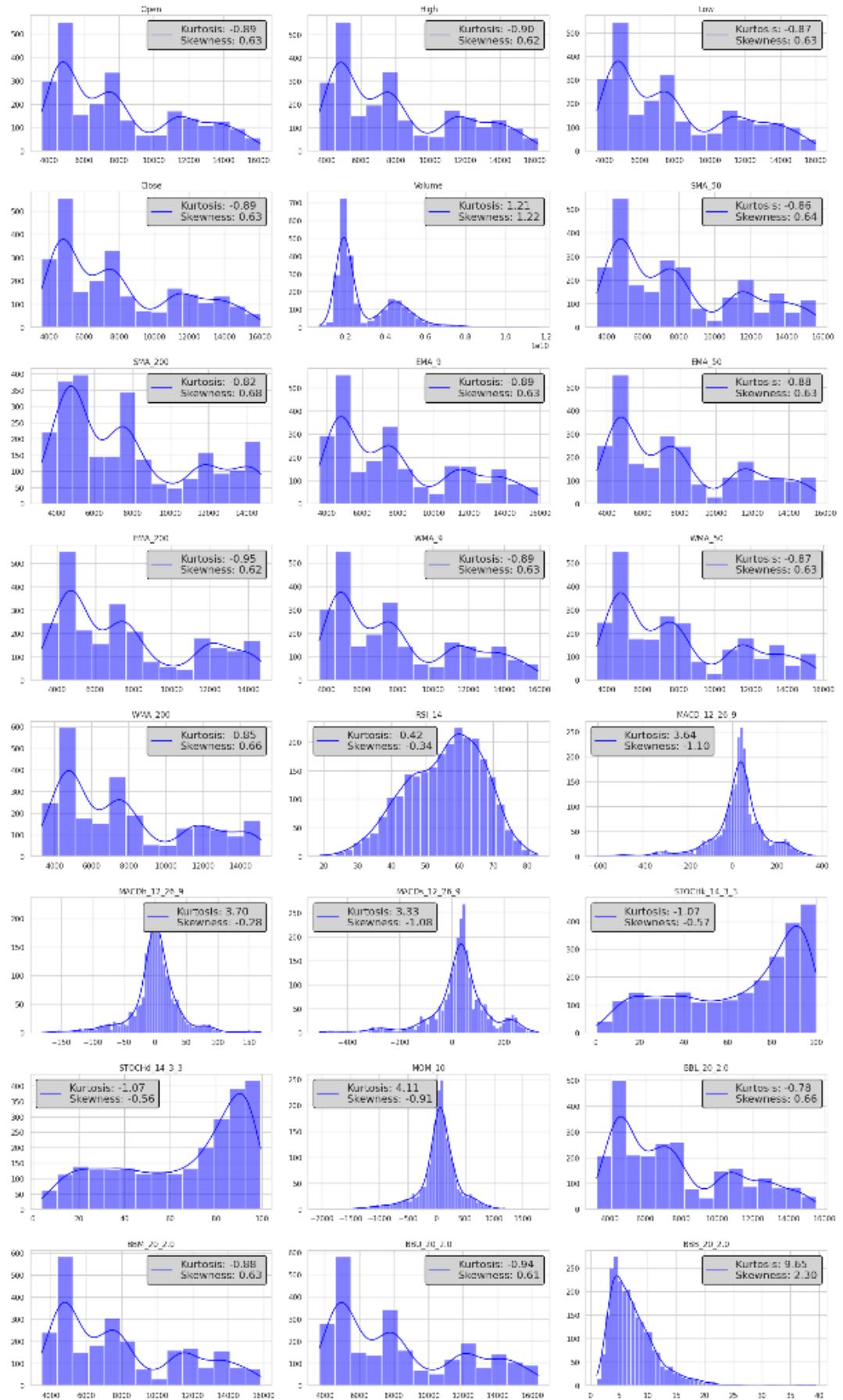


Figura 3.1: matriz de distribuciones de las variables con kurtosis y skewness

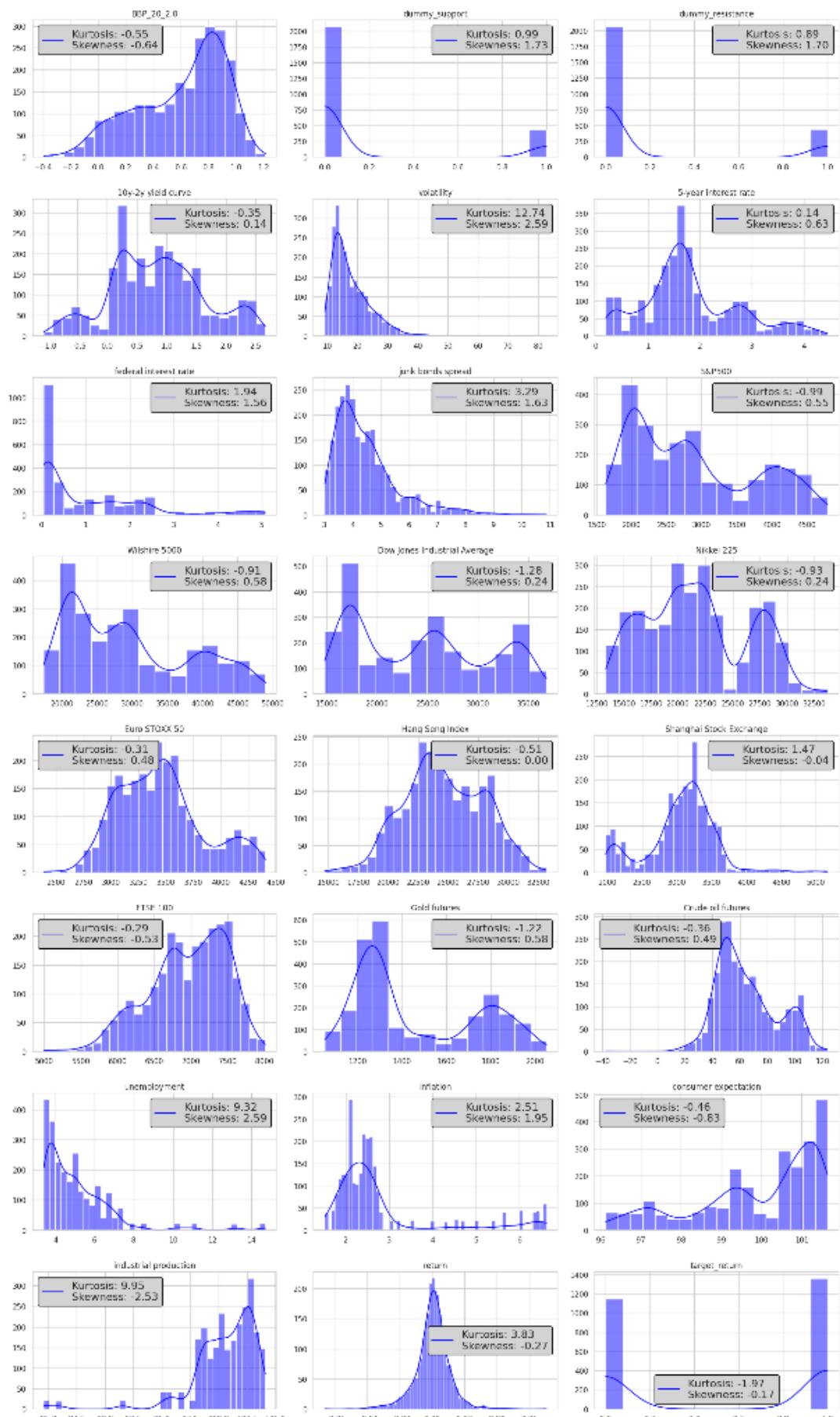


Figura 3.2: matriz de distribuciones de las variables con kurtosis y skewness

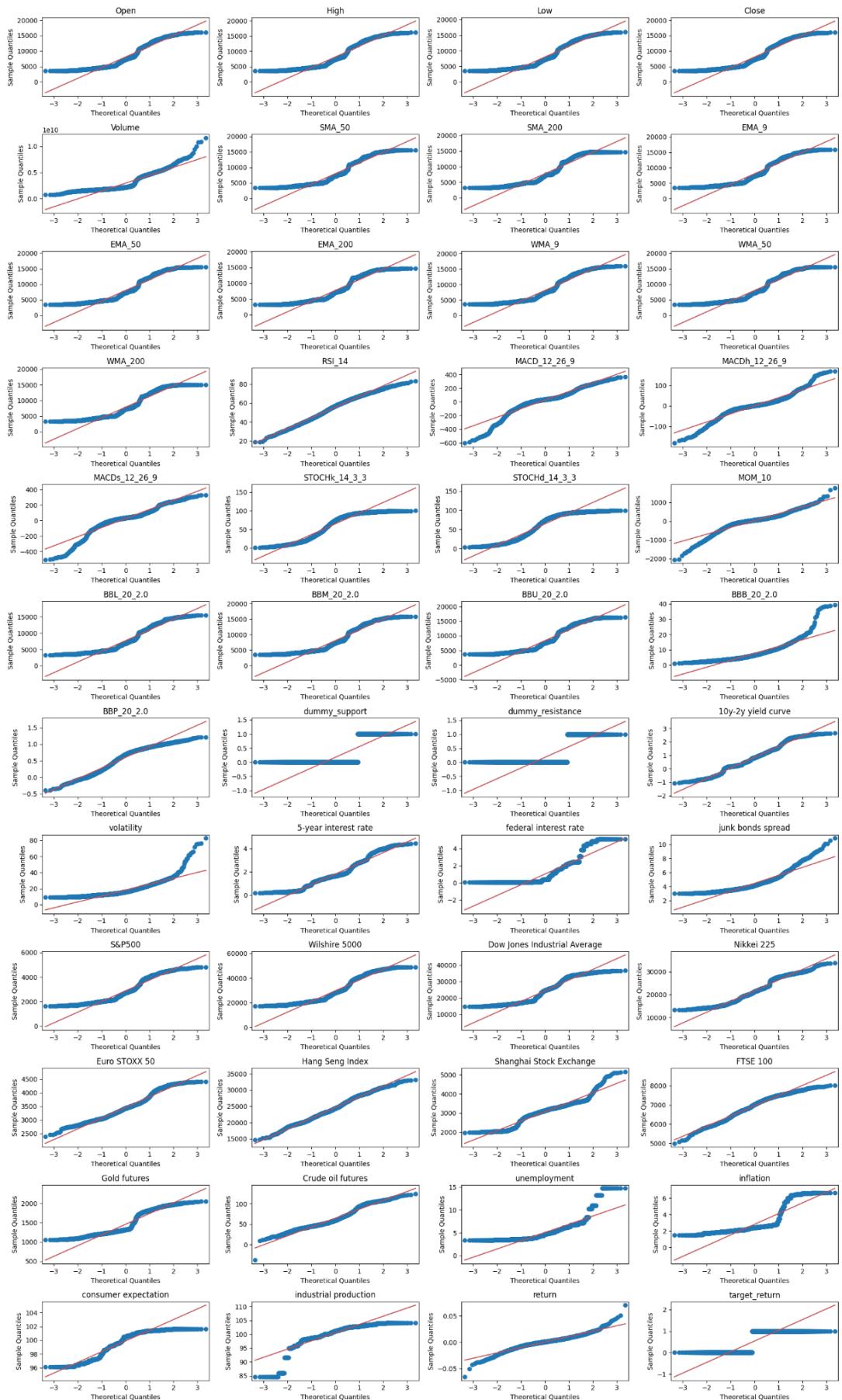


Figura 4: matriz de Q-Q plots de las variables

Variable	Statistic	p-value	Distribución Normal
Open	0,90296	3,26E-37	No
High	0,90203	2,39E-37	No
Low	0,90387	4,45E-37	No
Close	0,90292	3,22E-37	No
Volume	0,83261	2,80E-45	No
SMA_50	0,90102	1,71E-37	No
SMA_200	0,89394	1,74E-38	No
EMA_9	0,90224	2,56E-37	No
EMA_50	0,90078	1,58E-37	No
EMA_200	0,89526	2,63E-38	No
WMA_9	0,90248	2,78E-37	No
WMA_50	0,90123	1,83E-37	No
WMA_200	0,89774	5,82E-38	No
RSI_14	0,98655	1,23E-14	No
MACD_12_26_9	0,90800	1,86E-36	No
MACDh_12_26_9	0,91183	7,31E-36	No
MACDs_12_26_9	0,90321	3,56E-37	No
STOChk_14_3_3	0,89158	8,30E-39	No
STOChd_14_3_3	0,89345	1,49E-38	No
MOM_10	0,91296	1,11E-35	No
BBL_20_2.0	0,90661	1,14E-36	No
BBM_20_2.0	0,90203	2,39E-37	No
BBU_20_2.0	0,89828	6,95E-38	No
BBB_20_2.0	0,82784	1,40E-45	No

Tabla 2.1: Resultados del test de Shapiro-Wilk

Variable	Statistic	p-value	Distribución Normal
BBP_20_2.0	0,94176	2,54E-30	No
dummy_support	0,45765	0,00E+00	No
dummy_resistance	0,46170	0,00E+00	No
10y-2y yield curve	0,98258	5,76E-17	No
volatility	0,80191	0,00E+00	No
5-year interest rate	0,95055	2,86E-28	No
federal interest rate	0,76195	0,00E+00	No
junk bonds spread	0,86296	2,73E-42	No
S&P500	0,91302	1,13E-35	No
Wilshire 5000	0,91394	1,59E-35	No
Dow Jones Industrial Average	0,91881	1,01E-34	No
Nikkei 225	0,96043	1,35E-25	No
Euro STOXX 50	0,96862	5,79E-23	No
Hang Seng Index	0,99218	2,28E-10	No
Shanghai Stock Exchange	0,93663	2,08E-31	No
FTSE 100	0,97063	3,06E-22	No
Gold futures	0,86387	3,45E-42	No
Crude oil futures	0,95365	1,78E-27	No
unemployment	0,74640	0,00E+00	No
inflation	0,66646	0,00E+00	No
consumer expectation	0,88016	2,82E-40	No
industrial production	0,78571	0,00E+00	No
return	0,95095	3,61E-28	No
target_return	0,63391	0,00E+00	No

Tabla 2.2: Resultados del test de Shapiro-Wilk

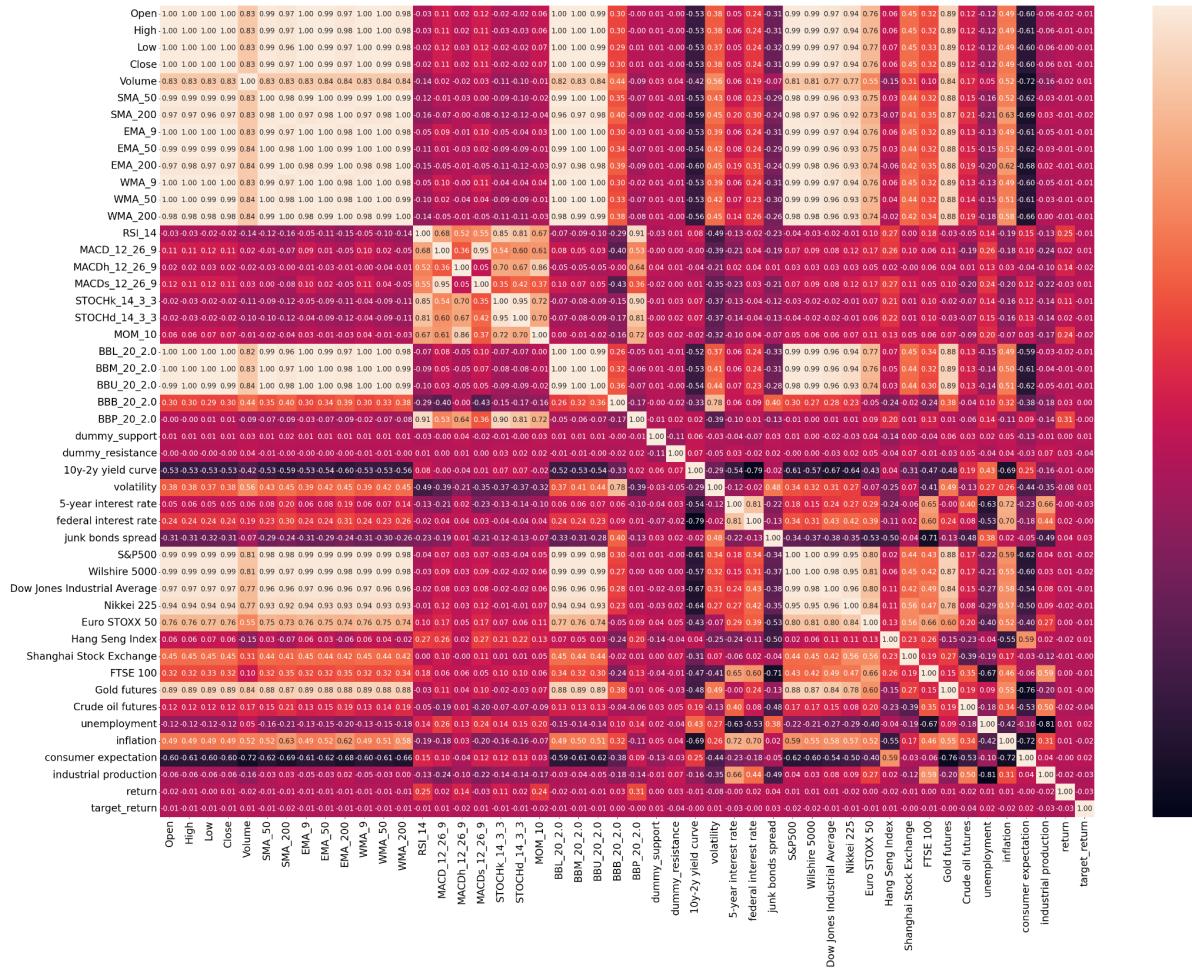


Figura 5: mapa de calor con las correlaciones entre variables

Variable 1	Variable 2	Correlación
industrial production	unemployment	-0,80703
10y-2y yield curve	federal interest rate	-0,78573
Gold futures	consumer expectation	-0,75932
consumer expectation	inflation	-0,71645
consumer expectation	Volume	-0,71561
junk bonds spread	FTSE 100	-0,71203
SMA_200	consumer expectation	-0,68601
10y-2y yield curve	inflation	-0,68577
EMA_200	consumer expectation	-0,67894
10y-2y yield curve	Dow Jones Industrial Average	-0,66855
unemployment	FTSE 100	-0,66623
WMA_200	consumer expectation	-0,65860
Nikkei 225	10y-2y yield curve	-0,63910
unemployment	5-year interest rate	-0,62704
S&P500	consumer expectation	-0,62237

Tabla 3: 15 pares de variables más negativamente correlacionadas

Variable 1	Variable 2	Correlación
EMA_9	WMA_9	0,99993
High	Open	0,99985
Low	Close	0,99981
Low	Open	0,99978
Close	High	0,99978
High	Low	0,99972
WMA_50	EMA_50	0,99962
SMA_50	EMA_50	0,99959
Close	Open	0,99958
WMA_9	High	0,99950
WMA_9	Open	0,99939
EMA_9	High	0,99937
WMA_50	SMA_50	0,99921
EMA_9	Open	0,99920
WMA_50	BBM_20_2.0	0,99920

Tabla 4: 15 pares de variables más positivamente correlacionadas

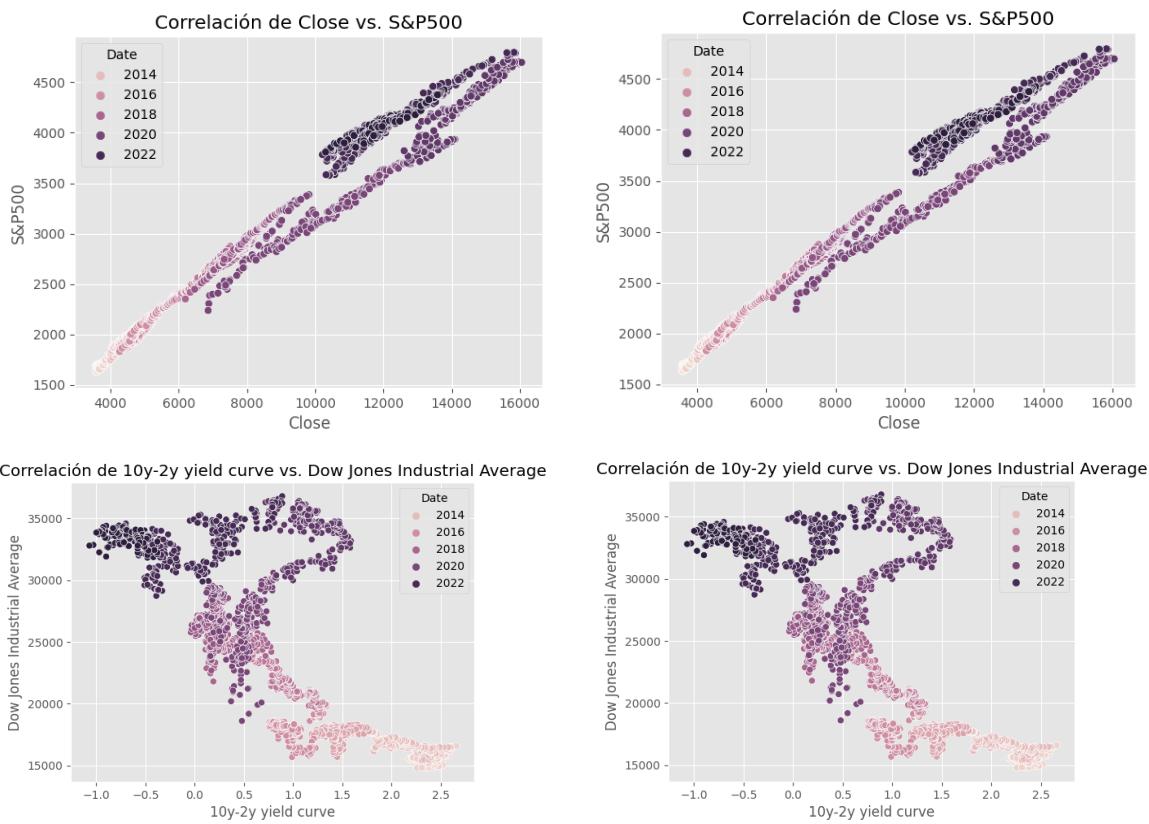


Figura 6: correlaciones temporales en gráficos de dispersión

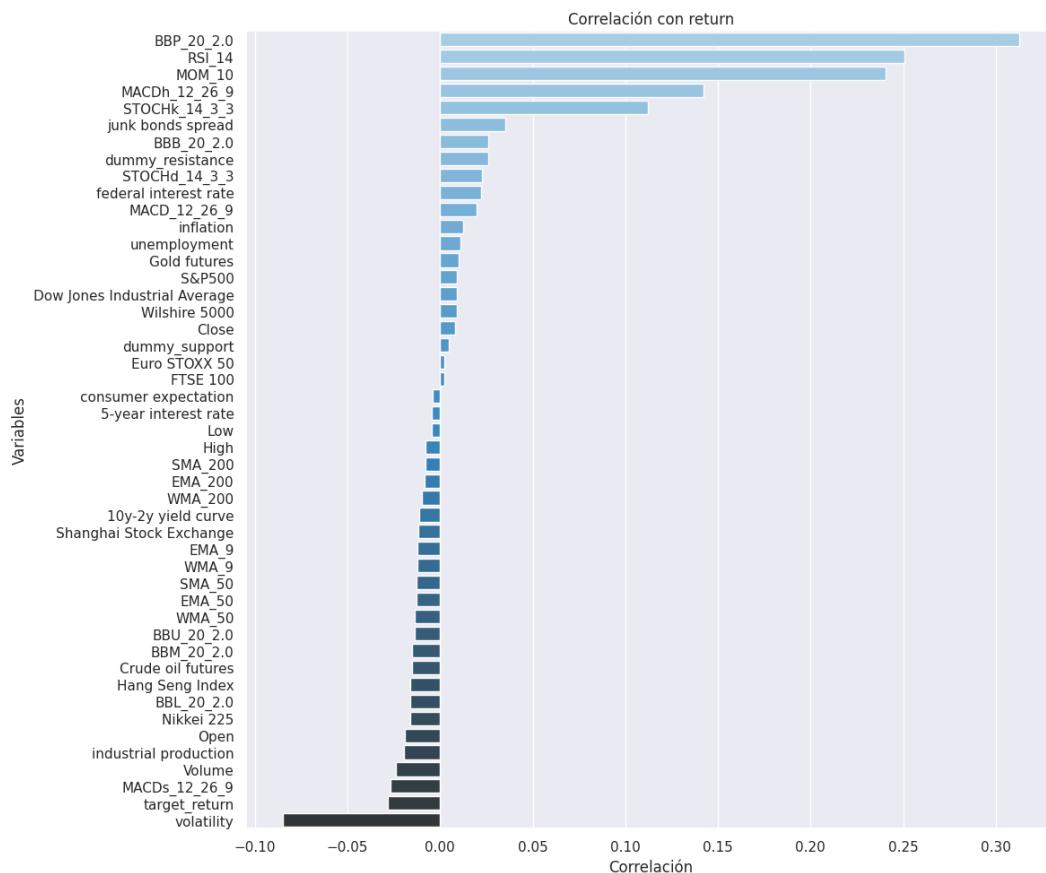


Figura 7: correlación de las variables con un proxy de la variable objetivo.

lag	lb_stat	lb_pvalue	Autocorrelación
1	4,26314	0,03895	Sí
2	4,40240	0,11067	No
3	4,94339	0,17599	No
4	4,95163	0,29230	No
5	4,95710	0,42114	No
6	5,00941	0,54261	No
7	5,38420	0,61319	No
8	5,38495	0,71575	No
9	7,07207	0,62962	No
10	7,41942	0,68534	No
11	7,45991	0,76071	No
12	8,07881	0,77894	No
13	8,31283	0,82264	No
14	8,34757	0,87044	No
15	11,82069	0,69255	No
16	11,93539	0,74842	No
17	13,04350	0,73327	No
18	16,11904	0,58424	No
19	18,49737	0,48948	No
20	18,89741	0,52850	No
21	23,63100	0,31128	No
22	23,63112	0,36686	No
23	23,66835	0,42236	No
24	23,68063	0,47998	No
25	23,70380	0,53654	No
26	24,58439	0,54259	No
27	24,63708	0,59479	No
28	25,41989	0,60490	No
29	25,42000	0,65628	No
30	25,73100	0,68879	No

Tabla 5: Resultados de la Prueba de Ljung-Box para cada rezago

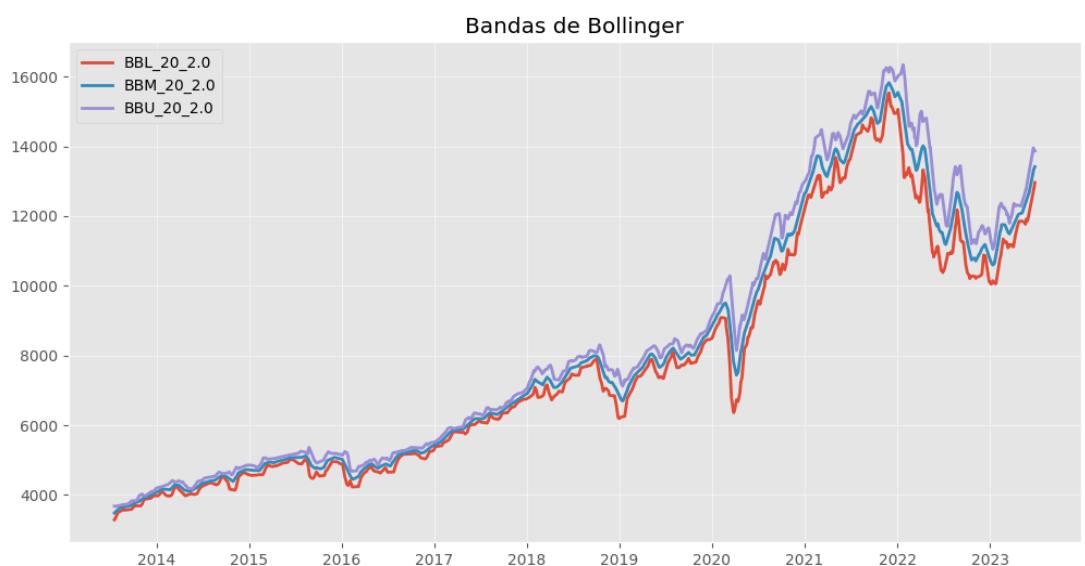
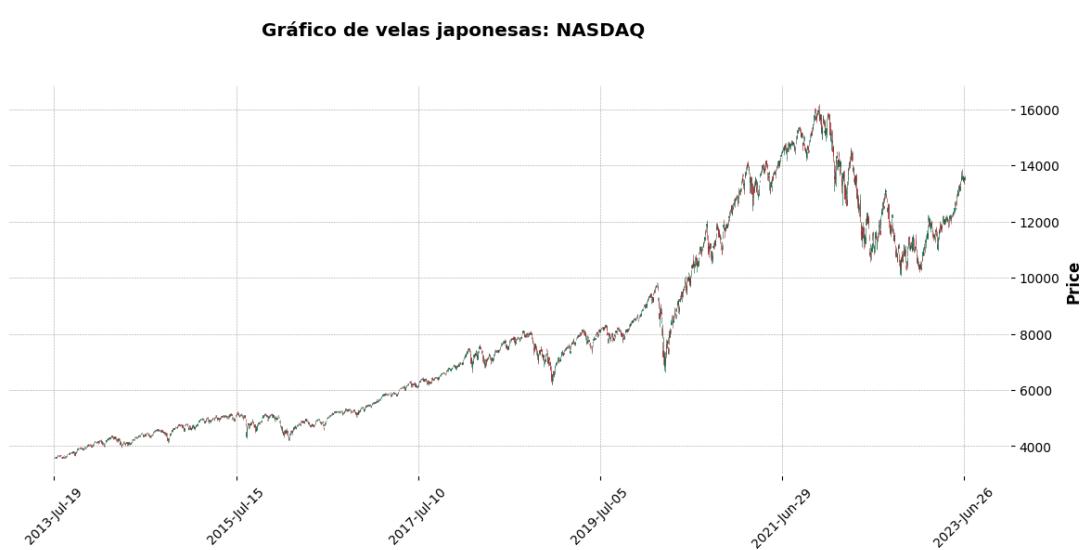
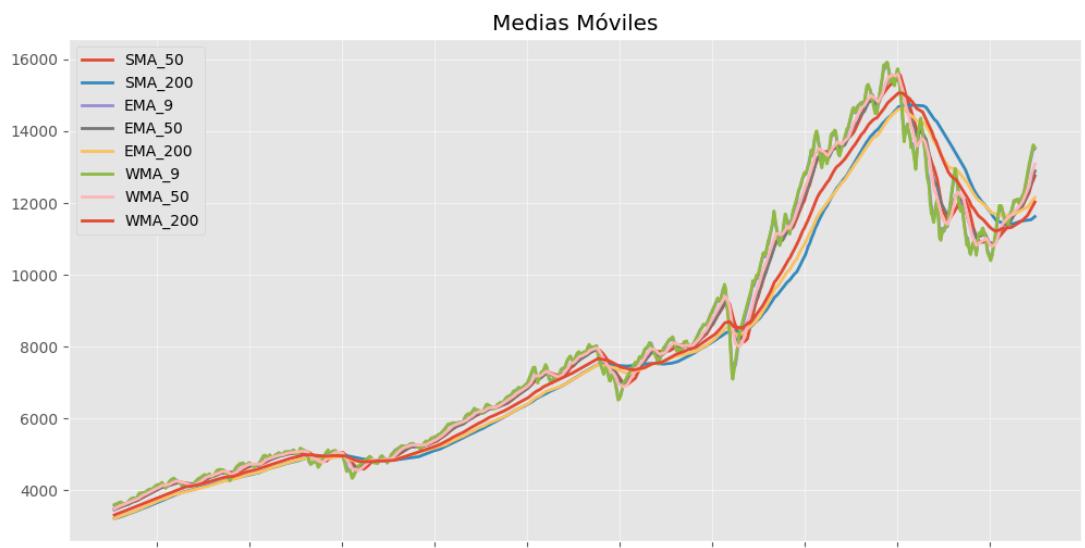


Figura 8: gráfico de las variables OHLC en forma de velas japonesas, medias móviles y bandas de bollinger para suavizar el ruido.

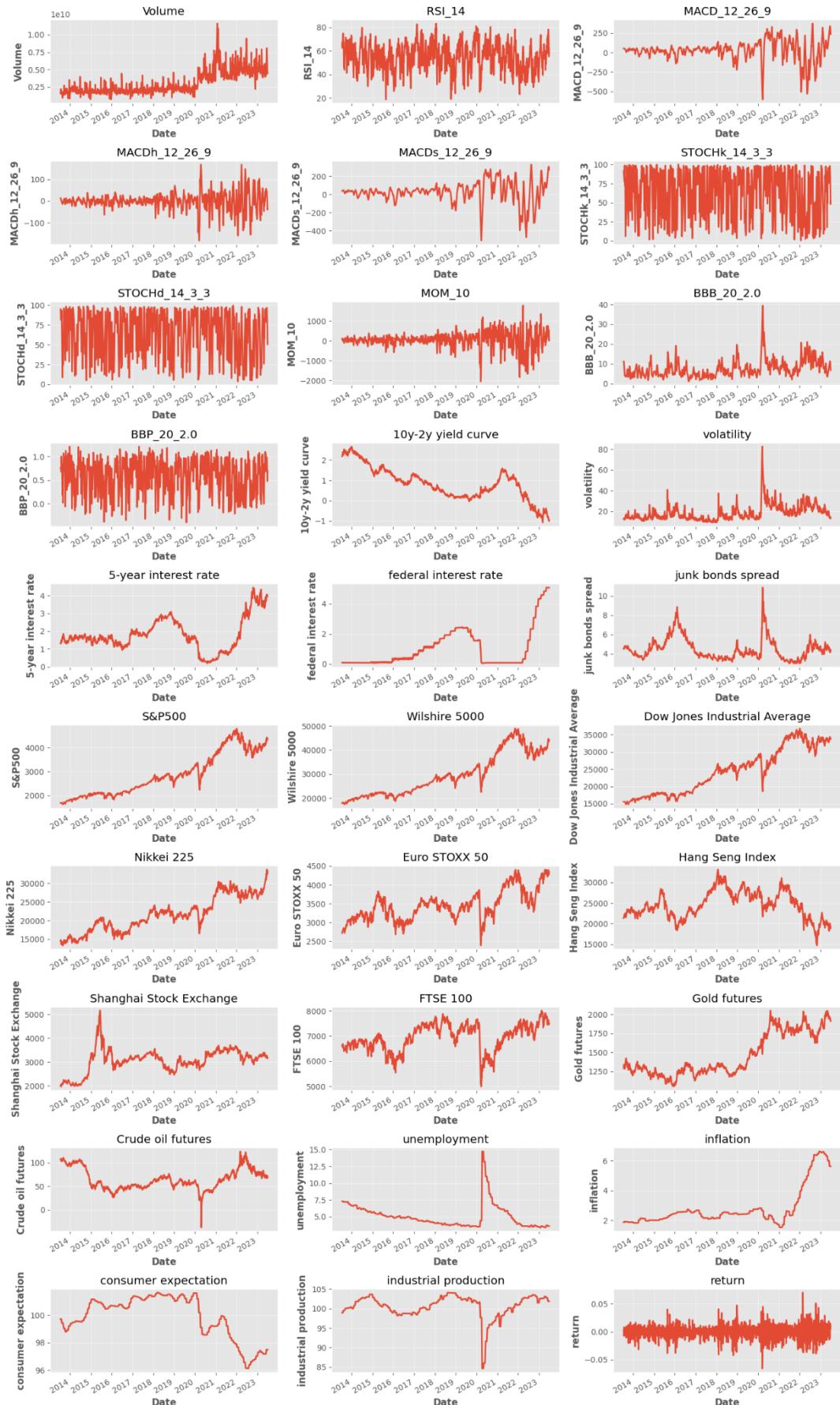


Figura 9: gráficos temporales para analizar la presencia de estacionalidad

Variable	ADF Statistic	p-value	Estacionalidad
Open	-0,588505	8,74E-01	No
High	-0,516633	8,89E-01	No
Low	-0,480129	8,96E-01	No
Close	-0,488308	8,94E-01	No
Volume	-1,445306	5,60E-01	No
SMA_50	-0,583774	8,75E-01	No
SMA_200	-1,211964	6,68E-01	No
EMA_9	-0,441712	9,03E-01	No
EMA_50	-0,508351	8,90E-01	No
EMA_200	-0,630411	8,64E-01	No
WMA_9	-0,483814	8,95E-01	No
WMA_50	-0,490776	8,94E-01	No
WMA_200	-0,798933	8,19E-01	No
RSI_14	-10,654765	4,56E-19	Sí
MACD_12_26_9	-6,119729	8,92E-08	Sí
MACDh_12_26_9	-11,087477	4,16E-20	Sí
MACDs_12_26_9	-6,138934	8,06E-08	Sí
STOChk_14_3_3	-7,971635	2,77E-12	Sí
STOChd_14_3_3	-7,945206	3,23E-12	Sí
MOM_10	-9,057917	4,65E-15	Sí
BBL_20_2.0	-0,553377	8,81E-01	No
BBM_20_2.0	-0,577721	8,76E-01	No

Tabla 6.1: Resultados de la prueba Dickey-Fuller Aumentada

Variable	ADF Statistic	p-value	Estacionalidad
BBU_20_2.0	-0,477127	8,96E-01	No
BBB_20_2.0	-4,990967	2,31E-05	Sí
BBP_20_2.0	-13,196591	1,12E-24	Sí
10y-2y yield curve	-0,41419	9,08E-01	No
volatility	-5,186021	9,39E-06	Sí
5-year interest rate	-0,55951	8,80E-01	No
federal interest rate	1,698014	9,98E-01	No
junk bonds spread	-2,883943	4,72E-02	Sí
S&P500	-0,570709	8,77E-01	No
Wilshire 5000	-0,737435	8,37E-01	No
Dow Jones Industrial Average	-0,928951	7,78E-01	No
Nikkei 225	-0,788178	8,23E-01	No
Euro STOXX 50	-2,436442	1,32E-01	No
Hang Seng Index	-2,045885	2,67E-01	No
Shanghai Stock Exchange	-2,834831	5,35E-02	No
FTSE 100	-3,085451	2,76E-02	Sí
Gold futures	-0,759308	8,31E-01	No
Crude oil futures	-2,268127	1,82E-01	No
unemployment	-3,319689	1,40E-02	Sí
inflation	-0,925136	7,80E-01	No
consumer expectation	-2,120791	2,36E-01	No
industrial production	-3,555197	6,68E-03	Sí
return	-11,714823	1,46E-21	Sí

Tabla 6.2: Resultados de la prueba Dickey-Fuller Aumentada

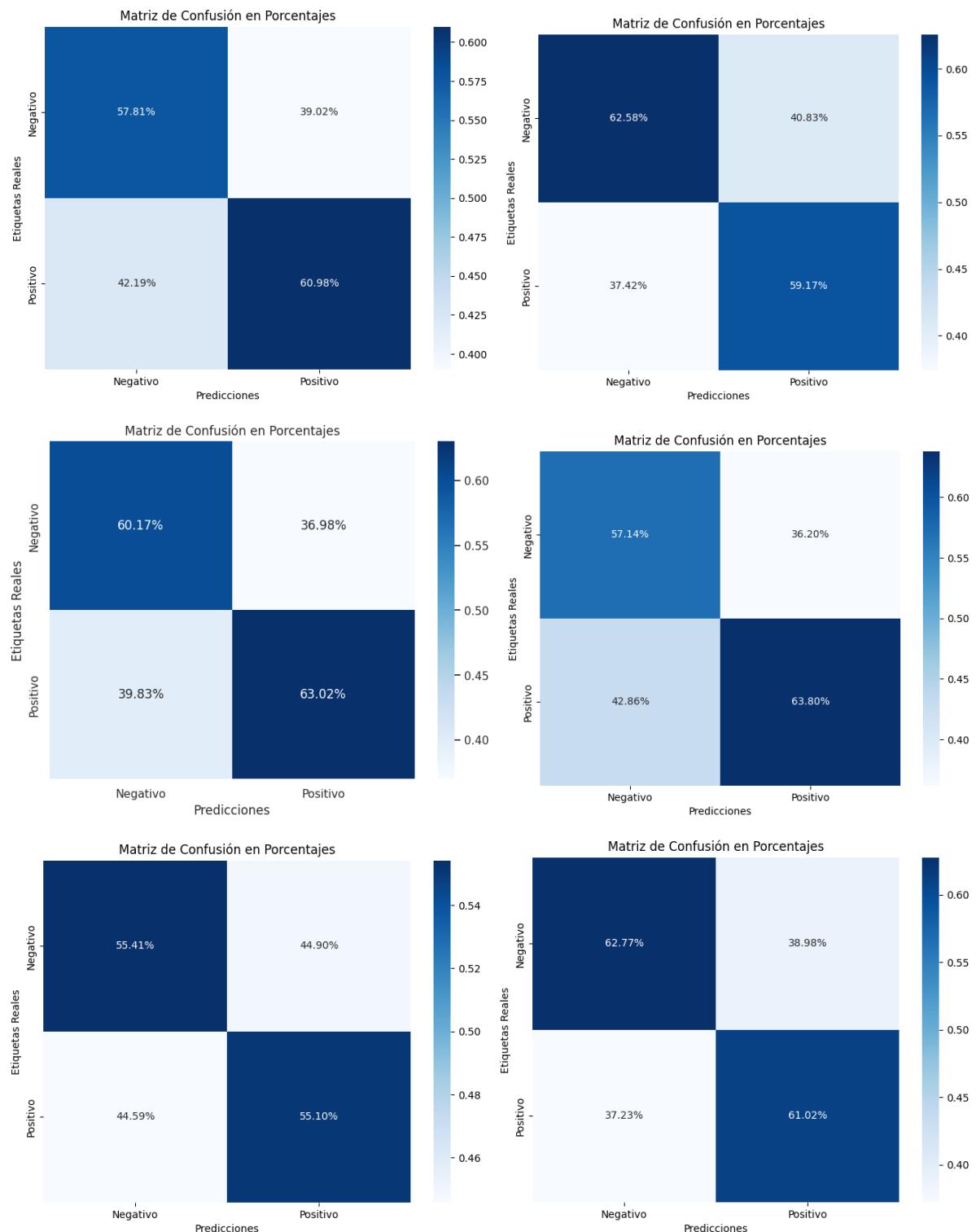


Figura 14: Matrices de confusión en porcentajes. De izquierda a derecha y de arriba hacia abajo: regresión logística, Random Forest, XGBoost, SVM, LSTM y KNN+XGBoost.

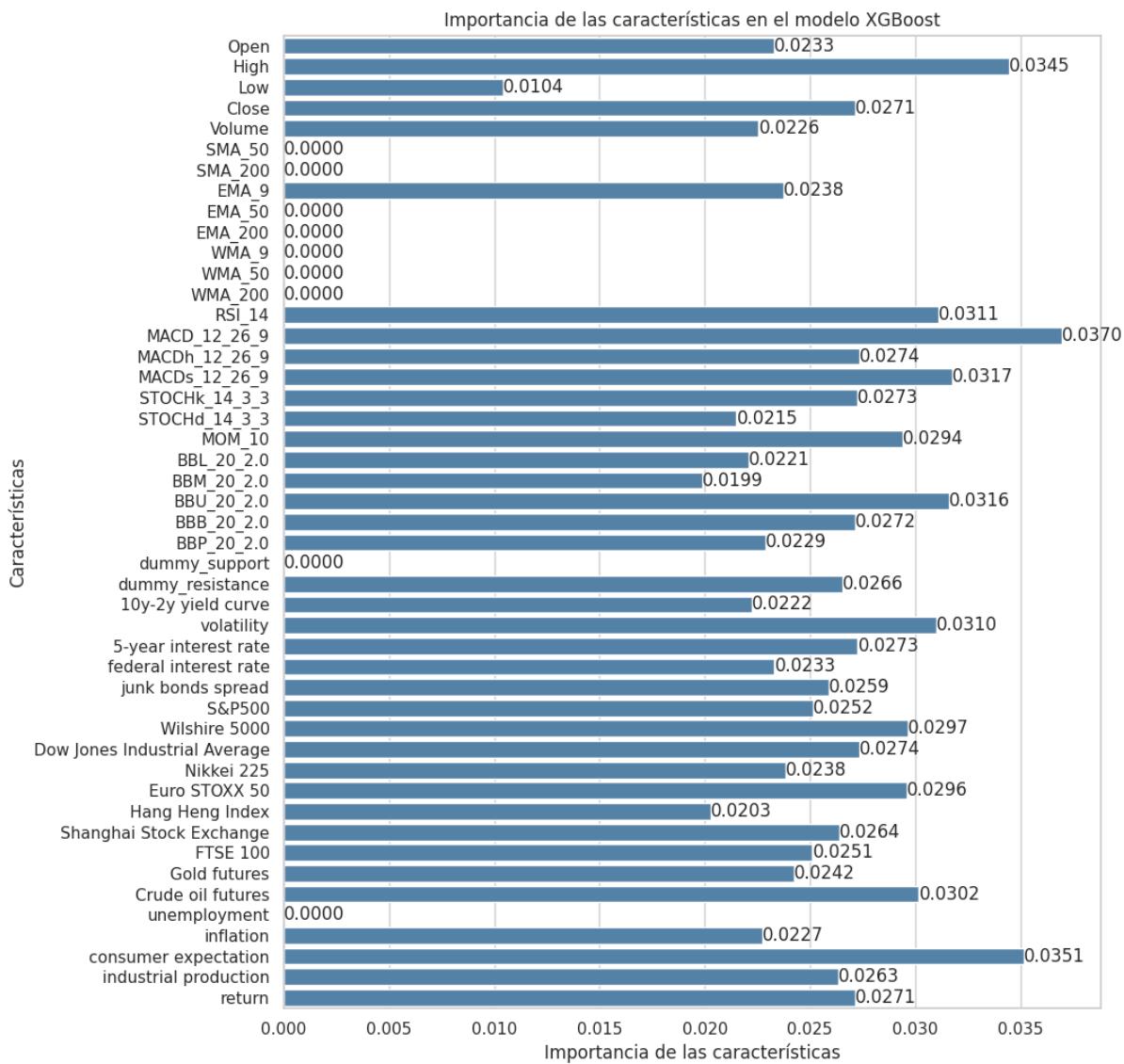


Figura 15: Importancia de las características en el modelo de XGBoost

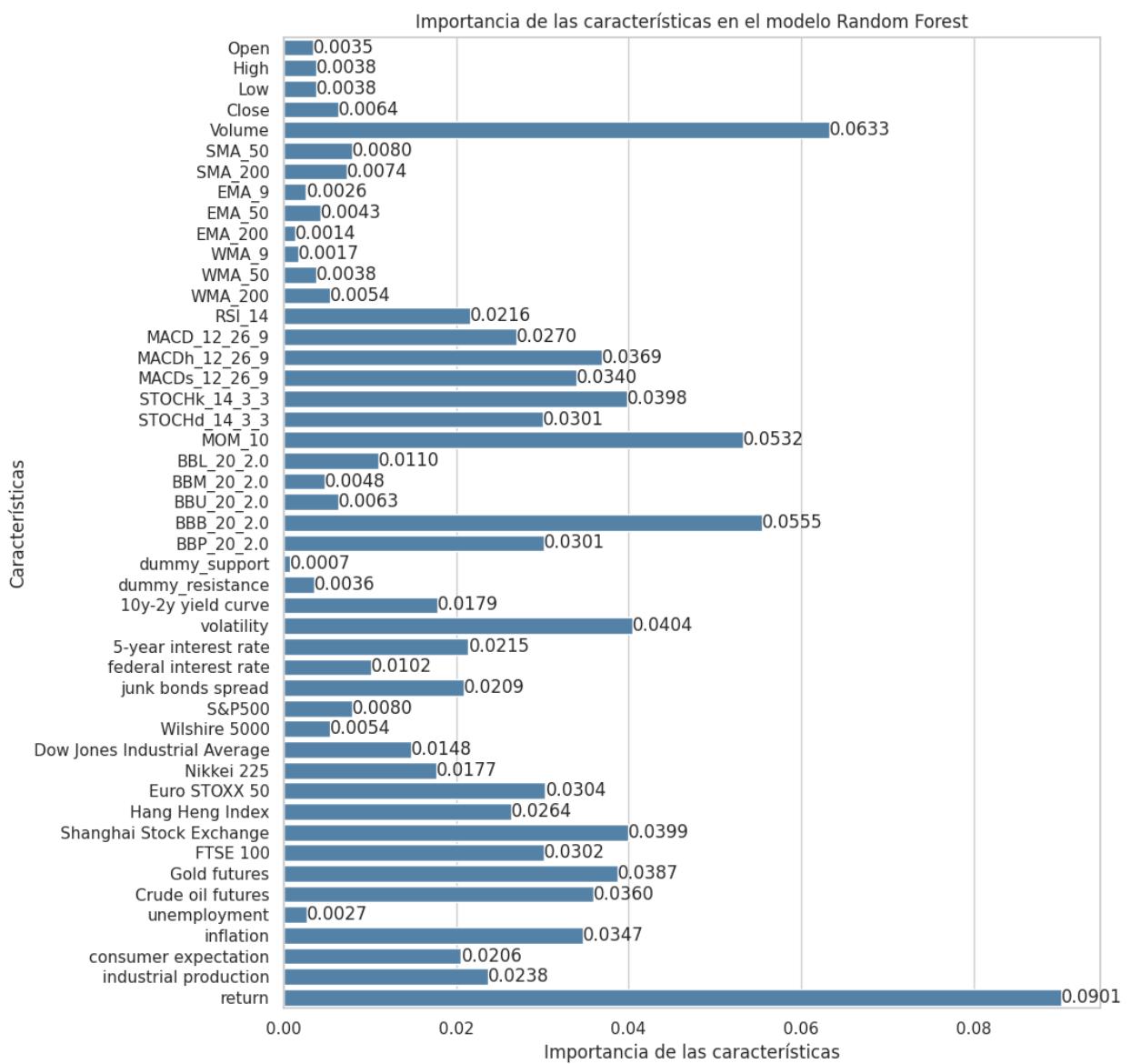


Figura 16: Importancia de las características en el modelo de Random Forest

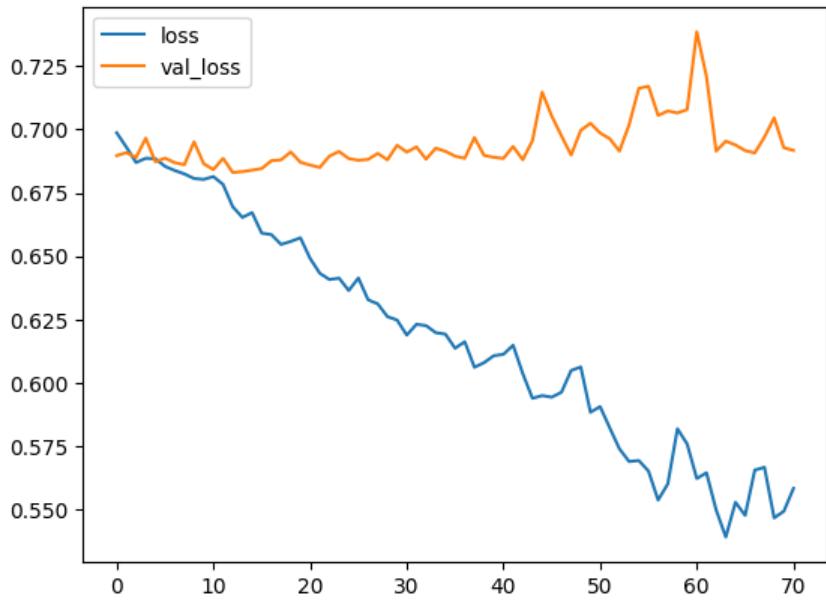


Figura 17: Sobreajuste en la función de pérdida (binary crossentropy) durante el entrenamiento de las redes neuronales LSTM.

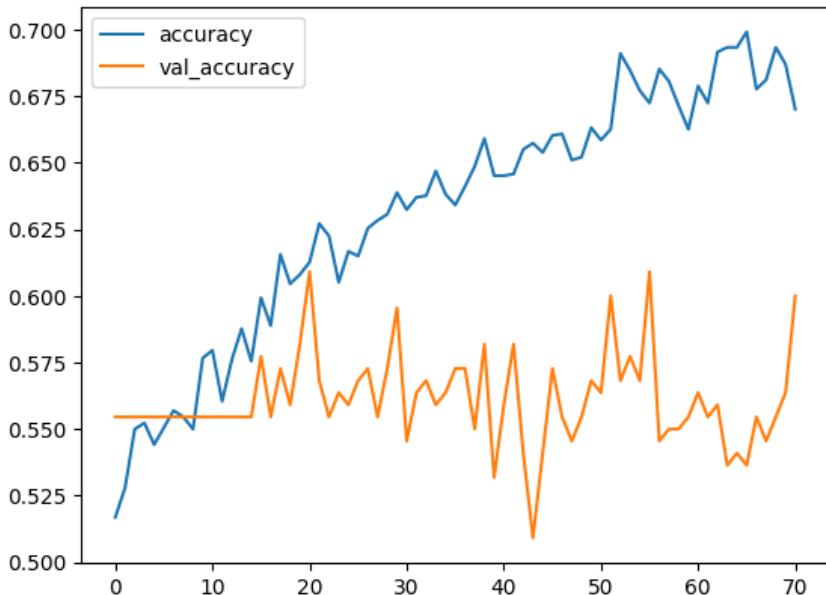


Figura 18: Sobreajuste en la precisión durante el entrenamiento de las redes neuronales LSTM.