Análisis comparativo de series de tiempo financieras usando modelos econométricos y de redes neuronales: el caso de mercado libre

Bulman Germann Gustavo

Tesis de grado

Licenciatura en Economía

Facultad de Ciencias Económicas

Universidad Nacional de Misiones

Director: Dr. Dip Juan Antonio

Diciembre de 2024

Resumen

Este trabajo realiza un análisis comparativo entre dos metodologías para la predicción de series de tiempo financieras: el modelo econométrico ARIMA y las redes neuronales recurrentes del tipo LSTM. El estudio se centra en la acción de Mercado Libre (MELI), cotizada en el *NASDAQ*, utilizando datos diarios desde su oferta pública inicial *(IPO)* en 2007 hasta 2024. Se desarrollan cuatro enfoques: (1) un modelo *ARIMA*, conocido por su capacidad para capturar relaciones lineales en los datos; (2) un modelo derivado de *ARIMA* el cual considerará los quiebres estructurales; (3) un modelo *LSTM*, que puede modelar relaciones no lineales y manejar secuencias temporales; y (4) un modelo híbrido *ARIMA-LSTM* que combina las fortalezas de ambos enfoques.

El objetivo principal es evaluar cuál de estos modelos ofrece un mejor rendimiento predictivo en términos de precisión, utilizando métricas como el error cuadrático medio *(MSE)* y el error absoluto medio *(MAE)*. Los resultados muestran que, mientras que el modelo ARIMA es efectivo para predicciones a corto plazo bajo dato lineales, las redes neuronales recurrentes, especialmente el modelo *LSTM*, presentan un mejor desempeño en escenarios no lineales. El modelo híbrido logra aprovechar las ventajas de ambos enfoques, ofreciendo predicciones más precisas en general.

El estudio concluye que, para el caso de la empresa de Mercado Libre, el uso de técnicas avanzadas de aprendizaje automático combinadas con métodos econométricos tradicionales permite obtener resultados más robustos y fiables, lo que es especialmente relevante para los inversores que buscan mejorar sus estrategias de predicción financiera.

Palabras clave: Series de tiempo, ARIMA, LSTM, Mercado Libre, predicción financiera.

1. Introducción

A lo largo de la historia, el mundo financiero ha captado un interés significativo tanto de agentes públicos como privados. Con el desarrollo de los mercados, han surgido diversos tipos de análisis destinados a predecir con mayor precisión los movimientos futuros de éstos. Dentro de este contexto, se han elaborado varias teorías y enfoques que buscan explicar y anticipar el comportamiento de las acciones en la Bolsa de Comercio.

Por un lado, el análisis fundamental, desarrollado en 1934 por los economistas Benjamin Graham y David Dodd, se centra en la inversión en valor. Este método analiza los datos financieros de una empresa, como sus balances y flujos de caja, para determinar el valor intrínseco de sus acciones y evaluar si están subvaloradas o sobrevaloradas en el mercado, lo cual puede representar una oportunidad de inversión.

Por otro lado, existen teorías alternativas como el análisis técnico, introducido a finales del siglo XIX por Charles Henry Dow (Dow, 1884), el cual, busca predecir los precios futuros de las acciones mediante la identificación de patrones repetitivos en los gráficos. Los defensores del análisis técnico sostienen que, aunque los movimientos del mercado puedan parecer aleatorios, existen patrones históricos que el mercado tiende a seguir.

La relevancia de estas teorías ha llevado a los economistas a desarrollar y perfeccionar modelos econométricos y de inteligencia artificial con el objetivo de mejorar la predictibilidad de los precios de los activos financieros. Este trabajo tiene como propósito la implementación de modelos paramétricos, utilizando herramientas estadísticas y econométricas, específicamente el modelo ARIMA y su derivado considerando quiebres estructurales, y modelos no paramétricos mediante el entrenamiento de redes neuronales recurrentes *(RNN)* bajo una arquitectura del tipo *LSTM*, además, se empleará un modelo hibrido que haga uso de las ventajas de cada estructura. El objetivo es determinar cuál de estos modelos se aproxima más a los datos reales futuros de los precios de cierre del activo en cuestión.

La motivación para emplear estas herramientas proviene del análisis de técnicas utilizadas en el análisis técnico, tanto chartista, basado en patrones gráficos, como de indicadores técnicos que emplean métodos estadísticos. Aunque estas metodologías se han basado en el valor histórico de las acciones para estimar sus precios futuros y han mostrado resultados relativamente aceptables, presentan ciertas deficiencias en el rigor metodológico, es por ello que este trabajo busca corregir algunas deficiencias mediante la implementación de técnicas más avanzadas y también complementar a las existentes.

En particular, el análisis se centrará en el mercado argentino, evaluando los modelos en la empresa líder de comercio electrónico Mercado Libre, la cual fue seleccionada debido a su magnitud medida en capitalización bursátil, siendo la más grande de Argentina. Para ello, se utilizarán datos de cotización desde el 10 de agosto de 2007 hasta el 30 de julio de 2024, abarcando así diversas crisis financieras y fluctuaciones en el precio de la acción.

Dado el acceso limitado, mediante las plataformas de acceso gratuito a datos de mercado de la empresa en la bolsa argentina (MELI.BA), se emplearán los datos de las cotizaciones diarias de la firma provenientes del *NASDAQ (MELI: NASDAQ)* bajo su cotización en dólares americanos, en el periodo considerado.

Para comparar los modelos se emplearán metodologías del tipo estadístico evaluando la capacidad predictiva de cada modelo mediante sus errores, entre ellos: *Mean Square error (MSE) y Mean Absolute Error (MAE)*. Estas métricas fueron seleccionadas debido a su fácil interpretación a la hora de comparar la precisión predictiva de diversos modelos. Además, una gran cantidad de algoritmos de *Machine Learning* demuestran una alta eficacia al minimizar los valores de *MSE* y *MAE*, lo que respalda su uso en el análisis comparativo de rendimiento.

El trabajo está organizado de la siguiente manera: se presentará el marco teórico el cual contiene conceptos claves y antecedentes, además se evaluará la aplicación de las técnicas de estimación, la comparación de los distintos enfoques y finalmente las conclusiones del trabajo.

2. Objetivos

2.1. Objetivo general

Comparar la performance de predicción entre un modelo econométrico del tipo *ARIMA*, un derivado del mismo considerando los quiebres estructurales, un modelo de redes neuronales recurrente y por último un modelo hibrido que contemple las fortalezas de los modelos antes planteados, para variaciones en el precio de la acción de Mercado Libre (MELI).

2.2. Objetivos específicos

1. Analizar y revisar la literatura existente para fundamentar el marco teórico sobre el mercado financiero y el precio de los activos.
2. Estudiar las técnicas de predicción habituales, con especial énfasis en los modelos ARIMA y las redes neuronales con sus variantes.
3. Desarrollar un modelo econométrico para analizar el comportamiento del precio de las acciones de Mercado Libre.
4. Construir una arquitectura de redes neuronales con la finalidad de entrenar valores pasados en búsqueda de predecir valores futuros del activo en cuestión.
5. Contrastar los resultados de ambas metodologías utilizadas para determinar cuál es más efectiva en términos de predicción.

3. Planteamiento del problema

Fundada en 1854, la Bolsa de Comercio de Buenos Aires es la más antigua de América Latina. Sin embargo, a pesar de su temprana creación, el mercado bursátil argentino presenta un desarrollo relativamente limitado en comparación con la magnitud del producto nacional (Heredia, 2008).

Históricamente, los inversores independientes han buscado predecir el valor futuro de las acciones utilizando diversos métodos de análisis desarrollados por distintos autores. No obstante, esta tarea requiere considerable tiempo, conocimientos y experiencia. Debido a estas limitaciones, la predicción de precios futuros de activos suele estar en manos de consultoras especializadas que cuentan con equipos de investigadores, analistas financieros y matemáticos.

Actualmente, varios autores estudian la predicción del valor futuro de los activos utilizando técnicas econométricas y de aprendizaje automático (Heredia, 2008). Aunque se han obtenido resultados específicos concluyentes, no se ha identificado un modelo perfecto que funcione de manera universal e independiente del mercado o del activo en cuestión.

Este estudio propone desarrollar y comparar tres enfoques metodológicos para la predicción de precios de acciones: en primer lugar, un modelo econométrico del tipo *ARIMA*; en segundo lugar, un modelo basado en técnicas de aprendizaje automático mediante redes neuronales recurrentes *LSTM;* y, finalmente, un enfoque híbrido que combina ambos métodos. Este modelo híbrido no ha sido aplicado hasta el momento para predecir el precio de una acción argentina. Por ello, el objetivo principal es evaluar la performance de estas metodologías en la predicción del precio de las acciones de Mercado Libre en el contexto del mercado argentino, contrastando sus resultados en términos de precisión.

Con base en los resultados, se procederá a contrastar la robustez del modelo utilizando otros activos. Para ello, se seleccionarán empresas representativas de sectores similares en diferentes países: Amazon (AMZN) en Estados Unidos, Alibaba (BABA) en China y Globant (GLOB) como un referente argentino adicional.

Un aspecto crucial en la evaluación de la predicción del precio de las acciones de Mercado Libre es verificar la hipótesis de eficiencia del mercado. Según esta hipótesis, los precios de las acciones reflejan toda la información disponible, lo que hace que sea imposible predecir los movimientos futuros basándose únicamente en la información histórica.

A partir de este análisis, el trabajo busca responder las siguientes interrogantes fundamentales:

1. ¿Se verifica la hipótesis de eficiencia de los mercados en el caso de las acciones de Mercado Libre?
2. ¿Es factible predecir el precio de las acciones de Mercado Libre utilizando un modelo econométrico?
3. ¿La aplicación de redes neuronales recurrentes mejora la precisión de la predicción en comparación con el modelo ARIMA?
4. ¿Cuál de las metodologías empleadas ofrece un mayor rendimiento en términos de precisión predictiva?

4. Marco teórico

4.1. Conceptos claves

4.1.1. Mercado financiero

Los mercados financieros están formados por todos los agentes que facilitan la transferencia de dinero a lo largo del tiempo. Generalmente, se distinguen dos tipos de agentes con características claramente definidas: las unidades con superávit de fondos y las unidades con déficit de fondos.

Las unidades con déficit de fondos obtienen dinero de las unidades con superávit, a cambio de compromisos de pago futuros, conocidos comúnmente como activos financieros. Al llegar el vencimiento del plazo acordado las unidades con déficit devuelven los fondos a las unidades con superávit y, adicionalmente, pagan una renta o interés por el uso de dichos fondos. (Lucero, 2012)

En la literatura sobre mercados financieros, se reconocen dos principales enfoques teóricos, por un lado, encontramos la teoría de las expectativas racionales, y por el otro lado, la teoría conductista de las finanzas. (Ramírez, 2007)

La primera se originó a partir de la Teoría de Dow para luego evolucionar hacia la Teoría de las Expectativas Racionales, fundamentada en la Hipótesis del Mercado Eficiente *(EMH por sus siglas en inglés).* Esta hipótesis postula que la mano invisible del mercado corrige cualquier anomalía presente en el mismo. (Mishkin, 2014)

Por otro lado, la teoría conductista (Watsons, 2013) de las finanzas se opone a los principios del mercado eficiente. Esta teoría explica cómo los patrones de conducta no completamente racional de los inversores influyen en los precios de las acciones de manera distinta a lo que plantea la *EMH.*

La teoría de las expectativas racionales y, por ende, la *EMH* sostienen que un mercado es eficiente si los precios de los activos reflejan rápidamente toda la información disponible. Fama (1969) establece dos supuestos para que esto se cumpla:

- Los precios actuales cambiaran rápidamente para ajustarse al nuevo valor intrínseco o teórico derivado de la nueva información.
- El periodo que transcurre entre dos ajustes sucesivos de precios o entre dos informaciones sucesivas de un mismo título, es una variable aleatoria independiente.

La teoría de las expectativas racionales afirma que los participantes en el mercado, al buscar su propio interés, basan sus decisiones en el supuesto de que los demás participantes harán lo mismo. (Soros, 2009)

Roberts (1967) estableció tres niveles de eficiencia en los mercados financieros, basados en el grado de información que los precios reflejan de manera rápida y precisa. Estos niveles son:

1. **Eficiencia débil**, que postula que los precios de los activos ya incorporan toda la información derivada de los precios y rendimientos históricos, por lo que el análisis técnico sería ineficaz para generar rendimientos anormales.
2. **Eficiencia semifuerte**, la cual afirma que, además de los precios históricos, los precios de los activos reflejan toda la información pública disponible, lo que haría inútil el análisis fundamental para obtener rendimientos extraordinarios.
3. **Eficiencia fuerte**, que sostiene que los precios de los activos incorporan toda la información disponible, tanto pública como privada, lo que implica que ni el análisis técnico ni el fundamental, ni siquiera la información privilegiada, permitirían obtener beneficios superiores de manera consistente.

Basándonos en lo anterior, entendemos que un mercado de valores es eficiente cuando la competencia entre los distintos participantes, guiados por el principio del máximo beneficio, conduce a una situación

de equilibrio en la que el precio de mercado de cualquier título constituye una buena estimación de su precio teórico o intrínseco. (Ramírez, 2007)

4.1.2. National Association of Securities Dealers Automated Quotations (NASDAQ)

En Estados Unidos, el *NASDAQ,* fundado en 1971, es una de las bolsas de valores más grandes del mundo. Para cotizar en el NASDAQ, las empresas deben cumplir con requisitos específicos como niveles mínimos de ingresos, ganancias, capitalización de mercado, flujos de caja, y normas de gobierno corporativo, incluyendo auditorías y la presencia de directores independientes (Latina, 2023).

4.1.3. Herramientas de análisis

Hay dos principales conjuntos de herramientas que los analistas emplean para intentar predecir el comportamiento de los mercados. El primero se conoce como Análisis Técnico el cual parte de la premisa de que el precio lo descuenta todo y que es la única variable relevante, junto con el volumen de negociación, por lo que para su uso se debe analizar solo la variable precio, a través de distintos procedimientos, para invertir o no dependiendo del movimiento de éste. En segundo lugar, el Análisis Fundamental, el cual no considera que el precio refleje toda la información necesaria, de hecho, estima que el precio por el que cotiza un valor no refleja el precio real de una empresa por lo que el objetivo con este análisis es encontrar ese valor real o intrínseco e invertir si un valor está infravalorado o desinvertir si está sobrevalorado.

4.1.4. Series de tiempo

Una serie temporal es una secuencia de N observaciones ordenadas y equidistantes cronológicamente sobre una característica (serie univariante o escalar) o sobre varias características (serie multivariante o vectorial) de una unidad observable en diferentes momentos. Este componente temporal distingue los conjuntos de datos de series temporales de otros tipos de datos, presentando tanto limitaciones como oportunidades únicas para extraer información.

Una serie temporal del tipo univariante se puede expresar de la forma:

$$y_1, y_2, \dots, y_N; (y_t)_{t=1}^{N}; (y_t : t = 1, \dots, N)$$

Ecuación 1: Serie de tiempo univariada

Donde y_t es la observación $n°t$ $(1 \leq t \geq N)$ de la serie y N es el número de observaciones de que consta la serie completa. Las N observaciones $y_1, y_2, \dots., y_N$ pueden recogerse en un vector columna $y \equiv [y_1, y_2, \dots, y_N]'$ de orden $Nx1$.

4.1.5. Pronóstico de series de tiempo

La previsión de series temporales representa un dominio crítico dentro del aprendizaje automático, particularmente porque muchos problemas de previsión implican inherentemente una dimensión temporal.

La metodología de la previsión de series temporales implica predecir valores futuros mediante el análisis de patrones de datos históricos y la suposición de la continuación de estas tendencias en el futuro. Este proceso conlleva ajustar modelos a las observaciones pasadas para hacer predicciones informadas sobre eventos futuros. La previsión de series temporales, por lo tanto, sirve como un enfoque basado en datos para la planificación efectiva y eficiente, abordando desafíos predictivos que incluyen un aspecto temporal.

Las aplicaciones de los modelos de series temporales son extensas, abarcando dominios como la previsión de ventas, las predicciones meteorológicas y las predicciones financieras. Estos modelos son particularmente hábiles en manejar escenarios caracterizados por la incertidumbre respecto a los resultados futuros. El objetivo principal de la previsión de series temporales es estimar valores futuros o clasificaciones en puntos específicos del tiempo futuro, aprovechando los datos históricos para informar estas proyecciones. (Latif et al. 2023).

Los datos de series temporales pueden tener los siguientes componentes:

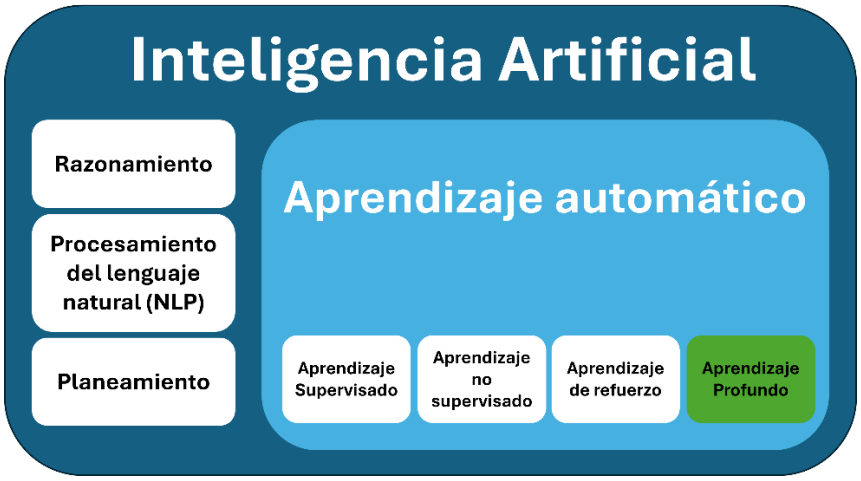
- Irregularidad: Este componente caracteriza los movimientos imprevisibles en cuanto a su ocurrencia, impacto y duración, los cuales son considerados como aleatorios.
- Estacionalidad: Este componente se caracteriza por patrones regulares y repetitivos que ocurren en intervalos fijos. Por ejemplo, en una serie temporal mensual con una periodicidad de 12, el patrón estacional se repite cada 12 meses.
- Tendencia: Se observa una tendencia cuando una serie temporal muestra un aumento persistente, una disminución o se mantiene constante durante un período prolongado.
- Ciclicidad: Esta componente refleja las fluctuaciones recurrentes a largo plazo que ocurren en la serie, pero que no tienen una periodicidad fija como los componentes estacionales.

Con el objetivo de entender el funcionamiento de las series temporales y la utilización de información histórica para la predicción de datos futuros, se emplean modelos econométricos y técnicas de aprendizaje automático basadas en inteligencia artificial. Estos enfoques, que se detallarán posteriormente, son seleccionados por su capacidad predictiva avanzada y su eficacia en la modelización de patrones temporales complejos.

4.1.6. Inteligencia artificial

La IA puede entenderse como la forma más amplia de describir sistemas que pueden pensar. Como se ilustra en la Figura 1, hay cuatro subconjuntos principales de IA. En este trabajo, se hará enfoque en el aprendizaje automático. Sin embargo, para entender el aprendizaje automático, es importante ponerlo en perspectiva.

Figura 1: Estructura de IA



Fuente: Tanco, 2020

El término Aprendizaje automático *(Machine Learning)* fue utilizado por primera vez en 1959 por Arthur Samuel, sin embargo, ha ganado relevancia en los últimos años debido al aumento de la capacidad de computación y al boom de los datos. (Kirsch et al., 2018).

El aprendizaje automático es una forma de inteligencia artificial que permite a un sistema aprender a partir de datos sin necesidad de programación explícita. Utiliza algoritmos que, al entrenarse con datos, pueden mejorar la precisión de los modelos predictivos. Un modelo de aprendizaje automático se genera al entrenar un algoritmo con datos, y luego, al proporcionarle un input, produce un output, como una predicción basada en los datos de entrenamiento. Este enfoque es esencial para crear modelos analíticos precisos (Kirsch et al., 2018).

Dentro del aprendizaje automático, el Aprendizaje profundo *(Deep Learning)* utiliza redes neuronales artificiales con múltiples capas para aprender de manera iterativa a partir de datos, especialmente útiles para identificar patrones en datos no estructurados. Estas redes, que imitan el funcionamiento del cerebro humano, permiten a las computadoras abordar abstracciones y problemas mal definidos (Kontopoulou et al., 2023).

En 1936, Alan Turing fue pionero en estudiar el cerebro desde la perspectiva de la computación. Sin embargo, los primeros en establecer los fundamentos de la computación neuronal fueron Warren McCulloch, neurofisiólogo, y Walter Pitts, matemático, quienes en 1943 desarrollaron una teoría sobre el funcionamiento de las neuronas.

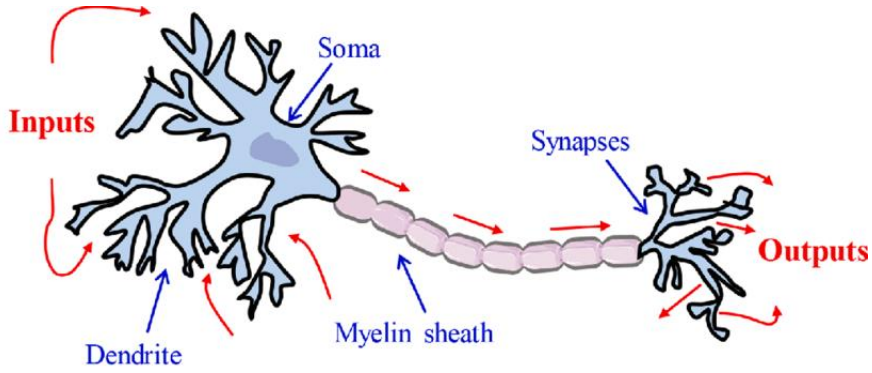
En 1959, Bernard Widrow y Marcial Hoff desarrollaron el modelo ADALINE *(ADAptive LINear Elements),* la primera red neuronal aplicada a un problema real, que consistía en filtros adaptativos para eliminar ecos en las líneas telefónicas (Tanco, 2020).

Finalmente, en 1986, Rumelhart, Hinton y Williams dieron a conocer el algoritmo *de back-propagation.* Ese mismo año, Rumelhart y McClelland publicaron el libro *Parallel Distributed Processing*, el cual se convirtió en una influencia clave para la aplicación generalizada del algoritmo de *back-propagation*.

4.1.7. Neuronas biológicas

El cerebro está compuesto por neuronas, elementos altamente conectados. Las neuronas tienen tres partes principales: dendritas, cuerpo celular (soma) y axón. Las dendritas reciben señales eléctricas, el cuerpo celular integra estas señales, y el axón las transporta hacia otras neuronas. La comunicación entre neuronas ocurre en la sinapsis, donde un axón hace contacto con la dendrita de otra célula. La longitud de la sinapsis está relacionada con la complejidad del proceso químico que regula la función de la red neuronal (Tanco, 2020). Un esquema simplificado de cómo se interconectan dos neuronas biológicas se muestra en la figura 2.

Figura 2: Neurona biológica



Fuente: Muhammed Nizem

Las neuronas transmiten información mediante potenciales de acción, impulsos eléctricos que recorren los axones. Estos potenciales de acción no pueden pasar directamente de una célula a otra, por lo que la comunicación entre neuronas se realiza a través de neurotransmisores liberados en las sinapsis. Existen sinapsis excitadoras, que facilitan la generación de impulsos, y sinapsis inhibidoras, que estabilizan el potencial de la membrana y dificultan la emisión de impulsos. La neurona recibe señales tanto de sinapsis excitadoras como inhibidoras, y la suma de estos impulsos en el cuerpo celular determina si la neurona será activada o no, dependiendo de si se supera el umbral necesario para generar un potencial de acción. (Tanco, 2020).

4.1.8. Neuronas artificiales

Existen redes que imitan las redes neuronales biológicas y se utilizan para aprender estrategias de resolución a partir de ejemplos de patrones de comportamiento típicos. El conocimiento de una red neuronal no se almacena en un conjunto de instrucciones. Su eficacia radica en su topología y en los valores de las conexiones (pesos) entre las neuronas.

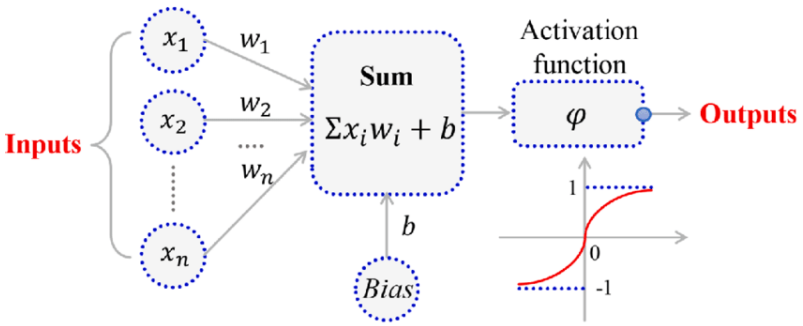
Las ventajas de las redes neuronales incluyen:

- Aprendizaje adaptativo: Capacidad de aprender a realizar tareas basadas en un entrenamiento o experiencia inicial.

- Autoorganización: Una red neuronal puede organizar y representar la información que recibe de manera autónoma a través de un proceso de aprendizaje.
- Generalización: Habilidad de las redes neuronales para responder correctamente a datos o situaciones nuevas, que no han sido previamente expuestas a la red.
- Tolerancia a fallos: Aunque una red neuronal sufra daños parciales, su estructura solo se degrada parcialmente, y algunas de sus capacidades pueden mantenerse. Además, estas redes son capaces de reconocer patrones incluso cuando los datos están ruidosos, distorsionados o incompletos. (Tanco, 2020).

El modelo de una neurona artificial es una imitación del proceso de una neurona biológica. En la figura 3 se observa una neurona artificial en forma general.

Figura 3: Neurona artificial



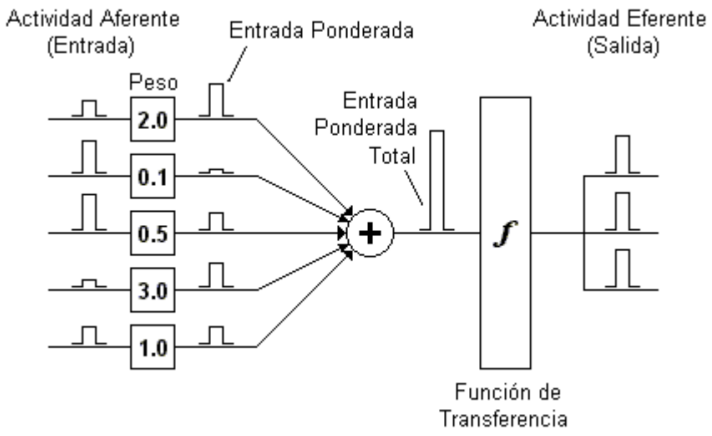
Fuente: Muhammed Nizam

A partir de esta, podemos observar las similitudes que tiene la versión artificial con su versión biológica.

- Las entradas X_i representan las señales que provienen de otras neuronas y que son capturadas por las dendritas, es decir, aquellas que alimentan la neurona.
- Los pesos W_i son la intensidad de la sinápsis que conecta dos neuronas. En este caso cada entrada X_i es multiplicada por un peso W_i cuya función es análoga a la de la función sináptica de la neurona biológica. Los pesos pueden ser positivos (excitatorios), o negativos (inhibitorios). En un proceso de entrenamiento, esos pesos son ajustados con el fin de minimizar el error de la predicción futura.
- En la tercera etapa, $\sum X_i W_i + \mathrm{b}$ se realiza la suma ponderada del proceso anterior agregándole un componente de sesgo el cual le permite a la neurona desplazar la función de activación y ayudar a la red a ajustarse mejor a los datos.
- En la última etapa, φ hace referencia a la función de activación o función umbral que la neurona debe sobrepasar para activarse; este proceso ocurre biológicamente en el cuerpo de la célula.

Para entender el proceso completo de una neurona artificial podemos observar la figura 4, donde se puede comprender el funcionamiento de los distintos pesos a las informaciones entrantes.

Figura 4: Proceso de una neurona artificial



Fuente: Tanco, 2020

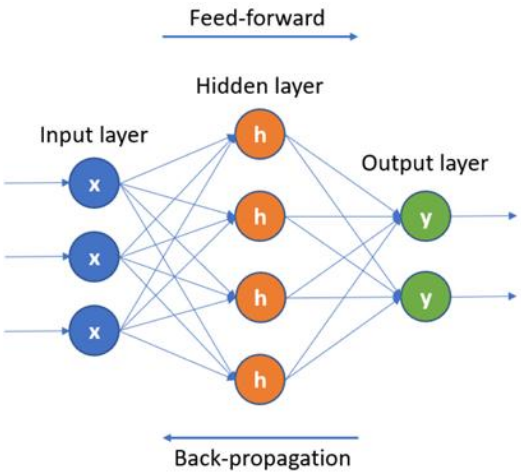
4.1.9. Redes Neuronales Recurrentes (RNN)

Esta clase de redes neuronales incluye al menos tres capas: la capa de entrada, una o más capas ocultas, y la capa de salida. El número de nodos en la capa de entrada depende de las características de los datos de entrada, y estos nodos se conectan a las capas ocultas mediante sinapsis, cada una con un factor de ponderación específico. Este factor determina qué señales pueden pasar.

En las capas ocultas, los nodos aplican una función de activación a la suma ponderada de las entradas para generar las señales de salida. La capa de salida produce un vector de probabilidades para las posibles salidas y selecciona aquella con el error mínimo. El proceso de *back-propagation* se utiliza para ajustar iterativamente los pesos de las sinapsis, minimizando la función de costo.

El modelo se considera entrenado cuando esta función de costo se ha minimizado. La capacidad y complejidad del modelo aumentan al añadir más capas ocultas y nodos a la arquitectura (Kontopoulou et al. (2023)).

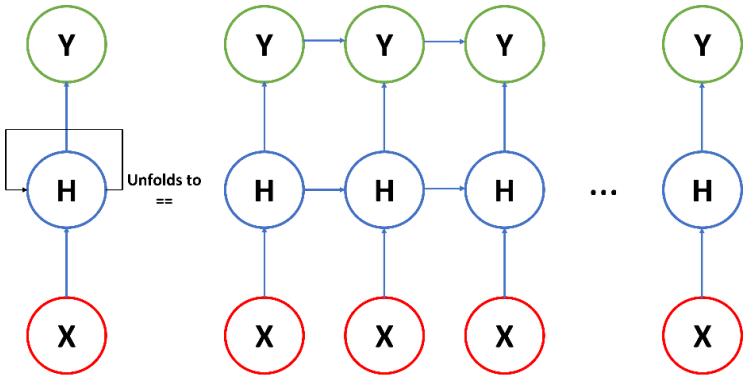
Figura 5: Estructura de RNN



Fuente: Kontopoulou

De manera análoga, una red neuronal recurrente consiste en un modelo de Deep Learning que esta entrenado para procesar y convertir una entrada de datos secuencial en una salida de datos secuencial específica. En las RNNs, las capas ocultas actúan como búferes para almacenar la información recopilada en etapas anteriores al leer los valores de datos sucesivos. La arquitectura y la lógica detrás de los bloques constructivos de las RNN se presentan en la Figura 6.

Figura 6: Estructura de RNN

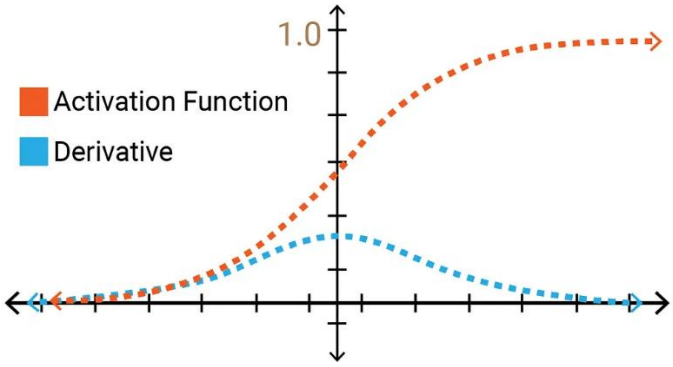


Fuente: Panagopoulos et al.

El principal desafío con una *RNN* típica es la limitación de la cantidad de datos que pueden mantener en su memoria en cada paso de entrenamiento, este problema se llama gradiente desaparecido. En el entrenamiento de redes neuronales, el objetivo es minimizar una función de pérdida ajustando los pesos de la red mediante el algoritmo de retropropagación *(backpropagation)*. Este algoritmo calcula los gradientes de la función de pérdida con respecto a cada peso de la red y utiliza estos gradientes para actualizar los pesos. Sin embargo, en redes neuronales profundas, donde hay muchas capas entre la entrada y la salida, los gradientes pueden disminuir exponencialmente a medida que se propagan hacia atrás desde las capas de salida hacia las capas de entrada. Esto puede suceder debido a la multiplicación repetida de valores menores que uno.

A modo de ejemplificación, podemos observar en la figura 7 como el uso de una función de activación incorrecta puede afectar en gran medida el poder de predicción mediante el problema del gradiente desaparecido.

Figura 7: Función de activación Incorrecta

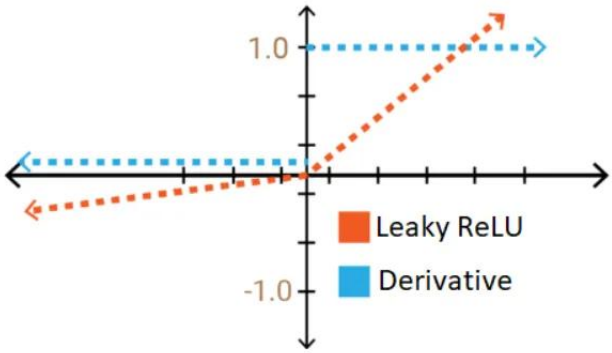


Fuente: Palak Jain

Como resultado de ello se puede producir un entrenamiento lento o estancado, además de computacionalmente costoso, además, la red puede no aprender representaciones útiles de los datos, lo que lleva a un rendimiento subóptimo.

Como solución a este inconveniente se utilizan arquitecturas específicas de RNN, las cuales contemplan funciones de activación como *Rectified Linear Unit (ReLU)* ya que ésta, al utilizar otro mecanismo, no sufre tanto la desaparición del gradiente como se puede observar en la figura 8.

Figura 8: Función de activación ReLU



Fuente: Palak Jain

Este problema se resuelve utilizando el carril de memoria, un concepto introducido por las redes de memoria recurrente de largo-corto plazo.

4.2. Antecedentes

En este apartado se definen los antecedentes y trabajos previos significativos que motivaron el presente trabajo.

Tras una exhaustiva investigación sobre modelos de predicción en series temporales, tanto financieras como no financieras, se ha identificado una variedad de enfoques y aportaciones de distintos investigadores. Sin embargo, los resultados son variados y no evidencian la predominancia de un tipo de modelo por sobre los demás.

4.2.1. Internacionales

En lo concerniente a investigaciones realizadas a nivel internacional, podemos nombrar aquellos autores que estudiaron diversos modelos y concluyeron que el modelo *Long Short Term Memory (LSTM)* es aquel que mejor se adapta a las circunstancias de predicción en series temporales.

Por un lado, nos encontramos con el trabajo de Kontopoulou et al. (2023) quienes realizaron comparaciones de distintos estudios poniendo a prueba los modelos en el sector financiero, sector de salud, en la predicción meteorológica, consumo de agua, energia, entre otros. Como resultados generales, arribaron a que los modelos de redes neuronales ganan mayor poder de prediccion en comparación a los modelos de prediccion estadísticos clásicos.

De manera similar, los autores Rhanoui et al. (2019) buscaron aplicar los modelos de predicción sobre la evolución del presupuesto, para ello, utilizaron un modelo *ARIMA* y un *LSTM* y tras su estudio concluyeron que la arquitectura de un modelo *LSTM* se ajusta mejor a los datos que uno de *ARIMA*.

Empleando modelos similares, Menacho (2014) decidio salirse del campo financiero y realizo sus mediciones aplicando sus modelos a 8 series de tiempo obtenidas del portal del Instituto Nacional de Estadística e Informática (INEI). Entre estas series se encuentran: Producción en la construcción, Producción de electricidad, Fabricación de papel y productos de papel, fabricación de productos textiles, producción minera e hidrocarburos, produccion de madera y productos de madera, fabricacion de prendas de vestir y producción sector fabril. Como herramientas de comparación utilizó las medidas para la medición del error del pronóstico *Mean Absolute Percentage Error (MAPE), Mean Absolute Deviation (MAD), Mean Squared Error (MSE)*. Al igual que los estudios anteriores, éste mostro que los modelos de redes neuronales tuvieron menores valores de *MAPE* en todas las series y menores valores de *MAD y MSE* en cuatro de las 8 series analizadas, concluyendo que los modelos basados en aprendizaje automático se ajustan más a los datos.

Estos resultados mayormente favorables para los modelos de redes se dan en general dado sus capacidades auto-adaptativas, generalización, aprendizaje y representación de funciones no lineales en las series de tiempo con patron de tendencia fluctuante en el tiempo.

De manera análoga, los autores Namini et al. (2019) fueron más alla y buscaron comparar tanto el modelo *ARIMA* y el *LSTM* como su versión más compleja *BiLSTM*, el cual a diferencia del modelo simple *LSTM*, emplea dependencias bidireccionales a largo plazo entre unidades de tiempo de series de tiempo. Con esto en juego, ellos buscaron conocer el grado de acierto en la predicción de valores financieros de distintos paises y sectores, para ello, utilizaron como base de datos el índice financiero japonés *Nikkei 225 index,* el índice financiero chino *Hang Seng Index*, los tres índices americanos: el

NASDAQ composite index, el S&P 500 commodity Price index y Dow Jones Industrial Average index y un activo individual el cual fue *IBM stock data*. Los datos fueron recolectados de manera diaria, semanal y mensual comprendidos entre el año 1985 y 2018, mientras que la variable analizada fue el precio de cierre ajustado y se empleo una ratio de 70:30, la cual implica que 70% de los datos fueron utilizados para entrenamiento, mientras que el 30% restante fue utilizado para el testeo de los resultados. Tras el estudio, se arribo mediante el análisis del *Root Mean Square Error (RMSE)* a que el modelo *BiLSTM* se ajustó mejor a los datos que sus alternativas en un contexto de series de tiempo financieras.

De manera análoga, existen autores que arribaron a conclusiones en las cuales el modelo *ARIMA* es aquel que se ajusta mejor a los datos de series temporales. Entre ellos encontramos a Yamak et al. (2019) quienes realizaron sus estudios, buscando comparar modelos *ARIMA, LSTM y GRU* aplicados a la cotización diaria del Bitcoin medidos en dólares americanos. Para ello, evaluaron datos desde el 28 de noviembre de 2014 hasta el 5 de junio de 2018, lo cual les permitió concluir que *ARIMA* tiene un mayor poder de predicción por sobre los otros modelos, medidos en términos de sus *RMSE*. Por otro lado, mediante el estudio los autores notaron que el modelo de pronóstico llevado a cabo mediante *ARIMA*, arrojó los resultados de manera más veloz que los otros modelos, ganando mayor eficiencia.

Por último, tenemos aquellos autores que concluyeron que ambos modelos son buenos para la predicción, sin embargo, cada uno cumple un rol especifico. Para ello, Hua (2020) estudio estos modelos aplicandolo sobre los precios de cierre de Bitcoin y Ethereum con un intervalo de 5 segundos, obteniendo de esta manera una base de datos con 10000 precios de cierre, dentro de los cuales 8000 fueron empleados en la fase de entrenamiento y los 2000 restantes en la fase de testeo. Como resultados obtuvieron dos situaciones:

- En primer lugar, cuando el objetivo es la predicción a corto plazo, el modelo *ARIMA* puede funcionar mejor, y a su vez hacerlo en menor tiempo.
- En segundo lugar, cuando se buscan predicciones a largo plazo, el modelo *LSTM*, si bien tarda más tiempo en realizar el entrenamiento, tiene un mayor grado de acierto en las predicciones futuras.

De manera similar, Navmeen Latif (2023) utilizó las cotizaciones de Bitcoin con una periodicidad de 10 minutos contemplados desde las 00hs del 21 de diciembre del 2020 hasta las 16hs del 21 de diciembre del 2021. Como resultados encontró que los tres modelos predijeron de manera acertada el precio del activo en cuestión, sin embargo tras otras comprobaciones se dió cuenta que para el caso de *ARIMA*, la predicción fue acertada dado que la tendencia futura fue la misma tendencia que la estudiada por el modelo, sin embargo , si la tendencia futura fuera distinta a la estudiada por el modelo, la estructura *ARIMA* tendría errores en la predicción, este resultado no se daría en los modelos de redes neuronales dada su capacidad de autoaprendizaje continuo.

Por último, Xingdan Huang et al. (2023), los autores compararon dos situaciones, por un lado, realizan una predicción estática, analizando los datos de precio de cierre de *Composite Index SSE* obtenidos de *China Stock Market Accounting Research (CSMAR).* Para ello, recopilan datos de 2288 días desde el 4 de enero de 2010 hasta el 8 de julio de 2019. Por otro lado, una situación dinámica con datos desde julio 9 de 2019 hasta diciembre 31 de 2019. Ellos, compararon ambas situaciones modelando *ARIMA, ARIMA-GARCH* y *LSTM* en cada una. Los resultados muestran que, en la predicción estática, la precisión de predicción general del modelo de series temporales es mayor que la del modelo de aprendizaje profundo. Sin embargo, en la predicción dinámica, el modelo de aprendizaje profundo tiene un mejor rendimiento.

Tras esta amplia indagación sobre los distintos modelos para predicción futura de valores, encontramos que no existe un modelo mejor que otro, sino que los modelos se adaptan a las situaciones y a los mercados para predecir mejor.

4.2.2. Evidencia para Argentina

A nivel local también existen autores que se dedicaron a estudiar estos modelos desde sus distintas aristas. Particularmente para el caso de Argentina, Pompilio et al. (2017) presentaron su análisis para el caso de la empresa argentina CAEX S.A. y el índice bursátil argentino MERVAL. Los autores buscaron comparar el grado de acierto entre el análisis fundamental, un modelo *ARIMA* y un modelo de redes neuronales. Como resultados obtuvieron buenas predicciones para los valores futuros utilizando la metodología de análisis funamental. Además, mediante el empleo de las otras metodologías obtuvieron resultados positivos en términos de la bondad de ajuste, permitiendo pronosticar la evolución promedio del precio de la acción, evidenciándose que los precios pronosticados siguen adecuadamente la evolución futura en los 33 días posteriores al último dato utilizado para la estimación, considerando en el pronóstico como variable dependiente el precio efectivizado de la acción en el mercado el día anterior.

Los autores agregan que estos pronósticos del precio de CAPX a un día, a pesar de ser muy cercanos a los precios que se efectivizaron en la realidad, no logran pronosticar cambios de tendencia de un día hacia el siguiente.

Por otro lado, Auza et al. (2019) aplicaron modelos híbridos *ARIMA-GARCH* con distintos valores para la predicción del valor futuro del índice Merval. Para su base de datos ellos contemplaron una serie histórica de retornos del Merval que dió inicio el 1 de enero de 2013 hasta el 6 de junio de 2016. Tras el estudio de las volatilidades pasadas para predecir retornos futuros concluyeron que para el caso de Argentina, como de muchos mercado emergentes, es necesario un modelo $E - GARCH \sim t(1,1)$ con modelos de media *ARMA (2,0)* y *ARMA (2,1)* resultando superiores dentro de la muestra.

Particularmente para el caso del mercado bursatil argentino, Dip et al. (2015) se interesó en la comparación entre un modelo *Arch-Garch* y uno de redes neuronales, específicamente un sistema de redes *backpropagation* con el objetivo de predecir el precio de la acción de Telecom Argentina S.A entre el periodo 2005-2012 obteniendo de esta manera 1666 datos.

Para ello, realizaron dos calibraciones de *ARCH-GARCH* y dos tipos de redes neuronales, como también, realizaron sus análisis dentro de la muestra y fuera de la muestra. Al analizar los porcentajes de aciertos y los errores de las cuatro propuestas de este trabajo se ha concluido en que el modelo *ARMA(1,1)-ARCH(2)* ha mostrado un mejor desempeño dentro de la muestra *(in sample).* Este modelo ha obtenido un porcentaje de aciertos del 99.29% y ha cometido un error promedio de 0.04. Por otro lado, se establece que la red 2 ha sido la que mejor ha estimado el precio de la acción fuera de la muestra *(out of sample).* Debido al hecho de que esta red ha obtenido un porcentaje de aciertos igual al 85.60% y ha alcanzado un error promedio de 0.27. Esto implica que las redes son aproximadoras de funciones universales, logrando modelar de mejor manera la tendencia. Con los resultados expuestos anteriormente, se puede concluir que las redes neuronales de retro propagación hacia atrás han manifestado su utilidad a la hora de trabajar con variables económicas e índices bursátiles, demostrando resultados aceptables.

4.3. Aporte del trabajo

Tras la amplia revisión de la literatura y enfoques de distintos autores, y entendiendo que no existe un modelo general que funcione de manera universal para todos los mercados y temporalidades, este trabajo buscara comprender mediante la aplicación de la metodología, aquel modelo que se ajuste en mayor medida a los datos en el mercado argentino, específicamente para el caso de la empresa Mercado Libre bajo su cotización en dólares americanos, empleando un modelo *ARIMA*, un *LSTM* y en especial un modelo hibrido que busque combinar ambas metodologías.

5. Metodología

En el desarrollo del presente trabajo, se utilizó Python como lenguaje principal de programación. Se empleó la *API* pública de *Yahoo Finance* a través de la librería *yfinance* (yfinance, (n.d.).) para la obtención de datos financieros. Para la construcción y predicción de modelos, se implementó la función *auto.arima* (Taylor, (n.d.).), lo que permitió explorar diversos parámetros del modelo. Asimismo, se llevaron a cabo pruebas manuales utilizando herramientas y librerías especializadas para modelos *ARIMA*, a fin de validar los resultados obtenidos.

Durante el análisis, se incorporó el *framework Keras* con *TensorFlow* (Developers T. , (n.d.)) como *backend*, y se *utilizó Scikit-learn* (Developers S.-l. , (n.d.)) en la etapa de preprocesamiento de datos, particularmente para la normalización mediante el método *MinMax*. Además, las librerías *Pandas* (Developers P. , (n.d.)) y *NumPy* (Developers N. , (n.d.)) fueron empleadas de forma extensiva para la manipulación y análisis de datos. Finalmente, *Matplotlib* (Developers M. , (n.d.)) se utilizó para la generación de gráficos y visualización de resultados.

5.1. Base de datos

Como se explicó anteriormente, dada las limitaciones para la obtención de datos de mercado para la firma en su cotización en pesos, se utilizara como base de datos las cotizaciones diarias de MERCADO LIBRE (NASDAQ:MELI) en su cotización en dólares americanos, entre el 10 de agosto de 2007 hasta el 30 de julio de 2024, esto es así dado que las cotizaciones diarias reflejan de mejor manera los movimientos de corto plazo para el activo en cuestión. Además, se consideró una base de datos que contemple 17 años (desde aproximadamente su *Inicial Public Offering (IPO*) hasta la actualidad) obteniendo así 4270 cotizaciones.

La precisión de un modelo de aprendizaje automático puede aumentar sustancialmente si se lo entrena con datos masivos. Sin datos suficientes, se intenta tomar decisiones sobre pequeños subconjuntos de los datos, lo que puede llevar a malinterpretar una tendencia o pasar por alto un patrón que recién comienza a surgir.

Para la capacitación de los modelos y su posterior evaluación, se utilizará una división de la serie de datos en un 80% para el entrenamiento y un 20% para la fase de testeo. Esta división se basa en una regla empírica que ha demostrado ser eficaz para garantizar un entrenamiento adecuado y una

evaluación confiable del modelo. Si bien es posible ajustar la proporción asignada a cada conjunto, incrementando el porcentaje de datos para el entrenamiento y reduciendo el destinado al testeo, una asignación excesiva de datos al entrenamiento puede resultar en un sobreajuste del modelo. Este fenómeno puede comprometer la capacidad del modelo para generalizar correctamente a nuevos datos, disminuyendo su desempeño en el conjunto de testeo (Kuhn et al., 2013).

Cabe destacar que en el periodo considerado ocurrieron una serie de crisis financieras, lo que nos permitirá evaluar con mayor detalle el comportamiento de nuestros modelos, como así también observar con que velocidad pudieron adaptarse a los cambios de tendencia.

A continuación, se puede observar la figura 9, la serie de tiempo completa en escala aritmética, la cual será empleada para el entrenamiento y testeo de los datos.

Figura 9: Precio de cierre MELI

Fuente: Elaboración propia con base a datos de yahoo finance

Tabla 1: Estadística descriptiva y precio de cierre MELI

Fecha	Precio de cierre		Estadística descriptiva	
2007-08-10	28,500000		Muestra	4270
2007-08-13	31,650000		Media	444,88
2007-08-14	30,030001		Desvío estandar	530,93
…	…		Min	8,28
2024-07-24	1642,550049		Max	1984,344
2024-07-25	1625,150024		25%	77,3
2024-07-26	1651,689941		50%	135,62
2024-07-29	1621,400024		70%	671,06

Fuente: Elaboración propia con base a datos de yahoo finance

El propósito de este análisis es predecir la cotización de la firma para los periodos siguientes observando como única variable independiente, las cotizaciones pasadas, entendiendo que las mismas se encuentras correlacionadas. Con el objetivo de aplicar el modelo *ARIMA*, vamos a seguir el *Box-Jenkins method.*

5.2. Modelo ARIMA

El término *ARIMA* hace referencia a Autoregresivo (AR) Integrado (I) y Media Móvil (MA), también conocido como el enfoque de Box-Jenkins. Un modelo *ARIMA* se especifica mediante 3 parámetros (p, d, q), tal que:

- p es el número de términos autorregresivos [AR (p)]: Un modelo de regresión que utiliza la relación de dependencia entre una observación y un número de observaciones rezagadas (parámetro del modelo p).
- d es el número de diferenciaciones [I (d)]: Calcula las diferencias entre observaciones en diferentes puntos de tiempo (parámetro del modelo d), con el objetivo de hacer la serie temporal estacionaria.
- q es el número de medias móviles [MA (q)]: Este enfoque considera la dependencia que puede existir entre observaciones y los términos de error creados cuando se utiliza un modelo de media móvil en observaciones que tienen un rezago temporal. (Rhanoui et al, 2019)

Un proceso del modelo AR de orden p, AR(p), puede ser escrito como:

$$x_t = c + \sum_{i=1}^{p} \phi_i x_{t-i} + \epsilon_t$$

Ecuación 2: proceso AR(p)

Donde x_t es la variable estática, c una constante, el termino ϕ_i son los coeficientes de autocorrelación con los retardos $1,2,\dots,p$ y ϵ_t son las muestras de la serie de ruido blanco gaussiano, con media cero y varianza σ^2.

Un modelo de medias móviles simples de orden q, MA(q), puede ser dado como:

$$x_t = \mu + \sum_{i=0}^{q} \Theta_i \epsilon_{t-i}$$

Ecuación 3: Proceso MA(q)

Donde μ es el valor esperado de x_t, Θ_i son los pesos aplicados a los valores actuales y pasados del término estocástico de la serie temporal.

Combinando los dos modelos, el de autorregresión y el de medias móviles, podemos crear un modelo ARMA del tipo (p,q):

$$x_t = c + \sum_{i=1}^{p} \phi_i x_{t-i} + \epsilon_t + \sum_{i=0}^{q} \Theta_i \epsilon_{t-i}$$

Ecuación 4: Proceso ARMA

Donde $\phi_i \neq 0, \Theta_i \neq 0, \sigma_\epsilon^2 > 0$. Los parámetros p, q constituyen el orden de los modelos AR y MA respectivamente.

La forma general de un modelo *ARIMA* es escrita como $ARIMA(p, d, q)$, incluyendo el término de integración el cual garantiza la estacionariedad de la serie de tiempo, y éste puede ser escrito como:

$$\Delta^d x_t = c + \sum_{i=1}^{p} \phi_i \Delta^d x_{t-i} + \sum_{i=0}^{q} \theta_i \epsilon_{t-i}$$

Ecuación 5: Proceso ARIMA

Donde Δx_t representa el modelo *ARIMA* diferenciado, Δ^d es un factor diferencial que introduce una diferencia de orden d, cuyo objetivo es eliminar la no estacionariedad de la serie temporal x_t (Kontopoulou, 2023).

En un modelo *ARIMA* se pueden realizar dos tipos de previsiones, por un lado, tenemos la previsión estática la cual implica una previsión de un paso adelante, donde se elige el valor verdadero para hacer el siguiente valor de la previsión, mientras que en la previsión continua o dinámica se elige reemplazar el valor verdadero con el valor ajustado para la previsión hacia adelante.

El modelo *ARIMA* funciona muy bien en la previsión a corto plazo de los precios de las acciones, sin embargo, la principal limitación que puede observarse al usar esta herramienta es su poder de predicción en valores de largo plazo. Además, posee una nula capacidad explicativa y por tanto simuladora, es decir, al ser una técnica donde se introduce una única variable no existe posibilidad de ponerla en relación con otras variables y, por tanto, no es posible soportar los resultados en ningún modelo económico.

Ahora bien, a pesar de estas limitaciones, la técnica, debido a su robustez estadística, es especialmente útil cuando deseamos realizar predicciones en el corto plazo, con un bajo ratio coste/eficiencia y con rapidez, de series que pueden tener un fuerte componente estacional, lo que la hace idónea para realizar predicciones de valores bursátiles.

De acuerdo a Dip y Romero (2015), las series financieras no presentan una varianza o variabilidad constante, por lo que las estructuras lineales son incapaces de explicar determinados aspectos que se encuentran presentes en las series financieras, como ser la volatilidad. Para ello, se incorpora un modelo que modelice esta características y son los modelos ARCH- GARCH. Estos autores mencionan que Engle (1982) introdujo los modelos ARCH (*Autoregressive Conditional Heteroskedasticity*), en los que

la varianza condicionada a la información pasada no es constante, sino que depende del cuadrado de las innovaciones anteriores. Su objetivo era demostrar que la incertidumbre variaba a lo largo del tiempo, un fenómeno conocido como heterocedasticidad. Más tarde, Bollerslev (1986) extendió este concepto con el modelo GARCH, incorporando la dependencia serial de la volatilidad, lo que permitió que las observaciones pasadas influyeran en la volatilidad futura. La formulación matemática es la siguiente:

1. Modelo de retorno: $y_t = \mu + \epsilon_t$. Donde Y_t es el retorno observado en el t , μ es la media de los retornos y ϵ_t es el término de error. Luego tenemos la Varianza Condicional $\epsilon_t = \sigma_t z_t$ en la cual σ_t es la desviación estándar condicional (volatilidad) en t y z_t es un ruido blanco.

2. El modelo GARCH modela la varianza condicional σ_t^2 como una función de los términos de error pasados y las varianzas pasadas:

$$\sigma_t^2 = \alpha_0 + \sum_{i=1}^{q} \alpha_i \epsilon_{t-i}^2 + \sum_{j=1}^{p} \beta_j \sigma_{t-j}^2$$

- σ_t^2 es la varianza condicional en el tiempo t,
- α_0 es una constante,
- ϵ_{t-i}^2 son los errores al cuadrado en momentos anteriores,
- σ_{t-j}^2 son las varianzas condicionales pasadas,
- α_i y β_j son los coeficientes estimados para los términos de error y varianzas pasadas, respectivamente,
- p es el orden de la varianza condicional (número de rezagos de la varianza pasada),
- q es el orden del error cuadrado (número de rezagos de los errores pasados).

La idea es modelizar esa volatilidad dentro del modelo, sin embargo, se pretende hacer hincapié más bien en modelizar la media de los retornos o precio que en la volatilidad en sí.

5.2.1. Enfoque Box-Jenkins

Esta metodología, tal y como se conoce en la actualidad, la desarrollaron los ingenieros George Box y Gwilym Jenkins en su obra *Time Series Analysis: Forecasting and Control* (Box et al. (2016) y desde sus inicios ha sido utilizada ampliamente en la predicción de variables económicas.

El análisis de una serie temporal se lleva a cabo en tres etapas. Primero, se evalúa si la serie es estacionaria utilizando la función de autocorrelación *(ACF),* la función de autocorrelación parcial *(PACF)* y la prueba de Dickey-Fuller aumentada *(ADF).* Si la serie no es estacionaria, se debe convertir en estacionaria mediante el proceso de integración. Luego, en la segunda fase, se identifica el modelo *ARMA* adecuado.

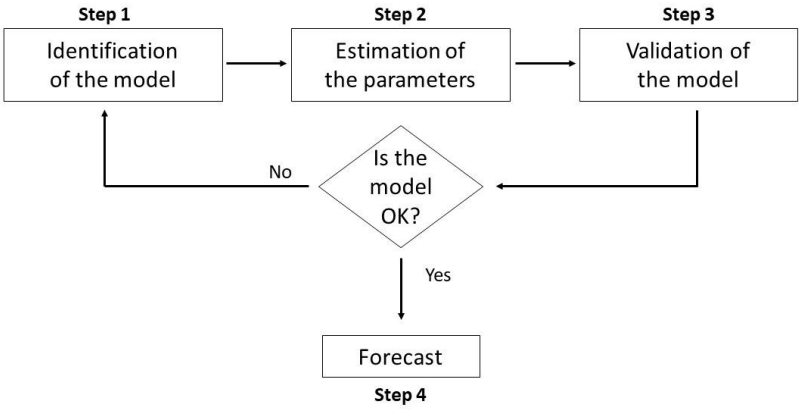
En general, las series temporales de origen económico tienden a no ser estacionarias, ya que suelen mostrar tendencias ascendentes o descendentes y rara vez fluctúan alrededor de una media constante. Por ello, es necesario transformar la serie para que sea estacionaria antes de realizar la inferencia estadística. Esto se logra aplicando la diferenciación a la serie, y el número de veces que se realiza hasta alcanzar la estacionariedad se denomina grado de integración. (Rhanoui et al, 2019).

Los gráficos derivados de la *ACF* deben mostrar una rápida disminución hacia cero a medida que aumentan los retardos. De manera similar, el gráfico de la *PACF* debe demostrar un decaimiento lento acercándose a cero a partir del primer retardo. Si esto no ocurre, la serie podría no ser estacionaria.

La figura 10 muestra que el método de Box-Jenkins resume el proceso *ARIMA* en tres pasos principales:

- Identificación: El primer paso es descomponer la serie temporal según los tres procesos: AR, I y MA. Este paso obviamente permite especificar los parámetros p, d y q, mientras se verifica primero la estacionariedad de la serie. La especificación de los parámetros p y q se realiza gracias a la *ACF* y la *PACF*. El parámetro d es el orden de diferenciación.
- Estimación: El segundo paso del procedimiento de Box-Jenkins es estimar los parámetros de los modelos apropiados proporcionando los órdenes p, d y q. La estimación se realiza utilizando métodos no lineales.
- Diagnóstico: El último paso del método de Box-Jenkins se refiere a la verificación de la relevancia del modelo. Es decir, verificar que el modelo estimado se adapte a los datos disponibles. Para ello, nos referimos a pruebas estadísticas.

Figura 10: Etapas Box-Jenkins



Fuente: Nishtha

Dadas las limitaciones del modelo paramétrico *ARIMA*, se realizará el análisis mediante un modelo de redes neuronales de tipo *Long Short Term Memory (LSTM),* el cual, dada su capacidad de aprendizaje automático, puede lograr predicciones precisas.

5.3. Redes Neuronales

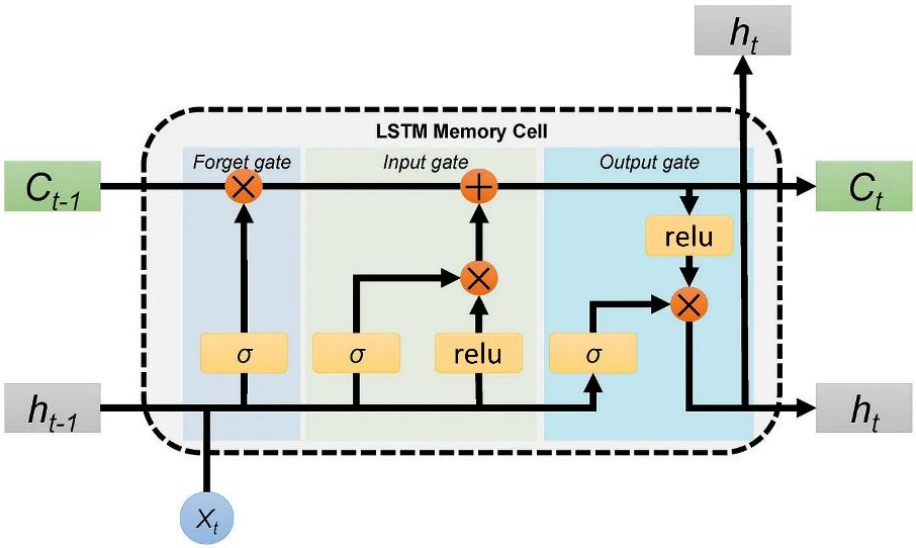
5.3.1. Long-Short Term Memory Networks (LSTM)

LSTM consiste en una arquitectura específica de *RNN* la cual contiene características para memorizar la secuencia de datos, es decir, recordar la tendencia de los datos hasta cierto punto en el tiempo es posible a través de algunas compuertas junto con una línea de memoria. Cada *LSTM* es un conjunto de celdas, o unidades del sistema, donde se registran y almacenan flujos de datos. Las celdas simulan una línea de transmisión que conecta una unidad con otra, llevando datos del pasado y recopilándolos para

el presente. La arquitectura de la celda *LSTM* se muestra en la Figura 11. En cada *LSTM* están involucrados tres tipos de compuertas para controlar el estado de cada celda:

- La compuerta de olvido (*Forget gate)* emite un número entre 0 y 1, donde 1 indica retención total del contenido, mientras que 0 indica el descarte total del mismo.
- La compuerta de entrada *(Input gate)* elige cuáles de los nuevos datos se almacenarán en cada celda.
- La compuerta de salida *(Output gate)* decide la salida de la celda. El valor de la salida se basa en el estado actual de la celda, junto con los nuevos datos filtrados. (Kontopoulou et al., 2023)

Figura 11: Diagrama LSTM



Fuente: Archit Saxena

Cell State (Memory State)

Es el primer componente de *LSTM* que recorre toda la unidad *LSTM*. El *Cell State* es el responsable de recordar y olvidar. Considerando que se trata de una arquitectura dentro de un modelo de red neuronal recurrente, existirán datos de la celda anterior que se agregarán a la celda nueva. Por ello, esta información debe ser clasificada la cual una parte de ella debe recordarse mientras que otra debe olvidarse como así también parte de la nueva información debe agregarse a la memoria.

La primera operación (x) es la operación puntual que no es más que multiplicar el estado de la celda por una matriz de [-1, 0, 1]. Como resultado de ello, el modelo *LSTM* olvidará la información multiplicada por 0, recordara si es multiplicada por 1, mientras que agregara la información invertida si es multiplicada por -1. Por otro lado, la segunda operación es (+) la cual es responsable de agregar información nueva al estado.

Forget Gate

Esta parte del proceso decide qué información se debe olvidar. Se utiliza una capa sigmoidea para tomar esta decisión. Esta capa sigmoidea se llama *hidden layer gate.*

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

Ecuación 6: forget gate

Algebraicamente se lo puede entender como un producto escalar de h_{t-1} y x_t, con la ayuda de la capa sigmoidea, genera un número entre 0 y 1 para cada número en el estado de celda C_{t-1}. Similar al caso anterior si la salida es un 1 mantendremos el valor, mientras que, si la salida es un 0, olvidaremos el valor.

Input Gate

A diferencia de la *Forget Gate*, la *Input Gate* se encarga de permitir o no permitir la entrada de nueva información al modelo.

Este proceso consiste en dos partes: Por un lado, una *sygmoid layer* decide que valores serán actualizados. Esta capa se llama *input layer gate.*

En segundo lugar, una capa de función de activación *relu* crea un vector de nuevos valores candidatos, $\tilde{C}_t$, que podrían agregarse al estado.

Por último, se busca combinar estos dos procesos $i_t * \tilde{C}_t$ para así actualizar el estado de la celda.

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

Ecuación 7: input gate

$$\tilde{C}_t = relu(W_c * [h_{t-1}, x_t] + b_c)$$

Ecuación 8: Cell candidate update

Para terminar con esta etapa del proceso, el nuevo estado de celda C_t se obtiene sumando la salida del *input gate* y *forget gate.*

$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t$$

Ecuación 9: Cell state update

Puerta de salida

La salida de la unidad *LSTM* depende del nuevo estado de la celda. Primero, una capa sigmoidea decide qué partes del estado de la celda vamos a generar. Luego, se usa una capa *relu* en el estado de la celda para aplastar los valores entre -1 y 1, que finalmente se multiplica por la salida de la puerta sigmoidea.

$$o_t = \sigma(W_o * [h_{t-1}, x_t] + b_0)$$

Ecuación 10: Output gate

$$h_t = o_t * relu(C_t)$$

Ecuación 11: hidden state update

El modelo combate el problema del gradiente desvanecido, el cual era una limitación de los *RNN* básicos. Esto le da a los *LSTM* una ventaja sobre los *RNN* teniendo la posibilidad de gestionar de manera más adecuada las predicciones futuras. (Archit, 2023).

Entre las ventajas de un modelo de redes neuronales en función de un modelo *ARIMA*, encontramos que el primero puede emplear un mayor nivel de variables de entrada como así también tiene un proceso de entrenamiento más detallado, lo que permite ganar mayor eficiencia cuando se trabaje con datos no lineales. Sin embargo, este proceso requiere un mayor poder de cómputo, lo que implica mayor tiempo de procesamiento.

Con el objetivo de determinar el número de neuronas requeridas por un modelo de entrenamiento de redes neuronales, diversos autores se encargaron de estudiar diversas aristas de esto y tras una amplia evidencia empírica arribaron a algunos resultados aproximados acerca de cómo determinar el número óptimo de neuronas.

La idea principal detrás de la ecuación 12, encargada de determinar el número de neuronas óptimo, consiste en balancear la complejidad del modelo con el tamaño del conjunto de datos y la estructura de la red neuronal. La ecuación considera los siguientes aspectos:

$$N_h = \frac{N_s}{(\alpha * (N_i + N_o))}$$

Ecuación 12: Neuronas óptimas

Siendo de esta manera:

- N_i el numero de neuronas de entrada
- N_o el numero de neuronas de salida
- N_s la cantidad de datos contenida en nuestra base de entrenamiento
- α es un factor de escala arbitrario, normalmente entre 2 y 10.

Aunque esta fórmula no garantiza la selección óptima de neuronas en todos los casos, proporciona un punto de partida razonable que puede ser ajustado mediante experimentación y validación cruzada.

5.4. Modelo Hibrido (ARIMA-LSTM)

Comprendiendo el funcionamiento tanto del modelo econométrico *ARIMA* como del modelo basado en redes neuronales *LSTM*, es posible desarrollar un tercer modelo híbrido que integre las fortalezas de ambos enfoques. Este modelo híbrido aprovecharía las ventajas inherentes de cada uno para abordar de manera más efectiva la complejidad de las series de tiempo.

El modelo *ARIMA* ha demostrado ser una herramienta sumamente eficaz en el tratamiento de datos lineales. Su capacidad para captar y modelar las características esenciales de una serie de tiempo cuando los datos presentan relaciones lineales es notable. Sin embargo, su eficacia disminuye significativamente en presencia de datos no lineales, donde las relaciones subyacentes entre las variables no siguen un patrón lineal claro. Es en este contexto donde el modelo *LSTM* juega un papel crucial.

Las redes neuronales *LSTM*, diseñadas específicamente para manejar secuencias de datos temporales, son particularmente adecuadas para captar las dependencias no lineales y las interacciones complejas que pueden existir en una serie de tiempo. A diferencia del *ARIMA*, el *LSTM* puede aprender y representar relaciones no lineales, lo que lo convierte en una herramienta poderosa para modelar series de tiempo con patrones no triviales.

La combinación de estas dos metodologías, a través de un enfoque híbrido, permitiría la creación de un modelo con la capacidad de capturar tanto las características lineales como no lineales presentes en los datos. Este modelo híbrido no solo se beneficiaría de la precisión del *ARIMA* en la captura de las dinámicas lineales, sino que también incorporaría la flexibilidad del *LSTM* para modelar las complejidades no lineales. De esta manera, se lograría una representación más completa y detallada de la serie de tiempo, lo que podría conducir a predicciones más precisas y robustas en contextos donde la estructura de los datos es mixta, es decir, donde coexisten elementos lineales y no lineales.

5.5. Métricas

La comparación de los resultados de pronóstico de los modelos mencionados anteriormente en el contexto de los estudios presentados en este trabajo se basa en varias métricas, cuyas fórmulas matemáticas se expresan a continuación.

5.5.1. Mean Squared Error (MSE)

Busca medir el promedio de los errores al cuadrado entre los valores observados y los valores predichos. Es decir, un *MSE* más bajo indica un mejor ajuste del modelo a los datos. Penaliza fuertemente los errores grandes debido a la operación de elevar al cuadrado.

$$MSE = \frac{1}{N}\sum_{t=1}^{N}(y_t - \hat{y}_t)^2$$

Ecuación 13: Error medio al cuadrado

Donde y_t es el valor real de la serie al momento de tiempo *t,* $\hat{y}_t$ es el valor de predicción correspondiente dada por el modelo, y N es el periodo de tiempo de la predicción.

5.5.2. Mean Absolute Error (MAE)

Se encarga de medir el promedio de los errores absolutos entre los valores observados y los valores predichos. Es decir, *MAE* más bajo indica un mejor ajuste del modelo. A diferencia del *MSE*, no penaliza los errores grandes de manera tan severa.

$$MAE = \frac{1}{N}\sum_{t=1}^{N}|y_t - \hat{y}_t|$$

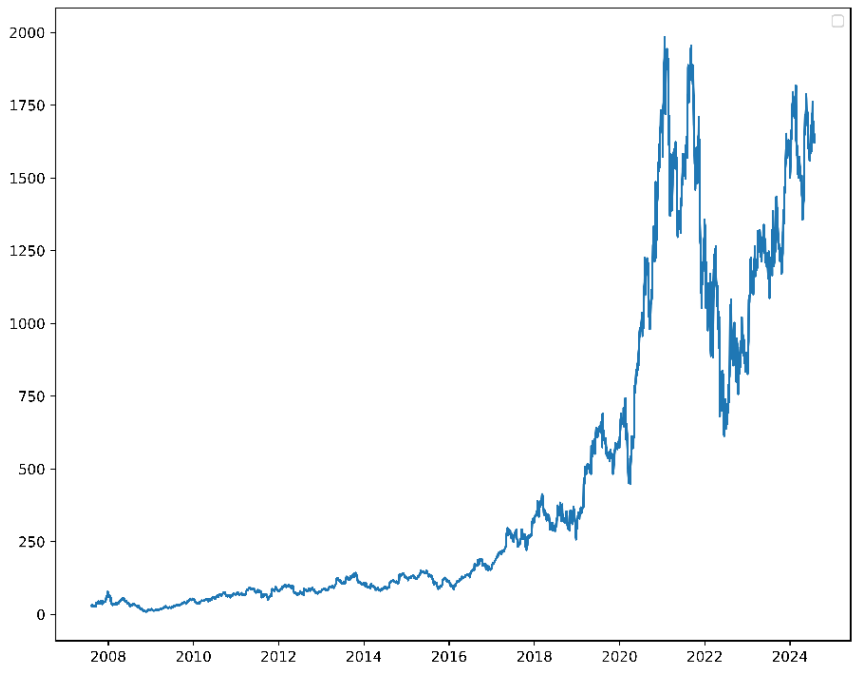
Ecuación 14: Error Absoluto Medio

6. Predicciones

6.1. Modelo ARIMA

El primer paso del método Box-Jenkins que seguimos para construir nuestro modelo *ARIMA* se refiere a la identificación de parámetros. El proceso *ARIMA* se aplica a una serie estacionaria, notamos a través de la figura 12 que la serie tiene periodos en los que su tendencia es creciente y periodos en los que la misma disminuye, por lo que a simple vista podríamos determinar que la serie no es estacionaria.

Figura 12: Precio de cierre aritmético



Fuente: Elaboración propia con base a datos de yahoo finance

Para sustentar lo anterior utilizaremos una prueba de *Augmented Dickey-Fuller (ADF)*. La prueba *ADF* es una prueba estadística llamada prueba de raíz unitaria. La idea detrás de esta prueba de raíz unitaria es que determina qué tan fuertemente una serie temporal está determinada por una tendencia.

La misma se basa en dos supuestos:

- La hipótesis nula (H_0): la serie puede ser representada por una raíz unitaria, por lo que no es estacionaria.
- La hipótesis alternativa (H_1): rechazar la hipótesis nula, sugiere que la serie no tiene raíz unitaria, lo que significa que es estacionaria.

Interpretación del p-valor

- P-valor > 0.05: Aceptamos la hipótesis nula
- P-valor < 0.05: Rechazamos la hipótesis nula

Podemos observar a partir de la tabla 2 los valores respectivos a la función de autocorrelacion y aquellos que hacen referencia a la funcion de autocorrelacion parcial. Como podemos notar en los datos, no existe por el lado de la funcion de autocorrelación un decaimiento a cero en los ultimos 30 periodos, por lo que la serie no seria estacionaria.

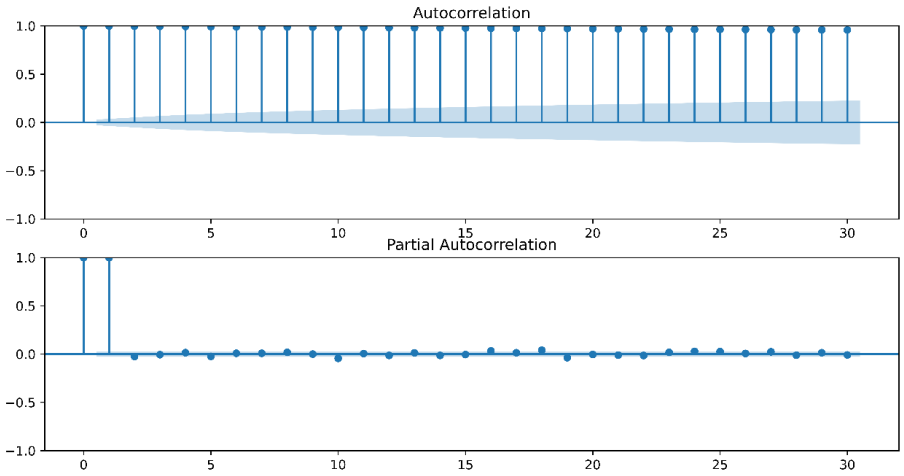
Tabla 2: Datos no diferenciados

Lag	ACF	PACF
0	1	1
1	0,999	0,999
2	0,997	-0,030
3	0,996	-0,006
4	0,994	0,018
5	0,993	-0,028
6	0,991	0,010
7	0,990	0,010
8	0,988	0,022
9	0,987	0,000
…		
20	0,971	-0,005
21	0,969	-0,012
22	0,968	-0,018
23	0,967	0,024
24	0,965	0,035
25	0,964	0,032
26	0,963	0,007
27	0,962	0,027
28	0,960	-0,014
29	0,959	0,016
30	0,958	-0,009

Fuente: Elaboración propia con base a datos de yahoo finance

Ademas, observando la figura 13, podemos notar graficamente como ocurre este resultado, el cual no tiende a cero a medida que aumenta el numero de retardos. Por ultimo, podemos contrastar estos resultados con el test de dickey-fuller en el cual el p-valor fue de 0.942560 el cual es mayor al nivel de significancia $\alpha = 0.05$ dando como resultado el no rechazo de la H_0.

Figura 13: Funciones de ACF y PACF no diferenciadas



Fuente: Elaboración propia con base a datos de yahoo finance

Tabla 3: Estadísticos no diferenciados

ADF Statistic	-0.165054
p-value	0.942560

Fuente: Elaboración propia con base a datos de yahoo finance

El siguiente paso para hacer la serie estacionaria consiste en eliminar la tendencia y la estacionalidad mediante la diferenciación, esto es realizado restando el valor actual de los valores anteriores. Este método estabiliza la media y aumenta las probabilidades de estacionariedad de la serie temporal. Es importante destacar que, al aplicar la diferenciación, el modelo ARIMA pasará de trabajar con niveles de precios a trabajar con los retornos del activo. Este procedimiento se utiliza para asegurar un desempeño óptimo del modelo. No obstante, una vez completado el análisis con el modelo ARIMA, estos rendimientos se convierten nuevamente en precios, lo que permite generar las predicciones de precios del activo.

Una vez realizada la diferenciación de los datos, podemos demostrar mediante la tabla 4 como los valores tanto de la función de autocorrelación como la función de autocorrelación parcial sufren un decaimiento a cero indicando la estacionariedad de la serie.

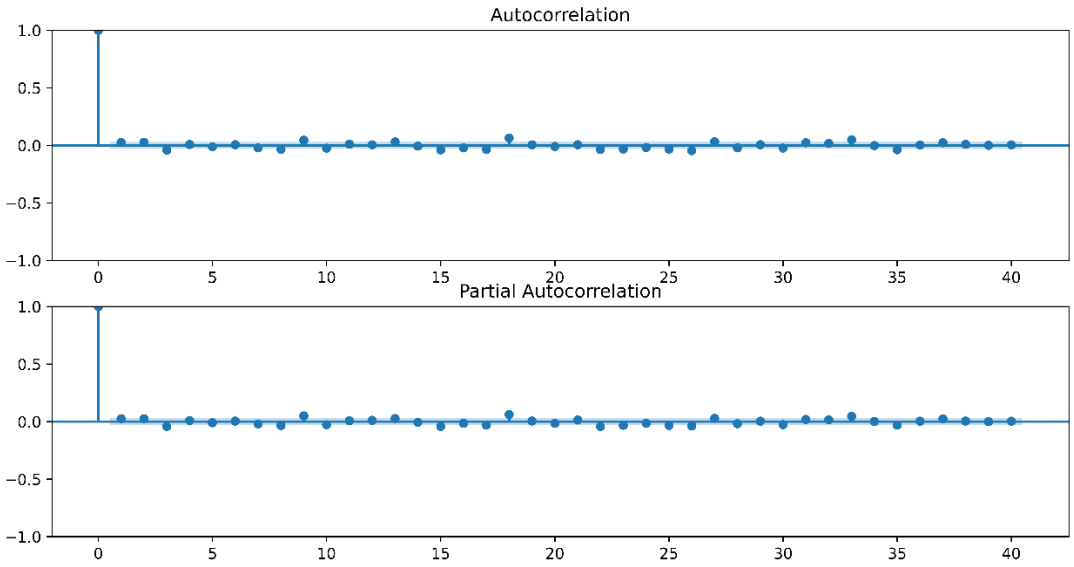
Tabla 4: Datos diferenciados

Lag	ACF	PACF
0	1	1
1	0,026	0,026
2	0,026	0,026
3	-0,040	-0,042
4	0,008	0,010
5	-0,011	-0,009
6	0,007	0,005
7	-0,021	-0,020
8	-0,033	-0,033
9	0,046	0,050
…		
20	-0,009	-0,015
21	0,007	0,015
22	-0,036	-0,042
23	-0,032	-0,031
24	-0,018	-0,014
25	-0,032	-0,034
26	-0,045	-0,038
27	0,032	0,029
28	-0,021	-0,018
29	0,006	0,004
30	-0,023	-0,026

Fuente: Elaboración propia con base a datos de yahoo finance

De manera análoga, podemos observar la figura 14 cómo la serie se vuelve estacionaria, tendiendo el grafico de *ACF* a cero rápidamente a medida que el numero de retardos aumenta, a su vez, se puede mostrar lo mismo con el gráfico de *PACF*, todo esto puede ser comprobado mediante el test de dickey-fuller en el cual el *p-valor* ≈ 0 siendo éste menor a nuestro nivel de significancia $\alpha = 0.05$

Figura 14: Funciones de ACF y PACF diferenciadas



Fuente: Elaboración propia con base a datos de yahoo finance

Tabla 5: Estadísticos diferenciados

ADF Statistic	-13,898210
p-value	5,79885e-19

Fuente: Elaboración propia con base a datos de yahoo finance

A pesar de que a partir de los gráficos podemos obtener los valores de *AR(p)* y *MA(q),* y de esta manera conocer el modelo *ARMA(p,q),* el cual según el grafico de *ACF* consideramos un modelo *AR(1)* y de manera similar, ocurre con el grafico de *PACF* siendo este *MA(1),* obteniendo consecuentemente dado el valor de la diferenciacion, un modelo de *ARIMA(1,1,1),* nos apoyamos de la función auto.arima y testeamos una serie de modelos arima con distintos parámetros.

De este modo, se probaron una amplia serie de parámetros, los cuales se pueden observar en la Tabla 6, comparándolos a través de sus respectivos criterios de información de Akaike *(AIC)* para determinar cuál es el modelo óptimo.

Tabla 6: Comparación de AIC entre diferentes modelos

Model	**AIC**
ARIMA(2,1,2)	-37.963
ARIMA(0,1,0)	-37.977
ARIMA(1,1,0)	-37.977
ARIMA(0,1,1)	-37.977
ARIMA(0,1,0)	-37.977
ARIMA(1,1,1)	-37.972

ARIMA(2,1,1)	-37.972
ARIMA(3,1,2)	-37.974
ARIMA(2,1,0)	-37.973
ARIMA(3,1,1)	-37.978
ARIMA(1,1,3)	-37.972
ARIMA(3,1,1)	-37.972
ARIMA(2,1,2)	**-37.962**
ARIMA(1,1,2)	-37.971
ARIMA(1,1,3)	-37.972
ARIMA(2,1,1)	-37.974
ARIMA(2,1,3)	-37.973
ARIMA(3,1,1)	-37.977
ARIMA(3,1,2)	-37.971
ARIMA(3,1,1)	-37.971

Fuente: Elaboración propia con base a datos de yahoo finance

El mejor modelo es uno con el menor nivel de *AIC*, por ello el *ARIMA (2,1,2)* es el mejor en este caso.

Una vez identificado los parámetros del modelo, aplicamos esta configuración a nuestros datos. El primer paso para implementar una arquitectura econométrica o de aprendizaje automático es dividir los datos en dos subconjuntos:

- Entrenamiento
- Prueba

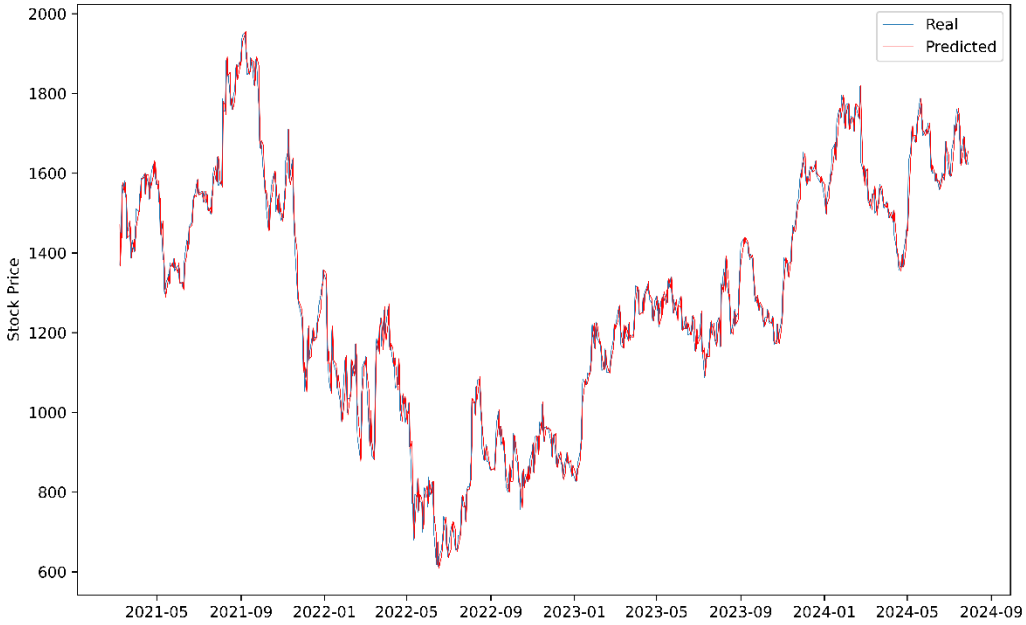
Luego aplicamos el modelo *ARIMA (2,1,2)* a nuestros datos de entrenamiento usando el 80% de los datos, el cual contempla los primeros 3416 datos y predecimos los datos de prueba con el 20% restante, siendo estos 854 datos, para de esta manera evaluar la precisión del modelo.

El propósito de este trabajo es la predicción del precio de cierre de las acciones de mercado libre en su cotización en dólares americanos para los próximos días con el fin de tener una idea clara y concisa hacia qué dirección pretende ir la tendencia.

Para esto, aplicamos el modelo construido para predecir la evolución del precio durante los próximos periodos. La figura 15 compara el conjunto de datos de prueba, los cuales refieren a los valores reales de la serie en la etapa de testeo con los valores generados por el modelo *ARIMA (2,1,2),* es decir, las predicciones.

Para obtener las predicciones bajo este modelo, se realizó una iteración, la cual únicamente genera el siguiente valor de la cotización, es decir en t+1, posterior a ello, utiliza la cotización real de t+1 para predecir la cotización en t+2, este proceso lo realiza de manera iterativa.

Figura 15: Predicciones de ARIMA vs valores reales



Fuente: Elaboración propia con base a datos de yahoo finance

Como se puede observar en la figura 15 los datos de las predicciones, en la mayoría de los casos, aciertan en la cotización del día siguiente, sin embargo, no lo hacen con una total precisión. Esto lo podemos corroborar mediante la tabla 7, en la cual podemos dinamizar los valores reales, los predichos y sus respectivos errores.

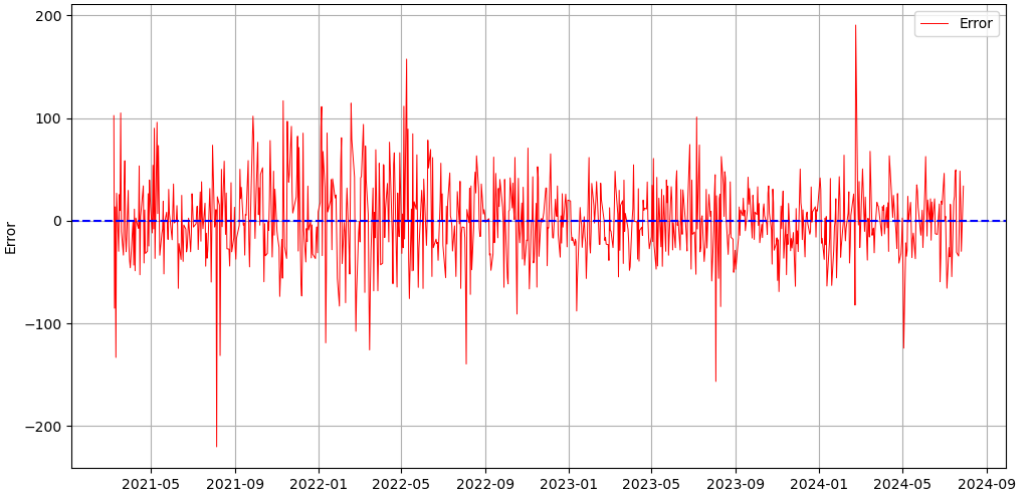
Tabla 7: Errores ARIMA

Fecha	Predicho	Real	Error
3/8/2021	1471,92	1369,54	102,38
3/9/2021	1366,73	1452,01	-85.28
3/10/2021	1449,14	1435,57	13,57
3/11/2021	1437,55	1570,78	-133,23
3/12/2021	1576,94	1550,15	26,79
3/15/2021	1551,49	1581,32	-29,83
3/16/2021	1576,00	1550,49	25,51
3/17/2021	1547,50	1537,62	9,89
3/18/2021	1541,19	1436,17	105,02
3/19/2021	1438,56	1448,89	-10,33
3/22/2021	1442,57	1476,11	-33,54
3/23/2021	1480,10	1446,65	33,45
	…		
7/19/2024	1619,19	1649,99	-30,80
7/22/2024	1652,96	1687,38	-34,42
7/23/2024	1687,29	1692,23	-4,94

7/24/2024	1690,95	1642,55	48,40
7/25/2024	1644,27	1625,15	19,12
7/26/2024	1621,87	1651,69	-29,82
7/29/2024	1655,16	1621,40	33,76

Fuente: Elaboración propia con base a datos de yahoo finance

Figura 16: Errores ARIMA

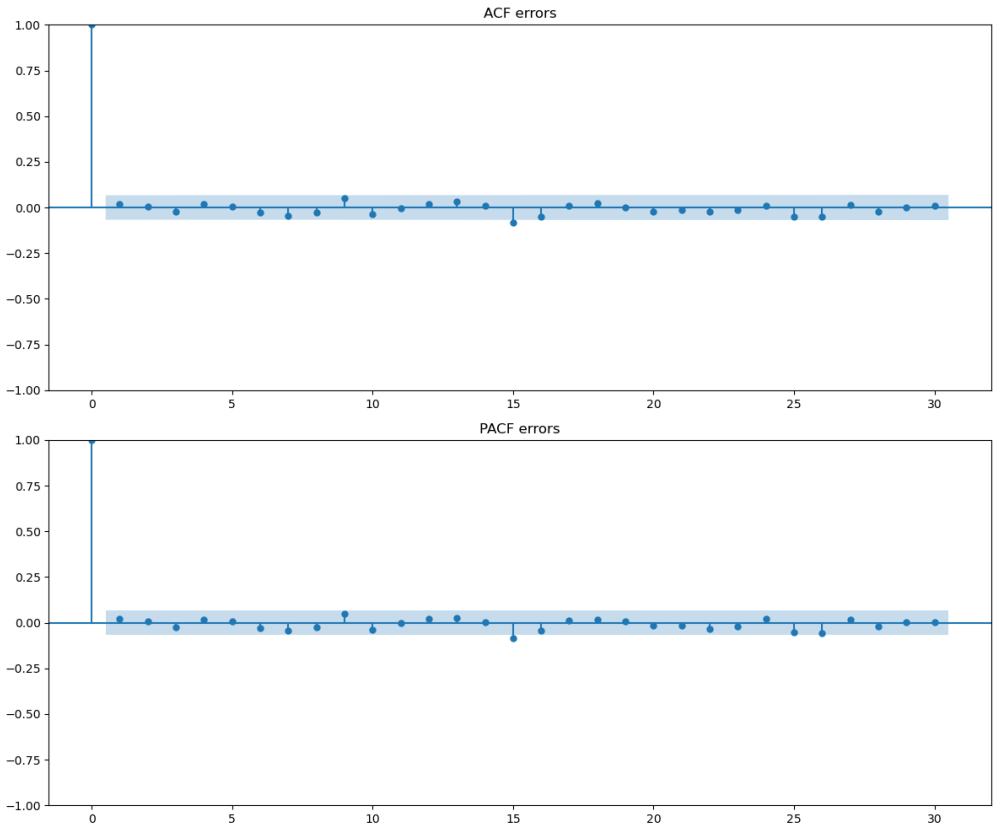


Fuente: Elaboración propia con base a datos de yahoo finance

Asimismo, podemos notar mediante la figura 16 como fluctúan los errores conforme va corriendo la serie temporal y nuestros valores predichos. Para concluir con el modelo utilizamos dos tipos de métricas el *MSE* y el *MAE*, siendo el valor de la primera 1527.794 y el valor de la segunda 29.308, lo cual deberemos comparar con los resultados arrojados por los otros modelos.

Además, podemos observar mediante la figura 17, las funciones de autocorrelación de los errores las cuales muestran que tanto la *ACF* como la *PACF* están prácticamente dentro de los intervalos de confianza a excepción del dato contenido en el primero rezago, lo cual es esperable. Esto indica que no hay autocorrelación significativa en los residuos, sugiriendo que el modelo *ARIMA* ha capturado bien la estructura de los datos y los errores son aproximadamente ruido blanco.

Figura 17: Funciones de ACF y PACF en errores ARIMA



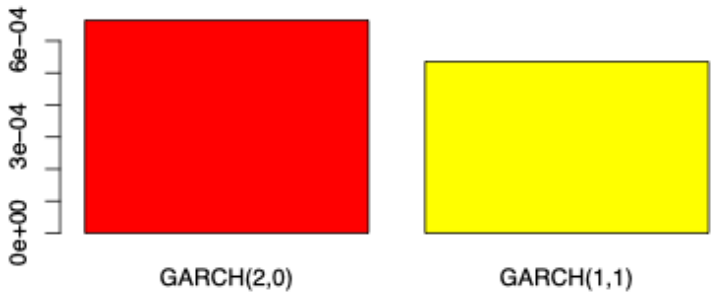
Fuente: Elaboración propia con base a datos de yahoo finance

6.1.1. Modelo ARCH-GARCH

Para modelizar la volatilidad seguimos el trabajo de Dip y Romero (2015) quienes toman en cuenta los retornos de la acción por lo que se trabajará la serie del precio de Mercado Libre (MELI) diferenciada y en logaritmo. Se trabaja con la muestra de entrenamiento (80% de los datos) y de testeo (20%) para calcular los errores de pronóstico dentro y por fuera de la muestra *(in-out sample errors)*, siendo estos últimos errores los que se presta más atención. En este ejercicio vamos a considerar varios modelos, consideraremos el modelo i.i.d *(identically Independently Distributed)*, el modelo *AR*, el modelo *ARMA* y algunos modelos *GARCH*.

En primer lugar, se estima un modelo *GARCH (2,0)* y un *GARCH (1,1)* a los fines de mostrar cuál de los dos modelos ajustan mejor a nuestra base de entrenamiento. Se estima un modelo *GARCH(2,0)* para modelar la volatilidad condicional de la serie temporal con 2 rezagos de la varianza pasada y sin incluir los rezagos de los errores pasados, mientras que en el GARCH (1,1) estima la volatilidad condicional de una serie temporal como una función de los errores pasados y la volatilidad pasada. Es el modelo más usado. Captura la volatilidad y su persistencia.

Figura 18: Error de la estimación de la Volatilidad /Varianza



Fuente: Elaboración propia con base a datos de yahoo finance

Se observa que el modelo *GARCH(1,1)* tiene menor error de estimación dentro de la base de entrenamiento. ¿Qué sucede con los errores de pronósticos cuando se estiman dentro y fuera de la muestra en distintos modelos? Para estos cálculos, se utiliza la base de datos de testeo que reúne el 20% de los datos de la muestra total.

Tabla 8: Errores de pronóstico por dentro y fuera de la muestra

Modelo	**in-sample**	**out-of-sample**
iid	0.001225748	0.001146600
AR(1)	0.001225516	0.001145756
ARMA(2,2)	0.001221798	0.001143502
ARMA(2,2)+GARCH(1,0)	0.001231521	0.001152818
GARCH(10,0)	0.001225748	0.001146600
ARMA(2,2) + GARCH(1,1)	0.001220690	0.001146188

Fuente: Elaboración propia con base a datos de yahoo finance

Se puede observar que, a medida que aumenta la complejidad del modelo, el error dentro de la muestra tiende a reducirse, como era de esperarse, debido al mayor número de grados de libertad para ajustar los datos. Sin embargo, la diferencia en este caso resulta ser insignificante. La cantidad realmente relevante es el error fuera de la muestra: se evidencia que un incremento en la complejidad del modelo puede producir resultados no deseados. En este contexto, parece que el modelo *ARMA(2,2)* es suficientemente adecuado en términos de error en la predicción de los rendimientos. Este modelo, había resultado elegido anteriormente. Se recuerda que acá ya se trabaja con la serie diferenciada. Este ejercicio ilustra el uso de modelos *ARCH* para modelar la varianza condicional de una serie temporal. Se ofrece una visión general de los modelos *ARCH* y sus variantes, pero es crucial considerarlo solo un punto de partida. La especificación de la heterocedasticidad y la dinámica de la volatilidad en los modelos *ARCH-GARCH* requiere un estudio más profundo para su adecuada comprensión y aplicación.

A continuación, se continua con el análisis de modelos *ARIMA*, incorporando los quiebres estructurales de la serie.

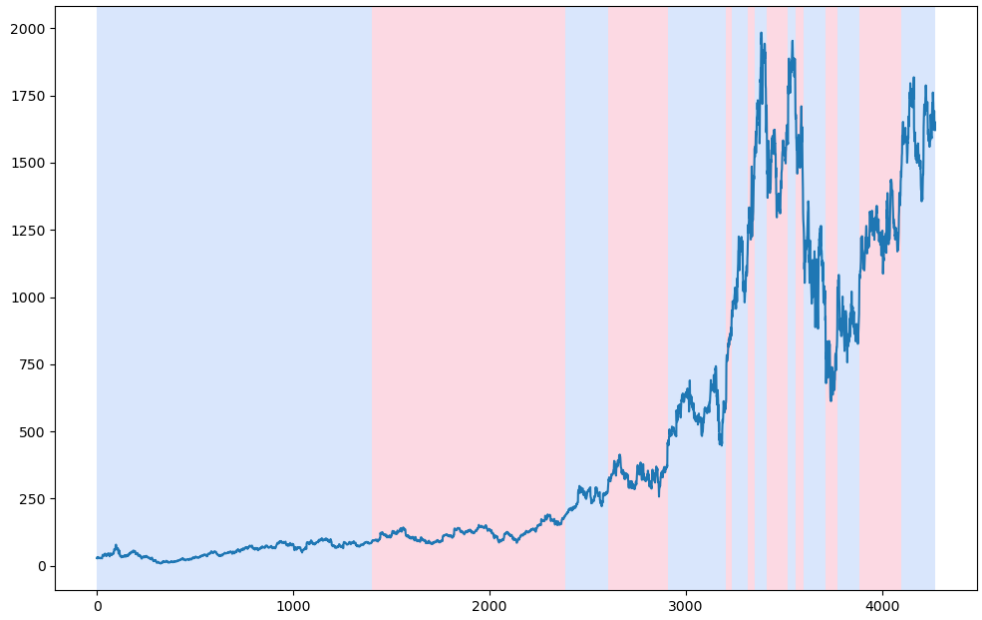
6.2. Modelo ARIMA Segmentado

Dado que las series temporales financieras suelen no ser estacionarias, es fundamental implementar una metodología que, además del proceso de diferenciación, considere los distintos tramos de la serie que presentan tendencias variables. Para ello, se desarrollará un enfoque que permita detectar cambios estructurales en la tendencia.

Los quiebres estructurales se identifican de manera automática y endógena basándose en los datos de la serie temporal. El algoritmo empleado para la segmentación tiene como objetivo determinar los niveles de media y varianza. Cuando se detecta una variación que supera un margen preestablecido, se considera que ha ocurrido un quiebre estructural. Estas variaciones pueden ser el resultado de cambios en la tendencia, aumentos en la volatilidad u otros factores significativos que afectan la estructura subyacente de los datos.

Como se ilustra en la Figura 19, se han delimitado los diferentes segmentos de tendencia utilizando colores diferenciados, lo que facilita la evaluación de parámetros específicos del modelo para cada uno de estos segmentos.

Figura 19: Puntos de quiebre



Fuente: Elaboración propia con base a datos de yahoo finance

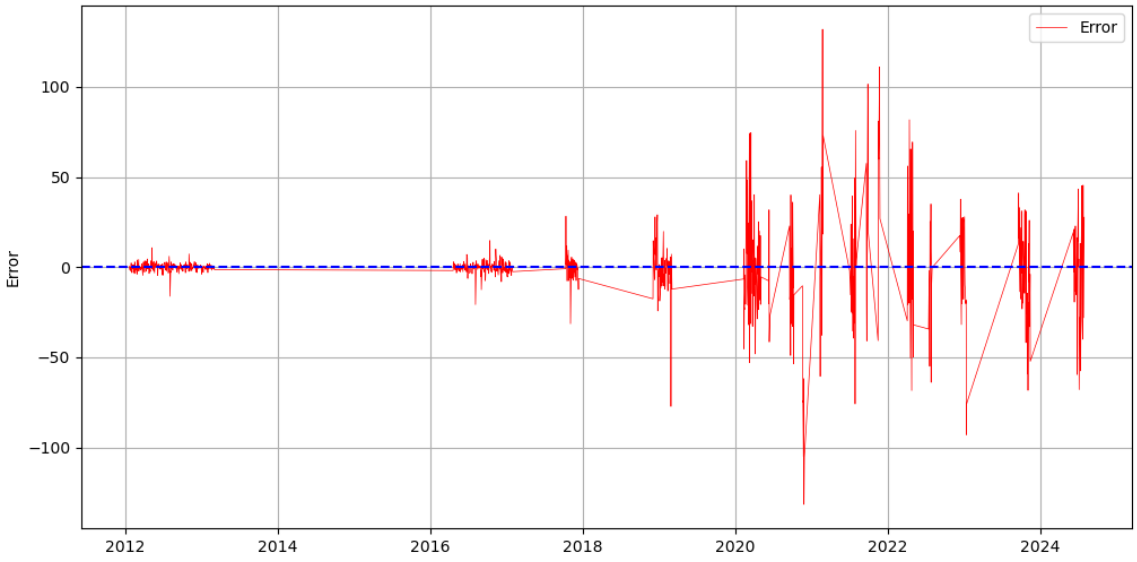
Tras las evaluaciones realizadas, se identificaron 17 segmentos distintos. En la Tabla 9 se presentan los parámetros correspondientes a cada uno de estos segmentos.

Tabla 9: Parámetros puntos de quiebre

Segmento	p	d	q
1	3	1	2
2	0	1	0
3	0	1	0
4	0	1	0
5	0	1	3
6	0	1	0
7	0	1	0
8	0	0	1
9	2	1	2
10	5	1	3
11	0	1	0
12	1	0	0
13	0	1	1
14	0	1	0
15	1	0	0
16	0	1	0
17	1	0	0

Fuente: Elaboración propia con base a datos de yahoo finance

Figura 20: Errores puntos de quiebre ARIMA

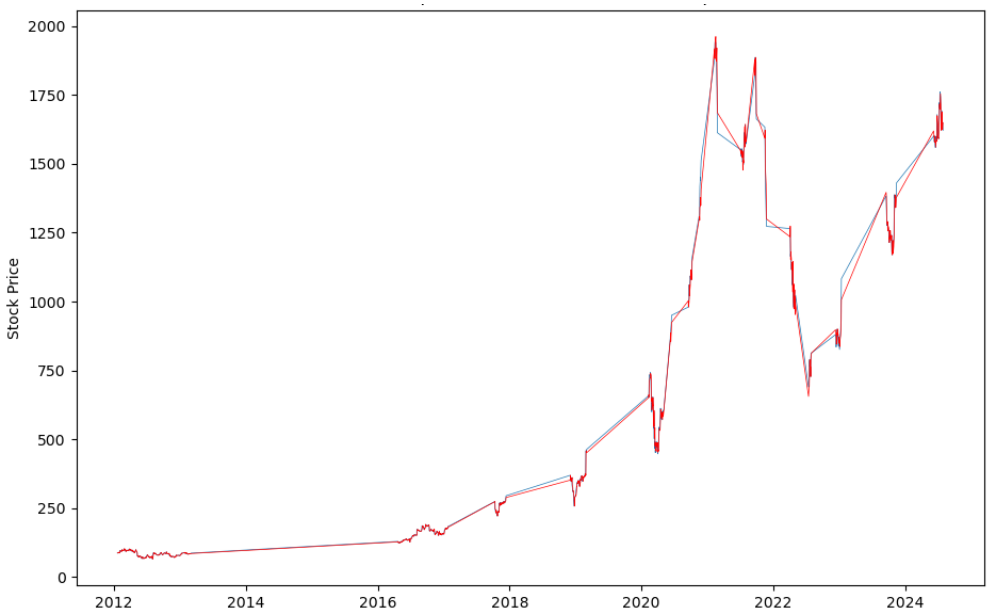


Fuente: Elaboración propia con base a datos de yahoo finance

Con estos parámetros, se observa una disminución en los errores, medidos como la diferencia entre las cotizaciones reales y predichas, en comparación al modelo ARIMA que no captura los quiebres estructurales, como se muestra en la Figura 20, sin embargo, se puede notar la existencia de una volatilidad baja hasta aproximadamente el 2020, la cual se incrementa para los años posteriores. Además, con el fin de comparar este modelo con otros, se calcularon los valores de *MSE* y *MAE*, obteniéndose 459,845 y 10,919, respectivamente.

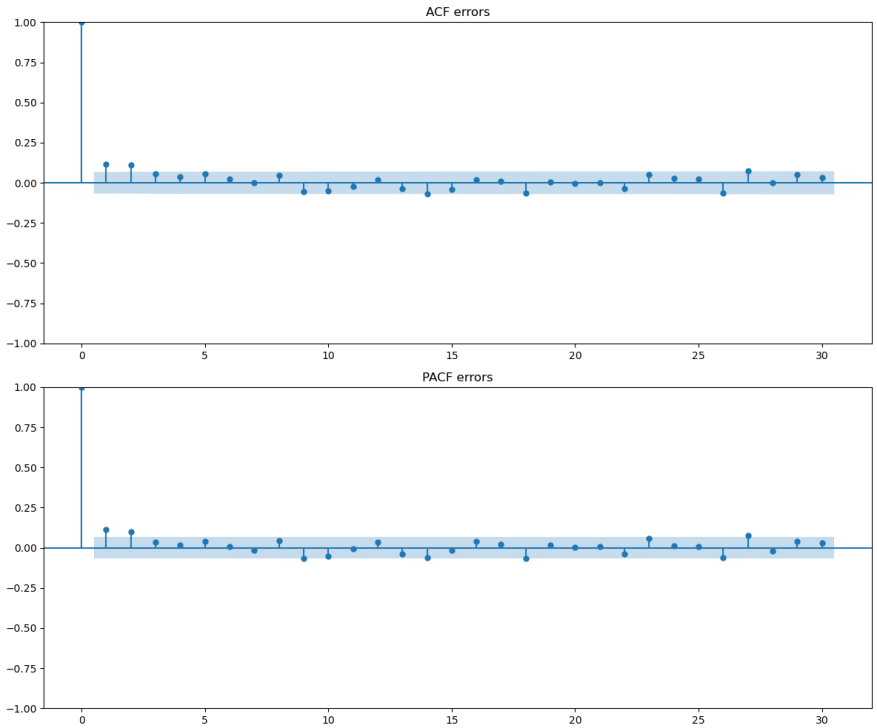
Con los niveles de error presentes, se procedió a realizar la predicción de los datos, la cual se presenta en la Figura 21, evidenciando un ajuste significativo a los datos observados.

Figura 21: Valores reales vs predicciones ARIMA con quiebres estructurales



Fuente: Elaboración propia con base a datos de yahoo finance

Figura 22: Funciones ACF y PACF de errores segmentados ARIMA



Fuente: Elaboración propia con base a datos de yahoo finance

Por último, podemos observar mediante la figura 22, las fluctuaciones de las funciones de autocorrelación y autocorrelación parcial sobre los errores, los cuales se encuentran en la mayoría de los casos dentro de los intervalos de confianza. Sin embargo, los primeros 2 rezagos salen un poco de los intervalos de confianza, implicando quizás que no se ha capturado la estructura de la serie temporal original en su totalidad.

6.3. Predicción LSTM

De manera similar al modelo *ARIMA*, este utiliza el 80% de los datos para el proceso de entrenamiento y evalúa los resultados con el 20% restante. Asimismo, y de manera análoga al *ARIMA*, también busca generar una predicción para el periodo t+1 utilizando los datos reales hasta t. Una vez llegado a t+1, se emplea el valor real de la cotización de cierre de ese periodo para predecir la cotización en t+2.

Inicialmente, se emplea la función MinMaxScaler para normalizar los datos en un rango entre 0 y 1, lo cual mejora la convergencia y estabilidad del modelo durante el entrenamiento. Tras entrenar el modelo, los datos se desescalan para obtener resultados acordes a la serie de tiempo original.

El modelo de redes neuronales utiliza la arquitectura de *LSTM* para el entrenamiento y posterior evaluación. Para ello, se emplea una capa de entrada, definida como `Sequential(1)`, una capa de salida definida como `Dense(1)`, y una capa oculta del tipo *LSTM*, que en este caso contiene 300 neuronas para maximizar la absorción de información proveniente de los datos de entrenamiento. La función de activación utilizada es la *Rectified Linear Unit (ReLU),* la cual introduce no linealidad y evita el problema del gradiente desvanecido.

Con el objetivo de determinar un parámetro de entrenamiento óptimo, se configura un optimizador del tipo *Adam (Adaptive Moment Estimation),* el cual busca minimizar el *Mean Squared Error (MSE),* definido en el modelo como `val_loss`, durante un periodo determinado. En este modelo, se especifica que la red entrene hasta alcanzar un nivel mínimo de *MSE* y, una vez alcanzado, continúe entrenando durante 100 épocas adicionales para corroborar que no se logra un nivel de error inferior en estos entrenamientos posteriores. Tras este periodo, se da por concluido el entrenamiento de la red, la cual queda lista para realizar predicciones.

Además, la red utilizará una configuración de `epoch = 1000`, siendo este el número máximo de veces que el algoritmo se entrenará en caso de no encontrar un error mínimo según los parámetros previamente definidos.

Tabla 10: Etapa de entrenamiento del modelo LSTM

Epoch 193/1000
35/35 ━━━━━━━━━━━━━━━━━━━━ **1s** 23ms/step - loss: 4.8005e-05 - val_loss: 5.8795e-04
Epoch 194/1000
35/35 ━━━━━━━━━━━━━━━━━━━━ **1s** 21ms/step - loss: 4.8182e-05 - val_loss: 5.4195e-04
Epoch 195/1000
35/35 ━━━━━━━━━━━━━━━━━━━━ **1s** 23ms/step - loss: 4.7184e-05 - val_loss: 5.5688e-04
Epoch 196/1000
35/35 ━━━━━━━━━━━━━━━━━━━━ **1s** 22ms/step - loss: 4.7681e-05 - val_loss: 5.6501e-04
Epoch 197/1000

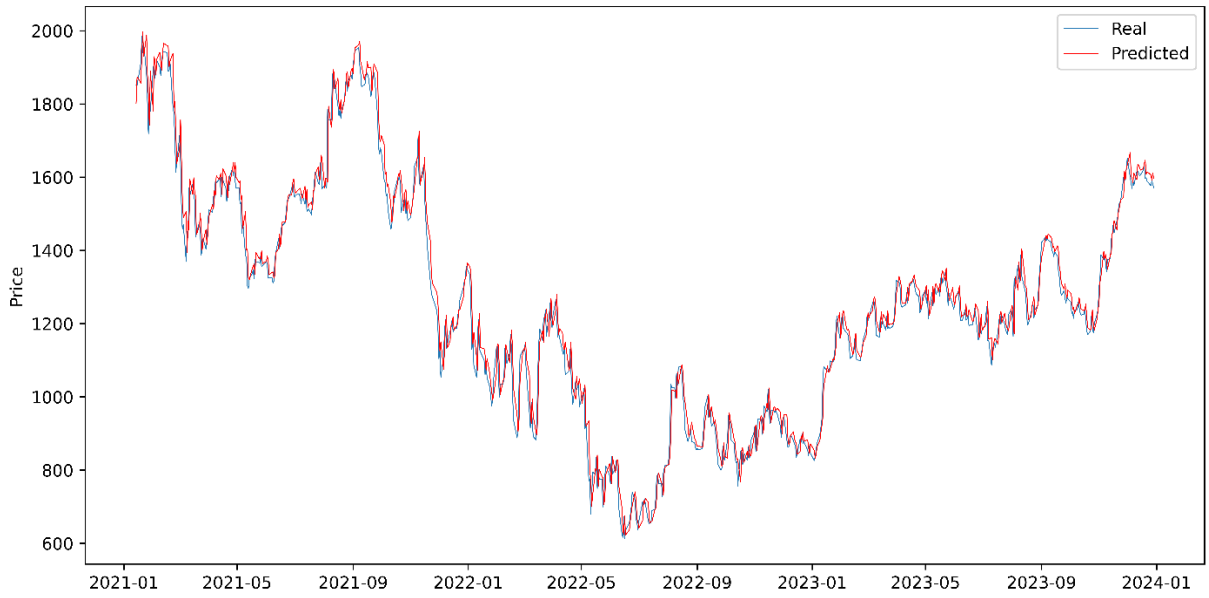
35/35 ━━━━━━━━━━━━━━━━━━━━ **1s** 26ms/step - loss: 4.7767e-05 - val_loss: 5.5197e-04
Epoch 197: early stopping
27/27 ━━━━━━━━━━━━━━━━━━━━ **1s** 13ms/step

Fuente: Elaboración propia con base a datos de yahoo finance

Como se puede observar en la tabla 10 la red necesito entrenarse 197 veces para lograr mantener el error minimizado.

Definida la etapa de entrenamiento, se lleva adelante el modelo con los parámetros determinados y se logra realizar una predicción, la cual es comparada con sus valores de testeo originales como se puede observar en la figura 23.

Figura 23: Predicciones LSTM vs valores reales



Fuente: Elaboración propia con base a datos de yahoo finance

De manera análoga al caso de *ARIMA*, se estudiaron los casos puntuales de cada dato junto a sus errores, como se muestra en la tabla 10. Además, se puede observar cómo fluctuaron estos errores a lo largo del tiempo en la figura 24.

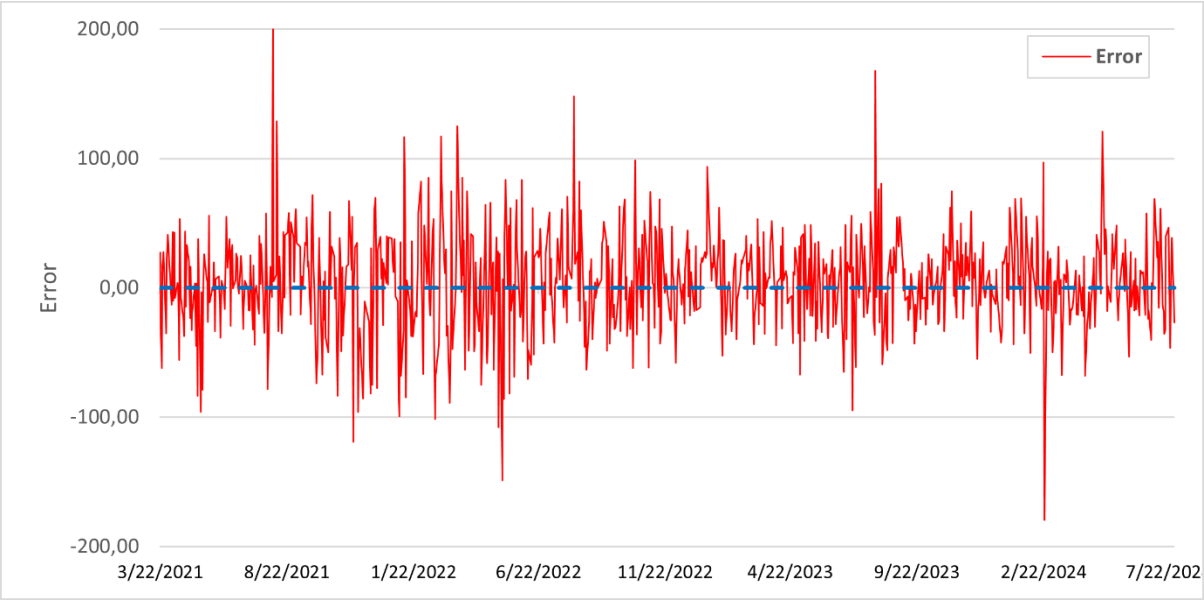
Tabla 8: Errores LSTM

Fecha	Predicho	Actual	Error
3/22/2021	1476,11	1448,94	27,17

3/23/2021	1446,65	1482,49	-35,84
3/24/2021	1387,85	1449,82	-61,97
3/25/2021	1402,50	1385,15	17,35
3/26/2021	1431,97	1404,50	27,47
3/29/2021	1403,42	1438,69	-35,27
3/30/2021	1428,03	1405,34	22,69
3/31/2021	1472,14	1431,14	41,00
4/1/2021	1510,50	1478,66	31,84
4/5/2021	1503,45	1516,50	-13,05
4/6/2021	1547,79	1504,59	43,20
4/7/2021	1542,00	1551,99	-9,99
		…	
7/19/2024	1649,99	1609,67	40,32
7/22/2024	1687,38	1640,91	46,47
7/23/2024	1692,23	1684,14	8,09
7/24/2024	1642,55	1689,27	-46,72
7/25/2024	1625,15	1631,09	-5,94
7/26/2024	1651,69	1613,17	38,52
7/29/2024	1621,40	1648,38	-26,98

Fuente: Elaboración propia con base a datos de yahoo finance

Figura 24: errores LSTM



Fuente: Elaboración propia con base a datos de yahoo finance

Para concluir con el modelo y hacerlo comparable de alguna manera con las otras metodologías, se llevó a cabo la medición de su *MSE* y *MAE*, siendo 1488.08 el valor del primero y 29.10 el valor del segundo.

6.4. Predicción ARIMA-LSTM (HIBRIDO)

Una de las características principales y más destacadas al evaluar un modelo del tipo *ARIMA* es su capacidad de predicción sobre datos lineales. Para implementar el modelo ARIMA en este trabajo, se suavizaron los datos con el fin de hacerlos estacionarios, lo que permite su análisis mediante un método *ARIMA*. Sin embargo, una de sus desventajas es la incapacidad para trabajar con datos no lineales.

En contraste, el modelo de redes neuronales puede manejar datos no lineales. No obstante, debido a la propia naturaleza del modelo, puede presentar errores al interpretar datos lineales, lo que afecta la exactitud de las predicciones.

Considerando las fortalezas y limitaciones de ambos modelos, se optó por explorar y experimentar con un modelo híbrido. Este modelo híbrido inicialmente aplica el enfoque *ARIMA* para analizar la serie de tiempo y capturar los rasgos lineales. Posteriormente, se incorpora al modelo *LSTM* las predicciones realizadas por el modelo *ARIMA* como *input*. Además, el modelo *LSTM* recibe un segundo *input*, que es la base de datos original, lo que permite una mayor aproximación a los datos reales al considerar tanto datos lineales, mediante el modelo *ARIMA*, como no lineales, mediante el modelo *LSTM*.

Al igual que en los modelos anteriores, este modelo utiliza el 80% de los datos para el proceso de entrenamiento y evalúa los resultados con el 20% restante. Asimismo, busca generar una predicción para el periodo t+1 utilizando los datos reales hasta t. Una vez alcanzado t+1, se emplea el valor real de la cotización de cierre de ese periodo para predecir la cotización en t+2.

Inicialmente, se emplea la función MinMaxScaler para normalizar los datos en un rango entre 0 y 1, mejorando así la convergencia y estabilidad del modelo durante el entrenamiento. Tras entrenar el modelo, los datos se desescalan para obtener resultados acordes a la serie de tiempo original.

El modelo de redes neuronales utiliza la arquitectura de *LSTM* para el entrenamiento y posterior evaluación. Para ello, se emplea una capa de entrada definida como Sequential(1), por la cual ingresa una base de datos previamente fusionada entre la base de datos original y las predicciones generadas por *ARIMA*. Además, se incluye una capa de salida definida como Dense(1) y una capa oculta del tipo *LSTM*, que en este caso contiene 252 neuronas para maximizar la absorción de información proveniente de los datos de entrenamiento.

La función de activación empleada es la *ReLU*. Para determinar un parámetro de entrenamiento óptimo, se configura un optimizador del tipo Adam. En este modelo, se especifica que la red entrene hasta alcanzar un nivel mínimo de MSE y, una vez alcanzado, continúe entrenando durante 100 épocas adicionales para confirmar que no se logra un nivel de error inferior en estos entrenamientos posteriores. Tras este periodo, se da por concluido el entrenamiento de la red, que queda lista para realizar predicciones. Además, la red utilizará una configuración de epoch = 1000.

Tabla 92: Etapa de entrenamiento modelo ARIMA-LSTM

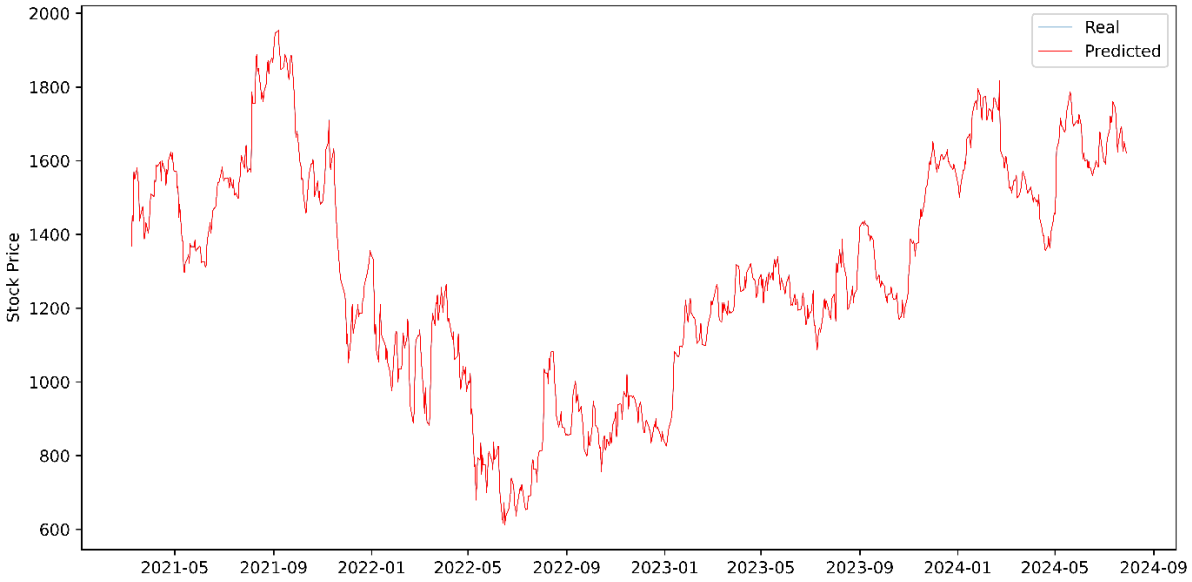
Epoch 194/1000
107/107 ━━━━━━━━━━━━━━━━━━━━ **0s** 3ms/step - loss: 1.0403e-07 - val_loss: 2.3899e-08
Epoch 195/1000

107/107 ━━━━━━━━━━━━━━━━━━━━ **0s** 3ms/step - loss: 4.4616e-09 - val_loss: 2.1265e-07
Epoch 196/1000
107/107 ━━━━━━━━━━━━━━━━━━━━ **0s** 3ms/step - loss: 2.0423e-08 - val_loss: 3.6815e-06
Epoch 197/1000
107/107 ━━━━━━━━━━━━━━━━━━━━ **0s** 3ms/step - loss: 1.6808e-07 - val_loss: 1.0065e-08
Epoch 198/1000
107/107 ━━━━━━━━━━━━━━━━━━━━ **0s** 3ms/step - loss: 5.7167e-08 - val_loss: 1.5837e-06
Epoch 198: early stopping
27/27 ━━━━━━━━━━━━━━━━━━━━ **0s** 8ms/step

Fuente: Elaboración propia con base a datos de yahoo finance

Como se puede observar en la tabla 12, la red necesitó entrenarse 198 veces para lograr mantener el error minimizado. Una vez definida toda la etapa de entrenamiento, se implementa el modelo con los parámetros determinados y se realiza una predicción. Esta predicción se compara con los valores originales de testeo, como se muestra en la figura 25.

Figura 25: Valores reales vs predichos ARIMA-LSTM



Fuente: Elaboración propia con base a datos de yahoo finance

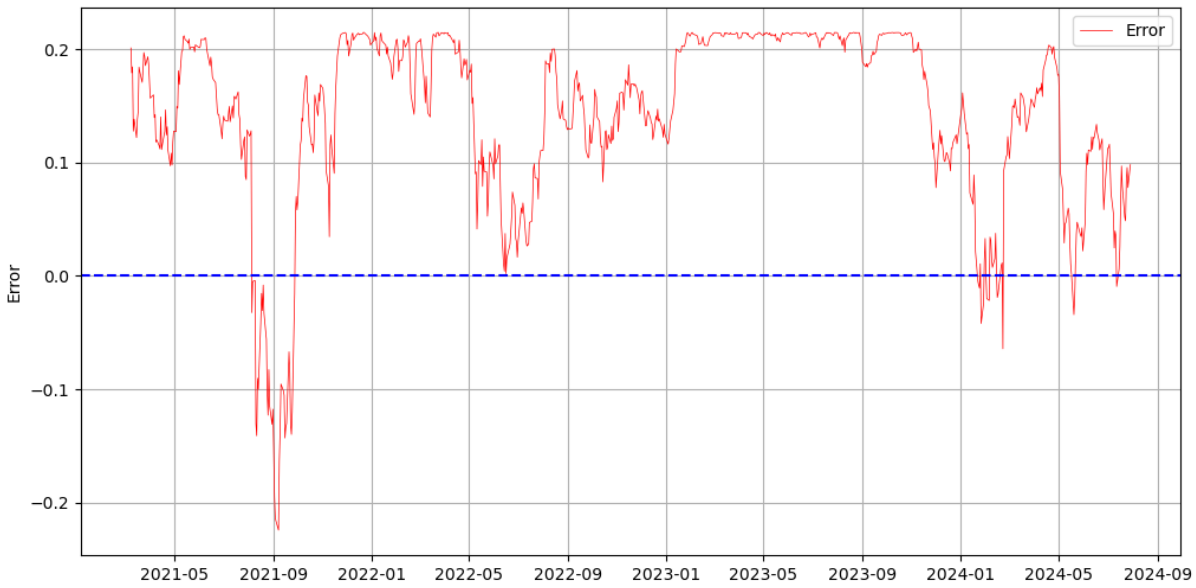
Tabla 103: Errores ARIMA-LSTM

Fecha	Real	Predicho	Error
3/8/2021	1369,54	1369,34	0,20
3/9/2021	1452,01	1451,83	0,18

3/10/2021	1435,57	1435,39	0,18
3/11/2021	1570,78	1570,65	0,13
3/12/2021	1550,15	1550,01	0,14
3/15/2021	1581,32	1581,20	0,12
3/16/2021	1550,49	1550,35	0,14
3/17/2021	1537,62	1537,48	0,14
3/18/2021	1436,17	1435,99	0,18
3/19/2021	1448,89	1448,71	0,18
3/22/2021	1476,11	1475,94	0,17
3/23/2021	1446,65	1446,47	0,18
		…	
7/19/2024	1649,99	1649,91	0,08
7/22/2024	1687,38	1687,33	0,05
7/23/2024	1692,23	1692,18	0,05
7/24/2024	1642,55	1642,47	0,08
7/25/2024	1625,15	1625,05	0,10
7/26/2024	1651,69	1651,61	0,08
7/29/2024	1621,40	1621,30	0,10

Fuente: Elaboración propia con base a datos de yahoo finance

Figura 26: Errores ARIMA-LSTM



Fuente: Elaboración propia con base a datos de yahoo finance

Para concluir con el modelo, se llevó a cabo la medición de su *MSE* y *MAE*, siendo 0.0269 el valor del primero y 0.1529 el valor del segundo, representando uno de los errores más bajos como se puede apreciar en la figura 26.

7. Comparación de resultados

Se presenta un análisis comparativo de los errores obtenidos con diferentes modelos. Se incluyen tanto el modelo *ARIMA(2,1,2)* como su variante ajustada considerando los quiebres estructurales. Asimismo, se evalúa el desempeño del modelo *LSTM*, entrenado con 300 neuronas, y del modelo híbrido *ARIMA-LSTM*, entrenado con 252 neuronas.

Como se puede observar en la tabla 14, los errores fueron medidos mediante su *MSE* y *MAE*, quedando el modelo hibrido como aquel con el mejor desempeño, el *ARIMA* con quiebres en segundo lugar y el *LSTM* básico en tercer lugar.

Tabla 114: Comparación de los errores

METRICA	ARIMA(2,1,2)	ARIMA C/ QUIEBRES	LSTM	ARIMA-LSTM
MSE	1527.794	459.845	1488.08	0.0269
MAE	29.308	10.919	29.10	0.1529

Fuente: Elaboración propia con base a datos de yahoo finance

Con base en estos resultados, se procedió a contrastar la robustez del modelo utilizando otros activos. Para ello, se seleccionaron empresas representativas de sectores similares en diferentes países: Amazon (AMZN) en Estados Unidos, Alibaba (BABA) en China y Globant (GLOB) como un referente argentino adicional. Tal como se observa en la figura 27, las tendencias de estos activos presentan, en promedio, comportamientos similares.

Figura 27: Evolución histórica de activos

Fuente: Elaboración propia con base a datos de yahoo finance

De este modo, se realizaron las predicciones teniendo en cuenta las características particulares de cada activo, obteniéndose los siguientes resultados:

Table 125: Comparación de los errores distintos activos

MSE	ARIMA(0,1,1)	ARIMA C/ QUIEBRES	LSTM	ARIMA-LSTM
AMZN	9.285	3.546	12.715	0.00039
BABA	5.486	11.897	7.886	$5.42e^{-6}$
GLOB	26.775	19.690	26.447	0.025

Fuente: Elaboración propia con base a datos de yahoo finance

De este modo, la tabla 15 permite observar la confirmación de la robustez del modelo híbrido *ARIMA-LSTM* al generar predicciones sobre los precios de los activos en distintos mercados.

Finalmente, respecto a la pregunta inicial sobre la teoría de la eficiencia de los mercados, puedo decir que, los resultados obtenidos a partir del análisis de las funciones de autocorrelación y autocorrelación parcial de los residuos de los modelos *ARIMA* y *LSTM* indican que dicha teoría se verifica para el activo en cuestión. Los residuos no presentan autocorrelaciones significativas en los gráficos de *ACF* y *PACF*, lo que sugiere que los valores se encuentran mayoritariamente dentro de los intervalos de confianza y exhiben un comportamiento de ruido blanco. Esta falta de autocorrelación residual implica que no existe información predecible en los datos que los modelos puedan aprovechar. En consecuencia, se puede inferir que toda la información relevante ya ha sido incorporada en los precios del activo, eliminando la posibilidad de identificar oportunidades sistemáticas para obtener rendimientos superiores mediante el uso de estos modelos.

8. Conclusión

En este trabajo se ha analizado la efectividad de los modelos predictivos *ARIMA, ARIMA con quiebres estructurales*, *LSTM* y el modelo híbrido *ARIMA-LSTM* en el análisis de las acciones de Mercado Libre en el mercado argentino bajo su cotización en dólares americanos. Mientras el modelo *ARIMA* se muestra eficaz para capturar dinámicas lineales a corto plazo, su alcance es limitado al tratarse de un enfoque univariado. El modelo *LSTM*, por otro lado, ofrece un potencial superior para captar patrones no lineales, aunque su precisión depende en gran medida de la cantidad y calidad de datos de entrenamiento, resultando en una complejidad que no siempre garantiza mejores resultados que el *ARIMA*.

La combinación de ambos modelos en un enfoque híbrido *ARIMA-LSTM* resultó ser la estrategia más robusta, ya que logró reducir errores. Este enfoque aprovecha tanto las características lineales como las no lineales presentes en los datos, lo que reafirma la utilidad de integrar técnicas de econometría y aprendizaje automático en escenarios de alta complejidad. Además, el modelo híbrido permitió reflejar con mayor precisión las variaciones en el precio de las acciones, destacando su aplicabilidad en la predicción de tendencias a corto plazo en mercados similares.

Por otro lado, los resultados obtenidos en este estudio proporcionan evidencia a favor de la teoría de eficiencia de mercado en el caso de las acciones de Mercado Libre, ya que los residuos de los modelos no presentaron autocorrelación significativa, lo cual sugiere que los precios de las acciones ya incorporan la información disponible. Esto implica que, si bien es posible anticipar ciertas tendencias a corto plazo, la posibilidad de obtener retornos consistentemente superiores mediante predicciones es limitada en un mercado que se comporta de manera eficiente.

Para futuras investigaciones, resulta relevante explorar la integración de variables adicionales, como indicadores macroeconómicos y factores externos que puedan influir en los precios de las acciones, a fin de robustecer los modelos y ampliar su capacidad predictiva. Estos hallazgos pueden ser de gran utilidad para analistas e inversionistas interesados en mejorar sus estrategias de inversión en mercados con características similares.

9. Referencias

Rhanoui et al. (2019). Forecasting financial budget time series: ARIMA random walk vs LSTM neural network. IAES International Journal of Artificial Intelligence, 11.

Archit. (2023, January 17). Medium. Retrieved from Introduction to Long Short-Term Memory (LSTM): https://medium.com/analytics-vidhya/introduction-to-long-short-term-memory-lstm-a8052cd0d4cd

Azura et al. (2019). Financial Modeling with Hybrid Arima-Garch Models: Evidence for Argentina. Ciencias Economicas, 17.

Box et al. (2016). Times series analysis: Forecasting and Control. United Stated of America: Willey & Sons.

CNV. (n.d.). Argentina.gob.ar. Retrieved from Comisión Nacional de Valores: https://www.argentina.gob.ar/cnv/institucional/quienes-somos

Cox. (2005). Electronic Trading and the Nasdaq Market. New York: Financial Times Press.

Crespo. (2007). Invertir y ganar en la bolsa. Buenos Aires: Gestión 2000.

Developers, M. ((n.d.)). Matplotlib: Python plotting library. Retrieved from https://matplotlib.org/

Developers, N. ((n.d.)). NumPy: The fundamental package for scientific computing with Python. Retrieved from https://numpy.org/

Developers, P. ((n.d.)). Pandas: Python data analysis library. Retrieved from https://pandas.pydata.org/

Developers, S.-l. ((n.d.)). Scikit-learn: Machine learning in Python. Retrieved from https://scikit-learn.org/

Developers, T. ((n.d.)). TensorFlow: Machine Learning for everyone. Retrieved from https://www.tensorflow.org/

Dip et al. (2015). A comparison of neural networks and arch-garch models to predict changes in share prices. An application to the case of sotck in the telecomunication industry. Centro de investigacion en metodos cuantitativos aplicados a la economia y la gestion, 29.

Dow. (1884). The Dow Theory. The Wall Street Journal.

Fama. (1969). Efficient Capital Markets: A Review of Theory and Empirical Work. American Finance Association, 36.

Fernandez et al. . (2010). Teoría y práctica de la bolsa. Madrid: Diaz de Santos.

Gadosey et al. (2019). A Comparison between ARIMA, LSTM , and GRU for Time Series Forecasting. Association for Computing Machinery, 7.

Heredia. (2008). La Bolsa de comercio de Buenos Aires. Research Gate, 69.

Hua. (2020). Bitcoin price prediction using ARIMA and LSTM. University of California San Diego, 5.

Huang et al. (2023). Stock Price Prediction Based on ARIMA-GARCH and LSTM. Atlantis Press, 11.

Kirsch et al. (2018). Machine Learning for dummies. New Jersey: John Wiley & Sons, Inc.

Kontopoulou et al. (2023). A Review of ARIMA vs. Machine Learning Approaches for Time Series Forecasting in Data Driven Network. Future internet, 31.

Kuhn et al. (2013). Applied Predictive Modeling. Springer.

Latif et al. (2023). Comparative Performance of LSTM and ARIMA for the Short-Term Prediction of Bitcoin Prices. Australasian Accounting, Business and Finance Journal, 21.

Latina. (2023, 07 11). Acción latina trading. Retrieved from Historia del NASDAQ: Como funcionala bolsa electronica: https://thevisionary.finamex.com.mx/the-visionary/que-es-el-nasdaq-y-como-invierto-en-el

Lucero. (2012). Mercados Financieros. San Luis: Universidad Catolica de Cuyo.

Macchi. (1998). La inversión bursátil. Buenos Aires: editorial tesis.

Menacho. (2014). Camparison of time series methods and neural networks. Universidad Nacional Agraria La Molina, Lima - Peru, 8.

Mishkin. (2014). Moneda, Banca y mercados financieros. Mexico: Pearson.

Morris. (2004). The evolution of the Nasdaq: From Over-theCounter to Electronic Exchange. Harvard Business Review, 56-63.

Namini et al. (2019). A Comparative Analysis of Forecasting Financial Time Series Using ARIMA, LSTM, and BiLSTM. Cornell Univerity, 8.

Ramírez. (2007). La hipótesis del mercado eficiente y la hipótesis conductista de los mercados financieros. Revista Universitaria de Administracion (novaRua), 12.

Sánchez. (2015). Análisis Bursátil. Madrid: ICADE BUSINESS SCHOOL.

Sartori et al. (2017). Pronostico del precio de una accion del mercado argentino: aplicacion del analisis fundamental y de modelos autorregresivos y de rezagos distribuidos. Repositorio digital universitario (RDU-UNC), 22.

Scherk. (2007). Manual de Análisis Fundamental. Madrid: Inversor Ediciones, S.L.

Soros. (2009). El nuevo paradigma de los mercados financieros. Madrid: Taurus.

Tanco. (2020). Redes Neuronales Artificiales. Buenos Aires: UTN.BA.

Taylor. ((n.d.).). pmdarima: Statistical library for ARIMA models in Python. Retrieved from https://alkaline-ml.com/pmdarima/

Watsons. (2013). BEHAVIORIST MANIFESTO. redalyc.

yfinance. ((n.d.).). Yahoo! Finance market data downloader. Retrieved from https://github.com/ranaroussi/yfinance