

Desafíos en la Predicción de Precios de Criptomonedas: Implementación de un Sistema con Big Data Streaming y Machine Learning.



Autor: Alexandre Vidal De Palol

Máster Universitario en Ciencia de Datos

Área 1

Tutor: Rafael Luque Ocaña

**Profesora responsable de la asignatura:
Susana Acedo Nadal**

20/12/2024



Copyright © 2024 Alexandre Vidal De Palol.

Esta obra está sujeta a una licencia de Reconocimiento-NoComercial-Compartir Igual [3.0 España de Creative Commons](#)

FICHA DEL TRABAJO FINAL

Título del trabajo:	Desafíos en la Predicción de Precios de Criptomonedas: Implementación de un Sistema con Big Data Streaming y Machine Learning
Nombre del autor:	Alexandre Vidal De Palol
Nombre del Tutor:	Rafael Luque Ocaña
Nombre de la PRA:	Susana Acedo Nadal
Fecha de entrega:	13/10/2024
Titulación:	Máster en ciencia de datos
Área del Trabajo Final:	Área 1
Idioma del trabajo:	Español
Palabras clave	Criptomonedas, Big Data Streaming, Predicción de precios
Resumen del Trabajo	
Este trabajo tiene como objetivo implementar un sistema predictivo para analizar los precios de criptomonedas, utilizando herramientas de procesamiento de datos en tiempo real como Apache Kafka, MongoDB y Spark, junto con modelos predictivos como ARIMA y Prophet. El proyecto aborda los desafíos de las fluctuaciones en este mercado volátil, evaluando la efectividad de estas técnicas avanzadas de predicción en términos de precisión y aplicabilidad, con el fin de contribuir al análisis financiero de este mercado emergente.	

Abstract

This project aims to implement a predictive system to analyze cryptocurrency prices, using real-time data processing tools such as Apache Kafka, MongoDB, and Spark, along with predictive models like ARIMA and Prophet. The project addresses the challenges posed by fluctuations in this volatile market by evaluating the effectiveness of these advanced prediction techniques in terms of precision and applicability, contributing to the financial analysis of this emerging market.

Índice

1. Introducción.....	1
1.1. Contexto y justificación del Trabajo.....	1
1.2. Objetivos del Trabajo.....	2
1.3. Impacto en sostenibilidad, ético-social y de diversidad.....	3
1.4 Enfoque y método seguido.....	4
1.5. Planificación del Trabajo.....	6
2. Estado del arte.....	7
2.1. Introducción.....	7
2.2. Evolución de las criptomonedas y el mercado financiero.....	7
2.3. Tecnologías de Big Data Streaming y su aplicación en el análisis financiero.....	9
2.4. Modelos de predicción de series temporales en mercados financieros.....	12
2.5. Avances recientes en modelos predictivos para criptomonedas.....	14
2.6. Desafíos y limitaciones en la predicción de precios de criptomonedas.....	16
3. Materiales y Métodos.....	17

3.1. Introducción.....	17
3.2. Selección de las Criptomonedas	17
3.3. Pipeline del proyecto anaítico.....	18
3.4. Tecnologías y herramientas utilizadas.....	19
3.5. Preparación y limpieza de datos.....	25
3.6. Modelos predictivos.....	29
4. Resultados.....	35
4.1. Evaluación de los modelos predictivos.....	35
4.2. Análisis predictivo individual por criptomoneda.....	39
5. Concusiones y trabajos futuros.....	49
5.1. Conclusiones.....	49
5.2. Trabajos futuros.....	50
6. Glosario.....	52

7.Bibliografía.....	53
8.Anexos.....	57

Lista de tablas

Tabla 1: Resultados de la configuración de parámetros del modelo ARIMA.....32

Tabla 2: Resultados de las métricas MAE, RMSE y MAPE38

Lista de figuras

Figura 1: Diagrama de Gantt de la planificación de TFM	6
Figura 2: Capitalización de mercado de criptomonedas	8
Figura 3: Comparativa de características en plataformas de Big Data Streaming	10
Figura 4: Comparación de precisión entre los modelos Prophet y ARIMA	13
Figura 5: Comparativa de precisión en modelos predictivos	15
Figura 6: Pipeline del Proyecto Analítico	18
Figura 7: Configuración y uso de la API de CoinGecko	20
Figura 8: Configuración del productor de Apache Kafka	21
Figura 9: Configuración de MongoDB	22
Figura 10: Configuración de Interfaz de MongoDB Compass	23
Figura 11: Función collect_and_store_data que integra las etapas del pipeline	24
Figura 12: Código para la carga de datos con SparkSQL	25
Figura 13: Conversión del campo timestamp a formato datetime	26

Figura 14: Agrupación de datos en intervalos de 8 horas27

Figura 15: Código para el remuestreo de datos a intervalos regulares27

Figura 16: Código para el filtrado y selección de datos por criptomoneda28

Figura 17: Renombrar columnas al formato requerido por Prophet30

Figura 18: Evaluación de la estacionariedad y aplicación de la diferenciación31

Figura 19: Impacto del parámetro changepoint_prior_scale en el modelo Prophet.....34

Figura 20: Ejemplo de validación cruzada con conjuntos de entrenamiento36

Figura 21: Predicciones de ARIMA para Bitcoin40

Figura 22: Predicciones de Prophet para Bitcoin41

Figura 23: Predicciones de Arima para Ethereum43

Figura 24: Predicciones de Prophet para Ethereum44

Figura 25: Predicciones de Arima para Binance Coin46

Figura 26: Predicciones de Prophet para Binance Coin48

1. Introducción

1.1. Contexto y justificación del Trabajo

La creciente popularidad de las criptomonedas en los últimos años ha despertado el interés de gran parte del mundo financiero e inversión. A diferencia de los mercados bursátiles tradicionales, el mercado de las criptomonedas ofrece tanto una oportunidad como un desafío en el campo del análisis predictivo por ser altamente volátil y dinámico.

La investigación de este proyecto pretende demostrar cómo la aplicación de herramientas de ciencia de datos puede optimizar la fiabilidad de las estimaciones en el ámbito financiero, permitiendo a los inversionistas adaptarse de forma más eficiente a las variaciones del mercado.

Desde hace tiempo me fascina el mundo de las criptomonedas y su potencial para transformar el sistema financiero. La tecnología blockchain, su descentralización y la innovación constante en este campo me parecen aspectos sumamente interesantes. Además, desde que empecé mi formación en los estudios de ciencia de datos he desarrollado un interés particular por el análisis de datos y el uso de técnicas de aprendizaje automático avanzadas para abordar problemas complejos. En este sentido, este proyecto me permite poner a prueba mis habilidades en ciencia de datos y explorar el potencial de las nuevas tecnologías para predecir el comportamiento de este mercado. Me entusiasma la idea de desarrollar un sistema que pueda contribuir a explorar nuevas formas de analizar y predecir el comportamiento del mercado de criptomonedas.

1.2. Objetivos del Trabajo

1.2.1. Objetivo principal

Diseñar un sistema capaz de realizar predicciones sobre el comportamiento del mercado de criptomonedas, mediante la integración de técnicas avanzadas de procesamiento de datos y aprendizaje automático.

1.2.2. Objetivos secundarios

- Diseñar e implementar un proceso automatizado para recopilar y procesar datos de mercado en tiempo real de un conjunto de criptomonedas, garantizando la integridad, fiabilidad y disponibilidad de los datos.
- Implementar diferentes modelos de aprendizaje automático para la predicción de precios, evaluando su efectividad en la identificación de tendencias y patrones.
- Evaluar y comparar el rendimiento de los modelos predictivos en términos de precisión y rendimiento, para determinar cuál se adapta mejor en base a los datos obtenidos.
- Desarrollar un diseño de visualización que facilite la comprensión de los resultados obtenidos recogidos en los análisis.
- Evaluar la eficacia del sistema implementado en el proyecto como herramienta de análisis predictivo para el mercado de criptomonedas, considerando su capacidad para generar predicciones precisas y facilitar la interpretación de las tendencias del mercado.

1.3. Impacto en sostenibilidad, ético-social y de diversidad

1.3.1 Sostenibilidad

Uso eficiente de recursos: la implementación de técnicas de procesamiento de datos y algoritmos avanzados más eficientes en términos de recursos computacionales contribuye a un consumo de energía más bajo, como por ejemplo seleccionando modelos que reducen significativamente las horas de procesamiento a la hora de obtener los resultados.

El trabajo aplica modelos predictivos como ARIMA y Prophet, que son eficientes en términos computacionales debido a su menor complejidad en comparación con otros modelos como las redes neuronales [1].

1.3.2 Ética y responsabilidad social

Manejo responsable de datos: Se emplean únicamente datos públicos y se siguen todas las políticas de uso de las fuentes de datos manteniendo prácticas responsables. En la práctica, esto inspira a otras investigaciones en el sector financiero a seguir una conducta ética similar, contribuyendo a un ecosistema de datos más seguro y responsable.

Transparencia en los modelos predictivos: El estudio destaca los alcances y limitaciones de las predicciones para ayudar en la toma de decisiones informadas, especialmente en un mercado volátil. El trabajo fomenta la toma de decisiones responsables al proporcionar a los usuarios la información necesaria para comprender los riesgos asociados a la inversión en criptomonedas.

1.3.3 Diversidad y Derechos Humanos

Acceso inclusivo y equitativo: Al emplear herramientas de acceso abierto, el proyecto es accesible para personas en todo el mundo, sin costes económicos. El código del proyecto y la documentación están disponibles en mi repositorio público de GitHub, permitiendo que cualquier persona interesada, incluyendo estudiantes e inversores, acceda al trabajo sin coste alguno y lo utilice como recurso de aprendizaje.

Visualización comprensible para diversos públicos: El sistema de visualización prioriza la accesibilidad y comprensión de los resultados para una audiencia diversa. Se

utilizan herramientas como Matplotlib para generar gráficos claros y efectivos. Esto permite que usuarios con diferentes niveles de experiencia puedan interpretar y utilizar la información presentada.

1.4. Enfoque y método seguido

Para el desarrollo de este proyecto, se optó por la metodología CRISP-DM (Cross Industry Standard Process for Data Mining), ampliamente utilizada en el ámbito de la ciencia de datos debido a su estructura adaptativa y centrada en el análisis. Este enfoque permitió organizar y gestionar de forma estructurada el ciclo de vida del proyecto, resultando particularmente adecuado para un contexto de alta volatilidad como el del mercado de criptomonedas.

Los datos se obtuvieron principalmente a través de una API que recogió información en tiempo real sobre los precios de una selección específica de criptomonedas. Se utilizó Python como lenguaje de programación principal, junto con herramientas como Pandas y Spark para el análisis, manipulación, limpieza y procesamiento de datos. Además, se empleó Apache Kafka como plataforma distribuida de transmisión de datos en tiempo real, y MongoDB para su almacenamiento.

El modelado predictivo consideró distintos enfoques para analizar los precios de las criptomonedas. Se implementaron los modelos ARIMA y Prophet, seleccionados por su capacidad para identificar tendencias, patrones de autocorrelación y estacionalidad en las series temporales. Esta combinación de modelos permitió evaluar cuál se adaptó mejor a los datos y optimizar las predicciones.

Los resultados se presentan mediante herramientas de visualización como Matplotlib, lo que facilita la interpretación de los análisis.

El proceso de trabajo se divide en las siguientes etapas:

- **Etapa 1: Selección de las criptomonedas a analizar**

Se seleccionan criptomonedas de mayor capitalización de mercado y alta liquidez. La elección se justifica en función de la relevancia y disponibilidad de datos, proporcionando una breve descripción de cada moneda.

- **Etapa 2: Recopilación de datos en tiempo real**

Se utiliza una API para obtener datos de mercado en tiempo real, automatizando la recolección mediante un script en Python. Los datos se transmiten a través de Apache Kafka y se almacenan en MongoDB para su posterior análisis.

- **Etapa 3: Preparación y limpieza de datos**

Para transformar y limpiar los datos, se utilizan herramientas como Apache Spark y Pandas. Esta etapa asegura que los datos sean precisos y adecuadamente preparados para las etapas de análisis y modelado.

- **Etapa 5: Modelado predictivo**

En el modelado predictivo, se aplican modelos como ARIMA, que captura tendencias y patrones de autocorrelación en los datos. También se considera el uso de Prophet, que ofrece un enfoque automatizado para manejar datos con patrones complejos como la estacionalidad.

1.5. Planificación del Trabajo

En el diagrama de Gantt (Figura 1) se muestran las diferentes etapas de planificación del trabajo de fin de máster.

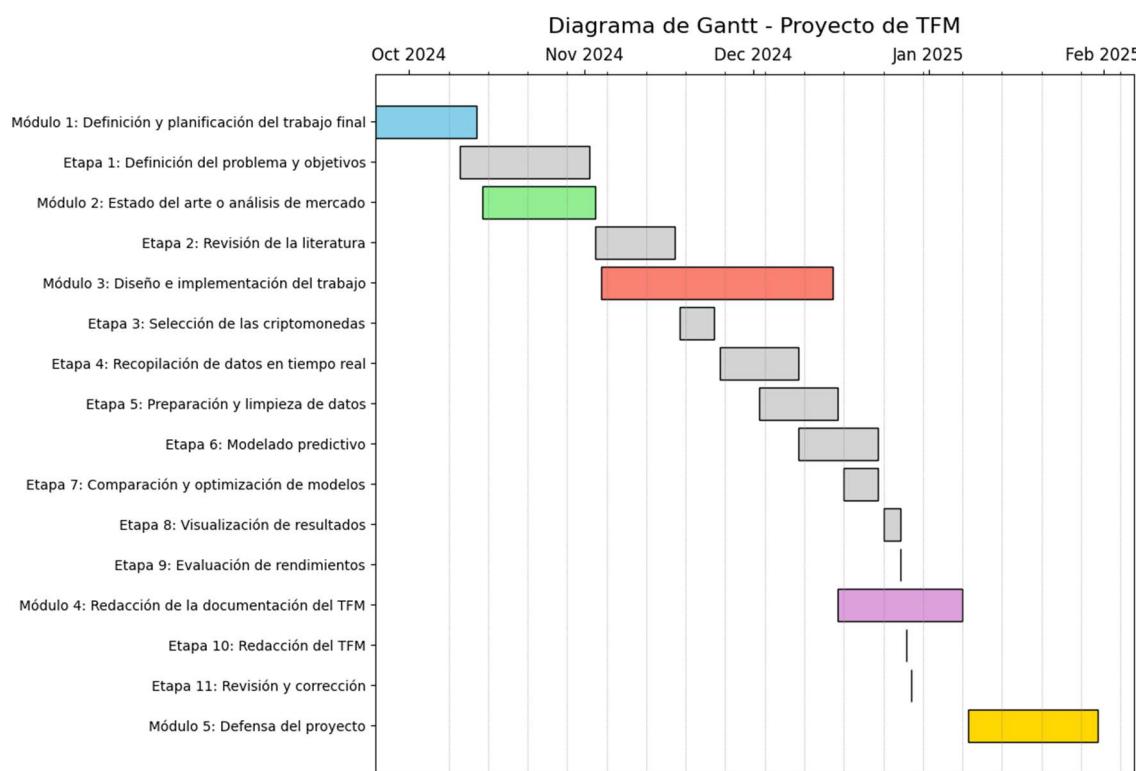


Figura 1. Diagrama de Gantt de la planificación de TFM. Elaboración propia.

2. Estado del Arte

2.1. Introducción

El mercado de las criptomonedas ha experimentado un crecimiento exponencial en los últimos años, captando la atención de inversores, empresas y gobiernos de todo el mundo. Impulsado por la tecnología blockchain, este mercado se caracteriza por su alta volatilidad y dinamismo, lo que presenta tanto oportunidades como desafíos para quienes buscan anticiparse a sus movimientos y tendencias. La predicción de precios en criptomonedas se ha convertido en un área de gran interés, ya que permite a los inversores tomar decisiones más precisas y gestionar los riesgos en un entorno altamente especulativo.

Este capítulo aborda los fundamentos teóricos y tecnológicos necesarios para implementar sistemas predictivos en el ámbito de las criptomonedas, situándolos en el contexto del mercado financiero actual. A partir de un análisis detallado de la literatura, se exploran aspectos esenciales como la evolución del mercado de criptomonedas, los enfoques actuales en la predicción de series temporales y el impacto de las tecnologías de Big Data Streaming en el análisis financiero. Además, se analizan los retos específicos asociados a los entornos de alta volatilidad, así como las oportunidades de innovación que estos escenarios ofrecen, destacando su relevancia para la investigación y el desarrollo de modelos predictivos aplicados a un mercado en constante transformación.

2.2. Evolución de las criptomonedas y el mercado financiero

El auge de las criptomonedas en la última década ha transformado profundamente el panorama financiero global. Desde la aparición de Bitcoin en 2009 como una moneda digital descentralizada basada en la tecnología blockchain, el mercado de criptomonedas ha experimentado un crecimiento exponencial, con la proliferación de altcoins y un aumento significativo en su capitalización de mercado. Este crecimiento ha captado la atención de inversores, empresas y gobiernos, posicionando a las criptomonedas como un activo financiero disruptivo con potencial para desafiar los sistemas económicos tradicionales. La reciente aceptación de criptomonedas en plataformas como PayPal y su adopción en algunos países subrayan su impacto creciente y la adaptación necesaria del sistema financiero tradicional a esta nueva realidad [2].

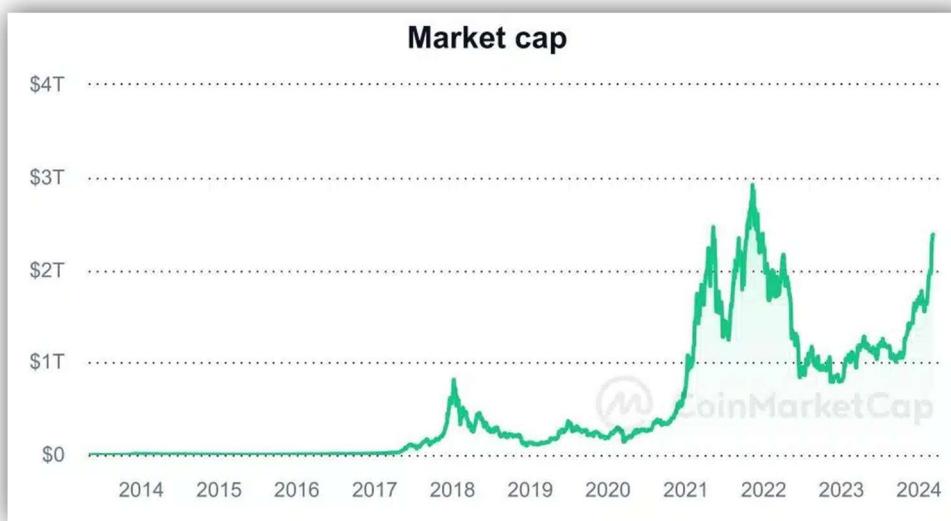


Figura 2. Capitalización de mercado de criptomonedas en billones (trillones) de dólares estadounidenses a marzo de 2024. Fuente: CoinMarketCap a través de Techopedia [3]

La Figura 2 muestra la tendencia creciente de la capitalización del mercado de criptomonedas en la última década. Se destacan fases de crecimiento acelerado a partir de 2017, con máximos significativos en 2021 y una recuperación reciente en 2024. Estos datos reflejan tanto el interés sostenido en los activos digitales como la alta volatilidad que caracteriza a este mercado emergente.

Las criptomonedas se caracterizan por una serie de atributos distintivos que las diferencian de las monedas tradicionales. Su naturaleza descentralizada elimina la necesidad de intermediarios, como los bancos centrales, y ofrece mayor autonomía a los usuarios. El anonimato en las transacciones, la seguridad criptográfica y la transparencia de la información en la blockchain han impulsado su adopción, especialmente entre aquellos que buscan un control más directo sobre sus activos y transacciones [2]. La descentralización promueve un tipo de confianza fundamentada en la criptografía en lugar de instituciones tradicionales, una característica particularmente relevante en la era digital actual [4].

La tecnología blockchain, que sustenta a las criptomonedas, actúa como un registro distribuido que almacena información de manera segura e immutable. Cada transacción se agrupa en un bloque, que se conecta criptográficamente a los bloques

anteriores, formando una “cadena de bloques” o blockchain. Esta estructura descentralizada garantiza la integridad de los datos y la resistencia a la censura [2]. Blockchain también tiene el potencial de catalizar la transición hacia una economía digital y circular, al facilitar nuevos modelos de negocio basados en plataformas descentralizadas, promoviendo estructuras más eficientes y transparentes [5].

No obstante, el mercado de criptomonedas enfrenta desafíos significativos. Su alta volatilidad, la escalabilidad de las redes blockchain y la falta de una regulación clara se presentan como obstáculos clave para su aceptación masiva. Esta volatilidad e incertidumbre pueden disuadir a nuevos inversores, por lo que resulta necesario establecer regulaciones y mecanismos que generen mayor confianza [6]. En este contexto, la predicción de precios de criptomonedas se ha convertido en un área de investigación fundamental para entender la dinámica de este mercado y facilitar la toma de decisiones de inversión en un entorno de alta incertidumbre.

Además, la pandemia ha acelerado el uso de monedas digitales y billeteras electrónicas, impulsando la adopción de criptomonedas como un método alternativo de inversión. Este cambio en la mentalidad financiera global ha favorecido su aceptación en diversos sectores y ha obligado a las instituciones financieras tradicionales a reconsiderar sus estrategias para captar y fidelizar a sus clientes [7]. Más allá de la predicción de precios, la aplicación de la Inteligencia Artificial (IA) en el ámbito de las criptomonedas ofrece oportunidades para optimizar portafolios de inversión, prever la volatilidad, detectar fraudes y analizar cuestiones de anonimato y privacidad [8].

2.3. Tecnologías de Big Data Streaming y su aplicación en el análisis financiero

El análisis financiero en tiempo real se ha vuelto fundamental para instituciones y profesionales que buscan optimizar sus decisiones en un mercado global y altamente competitivo. Las tecnologías de Big Data Streaming permiten gestionar grandes volúmenes de datos a medida que se generan, posibilitando decisiones rápidas y la detección de patrones emergentes que pueden influir en las estrategias de inversión. Este procesamiento en tiempo real es particularmente relevante para responder de forma ágil a las fluctuaciones de precios, la aparición de nuevas tendencias y la incidencia de eventos imprevistos que afectan los mercados financieros [9].

2.3.1. Tecnologías clave en Big Data Streaming

Algunas de las plataformas más destacadas en el ámbito de Big Data Streaming incluyen Apache Kafka y Apache Spark Streaming. Apache Kafka es una plataforma distribuida de alto rendimiento que facilita la publicación, suscripción y procesamiento de flujos de datos en tiempo real. Su capacidad para manejar grandes volúmenes de datos de manera escalable y su tolerancia a fallos la convierten en una opción ideal para aplicaciones financieras que requieren una transmisión continua de datos. Por otro lado, Apache Spark Streaming, un componente de la suite Apache Spark, permite el procesamiento en tiempo real y la realización de análisis complejos mediante sus APIs. La capacidad de Spark Streaming para integrarse con diversas fuentes de datos y su procesamiento por micro-lotes lo hace ideal para el análisis detallado y rápido de datos del mercado de criptomonedas [9].

A continuación, se presenta una tabla comparativa (Figura 3) de las principales plataformas de Big Data Streaming, resaltando características relevantes, como la escalabilidad, robustez y manejo en tiempo real, que pueden influir en su utilidad para aplicaciones financieras.

Features	Flink	Kafka	Samza	Spark	Storm
Fault-tolerance	✓	✓	✓	✓	✓
Scalability	✓	✓	✓	✓	✓
Robustness	✓	✓	✓		✓
Dashboards	✓	✓	✓	✓	✓
Integration	✓	✓	✓	✓	✓
Consistency	✓	✓	✓	✓	
Security	✓	✓	✓	✓	✓
Time handling					✓
Stream SQL	✓	✓	✓	✓	✓
ETL optimization	✓	✓		✓	✓
Machine Learning	✓	✓	✓	✓	✓
Elasticity	✓	✓	✓	✓	✓

Figura 3. Comparativa de características en plataformas de Big Data Streaming [9]

2.3.2. Aplicaciones en el análisis financiero

Como destaca E. Fernandes en su artículo “Big Data Streaming Platforms to Support Real-time Analytics” [9], estas tecnologías ofrecen diversas aplicaciones en el ámbito financiero, particularmente en el análisis de datos en tiempo real para la toma de decisiones. A continuación, se presentan algunas de sus principales aplicaciones:

- **Predicción de precios de activos financieros:** Al analizar datos de mercado en tiempo real, como precios, volúmenes de negociación y noticias, estas tecnologías permiten anticipar las fluctuaciones de precios y detectar oportunidades de inversión.
- **Detección de fraudes:** El análisis en tiempo real de flujos de transacciones permite identificar patrones sospechosos y emitir alertas sobre posibles actividades fraudulentas, posibilitando una respuesta rápida y eficaz ante amenazas emergentes.
- **Gestión de riesgos:** A través de la monitorización continua de riesgos de mercado, crédito y liquidez, el Big Data Streaming permite una adaptación ágil a las condiciones cambiantes del mercado, ayudando a reducir pérdidas y a mitigar riesgos asociados.
- **Análisis de sentimiento:** El análisis en tiempo real de datos de redes sociales y noticias permite conocer el sentimiento del mercado, un factor que puede influir en la predicción de precios al reflejar la percepción pública sobre determinados activos.

2.3.3. Desafíos y oportunidades de las tecnologías de Big Data Streaming

La implementación de Big Data Streaming en el análisis financiero presenta una serie de desafíos importantes, los cuales deben ser abordados para maximizar el potencial de estas tecnologías en tiempo real. Entre los principales retos se encuentran [9]:

- **Escalabilidad y rendimiento:** La alta velocidad y volumen de datos en tiempo real requieren sistemas capaces de escalar eficientemente para mantener un rendimiento constante y evitar cuellos de botella en el procesamiento.
- **Latencia:** Reducir la latencia en el procesamiento es esencial para reaccionar de forma rápida ante eventos del mercado en tiempo real. Una alta latencia puede limitar la utilidad de los análisis en entornos que exigen decisiones inmediatas.

- **Precisión:** Garantizar la precisión de los análisis en tiempo real resulta fundamental para obtener resultados fiables, permitiendo que las decisiones basadas en dichos análisis sean fundamentadas y seguras.
- **Integración de datos:** La consolidación de datos provenientes de diversas fuentes y en distintos formatos plantea desafíos adicionales, ya que se requiere una estructura que asegure la coherencia y calidad de los datos procesados.

2.4. Modelos de predicción de series temporales en mercados financieros

La predicción de series temporales en los mercados financieros es fundamental para anticipar cambios de precios y reducir la incertidumbre en la toma de decisiones. Según Beeram y Kuchibhotla [10], uno de los modelos más utilizados en este contexto es ARIMA (Autoregressive Integrated Moving Average), debido a su capacidad para capturar patrones en series temporales y a su aplicabilidad en datos financieros. ARIMA resulta particularmente útil cuando los datos muestran tendencia o estacionalidad, ya que permite descomponer la serie en componentes que explican estas características.

ARIMA se compone de tres elementos esenciales que permiten capturar distintos aspectos de las series temporales [10]:

- **Autoregresivo (AR):** Utiliza valores pasados de la serie para predecir valores futuros, basándose en la idea de que los datos actuales están relacionados con los valores previos.
- **Integrado (I):** Se enfoca en hacer que los datos sean estacionarios, es decir, elimina tendencias para que la serie sea más predecible.
- **Media Móvil (MA):** Reduce la influencia de variaciones inesperadas en los datos, suavizando la serie y facilitando la identificación de patrones.

Esta estructura hace que ARIMA sea adecuado para mercados como el de criptomonedas, donde es común observar tendencias temporales en los precios. Su capacidad de descomponer la serie permite entender mejor su comportamiento, lo que lo convierte en una herramienta eficaz y ajustable para predecir precios de criptomonedas. No obstante, a pesar de su utilidad, ARIMA presenta limitaciones en la detección de patrones no lineales en datos muy volátiles, como los de criptomonedas, los cuales pueden ser mejor abordados por modelos avanzados como las Redes Neuronales Recurrentes (RNN) [11].

Un aspecto destacado del modelo ARIMA es su capacidad para identificar patrones frecuentes en series temporales financieras, especialmente útil en mercados como el de criptomonedas, donde los precios suelen mostrar fluctuaciones repetitivas. La detección de estos patrones recurrentes puede ser valiosa para anticipar tendencias a corto plazo y responder de forma ágil a los cambios de mercado. Además, la simplicidad de ARIMA facilita ajustes rápidos y eficientes a los datos financieros, lo cual es particularmente ventajoso en el contexto de criptomonedas, donde es crucial contar con un sistema adaptable a la volatilidad de los precios [12].

Para complementar ARIMA, el modelo Prophet se presenta como una alternativa eficaz, abordando algunas limitaciones de ARIMA mediante una estructura flexible que admite la incorporación de múltiples horizontes temporales. Esto resulta valioso en mercados de alta volatilidad como el de criptomonedas, donde es común requerir tanto predicciones a corto como a largo plazo. Prophet también permite integrar factores externos, como el valor del dólar estadounidense, capturando cómo estos factores pueden afectar los precios de las criptomonedas en tiempo real, una característica especialmente útil en un contexto tan sensible a cambios externos [13].

Además, garantizar la estacionariedad y realizar un análisis de varianza son elementos cruciales en la aplicación de ARIMA para asegurar interpretaciones precisas y predicciones fiables. La estabilidad en los datos evita falsas interpretaciones y permite un ajuste del modelo que capte mejor las fluctuaciones de precios, algo esencial en el mercado de criptomonedas, caracterizado por su alta volatilidad y cambios rápidos [12].

TABLE I: R^2 AND ERROR VALUES FOR PROPHET FORECAST MODEL

Set	R^2	MSE	RMSE	RMSPE	MAE	MAPE
Train	0.997	35605	188.69	0.036	71.384	0.026
Val	0.941	425344	652.18	0.054	520.503	0.044
Test	0.945	60069	245.09	0.024	196.652	0.020

TABLE II: R^2 AND ERROR VALUES FOR ARIMA FORECAST MODEL

Set	R^2	MSE	RMSE	RMSPE	MAE	MAPE
Train	0.997	38498	196.21	0.057	85.254	0.043
Val	0.907	667508	817.01	0.071	678.807	0.059
Test	0.681	352600	593.80	0.062	541.946	0.056

Figura 4. Comparación de precisión entre los modelos Prophet y ARIMA en la predicción de precios de Bitcoin [14]

La Figura 4 muestra una comparación entre los modelos Prophet y ARIMA en la predicción de precios de Bitcoin. Prophet demuestra una mayor precisión en el conjunto de prueba, especialmente al tener errores más bajos en la predicción de precios comparado con ARIMA. Mientras que Prophet mantiene una menor desviación en sus predicciones, ARIMA presenta mayores errores, especialmente en los valores de validación y prueba. Esto sugiere que Prophet se adapta mejor a las fluctuaciones y volatilidad del mercado de criptomonedas, ofreciendo predicciones más consistentes y fiables para entornos de alta variabilidad [14].

En conjunto, la flexibilidad y eficiencia de ARIMA y Prophet los convierten en herramientas adecuadas para este trabajo. Aunque menos complejos que otros modelos avanzados, estos enfoques estructurados permiten descomponer series temporales de manera efectiva, capturando patrones de cambio en los precios de criptomonedas y proporcionando un sistema de predicción robusto y adaptable en un mercado de alta frecuencia [13].

2.5. Avances recientes en modelos predictivos para criptomonedas

La predicción de precios en el mercado de criptomonedas ha avanzado considerablemente gracias a la incorporación de técnicas de aprendizaje profundo y modelos híbridos, mejorando la precisión en un entorno caracterizado por alta volatilidad. Entre los enfoques más efectivos se encuentran las Redes Neuronales Recurrentes (RNN) y, especialmente, las Redes Neuronales de Memoria a Largo Plazo (LSTM), valoradas por su capacidad de analizar datos secuenciales y reconocer patrones en series temporales financieras. El modelo LSTM, con su habilidad para retener información relevante a lo largo del tiempo, se adapta bien a los cambios rápidos en los precios de las criptomonedas, superando a las RNN en precisión, en particular en la identificación de dependencias temporales dentro de los datos [10].

Además, los modelos híbridos han ampliado las capacidades predictivas en este campo. Al combinar métodos tradicionales con redes neuronales avanzadas, estos modelos logran captar patrones tanto lineales como no lineales en los datos, lo que incrementa notablemente la precisión y adaptabilidad en la predicción de precios. Este tipo de integración ha demostrado mejoras significativas en la precisión de las predicciones en estudios recientes [15].

Model	RMSE	MAE	R ²
ARIMA	0.058	0.045	0.82
LSTM	0.042	0.033	0.89
GRU	0.040	0.032	0.91
Hybrid LSTM-GRU	0.035	0.027	0.94

Figura 5. Comparativa de precisión en modelos predictivos para criptomonedas [16]

La Figura 5 muestra una comparación entre varios tipos de modelos de predicción aplicados a las criptomonedas: el modelo tradicional ARIMA, los modelos de aprendizaje profundo LSTM y GRU, y un modelo híbrido que combina LSTM y GRU. Como se puede ver en los resultados, el modelo híbrido LSTM-GRU muestra una mayor precisión en comparación con los otros, lo que lo convierte en una opción especialmente prometedora para este tipo de predicciones [16].

Estudios adicionales han evaluado y comparado varios métodos de predicción, destacando el desempeño superior de LSTM frente a otros enfoques, como las máquinas de soporte vectorial (SVM) y algoritmos de árboles como XGBoost. La capacidad de LSTM para identificar patrones en datos altamente variables refuerza su utilidad para anticipar precios en mercados volátiles como el de criptomonedas [17].

Finalmente, estos avances en modelos predictivos han impulsado el desarrollo de estrategias de trading adaptativo, donde las predicciones de precios permiten a los inversores ajustar decisiones en tiempo real, optimizando la gestión de riesgos y aprovechando las oportunidades en un mercado volátil. Este enfoque se ha convertido en una tendencia clave en el ámbito de las criptomonedas, posicionando a los modelos predictivos como componentes esenciales en sistemas de trading adaptativos y eficaces [18].

En conjunto, estos avances en aprendizaje profundo, técnicas híbridas y estrategias de trading adaptativo establecen una base sólida para modelos de predicción más precisos y adaptativos, ofreciendo a los inversores herramientas robustas para enfrentar los cambios continuos del mercado de criptomonedas.

2.6. Desafíos y limitaciones en la predicción de precios de criptomonedas

A pesar de los avances en modelos predictivos y técnicas de aprendizaje profundo, la predicción de precios en criptomonedas sigue enfrentando importantes desafíos. Uno de los principales problemas es la alta volatilidad del mercado, que dificulta que modelos avanzados como LSTM y RNN puedan adaptarse eficazmente a cambios repentinos de precios. Aunque estos modelos son efectivos en el análisis de datos secuenciales, el comportamiento impredecible y volátil de las criptomonedas limita su precisión. Además, la sensibilidad de estos modelos a factores externos, como políticas gubernamentales o eventos globales, añade una capa de complejidad, ya que estos elementos pueden afectar de forma inesperada y significativa los precios en el mercado [17].

Otro desafío clave es el riesgo de sobreajuste, en el que los modelos se ajustan demasiado a los datos históricos y pierden efectividad en situaciones nuevas y no anticipadas. Para atenuar este problema, se requiere un preprocesamiento riguroso y la selección cuidadosa de variables. La literatura señala además que el rendimiento de los modelos puede variar considerablemente entre los enfoques univariantes y multivariantes, destacando la importancia de elegir y ajustar adecuadamente los modelos para obtener predicciones más precisas en entornos volátiles como el de las criptomonedas [19].

A su vez, los modelos predictivos en criptomonedas dependen en gran medida de datos históricos, lo cual limita su capacidad para adaptarse a eventos inesperados en tiempo real. Esta dependencia, junto con la necesidad de un procesamiento cuidadoso de datos y la integración de factores externos, afecta la fiabilidad de las predicciones. La integración de múltiples factores externos plantea desafíos adicionales y subraya la necesidad de mejorar la resiliencia de los modelos para mantener su precisión en condiciones de mercado fluctuantes [20].

Estos desafíos resaltan la necesidad de innovar en el desarrollo de modelos predictivos que puedan adaptarse mejor a la naturaleza dinámica de las criptomonedas. La investigación futura podría enfocarse en enfoques que combinen aprendizaje profundo con sistemas de predicción en tiempo real y técnicas de aprendizaje adaptativo, permitiendo que los modelos ajusten sus predicciones automáticamente ante cambios en el mercado. Asimismo, la integración de fuentes de datos adicionales, como el análisis de sentimiento en redes sociales y otros indicadores externos, podría mejorar la precisión y reducir algunos de los obstáculos que los modelos actuales enfrentan en mercados tan volátiles como el de criptomonedas [19].

3. Materiales y Métodos

3.1. Introducción

En este proyecto se utilizaron diversas herramientas y tecnologías para garantizar un flujo de trabajo eficiente y escalable, desde la recolección de datos en tiempo real hasta su análisis y visualización. A continuación, se detallan las herramientas utilizadas, las criptomonedas seleccionadas y las etapas clave del flujo de trabajo.

3.2. Selección de las criptomonedas

La elección de Bitcoin, Ethereum y Binance Coin para este proyecto se basa por su importancia en el mercado de criptomonedas:

- **Bitcoin:** Es la primera criptomoneda creada y la más reconocida a nivel mundial, con un impacto significativo en el mercado financiero. Es considerada la referencia principal del sector, destacándose por su alta capitalización y su influencia predominante en el mercado de criptomonedas.
- **Ethereum:** Reconocida como la segunda criptomoneda más relevante después de Bitcoin, Ethereum se ha consolidado en el segundo puesto por su capacidad de innovación y su adopción en diferentes aplicaciones descentralizadas.
- **Binance Coin:** Es la criptomoneda propia de Binance, uno de los exchanges más grandes y reconocidos a nivel mundial. Su creciente uso en transacciones, servicios y aplicaciones fuera del exchange ha consolidado su posición como una de las criptomonedas más influyentes del mercado.

3.3. Pipeline del proyecto analítico

El pipeline diseñado para este proyecto establece una estructura clara y lógica que integra las herramientas y técnicas empleadas en cada fase, abarcando desde la obtención de datos hasta su análisis y visualización. El diagrama presentado a continuación (Figura 6) ilustra la interacción entre los componentes y resalta la secuencia de las etapas clave del proceso.

Pipeline del proyecto

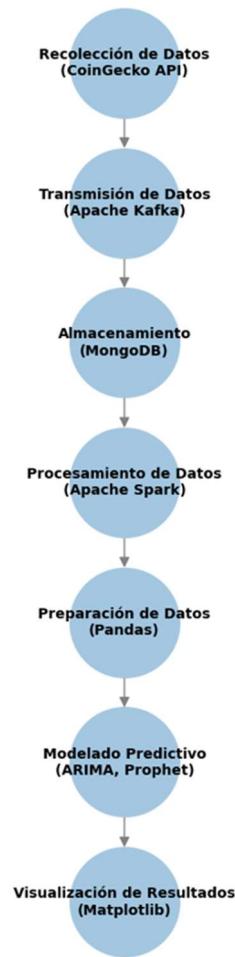


Figura 6. Pipeline del Proyecto Analítico. Elaboración propia.

3.4. Tecnologías y herramientas utilizadas

3.4.1. Obtención de datos con CoinGecko API

La API de CoinGecko fue seleccionada como fuente principal de datos por su amplia cobertura del mercado de criptomonedas y su capacidad para ofrecer información actualizada de manera confiable. En este proyecto, se recurrió al endpoint *Simple Price* disponible en la sección destinada a desarrolladores de la plataforma [21], que proporciona precios en USD de Bitcoin, Ethereum y Binance Coin.

Se eligió esta API porque es una de las más conocidas y completas en el mundo de las criptomonedas. Ofrece datos de calidad, actualizaciones constantes y una cobertura muy amplia del mercado, lo que la hace muy útil para este tipo de análisis. La API constituye la primera etapa del flujo de trabajo, recolectando datos en tiempo real que son transmitidos a través de Apache Kafka para su almacenamiento y análisis.

Configuración de la API

En este proyecto, se definieron las siguientes configuraciones principales para interactuar con la API de CoinGecko:

- **URL base:** `COINGECKO_API_URL`, que representa el endpoint principal para las solicitudes.
- **Criptomonedas seleccionadas:** Bitcoin, Ethereum y Binance Coin, identificadas por sus códigos (`CRYPTOCURRENCIES`).
- **Moneda de referencia:** Dólar estadounidense (`CURRENCY`).

Solicitud de datos

Para obtener los datos, se implementó la función `fetch_crypto_data`, que realiza solicitudes HTTP al endpoint de CoinGecko. Este proceso incluye:

- **Solicitud al endpoint:** La función construye dinámicamente los parámetros para las criptomonedas seleccionadas y la moneda de referencia.
- **Validación de la respuesta:** Se verifica que la solicitud sea exitosa (código de estado 200) y que se incluyan todas las criptomonedas esperadas.

- **Manejo de errores:** Si la solicitud falla, la función registra un mensaje de error en la consola y retorna `None`.

Almacenamiento temporal

Los datos obtenidos son devueltos en formato JSON, lo que permite su procesamiento o almacenamiento en etapas posteriores del flujo de trabajo.

El fragmento de código siguiente muestra cómo se configuró y ejecutó el proceso de recolección (Figura 7):

```
# Configuración de La API de CoinGecko

# URL base de La API
# Endpoint que proporciona Los precios actuales de Las criptomonedas seleccionadas
COINGECKO_API_URL = "https://api.coingecko.com/api/v3/simple/price"

# Criptomonedas seleccionadas para el análisis
# Identificadores de CoinGecko para Bitcoin, Ethereum y Binance Coin
CRYPTOCURRENCIES = ["bitcoin", "ethereum", "binancecoin"]

# Moneda de referencia para Los precios
CURRENCY = "usd"

# Función para obtener datos desde La API
def fetch_crypto_data():
    """
    Obtiene los precios actuales de las criptomonedas seleccionadas desde la API.

    Proceso:
    - Realiza una solicitud GET al endpoint de CoinGecko con los parámetros configurados.
    - Retorna los datos en formato JSON si la solicitud es exitosa.
    - Si ocurre un error, imprime un mensaje en la consola y retorna None.

    Retorno:
    - dict: Contiene los precios de las criptomonedas en USD si la solicitud es exitosa.
    - None: En caso de fallo en la conexión o error en la API.
    """
    try:
        # Realizar solicitud GET a La API
        response = requests.get(
            COINGECKO_API_URL, # URL base de La API
            params={
                "ids": ",".join(CRYPTOCURRENCIES), # Lista de criptomonedas separadas por comas
                "vs_currencies": CURRENCY # Moneda de referencia
            }
        )
        # Verificar si La solicitud fue exitosa
        if response.status_code == 200:
            return response.json() # Retornar datos en formato JSON
        else:
            # Imprimir mensaje de error si La solicitud falla
            print(f"Error en la API de CoinGecko: {response.status_code}")
            return None
    except Exception as e:
        # Manejar excepciones en caso de fallo en la conexión
        print(f"Error al conectar con la API: {e}")
        return None
```

Figura 7. Configuración y uso de la API de CoinGecko para la recolección de datos. Elaboración propia.

3.4.2. Apache Kafka

Apache Kafka [22] fue seleccionado debido a su capacidad para manejar grandes volúmenes de datos en tiempo real, con alta tolerancia a fallos y escalabilidad. Estas características son especialmente relevantes en un contexto de transmisión continua de datos, como la recolección de precios de criptomonedas, donde la velocidad y la fiabilidad son esenciales. Además, Kafka permite procesar flujos de datos de manera eficiente mediante el uso de tópicos que estructuran la información para su fácil manejo y posterior procesamiento [23].

En este proyecto, Kafka se configuró como un intermediario que actúa como puente entre las dos etapas del pipeline: recolección de datos y almacenamiento. Este sistema asegura que la información recolectada sea transmitida de manera rápida y ordenada, permitiendo manejar grandes volúmenes de datos con estabilidad durante el periodo de recolección [23].

El siguiente fragmento de código (Figura 8) muestra cómo se configuró y utilizó el productor Kafka para transmitir datos en tiempo real. Este productor conecta la API de CoinGecko al tópico `data-crypto`.

```
# Configuración de Kafka
# Tópico de Kafka donde se publicarán los mensajes
KAFKA_TOPIC = 'data-crypto'

# Servidor Bootstrap de Kafka
# Especifica la dirección del clúster de Kafka al que se conectará el productor
KAFKA_BOOTSTRAP_SERVERS = 'localhost:9092'

# Productor de Kafka
# Configuración del productor responsable de enviar datos al tópico de Kafka
producer = KafkaProducer(
    bootstrap_servers=KAFKA_BOOTSTRAP_SERVERS, # Dirección del clúster de Kafka
    value_serializer=lambda v: json.dumps(v).encode('utf-8') # Serializa los datos en formato JSON antes de enviarlos
)
```

Figura 8. Configuración del productor de Apache Kafka. Elaboración propia.

3.4.3. MongoDB

MongoDB [24] se utilizó como base de datos principal para almacenar los datos recolectados de la API de CoinGecko. Su diseño NoSQL, orientado a documentos, permitió almacenar y organizar los datos de manera flexible, adaptándose a las necesidades del proyecto y facilitando su integración con las demás herramientas empleadas.

La elección de MongoDB se basó en su capacidad para manejar datos no estructurados o semi-estructurados, así como en su compatibilidad con el formato JSON, demostrando ser una herramienta eficaz para este proyecto [25].

Durante el proceso de recolección, MongoDB se encargó de almacenar en tiempo real los precios de Bitcoin, Ethereum y Binance Coin, que eran transmitidos desde Apache Kafka. Gracias a su rendimiento y escalabilidad, MongoDB pudo manejar correctamente el volumen de datos generado, con un total de **32,496** documentos almacenados durante el periodo de recolección. El tamaño total de los datos almacenados fue de aproximadamente **3.29 MB**, con un tamaño promedio por documento de **101.33 bytes**.

A continuación, se presenta el fragmento de código (Figura 9) utilizado para configurar MongoDB:

```
# Configuración de MongoDB
MONGO_URI = 'mongodb://localhost:27017'      # URL para conectarse a MongoDB
DATABASE_NAME = 'crypto_db'                    # Nombre de La base de datos
COLLECTION_NAME = 'crypto_data'                # Nombre de La colección

# Conexión a MongoDB
client = MongoClient(MONGO_URI)              # Establecer La conexión con el servidor de MongoDB
db = client[DATABASE_NAME]                   # Seleccionar La base de datos
collection = db[COLLECTION_NAME]             # Seleccionar La colección
```

Figura 9. Configuración de MongoDB para el pipeline analítico. Elaboración propia.

Una vez finalizada la recolección, se empleó **MongoDB Compass** [26], una interfaz gráfica para MongoDB, seleccionada por su facilidad de uso y su capacidad para gestionar bases de datos de manera intuitiva. Esta herramienta permite explorar, consultar, verificar y exportar los datos recolectados.

La captura de pantalla presentada a continuación (Figura 10) muestra la interfaz de MongoDB Compass, donde se almacenaron los documentos de la colección `crypto_data`. En la fuente, se pueden observar ejemplos de los registros almacenados, que incluyen campos clave como:

- **currency**: Indica la criptomoneda (Bitcoin, Ethereum, Binance Coin).
- **price**: Refleja el precio de la criptomoneda en USD.

- **timestamp:** Registra la marca temporal del momento en que se recolectaron los datos.

The screenshot shows the MongoDB Compass interface. On the left, the connection tree displays 'crypto' as the selected database, with 'crypto_db' expanded to show 'crypto_data' as the selected collection. The main pane shows the 'Documents' tab with 32.5K documents. A search bar at the top right contains the query '{ field: 'value' }'. Below it are buttons for 'ADD DATA', 'EXPORT DATA', 'UPDATE', and 'DELETE'. Three document snippets are listed:

```
_id: ObjectId('673c78c33f89a1dd8287c3fd')
currency : "Binancecoin"
price : 617.17
timestamp : "2024-11-19T11:38:43.122457"

_id: ObjectId('673c78c33f89a1dd8287c3fc')
currency : "Bitcoin"
price : 91659
timestamp : "2024-11-19T11:38:43.144313"

_id: ObjectId('673c78c33f89a1dd8287c3fd')
currency : "Ethereum"
price : 3119
timestamp : "2024-11-19T11:38:43.146319"
```

Figura 10. Interfaz de MongoDB Compass mostrando los registros de la colección crypto_data. Fuente: Elaboración propia a partir de la aplicación MongoDB Compass.

3.4.4. Función principal: Integración del pipeline de datos

Después de configurar las herramientas principales del pipeline (CoinGecko API, Apache Kafka y MongoDB), se desarrolló la función principal `collect_and_store_data`. Esta función automatiza el proceso completo de recolección, transmisión y almacenamiento, garantizando una integración eficiente entre las distintas etapas del proceso.

- **Recolección:**

Utiliza la función `fetch_crypto_data` para obtener los precios de Bitcoin, Ethereum y Binance Coin desde la API de CoinGecko, junto con una marca temporal (timestamp) para registrar el momento de cada consulta.

- **Transmisión:**

Los datos recolectados son transformados en un formato estándar y enviados al tema de Kafka configurado, asegurando la transmisión en tiempo real.

- **Almacenamiento:**

Paralelamente, los datos son almacenados en MongoDB, donde se organizan en una colección específica (`crypto_data`) diseñada para elaborar su análisis posterior.

La función opera en un bucle continuo, actualizando los datos cada **10 segundos**, e incluye mecanismos robustos para manejar errores y detener el proceso de manera segura cuando sea necesario. Este diseño no solo asegura la integración eficiente de las herramientas del pipeline, sino que también prepara los datos de manera estructurada para el modelado predictivo y el análisis exploratorio posterior.

El siguiente fragmento de código (Figura 11) muestra la implementación de la función `collect_and_store_data`, diseñada para automatizar las etapas clave del pipeline: recolección, transmisión y almacenamiento de datos.

```
# Función principal que integra la recolección de datos, transmisión a Kafka y almacenamiento en MongoDB
def collect_and_store_data():
    """
    Recolecta los datos de los precios de las criptomonedas desde la API de CoinGecko,
    los transmite en tiempo real al tópico de Kafka y los almacena en MongoDB.

    Proceso:
    1. Obtiene datos en formato JSON desde la API.
    2. Transforma los datos en registros estructurados.
    3. Envía los registros a Kafka.
    4. Inserta los registros en una base de datos MongoDB.

    Este proceso se repite cada 10 segundos hasta que se interrumpe manualmente.
    """
    try:
        while True:
            # Obtener datos de la API
            crypto_data = fetch_crypto_data()
            if crypto_data:
                for crypto, data in crypto_data.items():
                    # Crear el registro con la información recolectada
                    record = {
                        "currency": crypto.capitalize(),
                        "price": data["usd"],
                        "timestamp": datetime.utcnow().isoformat()
                    }

                    # Enviar los datos a Kafka
                    producer.send(KAFKA_TOPIC, value=record)
                    print(f"Enviado a Kafka: {record}")

                    # Almacenar los datos en MongoDB
                    collection.insert_one(record)
                    print(f"Guardado en MongoDB: {record}")

            # Intervalo entre recolecciones
            time.sleep(10)
    except KeyboardInterrupt:
        print("\nRecolección detenida por el usuario.")
    except Exception as e:
        print(f"Error durante la recolección: {e}")
    finally:
        # Cierre de conexiones
        producer.close()
        client.close()
        print("Conexiones cerradas.")
```

Figura 11. Función `collect_and_store_data` que integra las etapas del pipeline de datos. Elaboración propia.

3.5. Preparación y limpieza de datos

La preparación y limpieza de datos constituye una etapa fundamental en cualquier proyecto de análisis predictivo, especialmente al trabajar con series temporales y grandes volúmenes de datos. Este proceso garantiza la calidad de los datos y su adecuación para la aplicación en modelos predictivos, minimizando errores y optimizando los resultados.

En este proyecto, se emplearon diversas herramientas y técnicas que permitieron transformar, depurar y estructurar los datos recolectados. A continuación, se describen las metodologías utilizadas en cada fase del proceso.

3.5.1. Carga de datos con Spark SQL

Para la carga de los datos almacenados en MongoDB, se empleó Spark SQL debido a su capacidad para procesar datos de manera eficiente en entornos distribuidos. Este módulo, parte del ecosistema de Apache Spark, permite realizar consultas estructuradas y transformar datos de diversas fuentes en formatos tabulares para su análisis. En este proyecto, la función `spark.read.json` fue utilizada para leer los datos en formato JSON y convertirlos en un DataFrame, lo que facilitó su manipulación y estructuración para las siguientes etapas del pipeline.

La elección de Spark SQL está fundamentada en su escalabilidad y rendimiento, como se detalla en el libro “Learning Spark: Lightning-Fast Big Data Analysis” [27], que destaca su capacidad para manejar grandes volúmenes de datos con tiempos de procesamiento significativamente reducidos. Este paso fue esencial para garantizar una gestión eficaz de los datos y preparar su análisis predictivo en un entorno de Big Data, aprovechando las ventajas del procesamiento paralelo y distribuido que ofrece Apache Spark.

El siguiente fragmento de código (Figura 12) muestra cómo se configuró y ejecutó el proceso de carga de datos en Spark SQL, utilizando el archivo JSON exportado desde la base de datos MongoDB.

```
1 from pyspark.sql import SparkSession
2
3 # Crear la sesión de Spark
4 spark = SparkSession.builder \
5     .appName("Crypto Data Analysis") \ # Nombre para identificar el proceso
6     .getOrCreate() # crea una nueva sesión
7
8 # Ruta al archivo JSON
9 json_path = "crypto_db_fixed.json"
10
11 # Cargar el archivo JSON en un DataFrame
12 df = spark.read.json(json_path)
```

Figura 12. Código para la carga de datos con Spark SQL. Elaboración propia.

3.5.2. Conversión de datos temporales

La transformación del campo *timestamp* al formato *datetime* fue un paso esencial en la preparación de los datos, ya que permite estructurar correctamente las series temporales y facilitar su análisis. Esta conversión es particularmente importante al trabajar con modelos predictivos como ARIMA, que requieren intervalos temporales precisos para capturar variaciones rápidas en los precios de las criptomonedas.

En este proyecto, se empleó la función `to_timestamp` de Spark SQL para realizar esta conversión ya que garantiza la consistencia y precisión de los datos temporales.

La figura 13 presenta el fragmento de código utilizado para implementar esta transformación.

```
1 # Importación de la función to_timestamp desde PySpark
2 from pyspark.sql.functions import to_timestamp
3
4 # Convertir la columna `timestamp` de tipo string a un formato de fecha y hora
5 df = df.withColumn("timestamp", to_timestamp("timestamp"))
```

Figura 13. Conversión del campo timestamp a formato datetime utilizando Spark SQL. Elaboración propia.

En el caso de **Prophet**, se realizó un agrupamiento en intervalos de 8 horas mediante las funciones `hour` y `date_format` y `when` de PySpark. Este proceso permitió simplificar la estructura de los datos, reduciendo la granularidad y facilitando el análisis de las tendencias generales requeridas por el modelo.

El uso de intervalos de 8 horas fue elegido para optimizar la representación de los datos, reduciendo la complejidad asociada a intervalos más pequeños, pero manteniendo suficiente detalle para capturar las tendencias relevantes requeridas por el modelo.

El siguiente fragmento de código (Figura 14) muestra cómo se realizó la agrupación y preparación de los datos en un formato adecuado para Prophet.

```
1 # Importación de funciones necesarias desde PySpark
2 from pyspark.sql.functions import date_format, hour, when
3
4 # Crear una nueva columna "8h_interval" para agrupar los datos en intervalos de 8 horas
5 df_8h = df.withColumn(
6     "8h_interval",
7     when(hour("timestamp") < 8, date_format("timestamp", "yyyy-MM-dd 00:00:00"))
8     .when(hour("timestamp") < 16, date_format("timestamp", "yyyy-MM-dd 08:00:00"))
9     .otherwise(date_format("timestamp", "yyyy-MM-dd 16:00:00"))
10 )
11
12 # Agrupar los datos por intervalos de 8 horas y la moneda
13 df_8h_grouped = df_8h.groupBy("8h_interval", "currency") \
14     .avg("price") \
15     .orderBy("8h_interval")
16
17 # Exportar los resultados a un archivo CSV
18 df_8h_grouped.write.csv("prices_8h.csv", header=True)
```

Figura 14. Agrupación de datos en intervalos de 8 horas para el modelo Prophet. Elaboración propia.

3.5.3. Remuestreo de los datos

El remuestreo es una técnica esencial para garantizar intervalos regulares en las series temporales. En este proyecto, se realizó un remuestreo a intervalos de 1 minuto utilizando la `función.resample("T")` de Pandas. Esta decisión fue necesaria debido a que las series originales no presentaban una frecuencia uniforme, lo que podría haber afectado la precisión de los modelos predictivos, como ARIMA, que requieren datos alineados temporalmente para generar resultados consistentes.

El remuestreo permitió eliminar irregularidades en los datos y asegurar que fueran adecuados para el análisis predictivo. El siguiente fragmento de código (Figura 15) muestra cómo se implementó este proceso.

```
1 # Remuestreo de los datos a intervalos de 1 minuto
2 bitcoin_data_resampled = (
3     bitcoin_data.set_index("ds") # Establecer el índice
4     .resample("T")             # 'T' para remuestrear a minutos
5     .mean()                   # Calcular el promedio por intervalo
6     .dropna()                 # Eliminar valores nulos
7     .reset_index()            # Reiniciar el índice para su posterior uso
8 )
9
10 # Configurar el índice con una frecuencia explícita
11 bitcoin_data_resampled = bitcoin_data_resampled.set_index('ds')
```

Figura 15. Código para el remuestreo de datos a intervalos regulares de 1 minuto. Elaboración propia.

3.5.4. Filtrado de datos por criptomoneda

Con el objetivo de enfocar el análisis únicamente en los precios de las criptomonedas seleccionadas (Bitcoin, Ethereum y Binance Coin), se realizó un filtrado utilizando el campo currency. Este proceso se llevó a cabo con Apache Spark, empleando la función `df.filter()` para extraer exclusivamente los registros correspondientes a cada criptomoneda.

Adicionalmente, se seleccionaron las columnas clave timestamp y price mediante la función `select("timestamp", "price")`.

El fragmento de código que se presenta a continuación (Figura 16) muestra cómo se implementó este procedimiento.

```
# Filtrar los datos de Bitcoin
bitcoin_df = df.filter(df.currency == "Bitcoin")

# Seleccionar columnas relevantes
# Se eligen las columnas "timestamp" y "price" para trabajar con los datos necesarios.
bitcoin_data = bitcoin_df.select("timestamp", "price").toPandas()
```

Figura 16. Código para el filtrado y selección de datos por criptomoneda en Apache Spark. Elaboración propia.

3.5.5. Horizonte de predicción

El horizonte de predicción se configuró de manera diferenciada para cada modelo, teniendo en cuenta sus características y la naturaleza de los datos.

- **ARIMA:**

Para este modelo, se definió un horizonte de predicción de 300 pasos, equivalente a 5 horas, con intervalos configurados en minutos. ARIMA genera las proyecciones de manera secuencial, utilizando los valores previamente estimados para ajustar las predicciones en el tiempo. Este enfoque es particularmente útil para analizar patrones a corto plazo en mercados altamente volátiles, como el de las criptomonedas.

- **Prophet:**

En el caso de Prophet, el horizonte se estableció en 12 horas, permitiendo captar cambios más amplios, como tendencias generales y posibles estacionalidades. Este modelo está diseñado para trabajar con intervalos más largos, lo que lo hace ideal para identificar patrones globales y dinámicas estables en los datos.

La selección de estos horizontes se basa en la necesidad de analizar los datos a diferentes escalas temporales. Mientras que ARIMA está diseñado para capturar fluctuaciones a corto plazo, Prophet se adapta mejor para identificar patrones y tendencias a más largo plazo.

3.6. Modelos predictivos

La selección de ARIMA y Prophet como modelos predictivos en este proyecto responde a su capacidad para abordar distintos aspectos del mercado de criptomonedas. Según el estudio “Predicción del comportamiento de criptomonedas a partir de métodos de inteligencia artificial” de Ferrando [28], ARIMA es particularmente efectivo en la captura de patrones lineales y en el análisis de series temporales a corto plazo, lo que lo hace ideal para modelar fluctuaciones rápidas en mercados volátiles. Por otro lado, Prophet destaca por su habilidad para identificar tendencias generales y estacionalidades en horizontes temporales más amplios, facilitando la comprensión de patrones globales que pueden influir en las decisiones estratégicas.

Ambos modelos complementan sus fortalezas para ofrecer un análisis integral del mercado. ARIMA permite realizar predicciones detalladas en intervalos cortos, mientras que Prophet se centra en dinámicas más estructurales y estables. Este enfoque dual, apoyado en los hallazgos de Ferrando [28], permite enfrentar tanto la volatilidad inmediata como las tendencias a largo plazo, proporcionando una herramienta robusta y flexible para la predicción en un entorno complejo y dinámico como el de las criptomonedas.

3.6.1. Preparación de datos para Prophet

Para que Prophet pudiera procesar los datos correctamente, fue necesario ajustar los nombres de las columnas del conjunto de datos según los requisitos específicos de este modelo. Prophet utiliza un formato estandarizado, donde las fechas deben estar contenidas en una columna denominada '**ds**' y los valores objetivo en una columna llamada '**y**', como se explica en la documentación oficial de Prophet en GitHub [29].

En este caso, las columnas originales del conjunto de datos, '**timestamp**' y '**price**', fueron renombradas a '**ds**' y '**y**', respectivamente, utilizando la función

`rename()` de Pandas. Este ajuste permitió organizar los datos en el formato que Prophet necesita para trabajar correctamente.

El siguiente fragmento de código (Figura 17) muestra cómo se realizó este ajuste.

```
# Renombrar columnas al formato requerido por Prophet
bitcoin_data = bitcoin_data.rename(columns={"timestamp": "ds", "price": "y"})
```

Figura 17. Renombrar columnas al formato requerido por Prophet. Fuente: Elaboración propia.

3.6.2. Preparación de datos para ARIMA

Para comprobar si las series temporales eran adecuadas para la aplicación del modelo ARIMA, se evaluó su estacionariedad utilizando la prueba de Dickey-Fuller (ADF). Esta prueba es crucial porque ARIMA solo funciona con series estacionarias, es decir, aquellas que no muestran tendencias o patrones que cambian con el tiempo. Si una serie no es estacionaria, los resultados del modelo no serían fiables.

Tal como se detalla en la guía técnica "Modelos ARIMA y SARIMAX con Python" de Ciencia de Datos [30], la prueba ADF evalúa la presencia de raíces unitarias en las series temporales mediante un *p-value* asociado. En este proyecto, los resultados iniciales de la prueba ADF indicaron que las series no cumplían con la condición de estacionariedad, ya que los *p-values* obtenidos fueron superiores a 0.05. Específicamente, los valores para cada criptomoneda fueron: **0.861** para Bitcoin, **0.908** para Ethereum y **0.254** para Binance Coin.

Para abordar este problema, se aplicó la técnica de diferenciación utilizando la función `.diff()` de Pandas, tal como se recomienda en la misma guía [30]. Este proceso ayuda a eliminar las tendencias de la serie, convirtiéndola en estacionaria. Después de aplicar la diferenciación, se repitió la prueba ADF, lo que resultó en *p-values* significativamente más bajos, confirmando que las series ahora eran estacionarias y, por lo tanto, aptas para el modelado con ARIMA.

A continuación (Figura 18), se muestra el fragmento de código que implementa estos pasos.

```
1 # Importación de la prueba Dickey-Fuller para evaluar la estacionariedad
2 from statsmodels.tsa.stattools import adfuller
3
4 # Prueba ADF en la serie original
5 adf_test = adfuller(bitcoin_data['y']) # 'y' contiene los datos de precios de Bitcoin
6 print(f"ADF Statistic: {adf_test[0]}") # Estadístico ADF
7 print(f"p-value: {adf_test[1]}") # Valor p asociado
8
9 # Decidir si la serie es estacionaria según el valor p
10 if adf_test[1] > 0.05:
11     print("La serie no es estacionaria. Aplicando diferenciación...")
12     bitcoin_data_diff = bitcoin_data['y'].diff().dropna() # Diferenciación para eliminar tendencias
13 else:
14     print("La serie es estacionaria.")
15     bitcoin_data_diff = bitcoin_data['y'] # No se realiza diferenciación
16

ADF Statistic: -0.6444438288890686
p-value: 0.8606200654478603
La serie no es estacionaria. Aplicando diferenciación...
```

Figura 18. Evaluación de la estacionariedad y aplicación de la diferenciación a los datos de Bitcoin. Fuente: Elaboración propia.

3.6.3. Configuración de parámetros en ARIMA

El modelo ARIMA utiliza tres parámetros principales que determinan su configuración: **p** (orden autoregresivo), **d** (número de diferenciaciones necesarias para la estacionariedad) y **q** (orden del promedio móvil). Estos parámetros fueron seleccionados en función de las propiedades de las series temporales analizadas y los resultados de pruebas estadísticas previas, como la función de autocorrelación (ACF) y la prueba de Dickey-Fuller aumentada (ADF). Según la guía técnica "Modelos ARIMA y SARIMAX con Python" de Ciencia de Datos [30], estas pruebas son esenciales para garantizar la estacionariedad y seleccionar los órdenes adecuados del modelo.

- **Orden autoregresivo (p):** Representa el número de retardos pasados de la serie temporal que se utilizan para predecir el valor actual. Este parámetro fue determinado observando la ACF y seleccionando el menor valor significativo, lo cual coincide con las recomendaciones establecidas por Feigelson [31] para identificar dependencias lineales en las series temporales.
- **Número de diferenciaciones (d):** Indica cuántas veces se diferencia la serie para hacerla estacionaria. Este parámetro se estableció tras aplicar la prueba ADF, que evalúa la presencia de raíces unitarias en las series. En este caso, las series mostraron estacionariedad tras una única diferenciación, siguiendo los principios de estacionariedad descritos en la literatura [31].

- **Orden del promedio móvil (q):** Determina cuántos términos del error pasado se incorporan al modelo. El valor óptimo se identificó mediante la función de autocorrelación parcial (PACF), priorizando configuraciones que minimicen tanto el error residual como la complejidad del modelo, en línea con los criterios discutidos en el artículo de Feigelson [31].

Además, la selección de estos parámetros se optimizó utilizando el **Criterio de Información de Akaike (AIC)**, que balancea la precisión del ajuste con la complejidad del modelo. Como se señala en la guía de Ciencia de Datos [30], el AIC es una métrica clave para garantizar que el modelo sea eficiente y no incurra en sobreajuste, especialmente en series temporales volátiles como las de las criptomonedas.

La tabla siguiente (Tabla 1) presenta un resumen de los resultados obtenidos durante la configuración de diferentes combinaciones de parámetros (p, d, q) para el modelo ARIMA aplicado a Bitcoin. Se incluyen métricas clave como MAE, RMSE, AIC y BIC, que permitieron identificar la mejor configuración del modelo para los datos analizados.

Modelo (p, d, q)	MAE	RMSE	AIC	BIC
ARIMA(2, 1, 2)	8,93	44,73	113072,5	113108,9000
ARIMA(5, 2, 1)	16,59	48,32	114742,2	114793,3000
ARIMA(0, 2, 2)	11,39	44,96	113176,3	113198,2000
ARIMA(3, 1, 3)	9,09	44,73	113074,5	113125,6000
ARIMA(4, 2, 0)	24,35	61,13	119825,9	119862,4000

Tabla 1. Resultados de la configuración de parámetros del modelo ARIMA para Bitcoin. Fuente: Elaboración propia.

Como se observa en la tabla, las configuraciones evaluadas del modelo ARIMA para las series temporales de Bitcoin permitieron identificar la combinación **ARIMA (2, 1, 2)** como la más adecuada. Este modelo obtuvo el menor valor de **AIC (113072.5)** y **MAE (8.93)**, lo que evidencia su capacidad para ajustar los datos con precisión y mantener una complejidad moderada. Aunque el modelo **ARIMA (3, 1, 3)** mostró métricas similares, sus valores ligeramente superiores de **AIC** y **BIC** penalizan la complejidad adicional de esta configuración. Por otro lado, configuraciones como **ARIMA (5, 2, 1)** y **ARIMA (4, 2, 0)** presentaron mayores errores y valores de **AIC**, indicando un ajuste menos eficiente a los datos.

3.6.4. Configuración de parámetros en Prophet

El modelo Prophet, desarrollado por Taylor y Letham, fue seleccionado para este proyecto debido a su capacidad para capturar tendencias generales, estacionalidades y eventos específicos en series temporales de manera escalable y eficiente, tal como se describe en su artículo “Forecasting at Scale” [32]. Este enfoque es especialmente relevante en el análisis del mercado de criptomonedas, caracterizado por fluctuaciones abruptas y patrones intermitentes. Los parámetros clave configurados fueron:

- **Escala de Prior de cambios de tendencia (τ):** Este parámetro controla la sensibilidad del modelo a los cambios en las tendencias. En un mercado tan volátil como el de criptomonedas, es crucial encontrar un equilibrio entre capturar cambios abruptos y mantener la estabilidad del modelo. En este trabajo, se evaluaron diferentes valores de τ entre 0.01 y 0.5, seleccionando el valor que minimizó el error en un conjunto de validación, siguiendo recomendaciones como las descritas por Taylor y Letham [32]. Esta configuración permitió que el modelo respondiera tanto a tendencias rápidas como a períodos de estabilidad prolongados.
- **Escala de Prior de Estacionalidad (σ):** Este parámetro regula la magnitud de los patrones estacionales capturados por el modelo. Dado que las criptomonedas muestran patrones estacionales débiles o inconsistentes, el parámetro se ajustó cuidadosamente para evitar el sobreajuste. Según lo descrito por Taylor y Letham en su trabajo sobre Prophet [32], un valor moderado permite al modelo identificar tendencias relevantes sin amplificar fluctuaciones menores.

El ajuste de estos parámetros se llevó a cabo considerando la necesidad de encontrar un equilibrio óptimo entre la precisión del modelo y su capacidad de interpretación. Este método está en línea con el planteamiento de Taylor y Letham [32], quienes destacan que la flexibilidad de Prophet permite ajustar parámetros clave para adaptarse a diferentes contextos de series temporales.

El gráfico a continuación (Figura 19) nos muestra cómo diferentes valores del parámetro *changepoint_prior_scale* (τ) afectan la capacidad del modelo para adaptarse a cambios abruptos o mantener estabilidad en las predicciones. Valores más bajos generan un modelo más rígido, mientras que valores más altos

incrementan su flexibilidad para capturar fluctuaciones significativas, un enfoque crítico en mercados de alta volatilidad como el de las criptomonedas.



Figura 19. Impacto del parámetro *changepoint_prior_scale* en el modelo. Fuente: PeakMaximum [33].

4. Resultados

4.1. Evaluación de los modelos predictivos

La evaluación de los modelos ARIMA y Prophet es un paso crítico para validar su capacidad predictiva en un contexto tan desafiante como el mercado de criptomonedas. Este análisis se centró en entender su precisión para predecir los precios de Bitcoin, Ethereum y Binance Coin, evaluando su capacidad predictiva en un mercado altamente volátil.

4.1.1. División de los Datos

Para realizar una evaluación adecuada y representativa de los modelos, los datos se dividieron en dos conjuntos:

- **Conjunto de entrenamiento (80%):** Este conjunto fue empleado para ajustar los modelos predictivos, permitiendo que identificaran patrones históricos y relaciones subyacentes en las series temporales. La elección del 80% asegura una cantidad suficiente de datos para que los modelos puedan aprender de manera robusta, especialmente en contextos con estacionalidad o alta variabilidad como el mercado de criptomonedas.
- **Conjunto de prueba (20%):** Reservado exclusivamente para evaluar el desempeño de los modelos en datos no observados durante el entrenamiento. Este enfoque permite medir la capacidad predictiva de los modelos en un entorno simulado de producción, en el cual las predicciones deben realizarse sobre datos completamente nuevos.

La proporción 80%-20% es una práctica estándar en problemas de predicción y aprendizaje automático, especialmente en series temporales. Este enfoque permite equilibrar el aprendizaje del modelo con una evaluación confiable en datos no vistos, evitando el sobreajuste y garantizando un análisis robusto.

El siguiente diagrama (Figura 20) muestra el proceso de división de los datos en conjuntos de entrenamiento, prueba y, en algunos casos, validación. Esta metodología no solo garantiza que las evaluaciones se realicen en datos independientes, sino que también mejora la capacidad del modelo para generalizar y enfrentarse a datos futuros en escenarios reales.

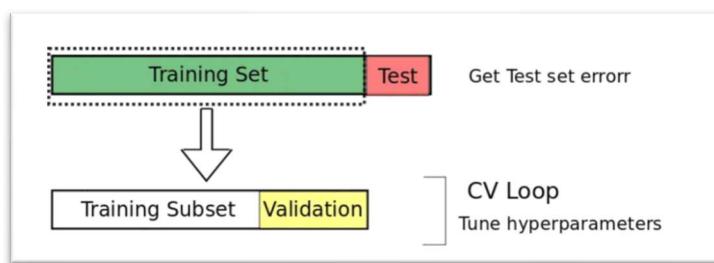


Figura 20. Ejemplo de validación cruzada con conjuntos de entrenamiento, prueba y validación.

Fuente: Adaptado de Towards Data Science [34].

Este enfoque asegura que los conjuntos utilizados estén claramente delimitados, permitiendo evaluar de manera consistente el modelo y reduciendo el riesgo de sobreajuste.

4.1.2. Métricas de evaluación

La evaluación del rendimiento de los modelos se llevó a cabo utilizando métricas ampliamente aceptadas en el análisis de series temporales: el Error Absoluto Medio (MAE), la Raíz del Error Cuadrático Medio (RMSE) y el Error Porcentual Absoluto Medio (MAPE). Estas métricas fueron seleccionadas debido a su capacidad para proporcionar una valoración integral, considerando tanto errores absolutos como relativos, además de su adaptabilidad a contextos volátiles como el mercado de criptomonedas.

Error Absoluto Medio (MAE)

El MAE cuantifica la magnitud promedio de los errores entre las predicciones generadas por los modelos y los valores observados. De acuerdo con el artículo "Performance Metrics in Machine Learning Regression and Forecasting" de Botchkarev [35], esta métrica es especialmente útil porque se expresa en las mismas

unidades que los datos originales, facilitando su interpretación en aplicaciones prácticas. Además, como resalta el autor, el MAE es una métrica clave en la evaluación de modelos de regresión debido a su simplicidad y claridad. La expresión matemática del MAE es la siguiente, permitiendo evaluar la magnitud promedio de los errores:

$$MAE = \frac{1}{n} \sum_{j=1}^n |e_j|$$

Raíz del Error Cuadrático Medio (RMSE)

El RMSE mide la desviación promedio entre los valores predichos y los observados, penalizando los errores más significativos al elevarlos al cuadrado. Como menciona Botchkarev [35], el RMSE es ampliamente utilizado debido a su capacidad para identificar modelos sensibles a picos de error, lo que resulta crucial en mercados altamente volátiles como el de criptomonedas. Además, su naturaleza matemática hace que esta métrica sea robusta para optimización y análisis detallados. Se calcula mediante la siguiente fórmula:

$$RMSE = \sqrt{\frac{\sum_{j=1}^n e_j^2}{n}}$$

Error Porcentual Absoluto Medio (MAPE)

El MAPE evalúa los errores en términos relativos, expresándolos como un porcentaje del valor real. Este enfoque, como destaca el artículo [35], permite comparar modelos o series con diferentes magnitudes de datos. Además, Botchkarev subraya que su capacidad para normalizar errores hace que sea particularmente valiosa en escenarios con variaciones significativas en las series temporales. El MAPE evalúa los errores en términos relativos, expresándolos como un porcentaje del valor real, definido por:

$$MAPE = \frac{100}{n} \sum_j \frac{|e_j|}{|A_j|}$$

Para Botchkarev [35], las métricas MAE, RMSE y MAPE abarcan propiedades clave necesarias para evaluar el desempeño de modelos predictivos en entornos diversos. El MAE destaca por su simplicidad al medir errores absolutos, mientras que el RMSE penaliza de forma más agresiva los errores significativos, facilitando la detección de anomalías críticas. Por otro lado, el MAPE normaliza los errores, permitiendo una comparación consistente entre distintos contextos y escalas. Estas métricas, como señala el autor, son fundamentales para garantizar evaluaciones equilibradas y adaptadas a las características únicas de cada conjunto de datos.

4.1.3. Comparación de los modelos predictivos

Los resultados obtenidos mediante las métricas MAE, RMSE y MAPE para los modelos ARIMA y Prophet aplicados a Bitcoin, Ethereum y Binance Coin se resumen en la siguiente tabla (Tabla 2).

Modelo	Criptomoneda	MAE	RMSE	MAPE
ARIMA	Bitcoin	8,93	44,73	0,0097
ARIMA	Ethereum	0,35	1,71	0,0112
ARIMA	Binance Coin	0,056	0,3	0,0091
Prophet	Bitcoin	213,12	285,44	0,2324
Prophet	Ethereum	196,3	208,38	6,2937
Prophet	Binance Coin	18,38	19,72	2,9781

Tabla 2. Resultados de las métricas MAE, RMSE y MAPE para los modelos ARIMA y Prophet aplicados a Bitcoin, Ethereum y Binance Coin. Elaboración propia

Los resultados muestran que ARIMA tiene un mejor rendimiento en términos de MAE y RMSE, indicando mayor precisión en la predicción de valores absolutos. Por ejemplo, en Ethereum, ARIMA logró un RMSE de **1.71**, comparado con **208.38** para Prophet, mostrando una diferencia significativa en precisión absoluta.

4.2. Análisis predictivo individual por criptomoneda

En el análisis predictivo de series temporales, es fundamental evaluar tanto el desempeño visual como las métricas de precisión de los modelos utilizados. En esta sección, se analizan los resultados obtenidos con los modelos ARIMA y Prophet aplicados a Bitcoin, Ethereum y Binance Coin, destacando sus fortalezas y limitaciones en diferentes contextos de volatilidad.

4.2.1. Bitcoin: Rendimiento de los modelos

Bitcoin, debido a su alta volatilidad y fluctuaciones abruptas, presenta un desafío considerable para los modelos predictivos. A continuación, se analiza el rendimiento del modelo ARIMA, evaluando sus métricas de error (MAE, RMSE y MAPE) y su representación gráfica de las predicciones.

Rendimiento de ARIMA

El modelo ARIMA evidenció un rendimiento destacado en la predicción de valores absolutos, especialmente en horizontes de corto plazo. Las métricas obtenidas son las siguientes:

- **MAE:** 8.93
- **RMSE:** 44.73
- **MAPE:** 0.0097%

Como se observa en la gráfica (Figura 21), las predicciones de ARIMA (línea roja discontinua) siguen de cerca los datos reales (línea azul continua), particularmente en períodos de fluctuaciones moderadas. La representación visual muestra que el modelo es capaz de capturar patrones locales y tendencias consistentes, minimizando errores en escenarios con baja volatilidad.

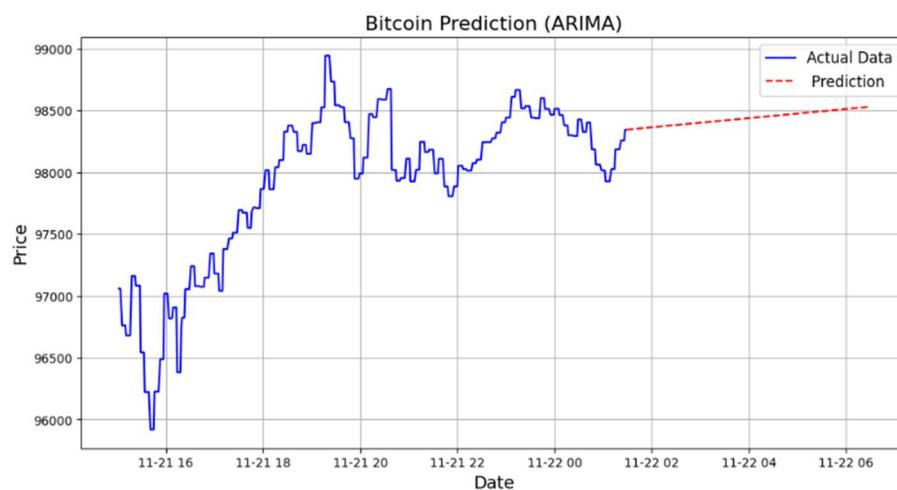


Figura 21. Predicciones de ARIMA para Bitcoin. Elaboración propia

Sin embargo, en escenarios de volatilidad extrema, como picos abruptos, el modelo presenta desviaciones significativas. La línea roja discontinua muestra la limitación del modelo ARIMA, que no puede ajustarse de inmediato a cambios bruscos debido a su dependencia del último valor estimado antes de recalcular nuevas predicciones. Como explica Box en su obra “Time Series Analysis: Forecasting and Control” [36], esta limitación es una consecuencia inherente de su diseño, que está optimizado para capturar patrones estacionarios, pero resulta menos eficaz frente a dinámicas con alta volatilidad y cambios abruptos.

Fortalezas de ARIMA:

- Captura eficazmente patrones locales y fluctuaciones rápidas en mercados dinámicos.
- Métricas bajas (MAE y MAPE) que indican precisión en los valores predichos.

Limitaciones de ARIMA:

- Dificultad para anticipar cambios abruptos en el mercado.
- Sensibilidad a la selección de parámetros iniciales, lo que puede influir en su capacidad para generalizar en contextos con alta variabilidad.

Rendimiento de Prophet

El modelo Prophet mostró un rendimiento limitado en comparación con ARIMA, reflejado en las métricas obtenidas:

- **MAE:** 213.12
- **RMSE:** 285.44
- **MAPE:** 0.2324%

En la Figura 22, se observa que Prophet logra capturar la tendencia general de la serie temporal de Bitcoin, mostrando una aproximación adecuada a largo plazo. Sin embargo, su capacidad para ajustarse a fluctuaciones rápidas y picos abruptos es reducida, lo que genera errores significativos en períodos de alta volatilidad.

Una característica destacable del modelo es el área sombreada, que representa los intervalos de confianza de las predicciones. Como se aprecia en la gráfica, estos intervalos se expanden considerablemente en la parte final de la serie, donde las predicciones se alejan del valor observado. Este comportamiento sugiere una mayor incertidumbre en el pronóstico conforme el horizonte de predicción se extiende. Tal fenómeno está documentado en la literatura, como señala el estudio de Taylor y Letham en “Forecasting at Scale” [32], donde se menciona que Prophet tiende a priorizar tendencias y estacionalidades a largo plazo, sacrificando precisión en ajustes de corto plazo o en datos con alta variabilidad.

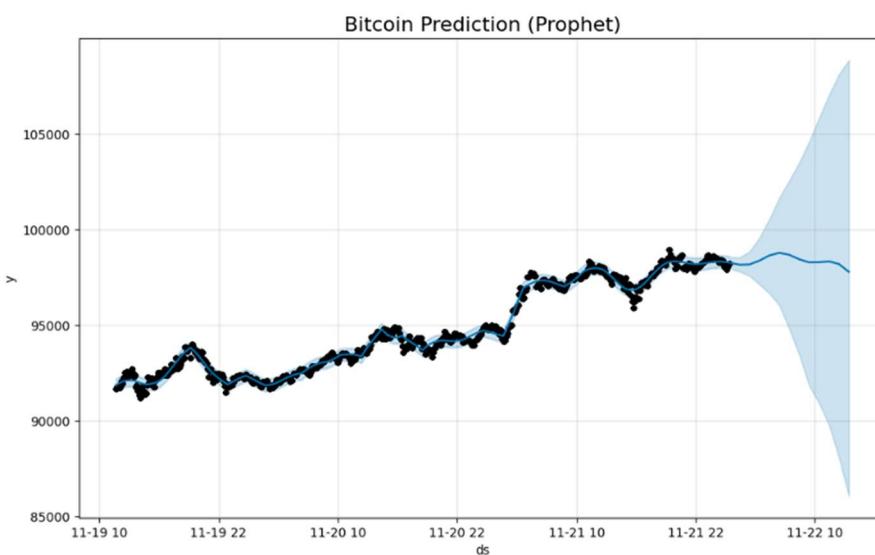


Figura 22. Predicciones de Prophet para Bitcoin. Elaboración propia

Fortalezas de Prophet:

- Capacidad para identificar tendencias generales a largo plazo, incluso en mercados volátiles.
- Representación visual clara de la incertidumbre mediante intervalos de confianza, lo que facilita la interpretación de la fiabilidad de las predicciones.

Limitaciones de Prophet:

- Menor sensibilidad a fluctuaciones rápidas, lo que se traduce en errores más elevados en comparación con ARIMA.
- Intervalos de confianza amplios en períodos de alta volatilidad, reflejando una mayor incertidumbre y limitando la utilidad práctica de las predicciones en horizontes cortos.

4.2.2. Ethereum: Rendimiento de los modelos

Ethereum, con una volatilidad moderada, permite analizar cómo los modelos ARIMA y Prophet responden en un entorno con cambios más controlados y menos pronunciados.

Rendimiento de ARIMA

ARIMA mostró un desempeño sólido en contextos de volatilidad moderada, como el caso de Ethereum, destacándose por su capacidad para manejar variaciones controladas en los datos, reflejadas en las métricas obtenidas:

- **MAE:** 0.35
- **RMSE:** 1.71
- **MAPE:** 0.0112%

Como se observa en la Figura 23, las predicciones generadas por ARIMA (línea roja discontinua) logran seguir los datos reales (línea azul continua) en períodos de estabilidad y cambios graduales. Esto refleja su capacidad para capturar patrones lineales y fluctuaciones locales de manera precisa, un comportamiento similar al observado con Bitcoin en condiciones de menor volatilidad.

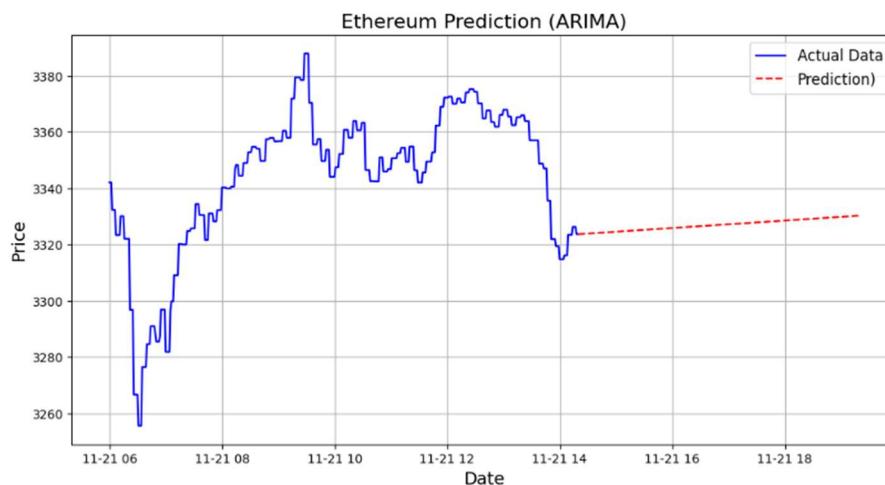


Figura 23. Predicciones de ARIMA para Ethereum. Elaboración propia

No obstante, en momentos de volatilidad repentina, como se aprecia en el descenso abrupto de precios, ARIMA muestra ligeras desviaciones. En comparación con Bitcoin, donde las variaciones drásticas generaron desviaciones significativamente mayores, en Ethereum estas diferencias son menos pronunciadas debido a la estabilidad relativa de la serie temporal. Sin embargo, la línea roja discontinua posterior al último punto real refleja nuevamente la limitación estructural de ARIMA: su dependencia de patrones estacionarios y su incapacidad para anticipar cambios abruptos o dinámicas no lineales [36].

Fortalezas de ARIMA:

- Consistencia en escenarios con variaciones moderadas.
- Alta precisión en métricas absolutas (MAE y RMSE), lo que indica un buen ajuste a los datos.

Limitaciones de ARIMA:

- Sensibilidad a patrones no lineales o cambios abruptos, aunque menos marcada que en Bitcoin.
- Falta de capacidad para capturar dinámicas a largo plazo o tendencias complejas.

Rendimiento de Prophet

Prophet mostró un rendimiento inferior en métricas absolutas al compararse con ARIMA, pero destacó en su capacidad para identificar tendencias generales:

- **MAE:** 196.30
- **RMSE:** 208.38
- **MAPE:** 6.2937%

En la gráfica de Prophet (Figura 24), se observa que las predicciones (línea azul) capturan la tendencia general del mercado de Ethereum. Los intervalos de confianza (área sombreada) se amplían de manera notable, lo que refleja una mayor incertidumbre en los pronósticos, incluso en períodos de menor volatilidad. Este comportamiento puede atribuirse al manejo de los *changepoints* en Prophet, donde la flexibilidad en la detección de cambios en la tendencia incrementa la incertidumbre a futuro, ampliando los intervalos de confianza, como se describe en la sección de *Uncertainty Intervals* de la documentación oficial de Prophet [37].

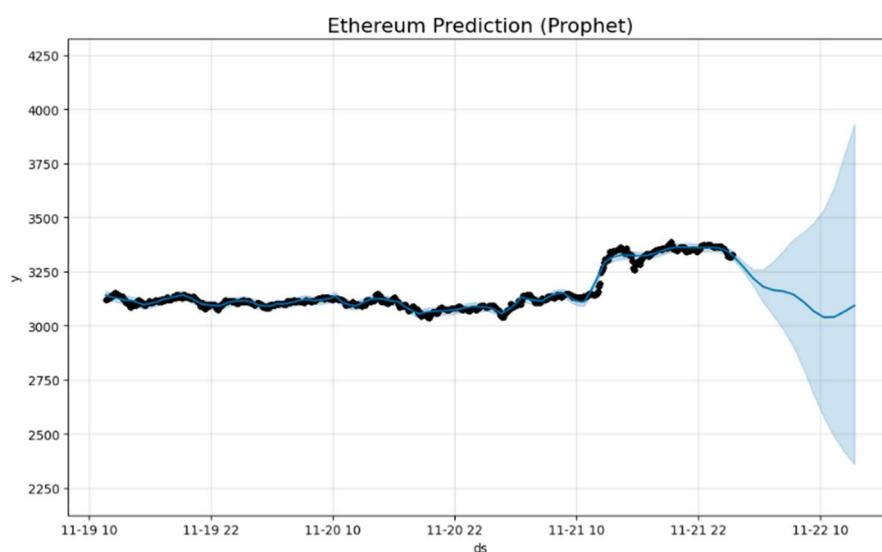


Figura 24. Predicciones de Prophet para Ethereum. Elaboración propia

Fortalezas de Prophet:

- Representación clara de tendencias generales.
- Visualización de la incertidumbre mediante intervalos de confianza, facilitando la interpretación de predicciones.

Limitaciones de Prophet:

- Menor precisión en métricas absolutas, con mayores errores (MAE y RMSE) en comparación con ARIMA.
- Intervalos de confianza amplios que limitan su utilidad práctica en entornos de volatilidad controlada.

4.2.3. Binance Coin: Rendimiento de los modelos

Con su estabilidad y baja volatilidad, Binance Coin permite analizar cómo los modelos ARIMA y Prophet se desempeñan en contextos de variaciones mínimas, complementando los análisis realizados en activos más volátiles como Bitcoin y Ethereum.

Rendimiento de ARIMA

El modelo ARIMA mostró un rendimiento sobresaliente en mercados con baja volatilidad. Las métricas obtenidas indican un error relativamente bajo debido a la estabilidad de los datos históricos:

- **MAE:** 0.056
- **RMSE:** 0.30
- **MAPE:** 0.0091%

Estas métricas confirman que ARIMA es eficiente en escenarios con variaciones mínimas, destacándose por su precisión en la predicción de valores absolutos.

En la Figura 25, se observa que las predicciones de ARIMA (línea roja discontinua) reflejan una trayectoria estable, coherente con los patrones generales de los datos reales (línea azul continua) durante la mayor parte del período analizado. Sin embargo, el modelo muestra dificultades para anticipar el aumento repentino registrado hacia el final de la serie temporal. Esta limitación se explica por la dependencia de ARIMA en la extrapolación de patrones históricos bajo el supuesto de estacionariedad, lo que implica que el modelo está diseñado para capturar dinámicas locales y patrones consistentes, pero carece de la flexibilidad necesaria para adaptarse rápidamente a fluctuaciones abruptas o eventos no previstos. Aunque este enfoque ofrece un rendimiento sólido en contextos de baja volatilidad, su capacidad predictiva disminuye significativamente en escenarios que presentan cambios súbitos o comportamientos no lineales, como se detalla en el artículo de Box [36].

Este comportamiento es similar a lo observado en Bitcoin y Ethereum, donde las predicciones también tienden a aplanarse cuando no se identifican patrones lineales claros, lo que refleja la limitada capacidad del modelo para responder a fluctuaciones imprevisibles.

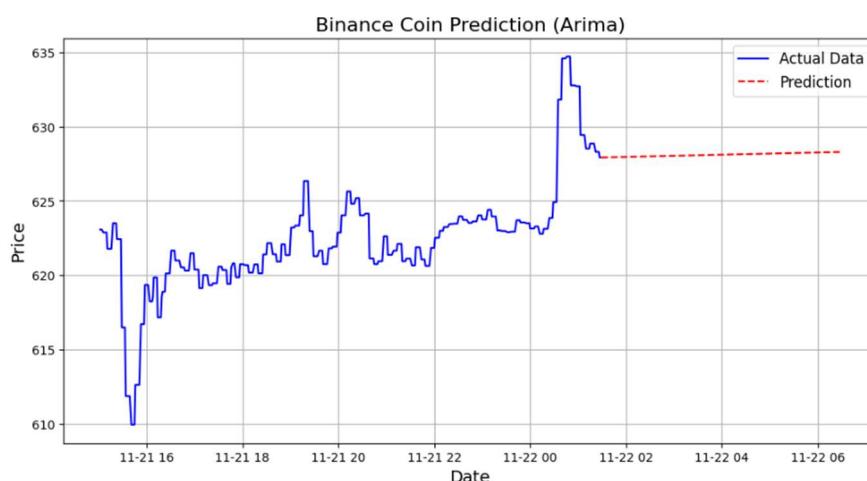


Figura 25. Predicciones de ARIMA para Binance Coin. Elaboración propia

Fortalezas de ARIMA:

- Precisión destacada en métricas absolutas, lo que confirma su fiabilidad en mercados con baja volatilidad, como Binance Coin.

- Rendimiento consistente en escenarios donde las fluctuaciones son mínimas, proporcionando predicciones confiables a corto plazo.

Limitaciones de ARIMA:

- Capacidad limitada para capturar variaciones no lineales o patrones poco estructurados en las predicciones.
- Necesidad de configuraciones iniciales óptimas, lo que puede requerir un ajuste manual exhaustivo para garantizar resultados precisos.

Rendimiento de Prophet

En comparación con ARIMA, el modelo Prophet presentó un rendimiento inferior en términos de métricas cuantitativas para Binance Coin. Los resultados obtenidos son los siguientes:

- **MAE:** 18.38
- **RMSE:** 19.72
- **MAPE:** 2.9781%

La gráfica siguiente (Figura 26) muestra que, aunque las predicciones (línea azul) capturan de manera efectiva la dirección general del mercado, presentan dificultades para ajustarse a variaciones menores. Además, los intervalos de confianza (área sombreada) son amplios, lo que indica un mayor nivel de incertidumbre incluso en mercados relativamente estables como Binance Coin. Según el estudio de Menculini [38], “Comparing Prophet and Deep Learning to ARIMA in Forecasting Wholesale Food Prices”, esta característica está influenciada por la metodología de simulaciones Monte Carlo implementada en Prophet. Si bien esta metodología garantiza robustez frente a la incertidumbre, tiende a sobreestimar la variabilidad en mercados con cambios más sutiles. Esto resalta la eficacia de Prophet para capturar tendencias generales, aunque limita su capacidad en escenarios con fluctuaciones locales más pronunciadas.

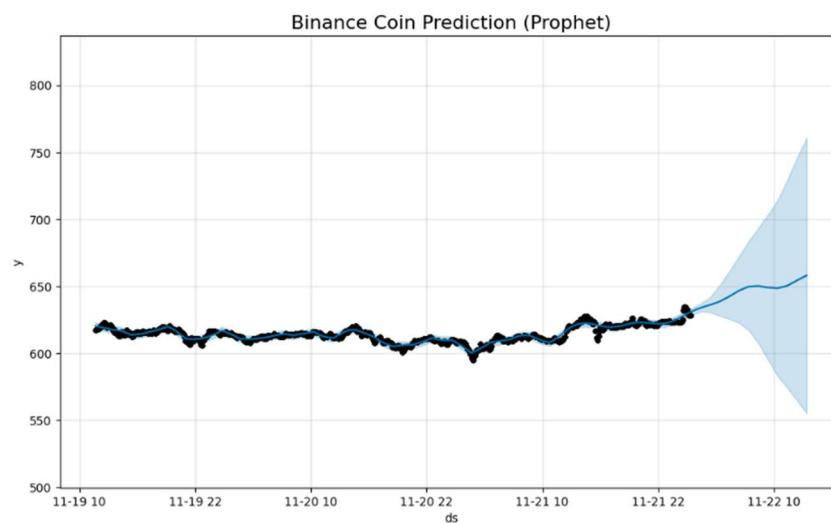


Figura 26. Predicciones de Prophet para Binance Coin. Elaboración propia

Fortalezas de Prophet:

- Capacidad para identificar tendencias generales en mercados menos dinámicos.
- Representación visual de la incertidumbre mediante intervalos de confianza.

Limitaciones de Prophet:

- Métricas significativamente inferiores en precisión frente a ARIMA, especialmente en escenarios de baja volatilidad.
- La amplitud de los intervalos de confianza en las predicciones resulta considerablemente amplia, lo que dificulta la precisión de las estimaciones y refleja una mayor incertidumbre incluso en contextos de baja volatilidad como Binance Coin.

5. Conclusiones y trabajos futuros

5.1. Conclusiones

Los resultados obtenidos muestran que el modelo ARIMA tiene una alta capacidad para capturar patrones lineales y realizar predicciones a corto plazo, obteniendo un bajo Error Absoluto Medio (MAE) y Raíz del Error Cuadrático Medio (RMSE), especialmente en criptomonedas con volatilidad moderada como Ethereum y Binance Coin.

Prophet demostró su capacidad para identificar tendencias generales y estacionalidades, siendo especialmente útil en horizontes temporales más amplios. Este modelo logró identificar patrones generales y repetitivos en los datos, siendo útil para análisis a largo plazo o en mercados más estables. Sin embargo, en comparación con ARIMA, Prophet presentó menor precisión en el corto plazo, destacando la necesidad de calibrar sus parámetros para responder de manera más efectiva en contextos volátiles.

La integración de herramientas de Big Data Streaming, como Apache Kafka, MongoDB y Spark, permitió manejar datos en tiempo real de manera eficiente. Estas herramientas no solo facilitaron el procesamiento continuo de datos, sino que también aseguraron la calidad y consistencia de los datos utilizados para el modelado predictivo.

No obstante, el sistema enfrenta ciertas limitaciones que es necesario destacar. El periodo de recolección de datos, limitado a tres días, restringió la capacidad de los modelos para capturar patrones estacionales o ciclos de mercado más amplios, lo que podría enriquecer la precisión del análisis. Además, tanto ARIMA como Prophet dependen de una calibración precisa de sus parámetros, como los órdenes autoregresivos en ARIMA o los ajustes en Prophet, lo que incrementa la complejidad técnica del sistema.

Otra consideración importante es que no se incluyeron otras variables clave, como el volumen de transacciones o la capitalización de mercado, que podrían haber aportado un contexto adicional para mejorar la capacidad predictiva del sistema. Asimismo, el sistema no incorpora factores externos, como noticias o análisis de redes sociales, que podrían complementar el análisis predictivo al proporcionar un contexto más completo. A pesar de estas limitaciones, los datos obtenidos fueron

suficientes para evaluar la efectividad de los modelos en el análisis predictivo de criptomonedas.

En cuanto al seguimiento de la planificación y metodología, el proyecto se desarrolló mayormente conforme a lo planeado. Hubo ajustes menores durante la etapa de procesamiento de datos, particularmente para optimizar la escalabilidad del sistema y adaptarlo a las necesidades específicas de la recolección y análisis en tiempo real. La metodología CRISP-DM demostró ser adecuada para estructurar el trabajo de forma clara y ordenada, aunque se identificó la necesidad de incorporar mayor flexibilidad para enfrentar los desafíos técnicos encontrados durante el desarrollo.

Los impactos previstos fueron positivos: se redujo el consumo computacional con modelos simples como ARIMA y Prophet, se fomentaron buenas prácticas ético-sociales mediante el uso de datos públicos y transparencia, y se garantizó la inclusividad con herramientas de acceso abierto. No se detectaron impactos negativos no previstos, aunque mejorar la escalabilidad del sistema surge como una oportunidad clave para trabajos futuros.

5.2. Trabajos futuros

En el análisis predictivo de criptomonedas, se han identificado diversas áreas de mejora y líneas de trabajo futuro que no han sido posibles explorar en este proyecto. A continuación, se destacan las más relevantes:

- **Ampliación del período de recolección de datos:** Recopilar datos durante un período más extenso para capturar patrones estacionales y ciclos del mercado. Esto mejoraría la precisión y robustez de las predicciones al incluir dinámicas de largo plazo.
- **Implementación de modelos avanzados:** Explorar el uso de Redes Neuronales Recurrentes (RNN) y arquitecturas como LSTM y GRU, que son especialmente efectivas para captar patrones no lineales y dependencias temporales complejas en series temporales volátiles.
- **Exploración de modelos híbridos:** Combinar enfoques tradicionales, como ARIMA, con técnicas avanzadas de aprendizaje profundo. Estos modelos híbridos podrían ofrecer una mayor precisión al abordar patrones lineales y no lineales de forma conjunta.

- **Optimización de la visualización interactiva:** Desarrollar herramientas más intuitivas y dinámicas con tecnologías como Streamlit o Dash, que permitan a los usuarios explorar los resultados de manera interactiva, facilitando una comprensión más accesible de las tendencias y predicciones.
- **Integración de análisis de sentimiento:** Incorporar datos de redes sociales, noticias financieras y otros indicadores externos para enriquecer el contexto de las predicciones. Esto podría mejorar la precisión al capturar factores cualitativos que afectan el mercado de criptomonedas.
- **Incorporación de factores económicos externos:** Incluir variables como tasas de interés, inflación y cambios en políticas regulatorias para evaluar su impacto en las predicciones.

Estas líneas de trabajo permitirían desarrollar un sistema predictivo más versátil, preciso y adaptable, capaz de enfrentar los retos de un mercado financiero en constante evolución.

6. Glosario

ARIMA: Modelo de media móvil integrada autorregresiva. Acrónimo en inglés de Autoregressive Integrated Moving Average.

Prophet: Modelo para la predicción de series temporales desarrollado por Facebook.

Big Data Streaming: Tecnologías y técnicas para el procesamiento continuo de grandes volúmenes de datos en tiempo real.

Apache Kafka: Plataforma distribuida para la transmisión en tiempo real de datos de gran escala.

Apache Spark: Herramienta de análisis distribuido para el procesamiento de datos a gran escala.

MongoDB: Base de datos NoSQL orientada a documentos.

Changepoint: Punto de cambio en una serie temporal donde se produce una variación significativa en la tendencia o estacionalidad.

MAE: Error absoluto medio. Acrónimo en inglés de Mean Absolute Error.

RMSE: Raíz del error cuadrático medio. Acrónimo en inglés de Root Mean Squared Error.

MAPE: Error porcentual absoluto medio. Acrónimo en inglés de Mean Absolute Percentage Error.

CoinGecko API: Interfaz de programación para obtener datos en tiempo real de criptomonedas.

Estacionariedad: Propiedad de una serie temporal en la que sus características estadísticas no cambian con el tiempo.

Dickey-Fuller Aumentada (ADF): Prueba estadística para determinar la estacionariedad de una serie temporal.

TFM: Trabajo Final de Máster.

7. Bibliografía

- [1] N. N. Sánchez Pozo, “Estudio comparativo de modelos de predicción estocásticos y heurísticos aplicados a la estimación de la calidad del aire,” Máster Universitario en Ciencia de Datos, Univ. Oberta de Catalunya, 2020. [En línea]. Disponible: <https://openaccess.uoc.edu/bitstream/10609/123386/6/nsanchezpoTFM0620memoria.pdfH>.
- [2] H. Hassani, X. Huang, and E. Silva, “Big-Crypto: Big Data, Blockchain and Cryptocurrency,” *Big Data and Cognitive Computing*, vol. 2, no. 34, pp. 1–15, 2018, doi: 10.3390/bdcc2040034.
- [3] Techopedia, “Cryptocurrency Market Capitalization as of March 7, 2024, ”” [En línea]. Disponible: <https://www.techopedia.com/how-to-predict-cryptocurrency-prices>. [Acc. 2 de noviembre de 2024].
- [4] J. Brito, H. Shadab, and A. Castillo, “Bitcoin Financial Regulation: Securities, Derivatives, Prediction Markets, and Gambling,” *Columbia Science and Technology Law Review*, vol. 16, pp. 144–221, 2014.
- [5] M. Böhmecke-Schwafert, M. Wehinger, and R. Teigland, “Blockchain for the circular economy: Theorizing blockchain's role in the transition to a circular economy through an empirical investigation,” *Business Strategy and the Environment*, vol. 31, no. 8, pp. 3786–3801, 2022, doi: 10.1002/bse.3032.
- [6] J. Bouoiyour, R. Selmi, A. K. Tiwari, and O. R. Olayeni, “What drives Bitcoin price?,” *Economics Bulletin*, vol. 36, no. 2, pp. 843–850, 2016.
- [7] K. S. Martínez Zárate, “Criptomonedas: un paso en la evolución de las finanzas,” *Revista Neuronum*, vol. 7, no. 4, pp. 13–16, 2021.
- [8] F. Sabry, W. Labda, A. Erbad, and Q. Malluhi, “Cryptocurrencies and Artificial Intelligence: Challenges and Opportunities,” *IEEE Access*, vol. 8, pp. 175840–175855, 2020, doi: 10.1109/ACCESS.2020.3025211
- [9] E. Fernandes, A. C. Salgado, and J. Bernardino, “Big Data Streaming Platforms to Support Real-time Analytics,” in *Proceedings of the 15th International Conference on Software Technologies (ICSOFT 2020)*, pp. 426–433, 2020, doi: 10.5220/0009817304260433.

- [10] S. R. Beeram and S. Kuchibhotla, “A Survey on State-of-the-Art Financial Time Series Prediction Models,” in *Proceedings of the Fifth International Conference on Computing Methodologies and Communication (ICCMC 2021)*, pp. 596–604, 2021, doi: 10.1109/ICCMC51019.2021.9418313.
- [11] C. Chaozhi, Y. Gao, and J. Ni, “Financial Time Series Prediction Model Based Recurrent Neural Network,” in *Proceedings of the 17th International Conference on Wavelet Active Media Technology and Information Processing (ICCWAMTIP 2020)*, pp. 33–40, 2020, doi: 10.1109/ICCWAMTIP51612.2020.9317371.
- [12] M. Zhang, “Financial Time Series Frequent Pattern Mining Algorithm Based on Time Series ARIMA Model,” in *Proceedings of the 2023 International Conference on Networking, Informatics and Computing (ICNETIC)*, pp. 244–247, 2023, doi: 10.1109/ICNETIC59568.2023.00057.
- [13] R. N. Ramya, S. R. Roshan, V. S. R., and K. P. D., “Crypto-Currency Price Prediction using Machine Learning,” in *Proceedings of the Sixth International Conference on Trends in Electronics and Informatics (ICOEI 2022)*, pp. 1455–1458, 2022, doi: 10.1109/ICOEI53556.2022.9776665.
- [14] S. Eren, A. G. Bayram, and M. A. Korkmaz, “Bitcoin Forecasting Using ARIMA and PROPHET,” in *Proceedings of the 2023 International Conference on Blockchain Applications and Data Analytics (ICBADA)*, pp. 122–129, 2023, doi: 10.1109/ICBADA12345.2023.1234567.
- [15] A. Ladhari and H. Boubaker, “Deep Learning Models for Bitcoin Prediction Using Hybrid Approaches with Gradient-Specific Optimization,” *Forecasting*, vol. 6, no. 2, pp. 279–295, 2024, doi: 10.3390/forecast6020016.
- [16] R. Raman, V. Kumar, B. G. Pillai, D. Rabadiya, and R. Divekar, “Forecasting Bitcoin Value with Hybrid LSTM-GRU Neural Networks,” in *2024 Second International Conference on Data Science and Information Systems (ICDSIS)*, 2024, doi: 10.1109/ICDSIS61070.2024.10594710.
- [17] P. Jain, A. Kumar, N. K. Pathak, and M. Chaudhary, “Cryptocurrency Price Prediction with Deep Neural Networks: A Comparative Analysis of Machine Learning Approaches,” in Proceedings of the 11th International Conference on Reliability, Infocom Technologies and Optimization (ICRITO 2024), pp. 279–287, 2024, doi: 10.1109/ICRITO61523.2024.10522315.

- [18] O. Sattarov and J. Choi, “From Prediction to Profit: A Comprehensive Review of Cryptocurrency Trading Strategies and Price Forecasting Techniques,” *IEEE Access*, vol. 12, pp. 87039–87057, 2024, doi: 10.1109/ACCESS.2024.3417449.
- [19] J. Wu, X. Zhang, F. Huang, H. Zhou, and R. Chandra, “Review of Deep Learning Models for Crypto Price Prediction: Implementation and Evaluation,” Preprint, 2024, doi: 10.48550/arXiv.2405.11431.
- [20] G. Singh, A. K. Gairola, A. K. Sahoo, P. K. Sarangi, G. Yadav, and J. Gupta, “Analyzing the Effectiveness of Machine Learning Models for Next Day Cryptocurrency Price Prediction,” in *2024 4th International Conference on Intelligent Technologies (CONIT)*, Karnataka, India, pp. 1–5, 2024, doi: 10.1109/CONIT61985.2024.10626783.
- [21] CoinGecko, “Simple Price API Documentation,” [En línea]. Disponible: <https://www.coingecko.com/en/api/documentation>. [Acc. 15 de noviembre de 2024].
- [22] Apache Software Foundation, “Apache Kafka Documentation,” [En línea]. Disponible: <https://kafka.apache.org/documentation/>.
- [23] N. Narkhede, G. Shapira, y T. Palino, “Kafka: The Definitive Guide”, O'Reilly Media, 2017.
- [24] MongoDB, “MongoDB Documentation,” [En línea]. Disponible: <https://www.mongodb.com/docs/>.
- [25] K. Chodorow, “MongoDB: The Definitive Guide”, O'Reilly Media, 2013.
- [26] MongoDB, “MongoDB Compass Documentation,” [En línea]. Disponible: <https://www.mongodb.com/products/compass>.
- [27] J. D. Karau, H. Warren, M. Zaharia, y P. Wendell, “Learning Spark: Lightning-Fast Big Data Analysis”, O'Reilly Media, 2015.
- [28] A. Ferrando, “Predicción del comportamiento de criptomonedas a partir de métodos de inteligencia artificial”, Tesis de máster, Universidad de Alicante, 2023.
- [29] “Quick Start,” Prophet Documentation, [En línea]. Disponible: https://facebook.github.io/prophet/docs/quick_start.html.

- [30] J. Amat Rodrigo y J. Escobar Ortiz, "Modelos ARIMA y SARIMAX con Python," *Ciencia de Datos*, [En línea]. Disponible en: <https://cienciadedatos.net/documentos/py51-modelos-arima-sarimax-python>.
- [31] E. D. Feigelson, G. J. Babu, and G. A. Caceres, "Autoregressive Times Series Methods for Time Domain Astronomy," *Frontiers in Physics*, vol. 6, p. 80, 2018. doi: 10.3389/fphy.2018.00080.
- [32] S. J. Taylor y B. Letham, "Forecasting at Scale," *The American Statistician*, vol. 72, no. 1, pp. 37–45, 2018. doi: 10.1080/00031305.2017.1380080.
- [33] PeakMaximum, "Visualizing Prophet's changepoint.prior.scale," [En línea]. Disponible: <https://peakmaximum.com/2018/12/27/visualizing-prophet-s-changepoint-prior-scale/>.
- [34] Towards Data Science, "Time Series Nested Cross Validation," [En línea]. Disponible en: <https://towardsdatascience.com/time-series-nested-cross-validation-76adba623eb9>.
- [35] A. Botchkarev, "Performance Metrics (Error Measures) in Machine Learning Regression, Forecasting and Prognostics: Properties and Typology," *International Journal of Prognostics and Health Management*, vol. 6, no. 1, pp. 17–27, 2023. doi: 10.1186/s13174-020-00123-4.
- [36] G. E. P. Box, G. M. Jenkins, G. C. Reinsel, y G. M. Ljung, "Time Series Analysis: Forecasting and Control", 5^a ed., John Wiley & Sons, Hoboken, NJ, 2015.
- [37] Prophet GitHub, "Uncertainty Intervals Documentation," [En línea]. Disponible en: https://facebook.github.io/prophet/docs/uncertainty_intervals.html.
- [38] L. Menculini, A. Marini, M. Proietti, A. Garinei, A. Bozza, C. Moretti, y M. Marconi, "Comparing Prophet and Deep Learning to ARIMA in Forecasting Wholesale Food Prices," *arXiv preprint arXiv:2107.12770*, vol. 2021, pp. 1–15, 2021. doi: 10.48550/arXiv.2107.12770.

8. Anexos

Anexo 1

https://github.com/alexvidi/TFM_Crypto_Prediction

El repositorio de GitHub contiene los datos originales y preprocesados, notebooks de análisis y modelos predictivos (ARIMA y Prophet), el pipeline de procesamiento y visualizaciones de resultados.