Escola d'Enginyeria de Telecomunicació i Aeroespacial de Castelldefels
UNIVERSITAT POLITÈCNICA DE CATALUNYA

TRABAJO FINAL DE GRADO

Título del TFG: Modelos de predicción del precio de Bitcoin mediante algoritmos de Machine Learning

Titulación: Grado en Ingeniería Telemática

Autor: Eric Jimenez Bujalance

Directora: Olga León Abarca

Fecha: 21 de octubre de 2022

Resumen

Bitcoin supone el origen del mercado de las criptomonedas en el año 2008, momento en el cuál se presenta un documento técnico bajo el pseudónimo de “Satoshi Nakamoto”, donde se expone el concepto, las características y la visión de esta criptomoneda.

Actualmente, Bitcoin es la criptomoneda líder en capitalización de mercado y la más popular. Por este motivo, a partir de 2020 la sociedad se interesó notablemente en comprender su funcionamiento y poder beneficiarse económicamente a partir de la inversión. Para ello, se usan principalmente dos metodologías: el análisis técnico o manual de mercado y el uso de Machine Learning.

El análisis técnico, permite a un ser humano analizar un gráfico de un activo en concreto, a partir del uso de un conjunto de herramientas que permiten predecir el siguiente movimiento y producir beneficios económicos. Aún así, es una metodología que conlleva una cantidad de tiempo considerable al necesitar mucha formación y tiempo para adquirir la experiencia necesaria y obtener rendimientos económicos de ello.

El uso de Machine Learning, permite crear modelos que realicen predicciones del precio de un activo de forma automatizada pudiendo generar beneficios económicos potenciales. Esta metodología, permite complementar y mejorar las predicciones realizadas por una persona que utiliza el análisis técnico o manual para determinar el siguiente movimiento de cualquier activo, ya que el Machine Learning permite procesar y tratar cantidades de información que un ser humano no es capaz de retener.

Por lo tanto, el objetivo principal de este trabajo de final de grado es crear modelos que permitan predecir el precio de Bitcoin mediante el uso de algoritmos automáticos de Machine Learning a partir de un archivo de datos que contiene atributos relacionados con el precio de la criptomoneda, un Data Set.

En este trabajo se han escogido los algoritmos Random Forest, Regresor Lineal y Series temporales (*Forecasting*) para crear los modelos de predicción.

Se han evaluado los diferentes modelos creados a partir de los errores en regresión y se han optimizado mediante el uso de las técnicas de Out of Bag, Cross Validation y Grid Search. Finalmente, se ha llevado a cabo la predicción del precio de Bitcoin con cada uno de los modelos entre los cuales destaca el de Regresión lineal, al obtener los mejores resultados de predicción respecto al precio real.

Title: Prediction models of Bitcoin price using Machine Learning

Titulación: Bachelor's degree in Telematics Engineering

Author: Eric Jimenez Bujalance

Director: Olga León Abarca

Date: October 21, 2022

Overview

Bitcoin was the origin of the cryptocurrency market in 2008, when a technical document was presented under the pseudonym "Satoshi Nakamoto", in which the concept, characteristics and vision of this cryptocurrency were set out.

Bitcoin is currently the leading and most popular cryptocurrency in terms of market capitalisation. For this reason, since 2020, society has become very interested in understanding how it works and being able to benefit economically from investment. To do this, two methodologies are mainly used: technical or manual market analysis and Machine Learning algorithms.

Technical analysis allows a human being to analyse a graph of a specific asset, based on the use of a set of tools that allow the prediction of the next movement and produce economic benefits. Even so, it is a methodology that involves a considerable amount of time as it requires a lot of training and time to acquire the necessary experience and obtain economic returns from it.

The use of Machine Learning allows the creation of models that make predictions of the price of an asset in an automated way. This methodology complements and improves the predictions made by a person who uses technical or manual analysis to determine the next movement of any asset, as Machine Learning can process and treat amounts of information that a human being is not capable of retaining.

Therefore, the main objective of this final degree work is to create models that allow predicting the price of Bitcoin by using automatic Machine Learning algorithms from a data file containing attributes related to the price of the cryptocurrency, a Data Set.

With that purpose, the set of algorithms that have been chosen are Random Forest, Linear Regression and Time Series (Forecasting).

The prediction models obtained by means of the previously mentioned algorithms have been evaluated in terms of regression errors and techniques such as Out of Bag, Cross Validation and Grid Search techniques have been applied in order to optimise the models. Finally, Bitcoin price prediction has

been carried out with each of the models, among which the Linear Regression model stands out, as it obtained the best prediction results compared to the real price.

A mi familia,
por apoyarme y ayudarme siempre en todo lo que he necesitado en mi vida

A los West,
por ser unos amigos increíbles y haberse convertido en mi família

A mi pareja,
por estar a mi lado todos estos años y apoyarme en todas las decisiones

A Olga,
por el apoyo, la paciencia y los consejos para realizar mi proyecto final

LISTA DE ACRÓNIMOS Y ABREVIATURAS

API	Application Programming Interfaces
BTC	Bitcoin
BTC/USD	Bitcoin/United States Dollar (Dólar estadounidense)
BTC/USDT	Bitcoin/Tether
CPU	Central Processing Unit
COVID-19	Coronavirus Disease 2019
LightGBM	Light Gradient Boosting Machine
MAE	Mean Absolute Error
P2P	Peer To Peer
RMSE	Root Mean Squared Error
SVM	Support Vector Machines
USDT	Tether
USD	United States Dollar (Dólar estadounidense)

ÍNDICE

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

CAPÍTULO 1. INTRODUCCIÓN

El origen de las criptomonedas se remonta a la crisis financiera de 2008, año en que se presentó el 1 de noviembre un documento técnico que revolucionó la visión del dinero, mostrando un nuevo sistema de efectivo electrónico llamado “Bitcoin” por parte de una identidad desconocida cuyo seudónimo fue “Satoshi Nakamoto” [1].

En 2009, Bitcoin salió a la luz oficialmente mediante el primer bloque que contenía 50 Bitcoins al que se le denominó “Génesis”. A partir de este momento, se convierte en una criptomoneda basada en el paradigma P2P *(Peer-to-Peer)* con la visión largoplacista de convertirse en la moneda digital pionera de una economía descentralizada.

La descentralización de la economía por parte de Bitcoin, persigue una revolución en el sistema económico actual en el cual las entidades financieras tradicionales, como los bancos, dejen de tener el control total sobre el capital de los usuarios. Su meta principal es otorgar el control de la economía de cada individuo a los propios usuarios sin tener que recurrir a terceros (bancos) para almacenar su capital o realizar transacciones. Esto es posible, ya que Bitcoin trabaja con un sistema basado en P2P *(Peer-to-Peer)*.

Un sistema P2P *(Peer-to-Peer)* es una red en la que un grupo de personas o máquinas participan de forma completamente descentralizada. Todas las partes actúan de forma autónoma respondiendo a un protocolo de comunicaciones y consenso común. De esta forma, los integrantes de la red pueden intercambiar información de forma directa y sin intermediarios [2].
Bitcoin proporciona anonimato, es decir, permite ocultar la identidad real del usuario, gracias al uso de este sistema y de la criptografía.

La gran evolución tecnológica desde su creación, ha permitido que sea posible realizar pagos en algunos establecimientos o locales utilizando tarjetas de casas de cambio de criptomonedas (*exchanges*). Adicionalmente, existen cajeros automáticos de criptomonedas que permiten al usuario cambiar sus Bitcoins a moneda local y recibir dinero en efectivo.

En los últimos años, el interés por el conocimiento y la posibilidad de invertir en el mercado de las criptomonedas y en concreto en el Bitcoin, se ha incrementado de forma exponencial, lo que ha supuesto un crecimiento extraordinario de la cantidad de información disponible sobre él.

Uno de los momentos clave fue el año 2020, en concreto, durante la pandemia de la COVID-19, cuando su precio se incrementó de aproximadamente 5200 $ (al declararse el estado de alarma el día 14 de marzo de 2020) hasta 29000 $ el 31 de diciembre del mismo año, creando su máximo histórico anual [3].

Un año después, la inversión en este cripto activo se incrementó considerablemente respecto 2020, tanto por parte de usuarios con poco capital como de grandes inversores, empresarios y corporaciones como Tesla. El

creador de esta última, Elon Musk, fue el gran protagonista durante el año 2021 al provocar periodos de volatilidad en esta criptomoneda mediante comentarios en redes sociales [4].

Actualmente, el Bitcoin se encuentra en un periodo denominado bajista, ya que el precio está realizando un retroceso respecto a la tendencia alcista que llevaba desde el inicio de la pandemia de la COVID-19. El 10 de noviembre de 2021, creó su máximo histórico registrado hasta la fecha en los 69100 $ aproximadamente y desde ese instante, el precio ha ido cayendo hasta el rango de los 20000 $ y 23000 $ en el que se encuentra en la actualidad.

Durante estos 3 últimos años, la sociedad se ha interesado por comprender la finalidad de la creación de Bitcoin, al observar la magnitud de su crecimiento y las posibilidades de beneficiarse de ello. Aun así, no es tarea sencilla, ya que requiere de formación y conocimientos muy sofisticados de análisis técnico de mercados que permitan a una persona ser capaz de analizar manualmente los movimientos anteriores y actuales para predecir movimientos futuros y sacar partido de ello.

Gracias a la evolución de la tecnología y a la aparición del Machine Learning y la Inteligencia Artificial [5], es posible utilizar algoritmos para crear modelos que permitan predecir el precio del Bitcoin y poder crear sistemas automáticos sofisticados que generen beneficios económicos respecto a este cripto activo.

La aplicación del Machine Learning en la predicción del precio del Bitcoin, puede ayudar a mejorar las predicciones que realiza un ser humano que utiliza como herramienta el análisis técnico o análisis manual aplicado a este cripto activo, ya que la tecnología puede procesar y tratar mayores cantidades de información.

Por todo lo comentado anteriormente, el objetivo de este trabajo es demostrar que mediante los algoritmos de Machine Learning es posible crear modelos para predecir el precio del Bitcoin. Para ello, se ha generado un fichero con extensión .CSV que contiene atributos relacionados con el precio del Bitcoin, de ahora en adelante, Data Set.
Para evaluar los distintos algoritmos, se ha usado el lenguaje de programación Python, puesto que tiene una curva de aprendizaje baja y una gran cantidad de librerías que facilitan el uso y el desarrollo de la programación con Machine Learning [6]. Por este motivo, todo el desarrollo de programación se ha llevado a cabo en Google Colaboratory [7], herramienta en la nube que permite a cualquier usuario escribir y ejecutar código arbitrario de Python a través del navegador. Es especialmente adecuado para tareas de aprendizaje automático y análisis de datos.

El documento se estructura en 5 capítulos (incluido el actual), tal y como se describe a continuación.

En el segundo capítulo, se introduce el concepto de criptomoneda y las características más importantes del Bitcoin.

En el tercer capítulo, se explican las dos metodologías posibles de análisis de mercado y se desarrolla detalladamente el concepto de Machine Learning y los algoritmos asociados que se han utilizado en el presente trabajo.

En el cuarto capítulo, se presenta el procedimiento seguido para crear el Data Set de Bitcoin, tratar los datos y crear modelos de predicción a partir de algoritmos de Machine Learning, variando diferentes parámetros para encontrar los valores óptimos. Asimismo, se evalúan los diferentes modelos en función del error de predicción y del tiempo de ejecución.

Finalmente, en el quinto capítulo se exponen las conclusiones del trabajo realizado.

CAPÍTULO 2. BITCOIN

2.1. Bitcoin

El Bitcoin [1], también conocido por las siglas de “BTC”, representa el origen de las criptomonedas al ser la primera moneda digital descentralizada. Es una moneda virtual que puede ser enviada a través de Internet.

El ecosistema de Bitcoin se basa en la tecnología blockchain (*cadena de bloques*) y en una red P2P con nodos interconectados entre sí, los cuales se encargan de recibir y procesar toda la información de las transacciones realizadas dentro de la red. Es una red totalmente pública a la que puede unirse cualquier persona.

La blockchain de Bitcoin, está formada por un conjunto (o cadena) de bloques unidos entre sí. Utiliza el sistema de Proof-Of-Work (*PoW*) a partir del cual cada uno de los nodos de la red, busca soluciones a problemas matemáticos que permiten introducir o generar un nuevo bloque en la blockchain. Este proceso se conoce como minería y los nodos que intentan solucionar los problemas matemáticos para introducir los bloques en la blockchain son los nodos mineros. Los mineros reciben recompensas en Bitcoin y de esta forma, se crea cada nuevo Bitcoin [8].

En definitiva, la labor de los mineros es de gran importancia, ya que son los encargados de introducir los bloques de transacciones en la blockchain a cambio de recompensas en Bitcoin y así mantener el funcionamiento del ecosistema.

2.2. Características de Bitcoin

Las principales características de Bitcoin son las siguientes:

- **Descentralización:** El Bitcoin no está controlado por ninguna entidad tradicional. Está formado por la tecnología blockchain y un sistema P2P (Peer-to-Peer) donde todos los usuarios mantienen el ecosistema.

- **Transparencia:** Posibilidad de consultar en todo momento las transacciones de la red de Bitcoin al ser pública [9].

- **Privacidad:** Cada usuario se identifica con una billetera que se asocia con una cadena de 25 a 34 dígitos en la que nunca aparecerá ningún dato sobre el propio usuario, por lo que nadie sabe quién hay detrás de cada billetera, a no ser que el usuario lo exponga públicamente.

- **Transacciones rápidas:** Cuando un usuario envía una cantidad de Bitcoin a otro, suele tardar unos pocos minutos en llegar a la billetera del usuario destino.

- **Comisiones bajas:** A cada transacción se le añade una comisión por utilización de los recursos de la red, normalmente pequeña.

- **Seguridad:** La blockchain es una tecnología altamente segura en la que es muy difícil provocar anomalías gracias a la utilización de las tecnologías de ciberseguridad.

- **Uso real:** Gracias a la evolución de la tecnología, actualmente puede utilizarse como medio de pago. Otras criptomonedas se utilizan para desarrollar proyectos tecnológicos innovadores que mejoren distintos ámbitos de la sociedad.

- **Volatilidad y riesgo:** El mercado de las criptomonedas es muy joven (poco más de 10 años) y hay muchos factores internos y sobre todo externos, que afectan al precio de Bitcoin significativamente creando mucha volatilidad y alto riesgo.

2.3. Factores internos y externos que afectan al precio

Aunque el conocimiento en torno al Bitcoin actualmente es más elevado que en sus inicios por parte de la sociedad, es necesario recalcar y comprender el valor de su precio, la volatilidad existente, el alto riesgo y sobre todo, los factores que afectan al precio [10].

Por un lado, los *factores internos* más importantes son los siguientes:

- *Halving:* Evento por el cual cada cierto tiempo se produce un cambio en la recompensa que reciben los mineros por bloque validado. El *Halving* se realiza cada 4 años aproximadamente, lo que equivale a unos 210000 bloques minados y se reducen a la mitad las recompensas de los mineros. Esto suele afectar positivamente al precio, ya que habrá un momento en el cual ya no se generen más Bitcoins al minar y su precio se basará principalmente en la ley de la oferta y demanda.

- *Oferta y demanda:* La *oferta* es la cantidad de Bitcoin disponible para comprar en el mercado. Por otro lado, la *demanda* es la cantidad de Bitcoin que se desea adquirir en un momento determinado.

  Tal y como se ha mencionado anteriormente, el hecho de que haya un límite en la creación de Bitcoins, le da mucho más valor. Por lo tanto, si crece la demanda en esta situación, es muy probable que incremente el precio. Es necesario destacar que, a diferencia de Bitcoin, en la economía tradicional los países pueden devaluar su moneda constantemente si sobrepasan el límite de impresión de efectivo, que es lo que permite tener una cierta estabilidad en el valor de una divisa. Se ha visto claramente en la época de la COVID-19 como muchos países, entre ellos Estados Unidos, han sobrepasado el límite de impresión de efectivo provocando que, a largo plazo, se devalúe cada vez más su moneda local [11].

- *Decisiones internas:* La introducción de mejoras de seguridad, de protocolo y nuevas funciones, da más valor a este cripto activo, hecho que se traduce en un incremento de su precio al aumentar la confianza de los usuarios. En cambio, si se anuncian o prometen mejoras y nuevas funcionalidades, sobre todo en fechas marcadas y éstas no se cumplen o se complica su implementación y salida, afectarán negativamente al precio de Bitcoin, creando mayor desconfianza entre los usuarios.

- *Ataques a la blockchain:* Los *ataques a la blockchain*, en concreto el ataque del 51 % [12], es el más temido por cualquier protocolo ya que si el 51 % de nodos de la red pasan a estar controlados por una persona o conjunto de personas, pueden tomar el control de toda la red y alterar el funcionamiento de la blockchain. No se ha detectado ningún ataque del 51 % en Bitcoin hasta la fecha pero, si ocurriera, afectaría drásticamente a su precio al crear inseguridad entre los usuarios provocando ventas masivas. Incluso es posible que el atacante o atacantes que tengan el control de la red, puedan realizar robos de Bitcoin de las billeteras de los usuarios.

  Aun así, se trata de un ataque cada vez más difícil de ejecutar, ya que, como se ha comentado, el atacante necesitaría tener más del 50% de los nodos de la red controlados y una red de mineros lo suficientemente potente como para que siempre se mine el bloque a favor del atacante.

- *Utilidad:* Actualmente, con Bitcoin es posible realizar pagos en la mayoría de establecimientos y en parte, tiendas online. Aun así, la visión con esta criptomoneda es largo placista y por este motivo los usuarios son más partidarios de realizar una inversión a largo plazo, por lo que mantienen intacta la cantidad de Bitcoin existente en sus billeteras e incluso realizan compras progresivas para acumular poco a poco.

  Es muy positivo para el precio del Bitcoin que los usuarios compartan la misma visión que se publicó cuando nació esta moneda digital, ya que hay una meta clara entre los usuarios y los creadores al querer que sea la moneda digital pionera para realizar pagos y transacciones por todo el mundo.

- *Capitalización de mercado:* La *capitalización de mercado* es el valor total de una criptomoneda que está en circulación. Esto suele indicar más estabilidad, es decir, cuanta más *capitalización de mercado* tenga una criptomoneda, más confianza se depositará en ella, por lo que la volatilidad debería reducirse.

  Por otro lado, también hay que valorar la distribución de esta *capitalización de mercado* (es decir, quién posee las criptomonedas), ya que, por ejemplo, una venta de una criptomoneda donde sus mismos creadores posean la mayoría de monedas, puede suponer una caída brusca de la capitalización y afectar negativamente al precio provocando que sea mucho más volátil (riesgo muy elevado en términos de inversión).

Bitcoin es, con diferencia, la criptomoneda líder con más *capitalización de mercado* y con una mayor distribución. Por lo tanto, da mucha más confianza que el resto, sobre todo a la hora de invertir, y esto afecta positivamente a su precio.

Por otro lado, los *factores externos* más importantes son los siguientes:

- *Medidas gubernamentales:* Aunque la naturaleza de las criptomonedas en general sea su descentralización al no depender de ninguna entidad gubernamental, en el caso de Bitcoin se ha demostrado que su precio es vulnerable a algunas decisiones, como por ejemplo la del pasado 24 septiembre de 2021 donde China declaró ilegales todas las transacciones con criptomonedas, provocando que el precio de Bitcoin cayera de los 44869 $ hasta los 42820 $, una diferencia de más de 2000 $ en cuestión de un solo día [13].

  Por otro lado, hay decisiones que también han impulsado al Bitcoin positivamente, como por ejemplo la decisión del presidente de El Salvador de adoptar Bitcoin como moneda legal en septiembre de 2021 [14]. Dos meses después, el Bitcoin crearía su nuevo máximo histórico hasta la fecha.

- *Figuras influyentes:* Principalmente, el fundador de Tesla Elon Musk, ha tenido mucha influencia en torno al mercado de las criptomonedas. En concreto, en marzo del año 2021, Musk anunció mediante la red social Twitter que Tesla aceptaría Bitcoin como medio de pago y el precio ascendió de 46000 $ a inicios de marzo, hasta los 61000 $ a mediados de mes, creando un nuevo máximo histórico [15]. El 12 de mayo de 2021, Musk publicó en Twitter que la compañía ya no aceptaría Bitcoins, indicando que la minería contaminaba el medio ambiente (algo que realmente es muy cuestionado). Este evento hizo que el precio descendiera de 57000 $ hacia los 49000 $ [16].

- *Otros eventos:*

  - *Crisis:* Cuando se declaró la pandemia de la COVID-19, el precio del Bitcoin y del resto de criptomonedas, sufrieron un fuerte impacto negativo repentino al mismo tiempo que la economía global. Pocos meses después, al ver que se avecinaba tanto una *crisis* sanitaria como económica, los grandes inversores comenzaron a comprar Bitcoin y el precio se impulsó desde los 4000 $ hasta los 35000 $ aproximadamente entre marzo y finales de 2020.

  - *Guerras:* El conflicto actual entre Rusia y Ucrania también ha afectado negativamente al precio de las criptomonedas, hasta el momento.

CAPÍTULO 3. ANÁLISIS TÉCNICO Y MACHINE LEARNING

En este capítulo, se exponen dos metodologías de análisis y las diferencias entre ambas para predecir la dirección del precio de Bitcoin: el análisis técnico y el uso de Machine Learning.

3.1. Análisis técnico

El análisis técnico se define como el estudio del movimiento de cualquier activo a través del uso de gráficas con el objetivo de predecir, con una probabilidad mayor, los posibles cambios en la estructura del mercado.

Un analista técnico normalmente utiliza tres tipos de gráfico:

- **Gráfico de velas japonesas**

Fig. 3.1 Gráfico Diario de BTC/USD (Bitcoin/Dólar estadounidense) de velas japonesas.

El gráfico de velas japonesas, como el mostrado en la Figura 3.1, es de los más utilizados por los analistas técnicos en muchos mercados diferentes (criptomonedas, forex o mercado de divisas, *commodities*, etc), ya que dan mucha información sobre el precio y se interpretan en diferentes temporalidades: segundos, minutos, horas, días, semanas y meses. Para comprender lo que representa cada una de ellas, se va a mostrar gráficamente una vela alcista y una vela bajista:

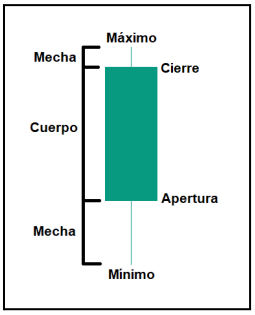

Fig. 3.2 Vela japonesa alcista de BTC/USD (Bitcoin/Dólar estadounidense).

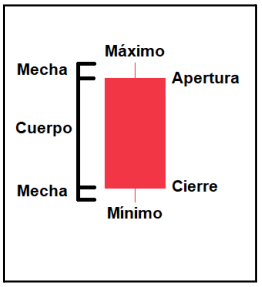

Fig. 3.3 Vela japonesa bajista de BTC/USD (Bitcoin/Dólar estadounidense).

Como se puede observar en las figuras anteriores, ambas velas están compuestas por las mismas características. Las diferencias entre ambas, están marcadas por la apertura y el cierre. Por este motivo, se exponen a continuación, las características que permiten identificar la formación de cada una:

- *Máximo (High):* El máximo de una vela se forma cuando un activo alcanza un precio superior al precio de apertura o de cierre según si la vela es bajista o alcista, respectivamente, durante un intervalo de tiempo concreto. El máximo se localiza en el extremo de la mecha superior.

- *Mínimo (Low):* El mínimo de una vela se forma cuando un activo alcanza un precio inferior al precio de apertura o de cierre según si la vela es alcista o bajista, respectivamente, durante un intervalo de tiempo concreto. El mínimo se localiza en el extremo de la mecha inferior.

- *Apertura (Open):* La apertura es el precio inicial de una vela japonesa. Después del cierre de la vela anterior, se forma una vela nueva, siendo el precio inicial el precio de cierre de la vela anterior.

- *Cierre (Close):* El cierre es el precio final de una vela japonesa cuando se completa un intervalo de tiempo determinado. Puede ser superior o inferior al precio de apertura según si la vela es alcista o bajista, respectivamente. Después del cierre, comienza la formación de una nueva vela.

- *Mecha (Spike):* La mecha indica que durante el intervalo de tiempo que se ha mantenido abierta una vela, el precio ha alcanzado precios superiores e inferiores al cierre/apertura según si la vela es alcista o bajista. Mediante las mechas, se identifican los máximos y mínimos.

- *Cuerpo (Body):* El cuerpo de una vela se forma mediante el precio de apertura y cierre cuando finaliza un intervalo de tiempo determinado. Si el precio de cierre es superior al de apertura, el cuerpo de la vela es alcista, mientras que si el precio de cierre es inferior al de apertura, el cuerpo de la vela es bajista.

Los colores más comunes de las velas son el verde, para las velas alcistas y el color rojo, para las velas bajistas. Aun así, ambos colores se pueden modificar a gusto del analista técnico.

La combinación de velas japonesas en el gráfico, permite al analista técnico identificar diferentes tipos de patrones que activen una señal de compra o de venta en un activo.

- **Gráfico de barras**

Fig. 3.4 Gráfico Diario de barras de Apple.

El gráfico de barras es muy utilizado por los analistas técnicos en acciones, principalmente. En la Figura 3.4 se muestra a modo de ejemplo, el gráfico de barras que refleja el precio de las acciones de Apple con temporalidad diaria.

De la misma manera que las velas japonesas, en las barras también se reflejan las direcciones alcistas y bajistas de cada una de ellas. A continuación, se muestra un ejemplo de una barra alcista y una barra bajista con sus características:

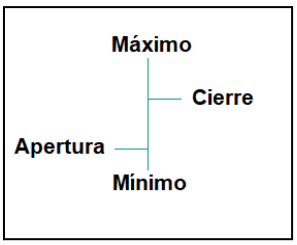

Fig. 3.5 Barra alcista de Apple.

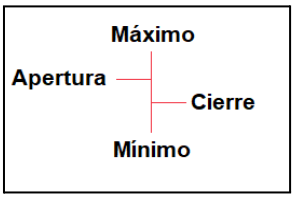

Fig. 3.6 Barra bajista de Apple.

Las barras contienen prácticamente las mismas características que las velas japonesas a excepción de que no se atribuye el concepto de cuerpo a las barras y, además, la forma es considerablemente diferente.

Las barras alcistas, son aquellas en las que el precio de cierre es superior al precio de apertura. Tal y como aparece en la Figura 3.5, las barras alcistas también dejan mechas en ambas direcciones creando el máximo y el mínimo.

En cuanto a las barras bajistas, el precio de cierre es inferior al precio de apertura. Como se puede observar en la Figura 3.6, las barras bajistas también pueden dejar mecha en ambas direcciones creando un máximo y un mínimo.

De la misma manera que las velas japonesas, el color más común de las velas alcistas es el verde y de las velas bajistas es el rojo.

El gráfico de barras no aporta tanta información como el gráfico de velas al no tener cuerpo y tener un grado de dificultad mayor para identificar patrones de precio que permitan detectar una posible señal de compra o de venta en un activo. Por este motivo, los analistas técnicos, suelen preferir el gráfico de velas japonesas al ser más atractivo y más fácil de interpretar [17].

- **Gráfico de líneas**

Fig. 3.7 Gráfico Diario de líneas de BTC/USD (Bitcoin/Dólar estadounidense).

Los gráficos de líneas, a diferencia del gráfico de barras y de velas japonesas, son utilizados para obtener una visión general de los movimientos del precio de un mercado en concreto, al mostrar exclusivamente los precios de cierre.

Una vez mostrados los tipos de gráficos más comunes a la hora de realizar un análisis de mercado, es conveniente centrarse en las características que aporta el gráfico de velas japonesas ya que, por una parte, es el que más información clara puede dar y por otra parte, es el más conveniente a la hora de analizar el mercado de las criptomonedas y, en este caso, el Bitcoin.

En el análisis técnico, se utilizan un conjunto de herramientas (líneas de tendencia, soportes, resistencias, zonas de liquidez, etc) [18], que permiten a los analistas tener un gráfico más claro y ver las posibilidades del siguiente movimiento. En la Figura 3.8 se muestra un ejemplo de análisis técnico en el gráfico diario de Bitcoin aplicando algunas de estas herramientas:

Fig. 3.8 Análisis técnico sencillo del gráfico de velas japonesas Diario BTC/USD (Bitcoin/Dólar estadounidense) del 26 de julio de 2022.

En este gráfico se observan tres zonas importantes. Las zonas se trazan cuando se detecta que el mercado se ha comportado de una forma similar en varias ocasiones del pasado. A continuación, se realiza una explicación detallada de la identificación de cada una de las tres zonas:

- *Zona superior* (color azul claro)*:* La zona superior se ha trazado al detectar que el precio rebotaba cada vez que la alcanzaba. Por ejemplo, entre los meses de enero y febrero de 2021, el precio de Bitcoin ascendía llegando a la zona marcada y realizó un movimiento bajista como si esa zona fuese un techo que no se pudiese superar. Posteriormente, se puede observar como entre mayo y agosto del mismo año, volvía a repetirse este comportamiento al entrar en la zona y no poder superarla hasta mediados de agosto y septiembre. Entre mediados de septiembre de 2021 e inicios de 2022, la zona actuó como si de un suelo se tratara, ya que el precio se volvía a incrementar cada vez que entraba en ella.

- *Zona intermedia* (color gris)*:* De la misma forma que la zona superior, se trazó la zona intermedia. En los meses de enero y febrero de 2021, el precio entró en la zona y la rechazó hacia arriba, repitiéndose este comportamiento entre mediados de mayo y agosto del mismo año. Entre mayo y junio de 2022, el precio intentó realizar un rebote pero finalmente acabó superando la zona hacia abajo.

- *Zona inferior* (color azul oscuro)*:* La zona inferior se ha trazado al identificar que en los meses de junio y julio de 2022, el precio entraba en ella en tres ocasiones consecutivas pero no consiguió superarla, provocando que el mercado fuera hacia arriba.

Una vez explicadas las zonas trazadas en la Figura 3.8, es posible realizar una interpretación de este análisis técnico para predecir los posibles movimientos futuros de Bitcoin. En este caso, al ver que el precio ha rebotado en tres

ocasiones consecutivas en la zona inferior, es posible que el Bitcoin suba hasta la zona intermedia trazada tal y como se ha marcado con la flecha de color azul.

Visto brevemente el significado y la esencia del análisis técnico, se puede llegar a la conclusión que se necesita una cantidad de tiempo considerable para aprender a analizar un gráfico, teniendo en cuenta que lo que se ha mostrado anteriormente sobre ello, es un porcentaje muy pequeño de todo el análisis que se debe realizar para predecir la dirección del precio de Bitcoin. Es por ello que existen algoritmos de Machine Learning que permiten crear modelos de predicción del precio y dirección de un activo de un mercado en concreto. A continuación, se van a exponer los algoritmos de Machine Learning más conocidos para valorar cuáles pueden ser útiles para crear un modelo de predicción del precio de Bitcoin.

3.2. Machine Learning

El Machine Learning o aprendizaje automático [5] , es una rama específica de las ciencias de la computación y la inteligencia artificial que permite crear sistemas capaces de aprender de forma automática.

Las características principales a destacar del Machine Learning son las siguientes:

- Existencia de algoritmos que permiten procesar cantidades de datos extensas y aprender de ellos.
- Capacidad de supervisar un sistema sin la intervención humana.
- Un sistema basado en Machine Learning, tiene que ser capaz de realizar analítica de datos y desarrollo de decisiones/respuestas que no tenía presentes en su inicio.

Gracias a la existencia de los algoritmos de Machine Learning, es posible realizar el análisis, la clasificación o la predicción de datos principalmente. Por este motivo, es esencial conocer las características y los tipos que existen.

3.3. Algoritmos de Machine Learning

Los algoritmos de Machine Learning o aprendizaje automático [19], permiten analizar conjuntos de datos extensos y complejos mediante la creación de modelos que permiten realizar predicciones o clasificar la información. Cada uno de los algoritmos tiene sus propios parámetros a partir de los cuales un modelo se puede optimizar mejorando sus prestaciones y así, su predicción o clasificación.

3.3.1. Tipos de aprendizajes automáticos

Las técnicas de aprendizaje automático [19] son las siguientes:

- *Aprendizaje supervisado:* El aprendizaje automático supervisado, permite utilizar un conjunto de datos de entrada y salida conocidos (etiquetados) y entrenar un modelo que genere predicciones lógicas respecto la salida.

  El aprendizaje supervisado se puede dividir en dos técnicas:

  - *Clasificación:* La técnica de clasificación, permite predecir una clase. Por ejemplo, si un correo es spam o no (2 clases), si va a hacer frío o no (2 clases) o si una publicación en una red social va a tener muchos “likes” o no (2 clases), entre otras. Hay algunos ejemplos de clasificación que tienen más de 2 clases.

  - *Regresión:* La técnica de regresión, permite predecir un valor numérico. Por ejemplo, predecir el precio del Bitcoin, el precio de venta de un inmueble o el tiempo que va a hacer durante la semana, entre otros ejemplos.

- *Aprendizaje no supervisado:* El aprendizaje automático no supervisado, permite utilizar un conjunto de datos de entrada sin conocer los datos de salida (no etiquetados). Es ideal para realizar agrupaciones de datos, por ejemplo, en el caso de que una compañía telefónica quiera optimizar la ubicación de las torres de telefonía móvil para estimar grupos de personas que usan cada torre.

- *Aprendizaje por refuerzo:* El aprendizaje por refuerzo, consiste en un aprendizaje por prueba y error y a partir de los resultados que se van obteniendo, se decide la siguiente acción. Un ejemplo podría aplicarse a un robot, en el que en vez de programar instrucciones para que se mueva, realice intentos y cada vez que lo consiga, realizar una acción de recompensa para que asocie el movimiento con lo correcto.

Mediante las técnicas anteriores, es posible clasificar o predecir datos. Para ello, se dividen en dos subconjuntos:

- *Entrenamiento:* Es el subconjunto de datos que se destina a ser entrenado para crear un modelo con uno de los algoritmos de aprendizaje automático. Cuantos más datos de entrenamiento se añaden, normalmente el algoritmo es más preciso.

- *Test:* Es el subconjunto de datos que se destina a validar el modelo entrenado. En el caso de clasificación, se evalúa el rendimiento del modelo entrenado a la hora de clasificar y se compara con los datos reales de test, en cambio en el caso de la predicción, se evalúa el rendimiento del modelo entrenado a la hora de predecir y se compara con los datos reales de test.

Es necesario destacar que en el aprendizaje automático no supervisado no se realiza entrenamiento, ya que contiene algoritmos que trabajan de forma directa con los datos disponibles.

Una vez vistos los 3 tipos, se ha escogido para el presente proyecto el aprendizaje supervisado de regresión, ya que se disponen de datos etiquetados para realizar el entrenamiento del modelo con cada uno de los algoritmos y permite predecir el precio de Bitcoin.

3.3.2. Algoritmos de aprendizaje supervisado de regresión

Los algoritmos más destacados que forman parte de las técnicas de aprendizaje automático supervisado de regresión se citan a continuación:

- Regresión lineal simple [20].
- Regresión lineal múltiple [21].
- Support Vector Machines (SVM) [22].
- Decision Trees [23].
- Random Forest [24].
- Deep Learning [25].
- Time Series Forecasting [26].

Entre todos los algoritmos anteriores, se han escogido Random Forest, Regresión lineal y Time Series Forecasting para crear los modelos de predicción del precio de Bitcoin.

En primer lugar, se ha escogido Random Forest ya que, entre otras ventajas, permite realizar un análisis exhaustivo a la hora de crear un modelo de predicción gracias a la cantidad de parámetros que contiene, por lo que facilita la optimización de un modelo inicial. En segundo lugar, se ha utilizado el Regresor Lineal ya que es un algoritmo sencillo de comprender, con un número menor de parámetros que el algoritmo de Random Forest pero ofrece muy buen rendimiento a la hora de realizar predicciones. Finalmente, se ha utilizado el Time Series Forecasting ya que asocia componentes temporales a las predicciones y principalmente porque permite añadir variables externas a un conjunto de datos, como por ejemplo el Halving de Bitcoin, para mejorar las predicciones realizadas.

En los siguientes apartados, se van a describir de forma detallada estos algoritmos junto con el Decision Tree, puesto que tanto Random Forest como Time Series Forecasting se basan en él. La explicación de los parámetros de cada uno de los algoritmos y los valores por defecto se extraen de las librerías utilizadas en el presente proyecto: Scikit-Learn.

3.3.3. Decision Tree

Los árboles de decisión o Decision Trees [23] permiten crear modelos tanto para regresión como para clasificación, con una estructura en forma de árbol tal

y como su nombre indica. Este algoritmo descompone el conjunto de datos original en varios subconjuntos y de este modo va creando el árbol de decisión.

Con el objetivo de comprender sus características, se va a exponer un ejemplo práctico. Este ejemplo está formado por un conjunto de datos que contiene precios de viviendas de la ciudad de Boston, así como información socioeconómica del barrio en el que se encuentran [23]. El conjunto de datos está formado por 13 atributos (características de cada uno de los pisos), aunque los más importantes son aquellos que intervienen en la construcción del árbol y son los siguientes:

- CRIM: Tasa de criminalidad per cápita por ciudad.
- RM: Promedio de habitaciones por vivienda.
- DIS: Distancias ponderadas a cinco centros de empleo de Boston.
- PTRATIO: Ratio alumno-profesor por municipio.
- LSTAT: % de la población con menos recursos.
- MEDV: Valor medio de las viviendas ocupadas por sus propietarios en miles de dólares.

El objetivo es predecir el precio medio de una vivienda (MEDV), en función de las variables anteriores o atributos. Dichos atributos se usan para crear las distintas ramas del árbol, como se muestra en la Figura 3.9.

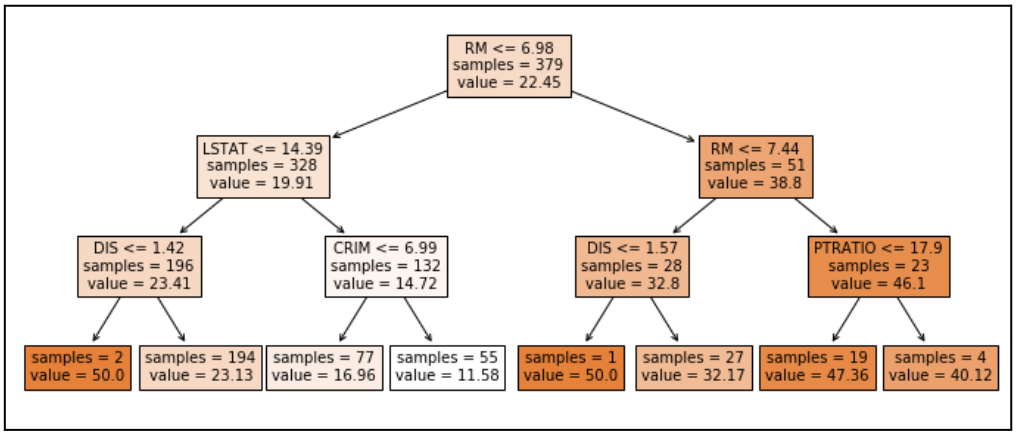

Fig. 3.9 Representación del árbol de decisión del ejemplo.

Los elementos que forman parte de éste son:

- *Nodo raíz o principal (Root node):* Nodo superior con dos o más ramas salientes. En la figura, el nodo principal está indicando una condición para RM <= 6.98, lo que indica un promedio de habitaciones por vivienda más pequeño o igual que 6.98.

- *Nodos de decisión (Decision nodes):* Nodo que tiene dos o más ramas salientes y una entrante. En este caso, hay seis nodos de decisión con diferentes condiciones (LSTAT <= 14.39, DIS <= 1.42, CRIM <= 6.99, DIS <= 1.57 y PTRATIO <= 17.9). Por lo tanto, según la decisión tomada, el árbol de decisión dará un resultado u otro.

- *Nodos hoja (Leaf nodes):* Nodo que no tiene ramas salientes y representa una decisión final. Dicho de otra forma, es el resultado final de las decisiones que ha ido tomando el árbol. Por ejemplo, si se observa la rama más a la izquierda del árbol de la figura anterior, se puede observar que el modelo ha predicho un precio promedio de 50.000 $ para viviendas que están en una zona con un RM <= 6.98, un LSTAT <= 14.39 y un DIS <= 1.42.

A la hora de crear u obtener un árbol de decisión, es necesario que se seleccionen los mejores atributos posibles en cada uno de los nodos. Es por ello que existen tres métodos principales para conseguirlo, que se explican brevemente a continuación:

- *Entropía (Entropy):* La *entropía* es una medida de aleatoriedad de la información o de los datos que se procesan.

  A partir de la siguiente figura se exponen las características principales de la entropía:

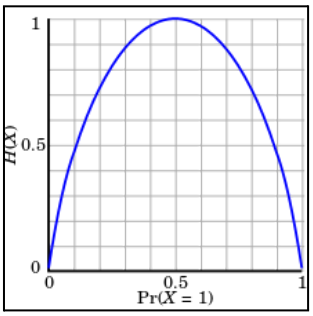

Fig. 3.10 Gráfico de la entropía de la información (H(X)).

  Como se puede observar en la Figura 3.10, la entropía es 0 cuando la probabilidad es 0 o 1, lo que indica que la decisión recae en un resultado u otro, pero cuando la entropía toma el valor máximo (1), indica que todos los resultados son igual de probables, por lo que la incertidumbre es muy grande y, por lo tanto, el resultado final es poco predecible.

- *Ganancia de Información (Information Gain):* La *ganancia de información* mide la importancia que toma un atributo dentro de la división de datos (clasificación o regresión).

  Una *ganancia de información* elevada se traduce en una mejor división de datos y los permite clasificar/predecir fácilmente. En cambio, una ganancia de información baja divide peor los datos y es más difícil de clasificarlos o realizar predicciones.

- *Índice de Gini (Gini Index):* El *índice de Gini*, permite evaluar la calidad de las divisiones en un conjunto de datos. En definitiva, este índice indica si los datos se clasifican/predicen correctamente o no.
  Cuanto más grande es el *índice de Gini*, más desigualdad, por lo que hay más heterogeneidad en la decisión de clasificación/predicción.

Una vez expuestas las características más destacadas del algoritmo de árboles de decisión, se van a mostrar las propiedades más importantes a tener en cuenta en el primer algoritmo escogido para realizar el modelo de predicción, el Random Forest.

3.3.4. Random Forest

El Random Forest [24], es un algoritmo automático que se puede aplicar tanto para clasificación como para regresión y combina las salidas de un conjunto de árboles de decisión para llegar a un resultado único.

Utiliza la técnica “Ensemble”, que consiste en la combinación de múltiples modelos en lugar de uno solo. Este algoritmo utiliza la técnica de Bagging, también conocida como Bootstrap Aggregation, que consiste en que cada árbol de decisión escoge una muestra aleatoria de un conjunto de datos utilizando reemplazamiento (Bootstrap) y cada árbol de decisión es entrenado de forma independiente generando un resultado. El reemplazamiento permite incluir un mismo dato varias veces.

El resultado final será escogido por mayoría en clasificación y promedio en caso de regresión (Aggregation).

Para su mayor comprensión, se muestra un esquema en la siguiente figura:

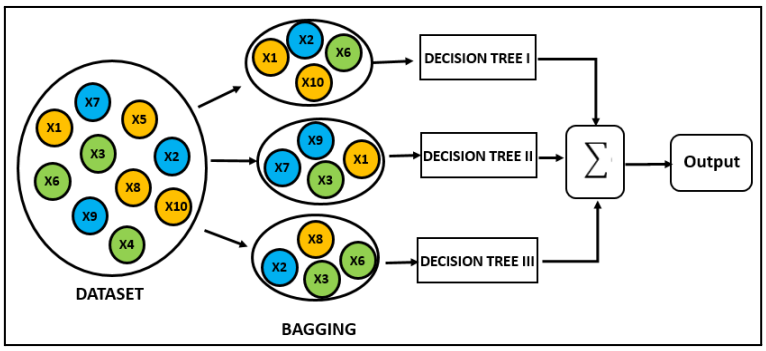

Fig. 3.11 Esquema del funcionamiento de Bagging.

Los parámetros que usa el algoritmo son modificables y son totalmente necesarios para optimizar un modelo de predicción. Los parámetros más importantes que contiene el algoritmo son los siguientes:

- *n_estimators:* Número de árboles que contiene el algoritmo. Normalmente, cuantos más árboles se usan más se optimiza el algoritmo, pero suele haber un límite donde el algoritmo empieza a

empeorar si se sigue aumentando este número. Por defecto, su valor se establece en 100.

- *criterion:* Función para evaluar la calidad de la división de datos. Por defecto está establecido el *squared error* o error cuadrático.

- *max_depth:* Profundidad máxima del árbol. La profundidad de los árboles, permite establecer el máximo número de nodos hoja. Por lo tanto, cuanta más profundidad, más divisiones del árbol hay y más nodos hoja, por lo que capta más información. El valor por defecto es None, lo que indica que los nodos se expanden hasta que se crea el número máximo de nodos hoja posibles.

- *min_samples_split:* Número mínimo de muestras requeridas para dividir un nodo interno. Por defecto, se establece en 2.

- *min_samples_leaf:* Número mínimo de muestras que debe haber en un nodo hoja o nodo final. Por defecto, su valor es 1.

- *max_features:* Número de atributos de un conjunto de datos que se toman en consideración para valorar la mejor división o el mejor resultado. Por defecto, se establece en 1.

- *max_leaf_nodes:* Número máximo de nodos hoja o nodos finales. Por defecto, su valor es None, lo que indica que no se limita el número de nodos hoja.

- *oob_score:* Parámetro que expresa el error de predicción medio en cada muestra de entrenamiento, por lo que permite evaluar el modelo construido. Por defecto, este parámetro está desactivado (False).

- *n_jobs:* Número de cores de la CPU que se pueden utilizar para entrenar árboles de forma paralela. Por defecto, su valor es 1, por lo que solo utiliza 1 core de la CPU. Existe la opción de introducir el valor -1, que permite utilizar todos los cores disponibles en un dispositivo para entrenar los árboles.

- *random_state:* Controla la aleatoriedad de una muestra. Por defecto, no se establece valor para este parámetro, por lo que cada vez que se entrena un modelo se utiliza un valor aleatorio diferente (None).

- *max_samples:* Cantidad de muestras de un conjunto de datos que se destinan a un árbol en particular. El valor por defecto es None, por lo que se utilizan todas las muestras de entrenamiento.

Expuesto el algoritmo de Random Forest, a continuación se muestran las características más importantes del segundo algoritmo utilizado: la Regresión lineal.

3.3.5. Regresión lineal

La regresión lineal [20], es un algoritmo automático supervisado utilizado en Machine Learning para realizar predicciones. Existen dos tipos de regresión lineal: la regresión lineal simple y la regresión lineal múltiple.

Por un lado, la regresión lineal simple [20] permite generar un modelo de regresión que muestra la relación lineal entre dos variables representada mediante la ecuación de una recta. Las dos variables se conocen como la variable independiente X (denominada también regresor, predictor o *feature*) y la variable dependiente Y (denominada también variable respuesta). La ecuación de la regresión lineal simple, se trata de la ecuación de una recta como se muestra a continuación:

$$Y = \beta_0 + \beta_1 X + \varepsilon \qquad \textbf{(3.1)}$$

Donde Y y X son la variable dependiente y la variable independiente, respectivamente. β_0 es la ordenada en el origen y se corresponde con el valor promedio de la variable respuesta cuando todos los predictores son cero. β_1 es la pendiente de la recta e indica cómo cambia Y al incrementar X en una unidad, aunque en este caso solo existe un valor para X, ya que solo tenemos una única variable independiente. Finalmente, ε es el error, la diferencia entre el valor observado y el estimado por un modelo.

Por otro lado, la regresión lineal múltiple [21] modela la relación entre una variable dependiente y más de una variable independiente. Con este tipo de regresión, es posible obtener predicciones más complejas al tener más variables. La ecuación de la regresión lineal múltiple, se muestra a continuación:

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \cdots + \beta_p X_p + \varepsilon \qquad \textbf{(3.2)}$$

Las diferencias entre el cálculo de la regresión lineal simple y múltiple, se encuentran simplemente en la cantidad de variables independientes y las β que se utilizan. Por lo tanto, la variable X_p son las variables independientes, mientras que β_p, es el efecto promedio que tiene sobre la variable respuesta el incremento en una unidad de la variable independiente X_p, manteniéndose constantes el resto de variables. Se conocen como coeficientes parciales de regresión.

Los parámetros que contiene este algoritmo son los siguientes:

- *fit_intercept:* Parámetro que calcula el valor de β_0 de la ecuación. Por defecto, su valor es True. Si se establece en False, β_0 no se calcula.

- *normalize:* Su valor por defecto es False, pero si se cambia a True, las variables independientes se normalizan restando la mediana y dividiendo por la regla L2 de la regresión lineal, donde L2 es la media del cuadrado de los coeficientes del modelo (β_p) [27]. Esto evita el sobreajuste del modelo de predicción.

- *copy_X:* El valor del parámetro por defecto es True. Esto quiere decir que a la hora de entrenar el modelo, las variables independientes establecidas como entrada serán exactamente las mismas que las de salida después de entrenar el modelo. En cambio, si el valor se establece en False, existe la posibilidad que las variables independientes que se han pasado en la entrada se modifiquen y no sean las mismas que las de salida después de entrenar el modelo.

- *n_jobs:* Número de cores de la CPU que se utilizan para los cálculos. Solo proporciona aceleración en caso de problemas grandes. Por defecto, su valor es None, es decir, que solo utiliza 1 core de la CPU.

- *positive:* En el caso que los coeficientes β_p sean negativos, si el valor es True se fuerza que los coeficientes sean positivos. Cuando un coeficiente es positivo, a medida que aumenta el valor de una variable independiente, la media de la variable dependiente también tiende a aumentar. En cambio, un coeficiente negativo sugiere que a medida que aumenta la variable independiente, la variable dependiente tiende a disminuir, creando así correlaciones negativas. Por lo tanto, según el trabajo que se quiera realizar con un Data Set, puede interesar que los coeficientes se fuercen a ser positivos o no.

3.3.6. Series temporales: *Forecasting*

Las series temporales [26], son sucesiones de datos ordenados cronológicamente, espaciados a intervalos iguales o desiguales. Mediante las series temporales, es posible realizar predicciones de valores futuros, ya sea mediante datos pasados conocidos (autorregresión) o utilizando otras variables (exógenas).

El proceso de predicción de valores expuesto en el párrafo anterior se denomina *Time Series Forecasting*. El *Forecasting*, se caracteriza por asociar una componente temporal a las predicciones, y porque permite crear modelos de predicción utilizando diferentes algoritmos. En este caso, se ha utilizado el regresor LightGBM, un algoritmo de potenciación de gradientes (*Gradient Boosting*) basado en árboles de decisión [28]. El funcionamiento de LightGBM se basa en minimizar una función objetivo mediante la técnica de *Gradient Boosting* combinando árboles de decisión sencillos. Se puede considerar que LightGBM es una versión optimizada del *Gradient Boosting*, puesto que permite manejar una mayor cantidad de datos, a mayor velocidad y con un menor uso de memoria. Por este motivo, constituye una de las técnicas de regresión que ha cobrado mayor popularidad durante los últimos años.

Existen 4 métodos de *Forecasting* que se describen a continuación [26]:

- **Forecasting autorregresivo recursivo:** En el Forecasting autorregresivo recursivo, se utiliza únicamente la variable que se desea predecir tanto como input como output del modelo de predicción. Es decir, la variable a predecir se utiliza también para entrenar el modelo.

  Para su mayor comprensión, se utiliza el siguiente ejemplo:

  Mediante una serie temporal con el gasto mensual en millones de dólares en fármacos con corticoides que tuvo el sistema de salud Australiano entre 1991 y 2008, se ha creado un modelo autorregresivo capaz de predecir el futuro gasto mensual. El conjunto de datos utilizado, se muestra a continuación:

fecha	y	exog_1	exog_2
1992-04-01	0.379808	0.958792	1.166029
1992-05-01	0.361801	0.951993	1.117859
1992-06-01	0.410534	0.952955	1.067942
1992-07-01	0.483389	0.958078	1.097376
1992-08-01	0.475463	0.956370	1.122199

Fig. 3.12 Parte del Data Set del gasto mensual en fármacos con corticoides del sistema de salud Australiano entre 1991 y 2008.

  De la misma forma que con los algoritmos de Regresión lineal y Random Forest, se especifican los conjuntos de entrenamiento y test de datos. La variable “y” es el gasto mensual en millones de dólares. En esta metodología, esta misma variable es la que se ha utilizado tanto para entrenar el modelo de forecasting como para predecir su valor.

- **Forecasting autorregresivo recursivo con variables exógenas:** Este método es similar al anterior, pero adicionalmente se incorporan otras variables cuyo valor a futuro se conoce y se utilizan como predictores adicionales en el modelo.

  En el ejemplo del método anterior, se ha utilizado como predictor únicamente la “y”. En este caso, se entrena el modelo teniendo en cuenta tanto la “y” como las variables exógenas que contiene el set de datos.

- **Forecasting autorregresivo recursivo con predictores custom:** El forecasting autorregresivo con predictores custom, permite incorporar otras características externas de la serie temporal, como por ejemplo una media móvil de los últimos n valores puede servir para capturar la tendencia de una serie.
  Utilizando el ejemplo expuesto, con este método es posible configurar los predictores que se deseen. En este caso se ha utilizado la media

móvil de los últimos 20 meses y se ha incorporado en el entrenamiento del modelo:

```
# Función para calcular los predictores a partir de la serie temporal
# ==============================================================================
def custom_predictors(y):
    '''
    Create first 10 lags of a time series.
    Calculate moving average with window 20.
    '''

    lags = y[-1:-11:-1]
    mean = np.mean(y[-20:])
    predictors = np.hstack([lags, mean])

    return predictors
```

Fig. 3.13 Función personalizada para utilizar como predictor adicional la media móvil.

- **Direct multi-step forecasting:** Este método utiliza tanto la variable a predecir ("y") como las variables exógenas como predictores complementarios si se desea. Sigue una estrategia de predicción recursiva en la que cada nueva predicción realizada se basa en la predicción anterior [29].

CAPÍTULO 4. CREACIÓN DE MODELOS DE PREDICCIÓN DEL PRECIO DE BITCOIN CON MACHINE LEARNING

En este capítulo, se va a exponer todo el proceso de creación de modelos de predicción del precio de Bitcoin utilizando los algoritmos de Machine Learning explicados en el capítulo anterior, junto con los resultados obtenidos. Para ello, se ha utilizado el lenguaje de programación Python con las librerías que facilitan el uso y el desarrollo de la programación con Machine Learning [6]. Todo el desarrollo de código se ha realizado mediante la herramienta Google Colaboratory [7].

4.1. Metodología

4.1.1. Generación y preparación del Data Set

El primer paso necesario para poder crear los modelos de predicción del precio de Bitcoin mediante los algoritmos de Machine Learning, ha sido obtener y generar un Data Set.

Un Data Set es un conjunto de datos tabulados en filas y columnas, donde cada columna representa una variable o atributo, y las filas, representan un grupo de datos específicos. Contiene todos los valores de cada una de las variables, como por ejemplo la altura y el peso de un conjunto de objetos.

En este caso, se han desarrollado un conjunto de funciones para extraer los datos en temporalidad diaria sobre el par de criptomonedas Bitcoin/Tether (BTC/USDT) e introducirlos en el Data Set, utilizando la API de Binance [30].

Binance es el mayor *exchange de criptomonedas* (plataforma digital de intercambio de monedas virtuales) en términos de volumen de operaciones. En un *exchange de criptomonedas* es posible realizar operaciones de compra, venta o intercambio. Binance va más allá y adicionalmente ha creado un ecosistema con multitud de aplicaciones. Una de las que más destaca es su API, a partir de la cuál es posible consultar y extraer los datos de cualquier criptomoneda listada, en este caso Bitcoin.

El Tether o USDT se define como una *stable coin* (criptomoneda estable) que está vinculada a una de las divisas más importantes del mundo real en una base 1 a 1, el dólar estadounidense o USD.

Los datos extraídos de la API y que forman el Data Set se detallan a continuación:

- *open_time:* Día y hora de apertura de las velas de Bitcoin.
- *open:* Precio de apertura diaria de la vela de Bitcoin.

- *high:* Precio máximo diario alcanzado de la vela diaria de Bitcoin.
- *low:* Precio mínimo diario alcanzado de la vela diaria de Bitcoin.
- *close:* Precio de cierre diario de la vela de Bitcoin.
- *volume:* Cantidad de Bitcoin negociada durante un día.
- *close_time:* Día y hora de cierre de las velas de Bitcoin.
- *quote:* Volumen en USDT.
- *trades:* Cantidad de operaciones diarias ejecutadas de Bitcoin.
- *takers_buy_base:* Cantidad del volumen total que representan las órdenes de compra del Bitcoin.
- *takers_buy_quote:* Cantidad del volumen total que representan las órdenes de compra de USDT.
- *ignore:* Es un "legacy field" o campo heredado. Se trata de un campo obsoleto que anteriormente contenía datos de interés, pero que actualmente está en desuso, por lo que su valor se puede ignorar, tal y como su nombre lo indica.

Para los atributos *"open_time"* y *"close_time"* se ha realizado la conversión de timestamp a día/hora.

Es necesario destacar que el Data Set está formado por las principales características del gráfico de velas expuestas en el apartado 3.1, ya que se han incluido el precio máximo (*high*), mínimo (l*ow*), apertura (*open*) y cierre (*close*).

Una vez se ha generado el Data Set y se ha descargado, se han cargado tanto las librerías necesarias de Python como el Data Set en el entorno de trabajo. El contenido del Data Set descrito anteriormente, se muestra en la siguiente figura:

	open_time	open	high	low	close	volume	close_time	quote	trades	takers_buy_base	takers_buy_quote	ignore
0	2017-08-18 02:00:00	4285.08	4371.52	3938.77	4108.37	1199.888264	2017-08-19 01:59:59.999000064	5.086958e+06	5233.0	972.868710	4.129123e+06	9384.141409
1	2017-08-19 02:00:00	4108.37	4184.69	3850.00	4139.98	381.309763	2017-08-20 01:59:59.999000064	1.549484e+06	2153.0	274.336042	1.118002e+06	9184.085529
2	2017-08-20 02:00:00	4120.98	4211.08	4032.62	4086.29	467.083022	2017-08-21 01:59:59.999000064	1.930364e+06	2321.0	376.795947	1.557401e+06	10125.414084
3	2017-08-21 02:00:00	4069.13	4119.62	3911.79	4016.00	691.743060	2017-08-22 01:59:59.999000064	2.797232e+06	3972.0	557.356107	2.255663e+06	11706.769970
4	2017-08-22 02:00:00	4016.00	4104.82	3400.00	4040.00	966.684858	2017-08-23 01:59:59.999000064	3.752506e+06	6494.0	423.995181	1.637188e+06	11773.279500
...	...	...	...	...	...	...	...	...	...	...	...	...
1702	2022-04-16 02:00:00	40551.90	40709.35	39991.55	40378.71	15805.447180	2022-04-17 01:59:59.999000064	6.382755e+08	423446.0	7642.382430	3.086424e+08	0.000000
1703	2022-04-17 02:00:00	40378.70	40595.67	39546.17	39678.12	19988.492590	2022-04-18 01:59:59.999000064	8.034142e+08	590241.0	9578.915330	3.851249e+08	0.000000
1704	2022-04-18 02:00:00	39678.11	41116.73	38536.51	40801.13	54243.495750	2022-04-19 01:59:59.999000064	2.153575e+09	1157741.0	27097.193750	1.076513e+09	0.000000
1705	2022-04-19 02:00:00	40801.13	41760.00	40571.00	41493.18	35788.858430	2022-04-20 01:59:59.999000064	1.472363e+09	934526.0	17806.815060	7.325392e+08	0.000000
1706	2022-04-20 02:00:00	41493.19	41631.47	41229.96	41312.24	8026.777940	2022-04-21 01:59:59.999000064	3.321694e+08	245315.0	3808.141970	1.575949e+08	0.000000

Fig. 4.1 Contenido del Data Set de Bitcoin/Tether.

El Data Set creado, está formado por los datos relacionados con el precio de Bitcoin/Tether (BTC/USDT) que abarca un periodo de tiempo que va desde la fecha inicial 18/08/2017 hasta la fecha final 20/04/2022 y contiene 1707 filas y 12 columnas.

Una vez cargado el Data Set en el entorno de trabajo, ha sido necesario añadir una columna denominada “y”, que hace referencia al precio de cierre de la vela del siguiente día. A partir de “y” se va a evaluar la predicción del modelo comparándola con los valores predichos. Además, se ha eliminado la última fila del Data Set ya que no contenía valor de “y” e impedía trabajar con el archivo.

A continuación, se muestra el Data Set modificado:

	open_time	open	high	low	close	volume	close_time	quote	trades	takers_buy_base	takers_buy_quote	y
0	2017-08-18 02:00:00	4285.08	4371.52	3938.77	4108.37	1199.888264	2017-08-19 01:59:59.999000064	5.086958e+06	5233.0	972.868710	4.129123e+06	4139.98
1	2017-08-19 02:00:00	4108.37	4184.69	3850.00	4139.98	381.309763	2017-08-20 01:59:59.999000064	1.549484e+06	2153.0	274.336042	1.118002e+06	4086.29
2	2017-08-20 02:00:00	4120.98	4211.08	4032.62	4086.29	467.083022	2017-08-21 01:59:59.999000064	1.930364e+06	2321.0	376.795947	1.557401e+06	4016.00
3	2017-08-21 02:00:00	4069.13	4119.62	3911.79	4016.00	691.743060	2017-08-22 01:59:59.999000064	2.797232e+06	3972.0	557.356107	2.255663e+06	4040.00
4	2017-08-22 02:00:00	4016.00	4104.82	3400.00	4040.00	966.684858	2017-08-23 01:59:59.999000064	3.752506e+06	6494.0	423.995181	1.637188e+06	4114.01
...	...	...	...	...	...	...	...	...	...	...	...	...
1701	2022-04-15 02:00:00	39942.37	40870.36	39766.40	40551.90	24026.357390	2022-04-16 01:59:59.999000064	9.668884e+08	616536.0	11755.544390	4.731004e+08	40378.71
1702	2022-04-16 02:00:00	40551.90	40709.35	39991.55	40378.71	15805.447180	2022-04-17 01:59:59.999000064	6.382755e+08	423446.0	7642.382430	3.086424e+08	39678.12
1703	2022-04-17 02:00:00	40378.70	40595.67	39546.17	39678.12	19988.492590	2022-04-18 01:59:59.999000064	8.034142e+08	590241.0	9578.915330	3.851249e+08	40801.13
1704	2022-04-18 02:00:00	39678.11	41116.73	38536.51	40801.13	54243.495750	2022-04-19 01:59:59.999000064	2.153575e+09	1157741.0	27097.193750	1.076513e+09	41493.18
1705	2022-04-19 02:00:00	40801.13	41760.00	40571.00	41493.18	35788.858430	2022-04-20 01:59:59.999000064	1.472363e+09	934526.0	17806.815060	7.325392e+08	41312.24

Fig. 4.2 Data Set modificado con la nueva columna “y” y la última fila eliminada.

Como se puede observar en la figura anterior, después de eliminar la última fila y añadir la columna “y”, la fecha final es el 19/04/2022.

Posteriormente, se ha eliminado la columna “ignore” del Data Set. Por este motivo, el Data Set pasa a tener 1706 filas y 12 columnas.

4.1.2. Cálculo de la correlación entre los atributos

Una vez preparado el Data Set, se ha calculado la correlación entre todos los atributos y el precio de cierre del día siguiente “y” (valor que queremos predecir) para identificar los atributos que tienen mayor importancia en la predicción.

A continuación, se muestran los resultados:

	open	high	low	close	volume	quote	trades	takers_buy_base	takers_buy_quote	y
open	1.000000	0.999298	0.998624	0.998359	0.234167	0.800176	0.764660	0.224989	0.801886	0.996824
high	0.999298	1.000000	0.998496	0.999282	0.243161	0.807709	0.772100	0.234989	0.810262	0.997694
low	0.998624	0.998496	1.000000	0.999077	0.215695	0.782769	0.750018	0.207687	0.785618	0.997401
close	0.998359	0.999282	0.999077	1.000000	0.230980	0.796688	0.762835	0.223608	0.800010	0.998361
volume	0.234167	0.243161	0.215695	0.230980	1.000000	0.633871	0.723463	0.997397	0.630902	0.233408
quote	0.800176	0.807709	0.782769	0.796688	0.633871	1.000000	0.965185	0.626939	0.999202	0.797439
trades	0.764660	0.772100	0.750018	0.762835	0.723463	0.965185	1.000000	0.714688	0.964667	0.763875
takers_buy_base	0.224989	0.234989	0.207687	0.223608	0.997397	0.626939	0.714688	1.000000	0.626043	0.226075
takers_buy_quote	0.801886	0.810262	0.785618	0.800010	0.630902	0.999202	0.964667	0.626043	1.000000	0.800805
y	0.996824	0.997694	0.997401	0.998361	0.233408	0.797439	0.763875	0.226075	0.800805	1.000000

Fig. 4.3 Correlación de las variables del Data Set.

Como se puede observar en la figura anterior, los parámetros “open”, “high”, “low” y “close” tienen una correlación del 99% con “y” aproximadamente y por este motivo, son los que se han utilizado para crear los modelos de predicción con cada uno de los algoritmos.

4.1.3. Herramientas utilizadas para evaluar los modelos de predicción

Las principales herramientas utilizadas para el desarrollo y la evaluación de los modelos de predicción del precio de Bitcoin son el entrenamiento y test de datos, la modificación de los parámetros de un algoritmo de forma manual, utilizando las técnicas de Out Of Bag, Cross Validation y Grid Search y los errores en regresión.

A continuación, se exponen detalladamente cada una de estas herramientas:

- **Entrenamiento y test de datos:** Como se ha explicado anteriormente, los algoritmos de Machine Learning con aprendizaje supervisado constan de dos fases: entrenamiento y test. Para ellos es necesario dividir el conjunto de datos en un subconjunto de entrenamiento, que servirá para entrenar un modelo y otro subconjunto de test, que servirá para comprobar la bondad de dicho modelo.

  Normalmente, el conjunto de datos se suele dividir en 80 % para entrenamiento y 20 % para test [31]. Aun así, esta división depende del conjunto de datos con el que se vaya a trabajar y siempre es necesario probar con diferentes distribuciones para valorar cuál es la óptima.

  Para tener un buen modelo de predicción, es necesario tener en cuenta el sobreajuste o “overfitting” y el subajuste o “underfitting”.

  Por un lado, el sobreajuste o *overfitting* es un efecto causado por el sobreentrenamiento de datos. Esto sucede cuando el algoritmo utilizado para crear un modelo de predicción queda demasiado ajustado a características específicas de los datos del subconjunto de

entrenamiento, lo que quiere decir que puede ser muy bueno para un determinado conjunto de datos, pero si se van introduciendo nuevos o si se utiliza el modelo para otro conjunto de datos, es muy probable que el rendimiento sea mucho menor.

Por otro lado, el subajuste o *underfitting* es el caso contrario. Sucede cuando el subconjunto de entrenamiento es insuficiente. Como consecuencia, el modelo creado es demasiado simple y su capacidad predictiva es muy baja.

Es posible detectar el sobreajuste cuando los errores y el coeficiente de determinación (que se describen más adelante) de un modelo son mucho mejores para el conjunto de entrenamiento que para el conjunto de test. En cambio, el subajuste puede suceder cuando los resultados de ambos conjuntos son muy malos.

Por lo tanto, en el proceso de creación del modelo de predicción es necesario llegar a un equilibrio para evitar ambas problemáticas.

- **Parameter Tuning:** La modificación de parámetros de un algoritmo, es necesaria para encontrar el modelo con mayor capacidad de predicción posible. Para ello se utilizan diferentes técnicas: manual, validación con Out of Bag, Cross Validation y Grid Search.

  La **modificación de parámetros manual** permite probar diferentes combinaciones de los distintos parámetros de un algoritmo para poder encontrar los valores óptimos, es decir, los que permiten obtener mejores predicciones. Aun así, es una técnica que requiere de mucho tiempo ya que existen múltiples combinaciones y encontrar el valor óptimo de cada parámetro de forma manual es complejo.

  Python dispone de la función **Grid Search** que permite automatizar la búsqueda de los parámetros óptimos. Para usar esta función, el usuario debe especificar un rango de valores para cada parámetro, y Grid Search evalúa el modelo creado para cada combinación de valores.

- **Técnicas Out of Bag y Cross-Validation:** Cuando se realiza una prueba para un conjunto específico de valores, es necesario dividir el Data Set en dos partes: una para la fase de entrenamiento y otra para la fase de test. Los resultados obtenidos pueden variar significativamente en función de esta división, y por ello es necesario realizar distintas pruebas variando las muestras usadas en cada fase para validar un modelo.

  La **técnica de Cross Validation** utiliza K-Fold, que consiste en la división de datos en K conjuntos, donde el K-1 conjuntos se utilizan para entrenamiento y el restante para la validación del modelo. K determina las veces que se entrena y prueba o valida el modelo, donde cada vez se utiliza un nuevo conjunto de pruebas y de entrenamiento. Una vez se completan todas las combinaciones de conjuntos, el resultado de la

validación cruzada con K-Fold es el promedio de los resultados obtenidos en cada serie.

Otra técnica que puede usarse para validar un modelo cuando el algoritmo usado es Random Forest, es la denominada **Out Of Bag.** Tal y como se ha explicado en el apartado 3.3.4, Random Forest utiliza diversos árboles de decisión y para cada uno de ellos se escoge una muestra aleatoria del conjunto de datos y se entrena de forma independiente generando un resultado. Aquellas muestras que no se han seleccionado para entrenar un árbol determinado, se les denomina Out Of Bag y son las que se utilizan para la validación del modelo obtenido mediante ese árbol de decisión. Esta técnica es especialmente útil cuando el Data Set es pequeño y no hay suficientes muestras como para poder aplicar Cross Validation.

- **Errores en regresión:** Los errores en regresión permiten identificar las carencias del modelo de predicción creado y poder así optimizarlo manualmente o mediante diferentes técnicas. Los errores más importantes a tener en cuenta al trabajar con algoritmos de regresión son el *RMSE, MAE* y R^2.

  El *RMSE (Root Mean Square Error)* o raíz cuadrada del error cuadrático medio, compara un valor predicho con el valor real de un conjunto de datos y se calcula mediante la siguiente fórmula:

$$RMSE = \sqrt{\sum_{i=1}^{n} \frac{(\hat{y}_i - y_i)^2}{n}} \qquad \textbf{(4.1)}$$

  La variable $\hat{y}_i$ hace referencia a los valores predichos, la variable y_i hace referencia a los valores reales y la variable n hace referencia al número de observaciones.

  Contra más pequeño es el valor del RMSE, menor es la diferencia entre los valores reales y predichos y por tanto, mejores son las predicciones del modelo. Aún así, es necesario tener en cuenta que la valoración del valor obtenido de este error depende mucho del Data Set.

  El *MAE (Mean absolute error)* o error absoluto medio, calcula el valor promedio de la diferencia entre el valor predicho y el valor real. Es un error complementario al RMSE que también permite valorar la precisión de un modelo de regresión. La fórmula para calcular el MAE se muestra a continuación:

$$MAE = \frac{\sum_i |y_i - \hat{y}_i|}{n} \qquad \textbf{(4.2)}$$

La variable $\hat{y}_i$ hace referencia a los valores predichos, la variable y_i hace referencia a los valores reales y la variable *n* hace referencia al número de observaciones.

Finalmente, el R^2 *(Coefficient of Determination)* o coeficiente de determinación, es una medida que muestra cómo se ajustan los datos a un modelo. Toma valores entre 0 y 1, por lo que contra más cercano esté del 0, menos se ajustan los datos al modelo y, en este caso, más errores de predicción hay. En el caso contrario, los datos se ajustan mejor al modelo y los errores de predicción se reducen a medida que el coeficiente de determinación incrementa. A la hora de utilizar esta medida en Python, hay ocasiones en las que el coeficiente de determinación da negativo para indicar que los datos no se ajustan bien al modelo.

Es necesario que se complemente esta medida con los dos errores anteriores para poder evaluar correctamente un modelo.

El coeficiente de determinación se calcula de la siguiente forma:

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(\hat{y}_i - y_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y}_i)^2} \quad \textbf{(4.3)}$$

La variable $\hat{y}_i$ hace referencia a los valores predichos, la variable y_i hace referencia a los valores reales, la variable $\bar{y}_i$ es el valor medio y *n* es la variable que hace referencia al número de observaciones.

4.2. Resultados con RandomForestRegressor

4.2.1. División de datos en conjunto de entrenamiento y test

Tras identificar los atributos con un valor de correlación elevado respecto “y”, se han dividido los datos en variables independientes y variable dependiente.

Por un lado, las variables independientes son las que se han utilizado para crear el modelo de predicción, que son aquellas que tienen el valor de correlación más alto con “y”: “open”, “high”, “low” y “close”.

Por otro lado, la variable dependiente es la variable objetivo, en este caso ha sido el precio predicho de la vela del día siguiente “y”.

A partir de ambos conjuntos de datos, se ha llevado a cabo la definición de los datos de entrenamiento y los datos de test. Para ello, se ha utilizado la función de Python train_test_split() para crear los dos conjuntos de datos de entrenamiento y test. Tal y como se ha mencionado en el apartado 4.1.3, lo más común es establecer un conjunto de entrenamiento del 80 % y un conjunto de test del 20 %. Aun así, es necesario variar los porcentajes para evaluar con qué valores se obtiene menos error.

Con el objetivo de identificar el rendimiento del modelo inicial, se han evaluado los errores en regresión: el *RMSE*, R^2 y *MAE* tanto en el subconjunto de entrenamiento como en el subconjunto de test. Para estas pruebas iniciales, se han usado los parámetros por defecto del algoritmo Random Forest que se muestran en la Tabla 4.1. En la Tabla 4.2, se muestran los resultados obtenidos al variar los % de ambos conjuntos en relación con el error obtenido al crear el modelo inicial con la clase de Python RandomForestRegressor().

Tabla 4.1. Tabla con los valores por defecto de los parámetros del modelo inicial.

Parámetros	Valores
n_estimators	100
Criterion	squared_error
max_depth	None
min_samples_split	2
min_samples_leaf	1
max_features	1
max_leaf_nodes	None
oob_score	False
n_jobs	None
random_state	None
max_samples	None

Tabla 4.2. Tabla con los resultados obtenidos al variar los % de entrenamiento y test.

% Training/ %Test	RMSE Training	RMSE Test	MAE Training	MAE Test	R^2 Training	R^2 Test
60/40	172.18	24161.64	105.37	19388.56	0.995987	-1.007979
70/30	166.74	27849.46	103.61	25678.08	0.996888	-5.617281
80/20	341.18	2483.54	168.68	1896.35	0.999296	0.914602

Tal y como se observa en la Tabla 4.2, se han obtenido los mejores valores del RMSE, MAE y R^2 cuando el % del conjunto de entrenamiento es 80 % y del conjunto de test es 20%, por lo que se ha iniciado la creación del modelo con esta combinación. Con el resto de valores, se puede observar como el R^2 de test da negativo, siendo esto un indicativo de que la división de datos no se ajusta correctamente al modelo.

Tanto el RMSE como el MAE del conjunto de entrenamiento no son muy elevados y el R^2 es muy cercano a 1, lo que indica que puede ser un buen conjunto de entrenamiento del modelo para realizar las predicciones del precio de Bitcoin posteriormente. En cuanto al conjunto de test, tanto el RMSE como el MAE son considerablemente más elevados y el R^2 es un poco más bajo que en el conjunto de entrenamiento. Esto es debido a que en el conjunto de test el precio de Bitcoin estaba entre los 30000 $ y 69000 $ aproximadamente, mientras que en el conjunto de entrenamiento el precio de Bitcoin estaba entre los 4000 $ y 65000 $.

Es necesario destacar que el 80% del Data Set destinado al conjunto de entrenamiento abarca un periodo de tiempo que va desde el 18/08/2017 hasta el 12/05/2021, mientras que el 20% restante destinado al conjunto de test va desde el 13/05/2021 hasta el 19/04/2022.

Para comprender los resultados anteriores y la variación en los valores del RMSE y el MAE de ambos conjuntos, se ha desarrollado una función que permite identificar cuando la diferencia en valor absoluto entre los valores reales y predichos es muy elevada, es decir, qué muestras son las que contribuyen en mayor medida a que los errores RMSE y MAE aumenten significativamente.

La Figura 4.4 y la Figura 4.5, muestran esta diferencia calculada sobre el conjunto de datos de entrenamiento y el de test.

- **Conjunto de entrenamiento:**

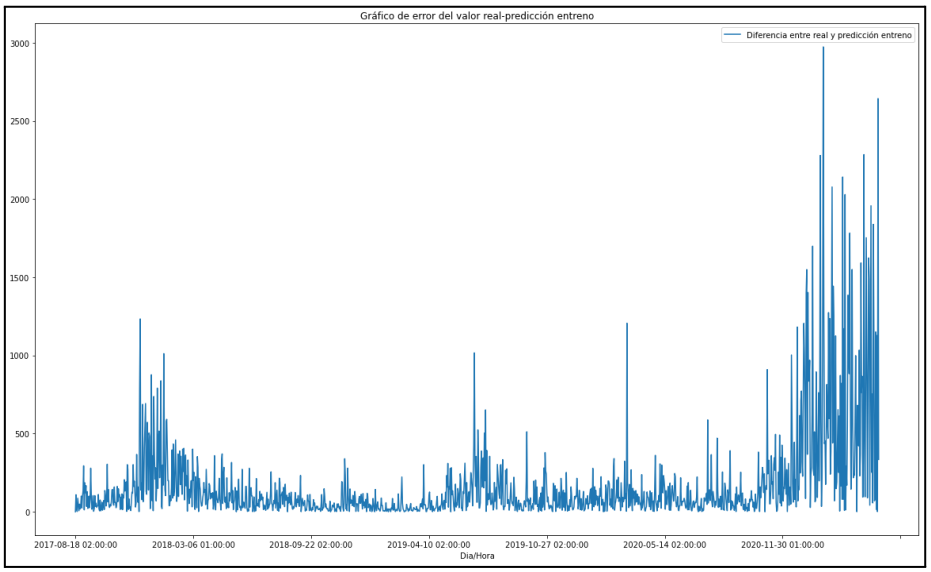

Fig. 4.4 Gráfica con la diferencia entre datos predichos y reales de entrenamiento.

En la siguiente tabla, se muestran los valores más importantes extraídos de la figura anterior:

Tabla 4.3. Tabla con los valores más importantes de la Figura 4.4.

Valor medio	Valor mínimo	Valor máximo	Mediana
168.67	0.018200	2974.26	73.11

A partir de los resultados anteriores, se puede detectar que hay algunos picos altos entre diciembre de 2017 y marzo de 2018 con valores que superan los 500 Tethers (USDT) y entre inicios de 2021 y abril de 2022 con valores que superan los 1000 Tethers, llegando incluso a un valor máximo de 2974.26 Tethers de diferencia entre los valores predichos y los reales. Justamente, este periodo de tiempo coincide tanto con las publicaciones de Elon Musk anunciando la aceptación de Bitcoin para la compra de sus vehículos, que posteriormente rectificó con una nueva publicación declarando que ya no sería posible comprarlos con esta criptomoneda. Estos hechos hicieron que el precio del Bitcoin se volviera mucho más volátil en ese periodo de tiempo.

- **Conjunto de test:**

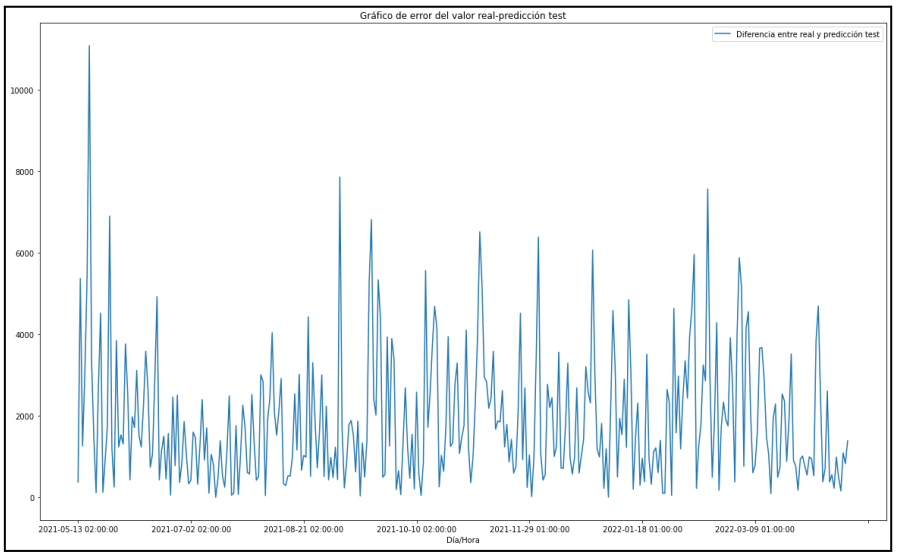

Fig. 4.5 Gráfica con la diferencia entre datos predichos y reales de test.

Los valores más destacados que se identifican en la Figura 4.5, se exponen a continuación:

Tabla 4.4. Tabla con los valores más importantes de la Figura 4.5.

Valor medio	Valor mínimo	Valor máximo	Mediana
1896.36	0.771900	11079.79	1459.34

En este caso, los picos de diferencia son más elevados que en el conjunto de entrenamiento, ya que en este conjunto de test los valores del precio de Bitcoin son considerablemente más elevados que en el conjunto de entrenamiento. Durante el periodo de test sucedieron eventos ya mencionados en el apartado 2.3., que provocaron un fuerte impacto negativo en el precio del Bitcoin, como la ilegalización del uso de las criptomonedas en China.

Llegados a este punto, se planteó considerar como *outliers* y eliminar aquellas muestras que superaran un valor medio de diferencia entre el valor predicho y real en concreto con el objetivo de reducir los errores RMSE, MAE y obtener un R^2 más elevado. Finalmente, se decidió no hacerlo, ya que se estarían eliminando datos de relevancia.

Las variaciones bruscas del precio pueden ayudar en un futuro si se implementara un sistema más sofisticado que incluya otro tipo de variables para realizar predicciones del precio del Bitcoin. Por lo tanto, se decidió trabajar con todos los datos.

4.2.2. Influencia de los parámetros del modelo

Analizados los errores en regresión del modelo inicial, se ha llevado a cabo la modificación manual de los parámetros más importantes para evaluar la influencia de cada uno de ellos en el modelo. Cuando se ha modificado el valor de un parámetro en concreto, los demás se han dejado con su valor por defecto.

- **Influencia del parámetro n_estimators**

  Tal y como se ha descrito anteriormente, este parámetro indica el número de árboles que contiene el algoritmo. Contra más árboles se introduzcan, más puede optimizarse el modelo, pero normalmente hay un límite donde el algoritmo empieza a empeorar si se sigue aumentando este número. Por este motivo, se va a ir incrementando el valor de n_estimators para comprobar su efecto.

  Se han llevado a cabo 3 modificaciones del valor de n_estimators (el valor por defecto es 100): 150, 200 y finalmente 400. En la siguiente tabla, se muestran los errores obtenidos al realizar estas modificaciones:

Tabla 4.5. Tabla con los resultados de los errores con los valores de 100, 150, 200 y 400 n_estimators.

n_ estimators	RMSE Training	RMSE Test	MAE Training	MAE Test	R^2 Training	R^2 Test
100	343.99	2422.14	170.12	1840.43	0.999284	0.918773
150	345.27	2425.08	169.77	1837.37	0.999278	0.918575
200	337.53	2424.49	167.98	1848.08	0.999310	0.918615
400	339.22	2476.38	168.50	1885.85	0.999303	0.915094

En primer lugar, para el caso de n_estimators = 150, el RMSE y R^2 de ambos conjuntos han empeorado y ambos MAE han mejorado ligeramente respecto los resultados con el valor por defecto.

En segundo lugar, se ha incrementado el valor del número de árboles a 200, obteniendo mejores resultados de RMSE y R^2 de ambos conjuntos en comparación con los resultados anteriores con 150 árboles excepto el MAE de test que es mayor.

Finalmente, en el caso de 400 árboles se puede observar cómo los errores de ambos conjuntos comienzan a empeorar.

Además, cuanto más se incrementa el número de árboles, mayor es el tiempo de entrenamiento de los datos, lo que conlleva tiempos de espera elevados. Por este motivo, es conveniente encontrar el mejor valor posible del número de árboles mediante técnicas que se han expuesto en el apartado 4.1.3, y que se mostrarán posteriormente.

- **Influencia del parámetro max_depth**

  Como se ha explicado anteriormente, el parámetro max_depth indica la profundidad máxima del árbol. Cuanta más profundidad, más divisiones en los árboles se crean y se consigue captar más información.

  Se han realizado 3 modificaciones del valor del parámetro max_depth (el valor por defecto es None): 50, 100 y 200. A continuación, se presenta el valor de los errores obtenidos con estas modificaciones:

Tabla 4.6. Tabla con los resultados de los errores con los valores de None, 50, 100 y 200 max_depth.

max_ depth	RMSE Training	RMSE Test	MAE Training	MAE Test	R^2 Training	R^2 Test
None	339.78	2454.16	169.40	1888.74	0.999301	0.916611
50	346.55	2444.45	171.32	1860.53	0.999273	0.917270
100	340.12	2451.26	170.70	1877.60	0.999300	0.916808
200	352.31	2510.22	171.29	1893.94	0.999249	0.912757

Como se puede observar en la tabla anterior, cuando el parámetro tiene un valor de 50, los errores del conjunto de entrenamiento son mayores que con el valor por defecto, por lo que el RMSE y MAE son mayores y el R^2 más pequeño. En cambio, los errores del conjunto de test han mejorado ligeramente obteniendo un RMSE y MAE menores y un R^2 mayor.

Cuando el valor del max_depth es 100, el RMSE, MAE y R^2 del conjunto de entrenamiento mejoran ligeramente respecto los anteriores resultados con el valor de 50, a diferencia del conjunto de test que tanto el RMSE y MAE como el R^2, han empeorado.

Por último, se ha incrementado la profundidad máxima considerablemente hasta un valor de 200. El RMSE, MAE y R^2 de ambos conjuntos ha empeorado, por lo que se puede deducir que a partir de este valor hay una tendencia a que los errores de ambos conjuntos se incrementen. En general no es conveniente introducir un

valor muy grande, ya que el tiempo de entrenamiento aumenta contra más grande sea el valor de la profundidad máxima.

- **Influencia del parámetro min_samples_split**

  El parámetro min_samples_split, indica el número mínimo de muestras requeridas para dividir un nodo interno. Se ha incrementado su valor (el valor por defecto es 2) a 10, 15 y 20 para analizar la influencia de estas modificaciones en los errores del modelo. En la siguiente tabla, se muestran los errores obtenidos al realizar estas modificaciones:

Tabla 4.7. Tabla con los resultados de los errores con los valores de 2, 10, 15 y 20 min_samples_split.

min_ samples_ split	RMSE Training	RMSE Test	MAE Training	MAE Test	R^2 Training	R^2 Test
2	345.61	2422.21	170.57	1821.06	0.999277	0.918768
10	547.49	2222.90	269.73	1675.43	0.998186	0.931587
15	626.43	2208.75	306.30	1676.75	0.997625	0.932454
20	666.85	2372.23	321.10	1837.35	0.997309	0.922086

Tal y como se puede observar en la tabla anterior, a medida que se ha ido incrementando el valor del parámetro, los errores RMSE y MAE de entrenamiento han ido aumentado considerablemente y el R^2 ha empeorado. En cambio, en el caso del conjunto de test, los valores de los errores RMSE y MAE y el R^2 han mejorado notablemente respecto el modelo inicial. Aún así, a partir del valor 20, el conjunto de test empieza a tomar una tendencia de empeoramiento del RMSE, MAE y R^2.

A partir de los resultados anteriores, se puede concluir que al incrementar el valor del parámetro min_samples_split, mejora el error del conjunto de test hasta el valor de 20, donde comienzan a incrementarse los errores. Adicionalmente, a medida que el valor del parámetro es mayor, los errores del conjunto de entrenamiento crecen notablemente.

- **Influencia del parámetro min_samples_leaf**

  Tal y como se ha explicado anteriormente, el parámetro min_samples_leaf indica el número mínimo de muestras que debe haber en un nodo hoja o nodo final.

  Con el objetivo de evaluar su influencia en los errores del modelo inicial, se ha llevado a cabo la modificación del parámetro incrementándolo (el

valor por defecto es 1) a 5, 10 y 15. Los resultados obtenidos son los siguientes:

Tabla 4.8. Tabla con los resultados de los errores con los valores de 1, 5, 10 y 15 min_samples_leaf.

min_ samples _leaf	RMSE Training	RMSE Test	MAE Training	MAE Test	R^2 Training	R^2 Test
1	351.56	2463.53	171.01	1882.23	0.999252	0.915973
5	623.62	2205.26	306.97	1656.22	0.997646	0.932668
10	715.58	2488.26	351.58	1904.88	0.996901	0.914278
15	621.28	2183.14	303.68	1662.38	0.997664	0.934012

Los resultados obtenidos con los valores 5 y 15 mejoran considerablemente el RMSE, MAE y R^2 del conjunto de test, mientras que los resultados del conjunto de entrenamiento empeoran. Con el valor 10, empeoran ambos conjuntos en comparación con los demás valores. Por este motivo es necesario el uso de técnicas que permitan encontrar el valor óptimo, las cuales se mostrarán posteriormente.

- **Influencia del parámetro max_leaf_nodes**

  El parámetro de max_leaf_nodes hace referencia al número máximo de nodos hoja o nodos finales. Se ha incrementado su valor (su valor por defecto es None) a 5, 50 y 100 para analizar cómo afectan estas modificaciones a los errores del modelo. En la siguiente tabla, se muestran los errores obtenidos al realizar estas modificaciones:

Tabla 4.9. Tabla con los resultados de los errores con los valores de None, 5, 50 y 100 max_leaf_nodes.

max_ leaf_ nodes	RMSE Training	RMSE Test	MAE Training	MAE Test	R^2 Training	R^2 Test
None	346.04	2489.39	169.87	1872.55	0.999275	0.914199
5	1767.41	5331.87	1176.12	4557.30	0.981095	0.606395
50	493.93	2408.60	297.55	1857.17	0.998523	0.919678
100	415.74	2491.67	256.35	1896.14	0.998954	0.914043

Como se puede observar en la tabla, tanto el RMSE, MAE y R^2 del conjunto de entrenamiento como de test han mejorado muchísimo al incrementar el valor de max_leaf_nodes a 50. En cambio, con el valor 100 vuelve a mejorar el RMSE, MAE y R^2 del conjunto de entrenamiento, pero empeoran de nuevo los errores y R^2 del conjunto de test.

No conviene incrementar considerablemente el valor del parámetro, ya que los tiempos de entrenamiento son mayores a medida que el valor es más grande y dificulta el desarrollo del modelo.

- **Influencia del parámetro max_samples**

El parámetro max_samples, indica la cantidad de muestras de un conjunto de datos que se destinan a un árbol en particular. En nuestro caso, el máximo número de muestras posible es 1364 ya que representa el 80 % del conjunto de datos original, que es el conjunto de entrenamiento. Por este motivo, mientras el valor sea más pequeño que 1364, el error del conjunto de entrenamiento incrementará a medida que se reduzca el valor.

Para analizar los errores, se ha realizado la modificación de los valores del parámetro (su valor por defecto es None, que equivale a 1364 muestras) a 1000 y 500 respectivamente. A continuación, se muestran los resultados obtenidos:

Tabla 4.10. Tabla con los resultados de los errores con los valores de None, 1000 y 500 max_samples.

max_ samples	RMSE Training	RMSE Test	MAE Training	MAE Test	R^2 Training	R^2 Test
None	340.86	2397.91	169.30	1835.83	0.999297	0.920390
1000	430.51	2337.44	214.49	1769.49	0.998878	0.924354
500	591.49	2240.30	294.43	1715.35	0.997883	0.930511

Tal y como se ha mencionado anteriormente, a medida que se reduce el valor de max_samples, los valores de los errores y el R^2 del conjunto de entrenamiento empeoran respecto el modelo inicial.

- **Influencia del parámetro max_features**

El parámetro max_features, indica el número de atributos del Data Set que se toman en consideración para encontrar el mejor resultado. Al tener 4 atributos, se ha realizado la modificación de los valores del parámetro (el valor por defecto es 1) a 2, 3 y 4. Los resultados obtenidos se exponen en la siguiente tabla:

Tabla 4.11. Tabla con los resultados de los errores con los valores de 1, 2, 3 y 4 de max_features.

max_ features	RMSE Training	RMSE Test	MAE Training	MAE Test	R^2 Training	R^2 Test
1	337.60	2504.46	167.45	1906.59	0.999310	0.913158
2	327.70	2461.40	163.95	1858.94	0.999350	0.916119
3	340.39	2382.09	167.29	1828.04	0.999299	0.921437
4	343.60	2404.44	170.48	1823.65	0.999285	0.919956

Como se puede observar en la tabla anterior, con los valores de 2, 3 y 4 el RMSE y MAE del conjunto de entrenamiento incrementan consecutivamente y el R^2 va empeorando. Si se compara con el valor por defecto, el RMSE, MAE y R^2 de entrenamiento con el valor de 2 es mejor, pero en cambio los resultados de entrenamiento con los valores de 3 y 4 son peores que el valor por defecto.

En cuanto al conjunto de test, a medida que crece el valor de max_features mejoran lo resultados del RMSE, MAE y R^2, pero con el valor de 4 empeora ligeramente. Por este motivo, es necesario utilizar técnicas que permitan encontrar el valor óptimo del parámetro para crear el mejor modelo posible.

Expuestos todos los resultados, se ha identificado la influencia de los parámetros más importantes que afectan al modelo de predicción. Adicionalmente, este análisis inicial de parámetros ha permitido identificar el posible rango de valores óptimo para cada uno de ellos.

En el siguiente apartado, se va a realizar la búsqueda de los parámetros óptimos utilizando dos de las técnicas anteriormente expuestas: Validación con Out Of Bag y Cross Validation.

4.2.3. Parameter tuning: Búsqueda de los valores óptimos de los parámetros del modelo

4.2.3.1. Valores óptimos de cada parámetro utilizando Out of Bag y Cross Validation

En el presente apartado, se van a exponer los resultados de los valores óptimos de cada parámetro para el modelo de predicción mediante el uso de las técnicas de Out Of Bag y Cross Validation. Con el uso de ambas técnicas, es posible realizar diferentes evaluaciones para cada uno de los parámetros y optimizar la búsqueda de sus valores óptimos.

En primer lugar, se ha definido un rango de valores a partir del cual se ha entrenado el modelo de predicción utilizando la técnica de Out Of Bag. Posteriormente, se ha evaluado el valor de cada parámetro óptimo relacionando cada valor del rango con el coeficiente de determinación o R^2. Por lo tanto, para cada prueba realizada, el valor del parámetro con mayor R^2 de Out Of Bag es el óptimo.

En segundo lugar, se ha utilizado la técnica de Cross Validation con K = 5 y se ha definido un rango de valores para cada parámetro a partir del cual se ha entrenado el modelo de predicción. Con el objetivo de identificar el valor óptimo de cada parámetro, se han evaluado los resultados relacionando el RMSE utilizando Cross Validation con cada uno de los valores del rango establecido para cada parámetro. En consecuencia, el valor con menor RMSE de Cross Validation es el óptimo.

A continuación, se van a mostrar los resultados de los valores óptimos de cada parámetro mediante el uso de ambas técnicas.

- **N_estimators**

  El rango de valores de n_estimators establecido ha sido de 1 a 400. Entre estos valores se han encontrado los óptimos en cada una de las pruebas realizadas mediante Out Of Bag y Cross Validation. Los resultados obtenidos con cada técnica se muestran en las Tablas 4.12 y 4.13.

Tabla 4.12. Tabla con los valores óptimos de n_estimators utilizando Out Of Bag.

Pruebas	n_estimators	R^2 Training	R^2 Out Of Bag
Primera prueba	92	0.999334	0.995073
Segunda prueba	114	0.999321	0.995159
Tercera prueba	86	0.999285	0.995088
Cuarta prueba	322	0.999311	0.995103

Para complementar los resultados anteriores, se muestra en la Figura 4.6, una gráfica con la evolución del R^2 de test de Out Of Bag en función de los valores de n_estimators**.** Tal y como se puede observar, a partir de un valor de 10 aproximadamente, el R^2 comienza a estabilizarse obteniendo los mejores resultados a partir de las pruebas expuestas en la tabla anterior.

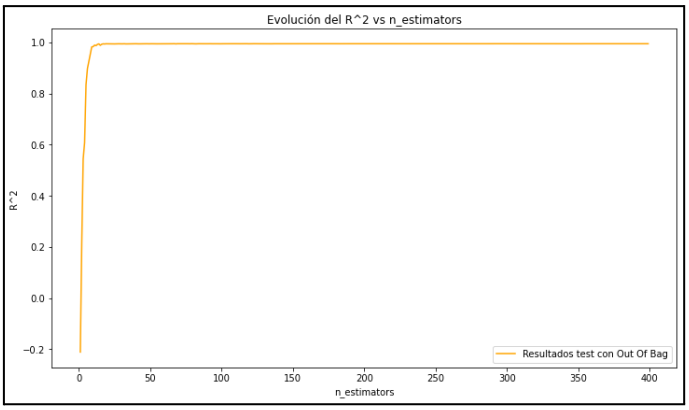

Fig. 4.6 Gráfica con la evolución del R^2 respecto el valor de n_estimators.

Tabla 4.13. Tabla con los valores óptimos de n_estimators utilizando Cross Validation.

Pruebas	n_estimators	RMSE Training	RMSE Cross Validation
Primera prueba	43	347.53	2835.27
Segunda prueba	7	443.41	2829.93
Tercera prueba	157	343.32	2837.80
Cuarta prueba	93	351.07	2837.67

De la misma forma que con Out Of Bag, se expone en la Figura 4.7 la evolución del RMSE de test con Cross Validation en función de los valores de n_estimators. En esta figura, se puede apreciar cómo se va reduciendo el RMSE a medida que aumenta el valor de n_estimators. Los mejores resultados de RMSE se obtienen con los valores de 7 y 93 n_estimators, tal y como se ha expuesto en las pruebas de la tabla anterior.

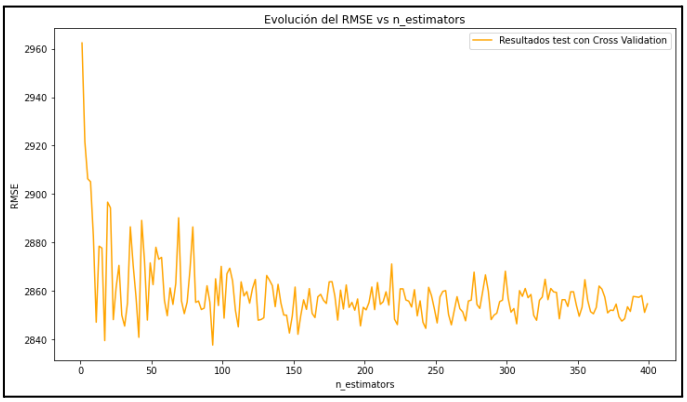

Fig. 4.7 Gráfica con la evolución del RMSE respecto el valor de n_estimators.

- **Max_depth**

  Para el presente parámetro, se ha utilizado un rango de valores entre 1 y 200. A continuación, se muestran las tablas con los valores óptimos obtenidos en cada una de las pruebas con ambas técnicas:

Tabla 4.14. Tabla con los valores óptimos de max_depth utilizando Out Of Bag.

Pruebas	max_depth	R^2 Training	R^2 Out Of Bag
Primera prueba	5	0.997658	0.995193
Segunda prueba	5	0.997681	0.995294
Tercera prueba	5	0.997668	0.995233
Cuarta prueba	5	0.997657	0.995286

La gráfica con la evolución del R^2 de test con Out Of Bag en función de los valores de max_depth se muestra a continuación:

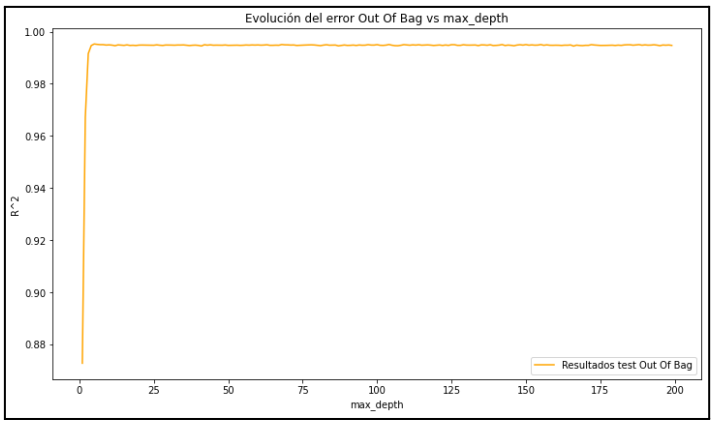

Fig. 4.8 Gráfica con la evolución del R^2 respecto el valor de max_depth.

Tal y como se indica en la Tabla 4.14, el valor óptimo de R^2 se consigue cuando max_depth es 5. En la Figura 4.8, se puede observar como hay un ligero pico antes de estabilizarse el R^2 a medida que incrementa el valor de max_depth.

Tabla 4.15. Tabla con los valores óptimos de max_depth utilizando Cross Validation.

Pruebas	max_depth	RMSE Training	RMSE Cross Validation
Primera prueba	7	479.42	2832.38
Segunda prueba	8	425.86	2832.48
Tercera prueba	7	471.78	2827.06
Cuarta prueba	130	336.23	2832.21

Los valores de RMSE de test con Cross Validation en función del parámetro max_depth se expone en la siguiente figura:

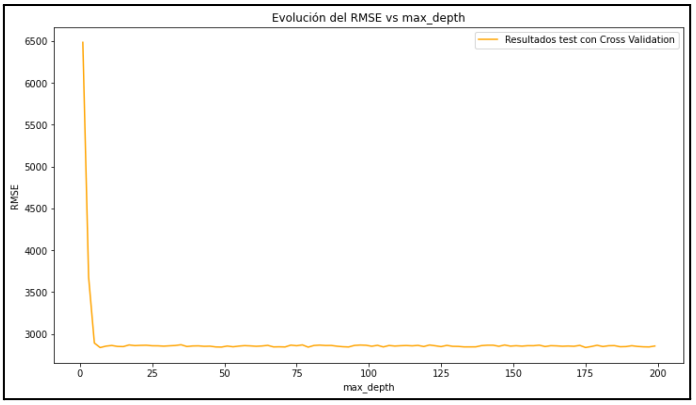

Fig. 4.9 Gráfica con la evolución del RMSE respecto el valor de max_depth.

Al comparar los resultados de las pruebas mostradas en la Tabla 4.15 y la gráfica de la Figura 4.9, se ha identificado como para los valores 7, 8 y 130 de max_depth se obtiene el menor RMSE. Adicionalmente, a partir de un max_depth de 10 aproximadamente, el RMSE se comienza a estabilizar.

- **Min_samples_split**

En el caso de min_samples_split, el rango de valores para cada prueba ha sido de 2 a 50. Los valores óptimos utilizando ambas técnicas se exponen en las Tablas 4.16 y 4.17:

Tabla 4.16. Tabla con los valores óptimos de min_samples_split utilizando Out Of Bag.

Pruebas	min_samples_ split	R^2 Training	R^2 Out Of Bag
Primera prueba	20	0.997379	0.995265
Segunda prueba	17	0.997519	0.995408
Tercera prueba	18	0.997449	0.995332
Cuarta prueba	20	0.997352	0.995272

El R^2 de test con Out Of Bag según el valor de min_samples_split se muestra en la siguiente gráfica:

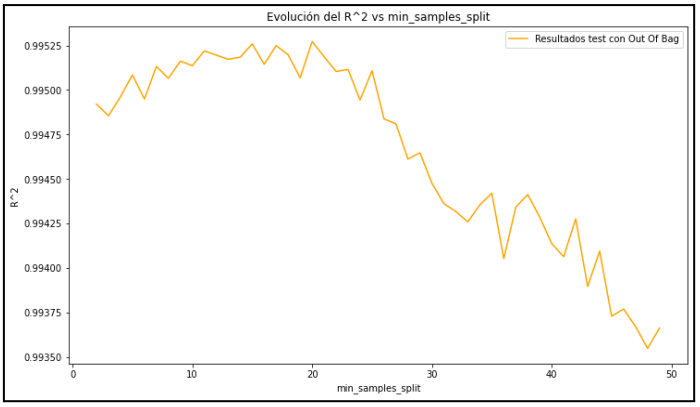

Fig. 4.10 Gráfica con la evolución del R^2 respecto el valor de min_samples_split.

Para los valores de 17, 18 y 20 de min_samples_split, se obtienen los mejores resultados de R^2. A partir de 20, el R^2 comienza a empeorar tal y como se puede observar en la Figura 4.10.

Tabla 4.17. Tabla con los valores óptimos de min_samples_split utilizando Cross Validation.

Pruebas	min_samples_ split	RMSE Training	RMSE Cross Validation
Primera prueba	2	330.14	2844.65
Segunda prueba	2	340.22	2858.70
Tercera prueba	2	343.09	2849.96
Cuarta prueba	3	370.00	2857.65

A partir de la siguiente figura se aprecia la evolución del RMSE de test con Cross Validation según el valor de min_samples_split:

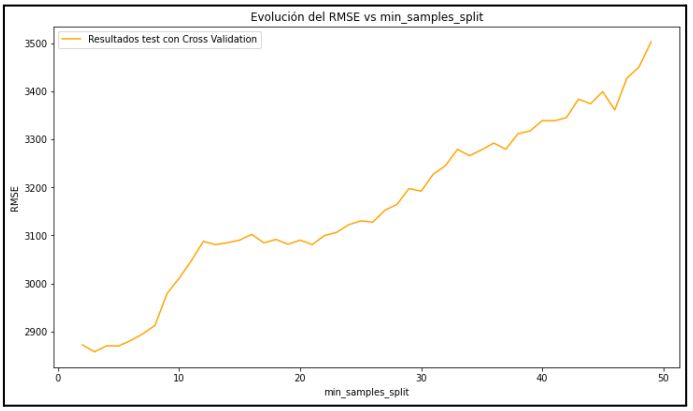

Fig. 4.11 Gráfica con la evolución del RMSE respecto el valor de min_samples_split.

Mediante la Figura 4.11, es posible detectar como para valores pequeños de min_samples_split se obtienen los mejores resultados de RMSE. Si se compara la Tabla 4.17 con esta figura, los valores óptimos son 2 y 3 ya que para valores mayores el RMSE incrementa.

- **Min_samples_leaf**

  El rango de valores establecido para las pruebas realizadas ha sido de 1 a 100. Los resultados de los valores óptimos se muestran a continuación:

Tabla 4.18. Tabla con los valores óptimos de min_samples_leaf utilizando Out Of Bag.

Pruebas	min_samples_ leaf	R^2 Training	R^2 Out Of Bag
Primera prueba	5	0.997680	0.995499
Segunda prueba	7	0.997304	0.995429
Tercera prueba	5	0.997668	0.995437
Cuarta prueba	6	0.997468	0.995388

El valor del R^2 de test con Out Of Bag a medida que incrementa el valor de min_samples_leaf se muestra a continuación:

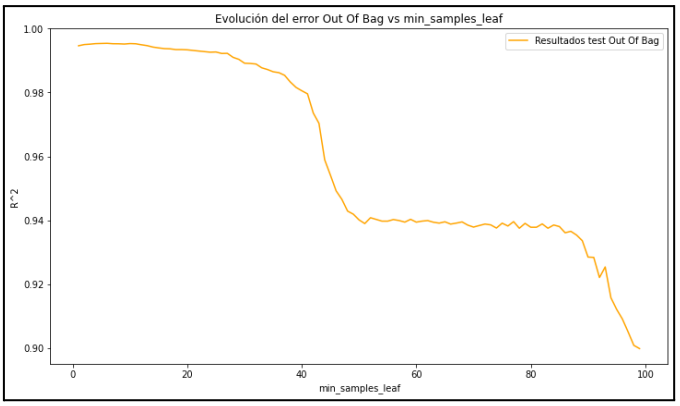

Fig. 4.12 Gráfica con la evolución del R^2 respecto el valor de min_samples_leaf.

Como se puede apreciar en la Figura 4.12, a medida que aumenta el valor de min_samples_leaf, el R^2 va disminuyendo. Al comparar la figura anterior con los resultados obtenidos en cada prueba de la Tabla 4.18, se identifican como los valores óptimos son 5, 6 y 7.

Tabla 4.19. Tabla con los valores óptimos de min_samples_leaf utilizando Cross Validation.

Pruebas	min_samples_ leaf	RMSE Training	RMSE Cross Validation
Primera prueba	1	338.25	2868.49
Segunda prueba	1	341.64	2852.61
Tercera prueba	1	339.94	2859.01
Cuarta prueba	2	454.28	2876.40

La evolución del RMSE según el valor de min_samples_leaf se expone en la Figura 4.13. A partir de esta figura, se aprecia cómo a medida que el valor de min_samples_leaf se incrementa, el RMSE cada vez es mayor, por lo que los valores óptimos deberían ser pequeños. Si se observa la anterior, los resultados con menor RMSE son 1 y 2, tal y como se puede observar en la Figura 4.13.

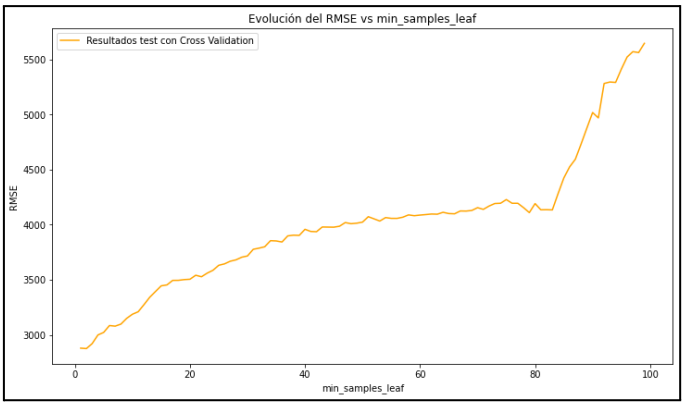

Fig. 4.13 Gráfica con la evolución del RMSE respecto el valor de min_samples_leaf.

- **Max_leaf_nodes**

  En cada una de las pruebas realizadas, el rango de valores utilizado para el presente parámetro ha sido de 2 a 200. En las siguientes tablas, se exponen los resultados obtenidos:

Tabla 4.20. Tabla con los valores óptimos de max_leaf_nodes utilizando Out Of Bag.

Pruebas	max_leaf_ nodes	R^2 Training	R^2 Out Of Bag
Primera prueba	26	0.997704	0.995359
Segunda prueba	47	0.998496	0.995304
Tercera prueba	21	0.997352	0.995314
Cuarta prueba	58	0.998681	0.995227

Adicionalmente, se muestran las variaciones del R^2 de test con Out Of Bag según el valor de max_leaf_nodes en la Figura 4.14. Como se puede observar, el R^2 se estabiliza a partir de un valor de 20 de max_leaf_nodes. A partir de la tabla anterior, es posible comparar los resultados obtenidos de cada una de las pruebas con la Figura 4.14 y comprobar que a partir de 20 max_leaf_nodes se obtienen los valores con menor R^2: 21, 26, 47 y 58.

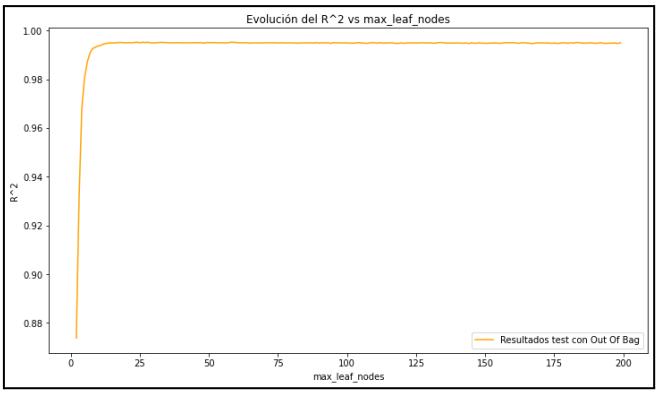

Fig. 4.14 Gráfica con la evolución del R^2 respecto el valor de max_leaf_nodes.

Tabla 4.21. Tabla con los valores óptimos de max_leaf_nodes utilizando Cross Validation.

Pruebas	max_leaf_ nodes	RMSE Training	RMSE Cross Validation
Primera prueba	146	385.73	2831.79
Segunda prueba	176	370.16	2832.30
Tercera prueba	115	403.47	2829.47
Cuarta prueba	163	374.95	2832.32

Mediante los resultados anteriores, se ha creado una gráfica que relaciona el RMSE de test Con Cross Validation con cada valor del rango establecido para el parámetro max_leaf_nodes:

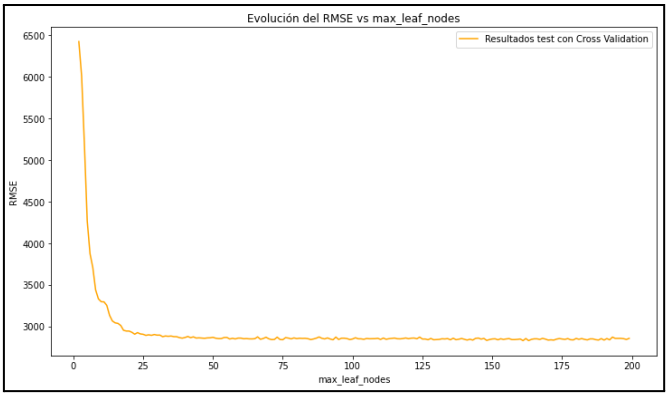

Fig. 4.15 Gráfica con la evolución del RMSE respecto el valor de max_leaf_nodes.

A partir de la Figura 4.15 y de la Tabla 4.21, se puede identificar que los valores óptimos del parámetro se pueden encontrar a partir de 50

max_leaf_nodes aproximadamente, ya que es cuando el RMSE comienza a estabilizarse. Los valores óptimos del parámetro con menor RMSE son 115, 146, 163 y 176.

- **Max_samples**

  Para el parámetro max_samples se ha utilizado un rango de valores entre 1 y 1364. Los valores óptimos del parámetro se muestran en la Tabla 4.22 y en la Tabla 4.23:

Tabla 4.22. Tabla con los valores óptimos de max_samples utilizando Out Of Bag.

Pruebas	max_samples	R^2 Training	R^2 Out Of Bag
Primera prueba	231	0.996982	0.995834
Segunda prueba	281	0.997218	0.995843
Tercera prueba	278	0.997185	0.995782
Cuarta prueba	452	0.997787	0.995806

El R^2 de test con Out Of Bag en función de los valores de max_samples se expone en la siguiente figura:

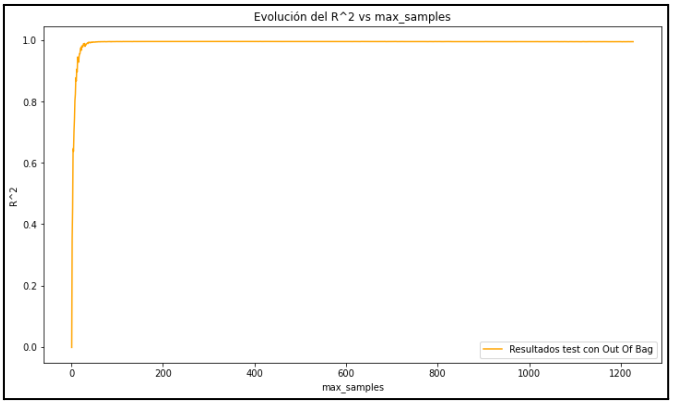

Fig. 4.16 Gráfica con la evolución del R^2 respecto el valor de max_samples.

Tal como se puede observar en la Figura 4.16, el R^2 se estabiliza rápidamente al incrementarse ligeramente el valor de max_samples, obteniendo como valores óptimos los mostrados en la Tabla 4.22.

Tabla 4.23. Tabla con los valores óptimos de max_samples utilizando Cross Validation.

Pruebas	max_samples	RMSE Training	RMSE Cross Validation
Primera prueba	1023	430.04	2839.63
Segunda prueba	1225	373.64	2837.97
Tercera prueba	1001	411.75	2860.14
Cuarta prueba	1069	419.93	2836.29

Los valores de RMSE de test con Cross Validation en función del parámetro max_samples, se muestra en el siguiente gráfico:

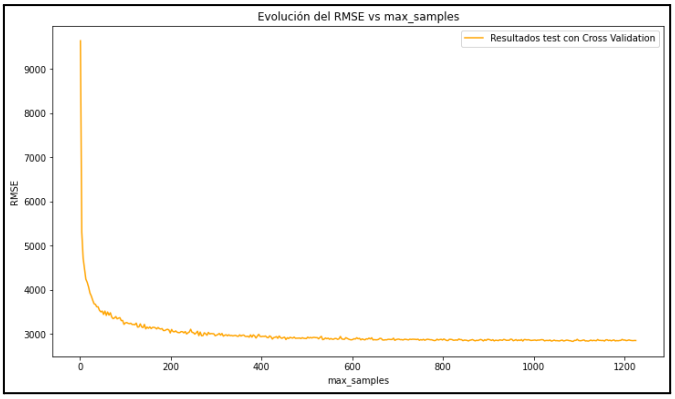

Fig. 4.17 Gráfica con la evolución del RMSE respecto el valor de max_samples.

A medida que incrementa el valor de max_samples se reduce el RMSE. Si comparamos los resultados óptimos mostrados en la Tabla 4.23 y la gráfica de la Figura 4.17, se puede observar cómo a partir de 800 aproximadamente, se estabiliza el RMSE obteniendo como valores óptimos del parámetro 1001, 1023, 1069 y 1225.

- **Max_features**

Por último, para calcular los valores óptimos del parámetro max_features, se ha utilizado un rango de valores entre 1 y 4. En las siguientes tablas, se muestran los resultados obtenidos:

Tabla 4.24. Tabla con los valores óptimos de max_features utilizando Out Of Bag.

Pruebas	max_features	R^2 Training	R^2 Out Of Bag
Primera prueba	1	0.999355	0.995299
Segunda prueba	1	0.999347	0.995198
Tercera prueba	1	0.999314	0.995033
Cuarta prueba	1	0.999318	0.995121

Para complementar los resultados anteriores, a continuación se añade una gráfica que muestra la evolución del R^2 de test con Out Of Bag en función del valor de max_features:

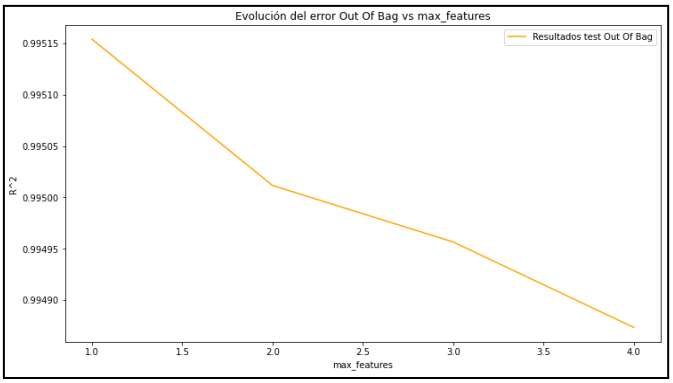

Fig. 4.18 Gráfica con la evolución del R^2 respecto el valor de max_features.

Las 4 pruebas realizadas han dado como resultado que el valor de max_features con mayor R^2 es 1. Al relacionar la Tabla 4.24 con la Figura 4.18, se observa claramente que a partir del valor 1 el R^2 empieza a empeorar.

Tabla 4.25. Tabla con los valores óptimos de max_features utilizando Cross Validation.

Pruebas	max_features	RMSE Training	RMSE Cross Validation
Primera prueba	3	330.89	2845.96
Segunda prueba	2	334.06	2839.22
Tercera prueba	2	340.02	2843.99
Cuarta prueba	2	342.52	2843.05

Seguidamente, se muestra una gráfica con los valores de RMSE en función del parámetro max_features:

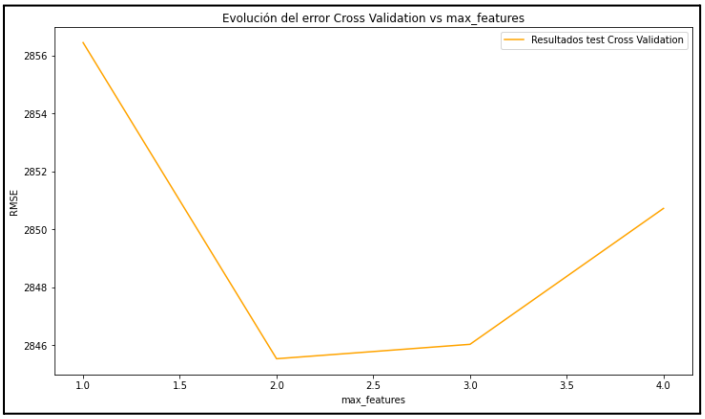

Fig. 4.19 Gráfica con la evolución del RMSE respecto el valor de max_features.

En la Figura 4.19, se puede identificar cómo, a partir de 2 max_features, aumenta considerablemente el RMSE. Si se relaciona la figura anterior con la Tabla 4.25, los resultados de las pruebas concuerdan, ya que las 3 últimas pruebas reflejan mejores resultados con un valor de 2.

A través de las técnicas de Out Of Bag y Cross Validation, se ha conseguido extraer los valores óptimos de cada uno de los parámetros del modelo creado. Para crear un modelo de predicción eficiente, es necesario que se tengan en cuenta los valores óptimos de cada parámetro al mismo tiempo y no sólo de forma individual, ya que dependen entre sí. Por este motivo, en el siguiente apartado se han utilizado algunos de los valores óptimos encontrados de cada parámetro y se ha utilizado la técnica de Grid Search con Cross Validation para encontrar la combinación óptima y obtener un modelo de predicción optimizado.

4.2.3.2. Grid Search con Cross Validation para la búsqueda de la combinación óptima de parámetros

Un Grid permite introducir diversos valores a cada parámetro del modelo a partir de los cuales se realizan combinaciones con todos ellos con el objetivo de encontrar aquella que optimice el modelo lo máximo posible.

Es necesario destacar que contra más valores se introduzcan en el Grid, más tiempo de ejecución se requiere para realizar todas las combinaciones. Por lo tanto, para cada uno de los parámetros, se han escogido algunos de los valores óptimos del apartado anterior.

Para ello, se ha utilizado Cross Validation en combinación con Grid Search para que la búsqueda de la combinación óptima de parámetros sea más efectiva.

Se han introducido en el Grid aquellos valores óptimos que mejores resultados de R^2 de Out Of Bag de test y RMSE de Cross Validation de test se hayan obtenido en el apartado anterior, teniendo en cuenta el valor más pequeño posible del parámetro. Esto quiere decir que si se ha obtenido un R^2 o RMSE de test muy similar entre un valor grande y pequeño de un parámetro en concreto, se escogerá el más pequeño ya que permite reducir considerablemente el tiempo de ejecución de cada prueba.

A continuación, se muestra una tabla a modo resumen de los parámetros que se han introducido en el Grid:

Tabla 4.26. Tabla con los valores de cada parámetro introducidos en el Grid Search.

Parámetros	Valores
n_estimators	7, 86, 114
max_depth	5, 7, 8
min_samples_split	2, 3, 17
min_samples_leaf	1, 5, 7
max_leaf_nodes	26, 115
max_samples	281, 1069
max_features	1, 2, 3

Mediante los valores introducidos en el Grid de parámetros se ha determinado la mejor combinación relacionándola con el coeficiente de determinación o R^2 de test.
Grid Search facilita la introducción de la mejor combinación en un modelo de predicción, ya que tiene una función que la escoge automáticamente, llamada “best_estimator_”.

En las siguientes tablas, se muestran los valores de la combinación de parámetros óptima obtenida y los resultados de los errores RMSE, MAE y el R^2 tanto de entrenamiento como de test:

Tabla 4.27. Tabla con los valores de los parámetros de la combinación óptima obtenida.

Parámetros	Valores
n_estimators	114
max_depth	7
min_samples_split	3
min_samples_leaf	1
max_leaf_nodes	115
max_samples	281
max_features	3

Tabla 4.28. Tabla con los resultados obtenidos de la combinación óptima de parámetros encontrada.

RMSE Training	RMSE Test	MAE Training	MAE Test	R^2 Training	R^2 Test
700.29	2208.44	355.41	1675.02	0.997032	0.932474

Como se puede observar en la Tabla 4.28, se obtienen mejores resultados de test en comparación con el modelo inicial creado gracias a la utilización de Grid Search en combinación con Cross Validation. Aunque el RMSE de entrenamiento haya incrementado, se consigue reducir el RMSE del conjunto de test a partir del cuál se realizan las predicciones.

Se concluye por tanto, que se obtiene un modelo en el cuál se ha conseguido reducir los errores del conjunto de test y aumentar su R^2 considerablemente respecto el modelo inicial. A la hora de realizar las predicciones, interesa que los errores de test sean los menores posibles y el coeficiente de determinación lo más alto posible para que los valores predichos sean mucho más cercanos a los valores reales.

En el siguiente apartado, se va a mostrar la predicción del precio de Bitcoin con el modelo final.

4.2.4. Predicción del precio de Bitcoin

Finalmente, se ha realizado la predicción del precio de la vela del día siguiente de Bitcoin y se ha comparado con el precio real mediante la siguiente gráfica:

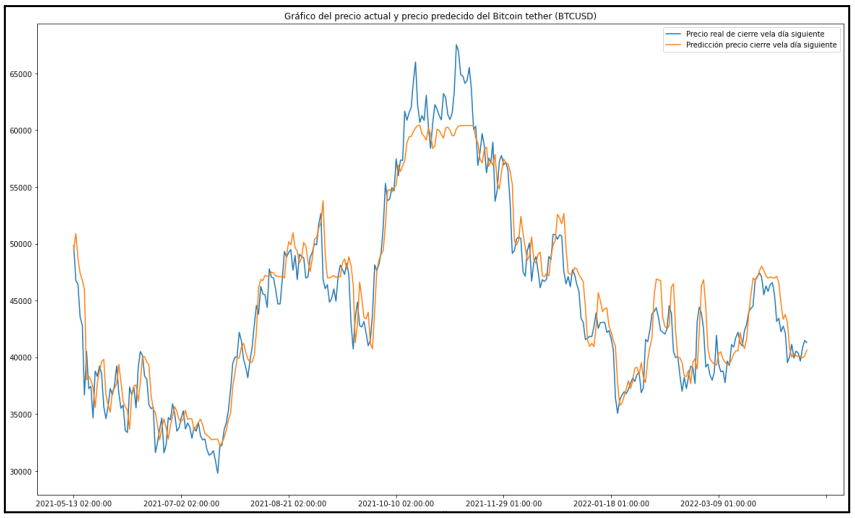

Fig. 4.20 Gráfica del precio diario real y la predicción mediante el modelo final.

Como se puede observar en la figura anterior, el precio real es el color azul y el precio predicho el color naranja. La precisión del modelo de predicción es bastante buena excepto el periodo entre el 10/10/2021 y el 29/11/2021, donde el modelo ha predicho que el precio del Bitcoin apenas variaría pero sucede todo lo contrario.

En conclusión, la predicción de los movimientos del precio de Bitcoin sigue la estructura del precio real, pero hay periodos en los que no acaba de ser del todo precisa debido a que no se tienen en cuenta factores externos como los expuestos en el apartado 2.3.

4.3. Resultados con LinearRegression

Una vez mostrada la creación del modelo de predicción del precio de Bitcoin con Random Forest, se ha utilizado la regresión lineal como algoritmo para crear un nuevo modelo.

4.3.1. División de datos en conjunto de entrenamiento y test

De la misma forma que en el apartado 4.2.1, ha sido necesario realizar la división de datos en el conjunto de entrenamiento y en el conjunto de test. Se han utilizado las mismas variables independientes y variable dependiente que en el caso de Random Forest.

El modelo inicial de regresión lineal creado, utiliza los parámetros por defecto que se muestran en la Tabla 4.29. Para estos valores por defecto, se han llevado a cabo distintas pruebas con el objetivo de escoger el modelo con mejores prestaciones en relación a la división de los datos en conjunto de entrenamiento y test respectivamente. Los resultados de los errores del modelo se exponen en la Tabla 4.30.

Tabla 4.29. Tabla con los valores por defecto de los parámetros del modelo inicial.

Parámetros	Valores
fit_intercept	True
normalize	False
copy_X	True
n_jobs	None
Positive	False

Tabla 4.30. Tabla con los resultados obtenidos al variar los % de entrenamiento y test.

% Training/ %Test	RMSE Training	RMSE Test	MAE Training	MAE Test	R^2 Training	R^2 Test
60/40	403.78	1581.87	240.39	1065.18	0.977931	0.991393
70/30	392.72	1760.02	234.08	1286.30	0.982736	0.973571
80/20	769.44	1676.53	376.70	1234.66	0.996417	0.961084

A partir de los resultados anteriores, se ha escogido el 60 % para el conjunto de entrenamiento y el 40 % para el conjunto de test puesto que el RMSE y MAE de test son inferiores a los otros casos y además se obtiene un R^2 muy cercano a 1.

Destacar que, en este caso, el 60% del Data Set destinado al conjunto de entrenamiento abarca un periodo de tiempo que va desde el 18/08/2017 hasta el 05/06/2020, mientras que el 40% restante destinado al conjunto de test va desde el 06/06/2020 hasta el 19/04/2022.

4.3.2. Análisis de errores mediante la modificación de los parámetros del modelo

Una vez estudiado el modelo inicial, se ha considerado modificar los parámetros para analizar la variación en los errores RMSE, MAE y en el coeficiente de determinación mediante la técnica de Grid Search en combinación con Cross Validation para buscar la mejor combinación de parámetros. Aun así, para el caso del Data Set utilizado, no es necesario variar todos los parámetros de la Regresión lineal.

Los parámetros fit_intercept, n_jobs y positive, sí que se han introducido en el Grid Search ya que son los únicos que interesa variar en este caso para analizar los resultados del RMSE, MAE y R^2 de ambos conjuntos. Se ha decidido no variar los parámetros de normalize y copy_X, ya que en este Data Set no es necesario normalizar los datos puesto que todos los atributos que se han usado tienen el mismo rango de valores. Asimismo, se ha optado por no variar el parámetro copy_X para mantener intactas a la salida las variables de entrada.

En la siguiente tabla, se muestran los valores de los tres parámetros que se introducen en el Grid Search manteniendo el resto por defecto:

Tabla 4.31. Tabla con los valores de cada parámetro introducidos en el Grid Search.

Parámetros	Valores
fit_intercept	False, True
normalize	False
copy_X	True
n_jobs	None, -1
positive	False, True

A partir de los valores de la tabla anterior, se ha entrenado el modelo utilizando todas las combinaciones posibles del Grid.

La combinación con mejor coeficiente de determinación obtenido es la expuesta en la siguiente tabla:

Tabla 4.32. Tabla con los valores de la mejor combinación de parámetros mediante Grid Search y Cross Validation.

Parámetros	Valores
fit_intercept	True
normalize	False
copy_X	True
n_jobs	None
positive	True

Adicionalmente, los resultados en relación a los errores del modelo con esta combinación son los siguientes:

Tabla 4.33. Tabla con los valores de los errores del modelo con la mejor combinación encontrada mediante Grid Search y Cross Validation.

RMSE Training	RMSE Test	MAE Training	MAE Test	R^2 Training	R^2 Test
404.01	1574.74	240.29	1058.94	0.977907	0.991470

Como se puede observar en la tabla anterior, los valores del RMSE, MAE y R^2 de entrenamiento son ligeramente peores al modelo inicial pero realmente apenas se ven afectados. En cambio, en el caso del conjunto de test, tanto el RMSE, MAE y R^2 son ligeramente mejores que el modelo inicial, por lo que esto aumenta las posibilidades de que el modelo creado realice mejores predicciones del precio de Bitcoin.

4.3.3. Predicción del precio de Bitcoin

A partir del modelo anterior, se ha realizado la predicción del precio de la vela del día siguiente de Bitcoin y se ha comparado con el precio real mediante la siguiente gráfica:

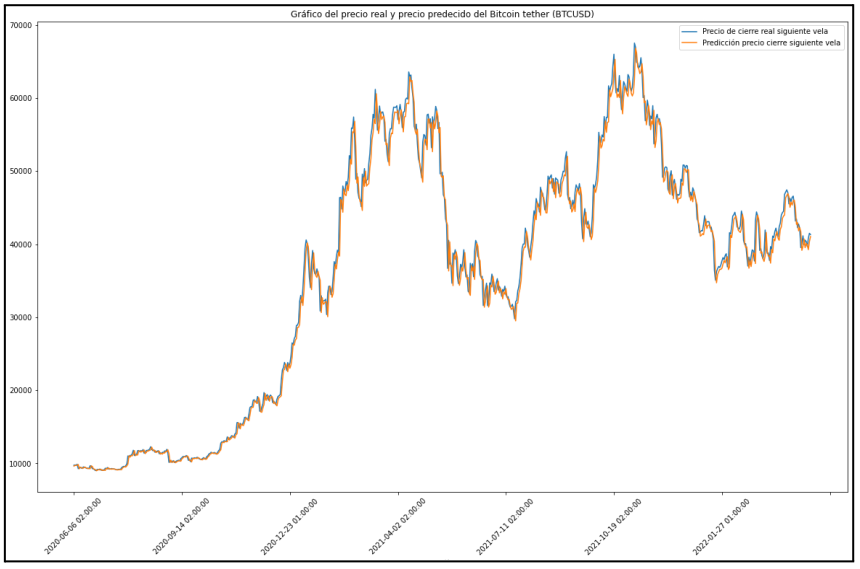

Fig. 4.21. Gráfico con el precio real de la vela del día siguiente y la predicción del precio de la vela del día siguiente.

Como se puede observar en la figura anterior, el precio real es el color azul y el precio predicho el color naranja. Tanto la estructura del precio de Bitcoin como

la precisión de las predicciones es muy buena, llegando a ser prácticamente igual el precio predicho y el precio real del cripto activo.

4.4. Resultados con series temporales: Forecasting

4.4.1. División de datos en conjunto de entrenamiento y test

De la misma forma que con los algoritmos anteriores, ha sido necesario realizar la división de datos en el conjunto de entrenamiento y en el conjunto de test.

Tal y como se ha mostrado con los algoritmos anteriores, al dividir los datos en conjunto de entrenamiento y test se evalúan los errores en regresión (RMSE y MAE) y el coeficiente de determinación o R^2. Los modelos con forecasting contienen un parámetro muy importante a la hora de realizar el entrenamiento: los lags.

Los lags permiten utilizar valores de variables anteriores para realizar predicciones futuras. En este caso, es necesario especificar el número de variables anteriores (“y”) que se quieren utilizar para entrenar el modelo.

Por lo tanto, al dividir los datos en conjunto de entrenamiento y test, se ha entrenado el modelo con 10 lags ya que el tiempo de entrenamiento es muy reducido.

Con forecasting, se ha centrado el análisis y la creación del modelo en la variación de los lags, los errores y el coeficiente de determinación del conjunto de test para comprobar y validar las predicciones del precio de Bitcoin, por lo que interesa optimizar al máximo el RMSE, MAE y R^2 del conjunto.

En la siguiente tabla, se muestran los resultados de los errores de test obtenidos:

Tabla 4.34. Tabla con los errores de test.

% Training/ %Test	RMSE Test	MAE Test	R^2 Test
60/40	2224.03	1490.05	0.982986
70/30	2535.33	1821.90	0.945158
80/20	2068.71	1520.98	0.940749

Tal y como se puede observar en la tabla anterior, la división de datos 60/40 es la que mayor R^2 tiene. Aun así, como se ha comentado anteriormente, también se debe tener en cuenta el resultado del RMSE obtenido, que da mejores

resultados para la división 80/20. Por lo tanto, se ha elegido la combinación de 80/20 ya que nos da el RMSE más bajo y el R^2 sigue siendo bueno.
Es necesario destacar que el 80% del Data Set destinado al conjunto de entrenamiento abarca un periodo de tiempo que va desde el 18/08/2017 hasta el 12/05/2021, mientras que el 20% restante destinado al conjunto de test va desde el 13/05/2021 hasta el 19/04/2022.

A continuación, se muestra una gráfica con la división de los datos en ambos conjuntos:

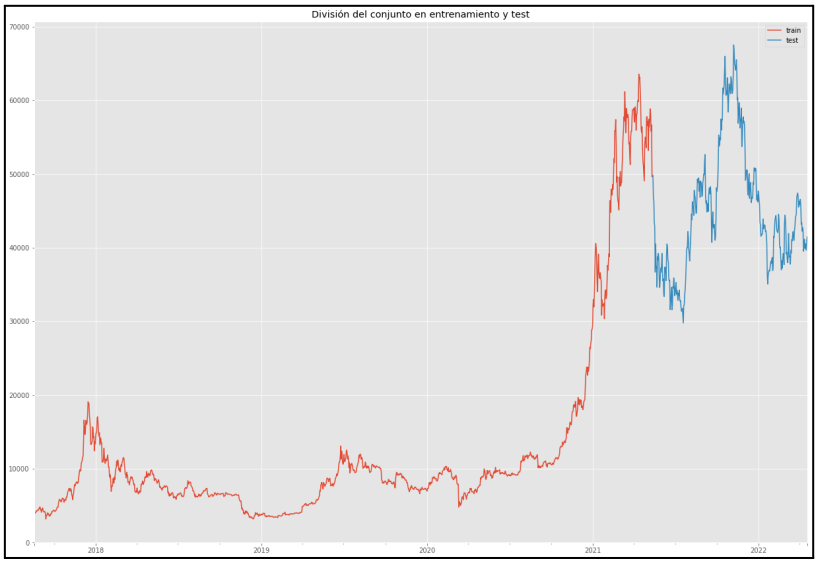

Fig. 4.22. Gráfico con la división de datos en conjunto de entrenamiento (rojo) y conjunto de test (azul).

4.4.2. Análisis de los errores del modelo y predicción del precio de Bitcoin

Anteriormente, se han expuesto diferentes métodos de forecasting. En este trabajo se han evaluado dos de ellos: el forecasting autorregresivo sin variables exógenas y utilizando como variable exógena el Halving de Bitcoin.

- **Forecasting autorregresivo sin variables exógenas:** En este método únicamente se ha utilizado la variable "y" y como regresor el LightGBM con los parámetros por defecto [32].

  Se han utilizado diferentes valores de lags para predecir el precio de Bitcoin y poder realizar una evaluación de las predicciones obtenidas junto con el valor de los errores RMSE, MAE y el coeficiente de determinación R^2. Los resultados obtenidos en relación con el número de lags establecido, se muestra en la siguiente tabla:

Tabla 4.35. Tabla con los resultados de RMSE, MAE y R^2 en función del número de lags establecido.

Lags	RMSE Test	MAE Test	R^2 Test
10	2068.71	1520.98	0.940749
100	1987.72	1478.30	0.945297
200	1926.05	1462.73	0.948638
342	1904.64	1419.36	0.949774

Como se muestra en la tabla anterior, a medida que se van aumentando el número de lags a la hora de entrenar el modelo, los resultados de test mejoran. Se ha establecido como límite 342, ya que el tiempo de entrenamiento con más lags puede comportar tiempos de entrenamiento superiores a 40 minutos, llegando a sobrepasar la hora de simulación.
Al obtener mejores resultados con 342 lags, se ha escogido para realizar las predicciones del precio.

En la siguiente figura, se muestra la predicción del precio de Bitcoin respecto el precio real:

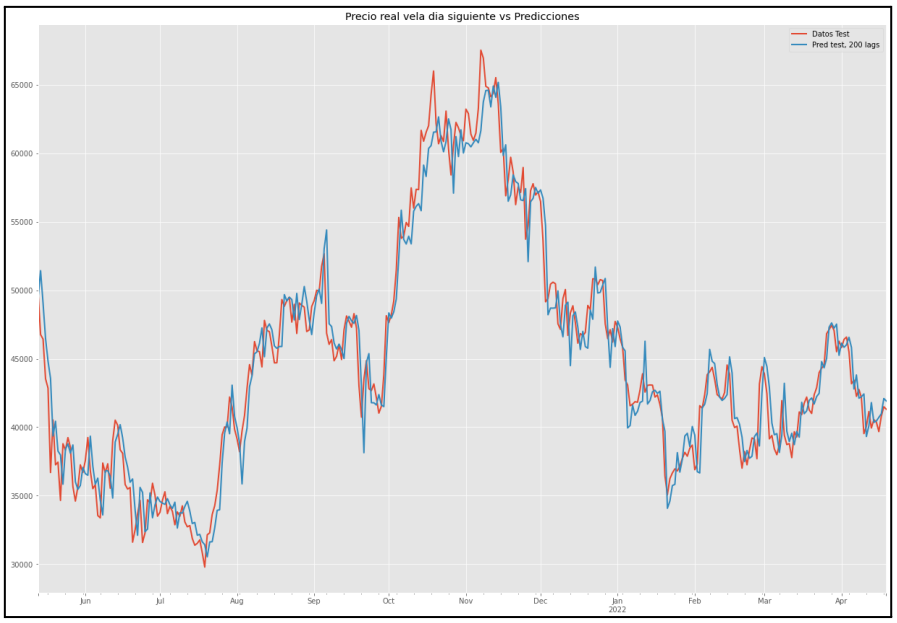

Fig. 4.23 Gráfico con el conjunto de test (rojo) y predicciones (azul).

Tal y como se puede observar en la Figura 4.23, en general la predicción es buena ya que la estructura del precio predicho sigue la estructura del precio real de Bitcoin. Aún así, hay intervalos donde la predicción no es del todo precisa, sobre todo entre octubre y finales de diciembre de 2021, donde Bitcoin realizó un movimiento agresivo al alza y entre mediados de octubre y noviembre el precio no tenía una dirección clara.

En cambio, a partir de enero de 2022, se puede observar que las predicciones mejoran en cuanto a precisión se trata.

- **Forecasting autorregresivo con variables exógenas:** En este método, se ha utilizado la misma variable y regresor que en el caso anterior, pero se ha utilizado como variable exógena el Halving.

  Tal y como se ha explicado brevemente al inicio del apartado 2.3, el Halving es un evento que ocurre aproximadamente cada 4 años en el cuál se reducen las recompensas que reciben los mineros por bloque validado.

  Se ha incorporado en el Data Set las fechas de los Halvings, las recompensas que reciben los mineros cada vez que ocurre el evento y el bloque en el que sucede [33]. Con esta información es posible predecir cuándo será el siguiente Halving.

  En la siguiente tabla, se muestran los datos desglosados de cada uno de los Halvings:

Tabla 4.36. Tabla con los datos de los Halvings registrados hasta la fecha.

Número halving	Fecha	Recompensa	Bloque
1	03/01/2009	50	0
2	28/11/2012	25	210000
3	09/07/2016	12.5	420000
4	11/05/2020	6.25	630000
5	?	3.125	840000

Con esta información, se ha realizado la predicción del siguiente Halving teniendo en cuenta los bloques que faltan por validar en total de los 840.000 y los bloques por día antes de que ocurra el evento (este cálculo se ha realizado a fecha de 21/09/2022):

Tabla 4.37. Tabla con los detalles respecto al siguiente Halving.

Número halving	Bloques restantes	Bloques/dia	Recompensa	Fecha predicha
5	84904	144	3.125	02/05/2024

Para obtener la fecha predicha, se ha realizado la división de los bloques restantes entre los bloques por día para encontrar el número de días que faltan para el siguiente Halving. Aproximadamente son 589 días, por lo que desde el 21/09/2022 el evento ocurriría el 02/05/2024.

Una vez se ha averiguado la fecha, se ha incorporado en el Data Set los días restantes para el siguiente Halving y las recompensas. Es necesario recordar que el Data Set abarca un periodo de tiempo desde el 18/08/2017 hasta el 19/04/2022, por lo que el número de bloques restantes y las recompensas están limitadas a estas fechas.

Llegados a este punto, se han incorporado diferentes valores de lags para predecir el precio de Bitcoin y comparar los resultados obtenidos con variables exógenas y sin variables exógenas evaluando los errores y las predicciones de ambos métodos. Los resultados obtenidos en relación con el número de lags establecido, se muestran en la siguiente tabla:

Tabla 4.38. Tabla con los resultados de RMSE, MAE y R^2 con variables exógenas, en función del número de lags establecido.

Lags	RMSE Test	MAE Test	R^2 Test
10	2011.62	1473.53	0.943973
100	1965.93	1462.83	0.946489
200	1927.42	1464.86	0.948565
342	1910.99	1422.14	0.949439

Como se puede observar en la tabla anterior, aumentando el número de lags disminuyen los errores y mejora el R^2. Si se comparan los resultados obtenidos con el caso sin variables exógenas, se aprecia como con los valores de 10 y 100 lags se mejoran las prestaciones del modelo, pero en cambio con 200 y 342 empeoran ligeramente.

El modelo escogido en este caso es el de 342 lags para realizar las predicciones del precio al obtener el mejor resultado con variables exógenas.

A continuación, se muestra la predicción del precio de Bitcoin respecto el precio real sin variables exógenas y con variables exógenas:

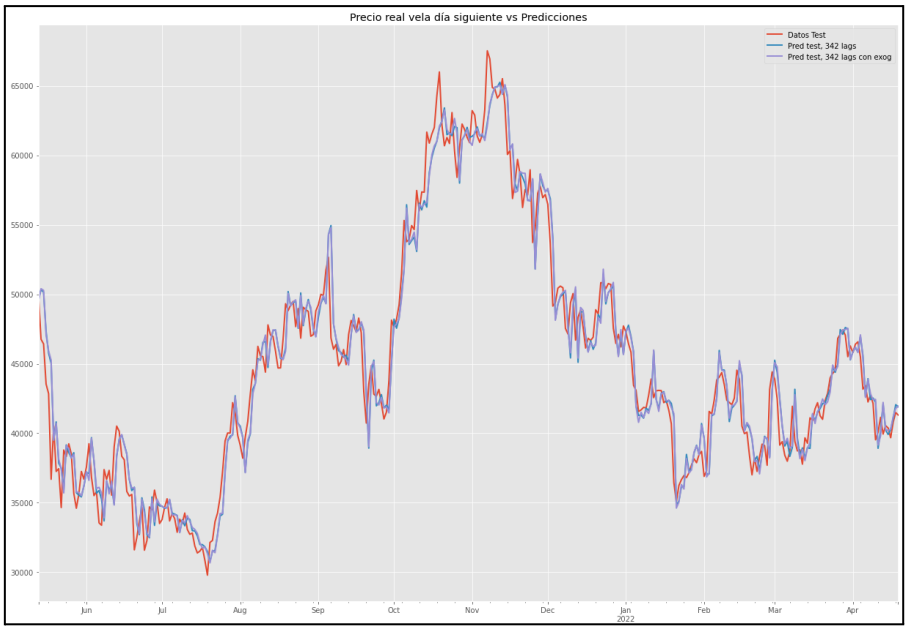

Fig. 4.24. Gráfico con el conjunto de test (rojo), predicciones sin variables exógenas (azul) y con variables exógenas (lila).

Tal y como se puede observar en la figura anterior, la predicción del precio sin variables exógenas es ligeramente peor que utilizando como variable exógena el Halving de Bitcoin en periodos de tiempo donde el precio no presenta variaciones agresivas como por ejemplo entre mediados de octubre y mediados de noviembre de 2021, donde la predicción sin variables exógenas es menos precisa y más sensible a las variaciones del precio que con variables exógenas.

4.5. Tiempo de ejecución de las pruebas

En este apartado se expone el tiempo de ejecución de las pruebas más importantes realizadas con los algoritmos utilizados en el presente trabajo. Es necesario comentar que la herramienta Google Colab tiene establecido como tiempo mínimo 1 segundo, por lo que aunque una ejecución de código dure milisegundos, la herramienta indica siempre 1 segundo.

A continuación, se muestra una tabla con el tiempo de ejecución en cargar las librerías, el Data Set y la preparación de este:

Tabla 4.39. Tabla con el tiempo de ejecución al cargar librerías, el Data Set, su preparación y la división de los datos en conjunto de entrenamiento y test.

Pruebas	Tiempo (h min s)
Librerías y cargar Data Set	24 s
Preparación del Data Set	3 s
División entrenamiento y test	1 s
Total	**28 s**

4.5.1. Random Forest

En la siguiente tabla, se exponen los tiempos de ejecución de las pruebas llevadas a cabo con Random Forest y el tiempo total:

Tabla 4.40. Tabla con el tiempo total de ejecución de las pruebas realizadas con Random Forest.

Pruebas	Tiempo (h min s)
Visualizar errores modelo inicial	3 s
Valoración de la influencia de cada parámetro	25 s
Valores óptimos de los parámetros con Out of Bag y Cross Validation	7 h 13 min 26 s
Parámetros óptimos con Grid Search usando Out Of Bag y Cross Validation	43 min 13 s
Predicción precio Bitcoin	5 s
Total	**9 h 18 min 4 s**

Tal y como se muestra en el apartado 4.2.3.1, se han llevado a cabo 4 pruebas para cada uno de los parámetros del modelo para encontrar los valores óptimos mediante Out Of Bag y Cross Validation e introducir algunos de ellos en el Grid Search para buscar la combinación óptima. A continuación, se expone el tiempo de ejecución medio de cada prueba:

Tabla 4.41. Tabla con el tiempo de ejecución medio de cada prueba realizada para encontrar los valores óptimos de cada parámetro con Out Of Bag y Cross Validation.

Parámetro	Out Of Bag	Cross Validation
n_estimators	6 min 52 s	29 min 30 s
max_depth	2 min 1 s	8 min 27 s
min_samples_split	22 s	3 min 19 s
min_samples_leaf	38 s	5 min 56 s
max_leaf_nodes	1 min 29 s	13 min 28 s
max_samples	8 min 45 s	27 min 23 s
max_features	4 s	20 s

4.5.2. Regresión Lineal

El tiempo de ejecución de las pruebas llevadas a cabo con Regresión Lineal y el tiempo total, se muestra en la siguiente tabla:

Tabla 4.42. Tabla con el tiempo de ejecución de las pruebas realizadas con Regresión Lineal.

Pruebas	Tiempo (h min s)
Visualizar errores modelo inicial	3 s
Parámetros óptimos utilizando Grid Search con Cross Validation	5 s
Predicción precio Bitcoin	52 s
Total	**1 min**

Como el modelo con el algoritmo de Regresión Lineal no contiene el parámetro random_state, se han realizado sólo 2 pruebas. La segunda únicamente se ha llevado a cabo como comprobación de que se obtienen los mismos resultados de combinación de valores óptimos de los parámetros utilizados, por lo que el tiempo medio de cada prueba ha sido de 2 min 30 s.

4.5.3. Series temporales: Forecasting

Por último, se plasma el tiempo de ejecución de las pruebas de Forecasting sin variables exógenas y con variables exógenas y el tiempo total:

- **Sin variables exógenas:**

Tabla 4.43. Tabla con el tiempo de ejecución de las pruebas realizadas con Forecasting sin variables exógenas.

Pruebas	Tiempo (h min s)
Visualizar errores modelo con cada lag y predecir el precio de Bitcoin	36 min 47 s
Total	**30 min 28 s**

- **Con variables exógenas:**

Tabla 4.44. Tabla con el tiempo de ejecución de las pruebas realizadas con Forecasting con variables exógenas.

Pruebas	Tiempo (h min s)
Establecer Halving como variable exógena y calcular fecha siguiente halving	4 s
Incluir los datos de Halving en el Data Set	2 s
Visualizar errores modelo con cada lag y predecir el precio de Bitcoin comparándolo con el modelo sin variables exógenas	31 min 30 s
Total	**31 min 36 s**

Tanto para el modelo sin variables exógenas como para el modelo con variables exógenas, se han realizado 4 pruebas utilizando diferentes lags.

En la siguiente tabla, se muestra el tiempo de ejecución medio de cada prueba:

Tabla 4.45. Tabla con el tiempo de ejecución de cada prueba realizada con diferentes valores de lags sin variables exógenas y con variables exógenas.

Lags	Sin variables exógenas	Con variables exógenas
10	40 s	47 s
100	4 min 33 s	5 min 5 s
200	9 min 15 s	9 min 19 s
342	16 min	16 min 19 s

Tal y como se puede observar, el algoritmo con más tiempo de ejecución es el Random Forest. Esto es debido a la cantidad de parámetros y al rango de valores establecido en cada uno de ellos ya que, por ejemplo, el rango de n_estimators y max_samples va de 1 a 400 y de 1 a 1364 respectivamente.

En el caso de la Regresión Lineal, el tiempo de ejecución de las pruebas es mucho menor debido a que contiene menos parámetros que Random Forest. Además, solo ha sido necesario valorar la modificación de los parámetros fit_intercept, n_jobs y positive, por lo que el tiempo de ejecución se reduce considerablemente.

Finalmente, en las series temporales se han realizado pruebas sin utilizar variables exógenas y utilizando variables exógenas con el algoritmo LightGBM con los valores por defecto [28]. El análisis y la predicción del precio de Bitcoin, se ha centrado en la variación del número de lags para ver la diferencia entre el uso de variables externas (Halving) y sin utilizar variables externas. Como se puede apreciar en la Tabla 4.45, a medida que se ha ido aumentando el número de lags el tiempo de ejecución ha ido incrementando en ambos casos, aunque la diferencia principal es que con variables exógenas, el tiempo de ejecución es ligeramente superior al tener en cuenta una variable adicional (Halving).

CAPÍTULO 5. CONCLUSIONES Y LÍNEAS FUTURAS

Bitcoin es una criptomoneda muy joven, con poco más de 10 años, y hasta el momento ha demostrado ser muy volátil, con un precio que depende en muchos casos de decisiones políticas u otros factores externos.

Este escenario hace que sea complejo invertir en este criptoactivo para poder obtener beneficios económicos utilizando exclusivamente el análisis técnico o manual de mercado, ya que un ser humano no es capaz de retener y tratar con la gran cantidad de información que procesa la tecnología. Por ello, en este proyecto se han utilizado técnicas de Machine Learning y se ha conseguido desarrollar diferentes modelos de predicción del precio de Bitcoin utilizando tres algoritmos supervisados de regresión: Random Forest, Regresor Lineal y Series temporales *(Forecasting)*.

Se ha podido observar que, utilizando las técnicas de Out Of Bag, Cross Validation y Grid Search, es factible obtener los valores óptimos de los parámetros de Random Forest y Regresión lineal. Este hecho demuestra que es posible optimizar los modelos iniciales, permitiendo una mejora en la predicción del precio de Bitcoin y una reducción de los errores.

Cabe destacar la relevancia que tiene la introducción de variables externas en el modelo de Series temporales (*Forecasting)* como el Halving de Bitcoin, que mejora las predicciones respecto al modelo que no incluye variables externas.

Analizados los resultados obtenidos con los modelos creados, se puede concluir que Machine Learning es una herramienta muy potente para realizar predicciones, siendo el modelo de Regresión Lineal el que mejores predicciones proporciona. Aún así, el rendimiento de Random Forest y Series temporales *(Forecasting)* también ofrecen resultados notables.

Este trabajo de final de grado es el primer paso para facilitar el análisis de un mercado joven y volátil, que hasta hace poco no era muy conocido en nuestra sociedad, a todos aquellos usuarios que estén interesados en el mundo de las criptomonedas.

Aprender cómo funciona el análisis técnico de mercados, supone invertir una gran cantidad de tiempo en asimilar todas las herramientas y conceptos necesarios para ser capaz de entender y predecir los posibles movimientos futuros de la criptomoneda. A partir de este trabajo, no solo se puede descubrir el potencial del Machine Learning para realizar predicciones del precio de Bitcoin, sinó también la metodología a seguir con cualquier tipo de criptomoneda e incluso, cualquier tipo de mercado.

Como líneas futuras, se podría desarrollar modelos de predicción mucho más sofisticados que permitan predecir el precio del Bitcoin o de cualquier criptomoneda, sin tener en cuenta datos conocidos. Para ello, se podría trabajar con algoritmos no supervisados de regresión, que permitan realizar predicciones del precio de Bitcoin sin conocer datos futuros.

Para que un trabajo de estas características pueda generar beneficios económicos, se debería desarrollar un software de Machine Learning aplicado al mercado de las criptomonedas que permita realizar operaciones de forma automática.

ANEXO: CÓDIGO DE PROGRAMACIÓN

El código de programación desarrollado para realizar el presente trabajo, ha sido almacenado en una carpeta de Google Drive. En esta carpeta se puede encontrar el código para descargar el .CSV de Bitcoin (Descargar CSV BTC), el desarrollo del modelo con Random Forest (BTCUSDT 1D RF), con Regresión lineal (BTCUSDT 1D LinearRegression) y finalmente, con Time Series Forecasting sin variables exógenas (Series temporales sin exog) y con variables exógenas (Series temporales Halving).

Se adjunta el enlace a partir del cual se pueden consultar los desarrollos de código mencionados:

https://drive.google.com/drive/folders/1DLcxBWJjWeSHYxGPzTn26D7ycfsIWS6u?usp=sharing

BIBLIOGRAFÍA

[1] Satoshi Nakamoto, “Bitcoin: A Peer-to-Peer Electronic Cash System”, https://bitcoin.org/bitcoin.pdf, [Última visita: 23/07/2022].

[2] Bit2Me Academy - Formación de Bitcoin y Criptomonedas , “¿Qué es una red P2P?”, https://academy.bit2me.com/que-es-una-red-p2p/#:~:text=Bitcoin%2C%20una%20red%20P2P%20para,usuarios%20manejar%20valor%20sin%20intermediarios, [Última visita 23/08/2022].

[3] Investing.com Español - Finanzas, Noticias y Bolsa de Valores, “Bitcoin”, https://es.investing.com/crypto/bitcoin/historical-data, [Última visita: 20/07/2022].

[4] Emily Rella, “Los 5 tweets de Elon Musk que sacudieron al mundo de las criptomonedas este año”, https://www.entrepreneur.com/article/410838, [Última visita: 15/07/2022].

[5] Denniye Hinestroza Ramírez, “El Machine Learning a través de los tiempos, y los aportes a la humanidad, https://repository.unilibre.edu.co/bitstream/handle/10901/17289/EL%20MACHINE%20LEARNING.pdf?sequence=1&isAllowed=y, [Última visita: 20/07/2022].

[6] Scikit Learn, “Machine Learning in Python”, https://scikit-learn.org/stable/, [Última visita 16/07/2022].

[7] Google, “Colaboratory”, https://research.google.com/colaboratory/faq.html, [Última visita: 16/07/2022].

[8] Bit2Me Academy - Formación de Bitcoin y Criptomonedas, “¿Cómo se crea o genera un bitcoin en la blockchain?”, https://academy.bit2me.com/como-se-crea-un-bitcoin/#:~:text=Es%20por%20medio%20del%20proceso,la%20econom%C3%ADa%20que%20lo%20sostiene, [Última visita: 20/07/2022].

[9] Blockchain.com | Buy Bitcoin, Ethereum and more with trust, “Bitcoin”, https://www.blockchain.com/explorer/assets/btc, [Última visita: 16/07/2022].

[10] Derliz Machado, “¿Qué factores determinan el precio de bitcoin?”, https://www.criptonoticias.com/mercados/factores-determinan-precio-bitcoin-explicamos/, [Última visita: 10/07/2022].

[11] El Confidencial, “Por qué Estados Unidos decidió imprimir más dólares que nunca en 2020”, https://www.elconfidencial.com/mercados/2020-12-22/estados-unidos-imprimir-dinero-2020_2882452/, [Última visita: 11/07/2022].

[12] Bit2Me Academy - Formación de Bitcoin y Criptomonedas, “Ataque del 51% en Bitcoin”,

https://academy.bit2me.com/ataque-51-bitcoin/#:~:text=Un%20ataque%20del%2051%25%20se,votaciones%E2%80%9D%20que%20el%20resto%20junto., [Última visita: 11/07/2022].

[13] BBC News Mundo, “Bitcoin: China declara ilegales todas las transacciones con criptomonedas y se desploma el precio de la más popular”, https://www.bbc.com/mundo/noticias-58683341, [Última visita: 12/07/2022].

[14] BBC News Mundo, “Bitcoin: El Salvador se convierte este martes en el primer país del mundo en adoptar la criptomoneda como divisa de curso legal”, https://www.bbc.com/mundo/noticias-america-latina-58441561, [Última visita: 12/07/2022].

[15] Álvaro Sánchez, “Elon Musk anuncia que ya se pueden comprar coches Tesla con bitcoins en EE UU”, https://elpais.com/economia/2021-03-24/elon-musk-anuncia-que-ya-se-pueden-comprar-coches-tesla-con-bitcoins-en-ee-uu.html, [Última visita: 18/07/2022].

[16] BBC News Mundo, “Bitcoin y Elon Musk: el CEO de Tesla anuncia que ya no aceptarán la criptomoneda y esta sufre una fuerte caída”, https://www.bbc.com/mundo/noticias-57096818, [Última visita: 25/07/2022].

[17] XTB Online Trading - Los Mejores Expertos En Bolsa, “Tipos de gráficos de bolsa”, https://www.xtb.com/es/educacion/tipos-de-graficos-de-bolsa#:~:text=La%20%C3%BAnica%20diferencia%20es%20la,o%20un%20bajista%20(ca%C3%ADda), [Última visita: 01/08/2022].

[18] AvaTrade-Trade with confidence, “Herramientas del análisis técnico”, https://www.avatrade.es/informacion-de-trading/analisis/analisis-tecnico,[Última visita: 05/08/2022].

[19] Azure Microsoft, “Algoritmos de aprendizaje automático”, https://azure.microsoft.com/es-es/resources/cloud-computing-dictionary/what-are-machine-learning-algorithms/#overview, [Última visita: 05/08/2022].

[20] Universidad de Jaén, “Regresión lineal simple”, http://www4.ujaen.es/~dmontoro/Metodos/Tema%209.pdf, [Última visita: 15/09/2022].

[21] Universidad de Santiago de Compostela, “Regresión Lineal Múltiple, El modelo, estimación de los parámetros, contrastes”, http://eio.usc.es/eipc1/BASE/BASEMASTER/FORMULARIOS-PHP-DPTO/MATERIALES/Mat_50140128_RegresionMultiple.pdf, [Última visita: 15/09/2022].

[22] Scikit Learn, “Support Vector Machines”, https://scikit-learn.org/stable/modules/svm.html, [Última visita: 15/09/2022].

[23] Joaquín Amat Rodrigo, “Árboles de decisión con Python: regresión y clasificación”,

https://www.cienciadedatos.net/documentos/py07_arboles_decision_python.html, [Última visita: 02/08/2022].

[24] Joaquín Amat Rodrigo, “Random Forest con Python”, https://www.cienciadedatos.net/documentos/py08_random_forest_python.html, [Última visita: 25/07/2022].

[25] Analítica, Inteligencia Artificial y Gestión de Datos-SAS, “Deep Learning, Qué es y por qué es importante”, https://www.sas.com/es_es/insights/analytics/deep-learning.html, [Última visita: 01/08/2022].

[26] Joaquín Amat Rodrigo, “Skforecast: forecasting series temporales con Python y Scikit-learn”, https://www.cienciadedatos.net/documentos/py27-forecasting-series-temporales-python-scikitlearn.html, [Última visita: 01/09/2022].

[27] Jose Martínez Heras, “Regularización Lasso L1, Ridge L2 y ElasticNet”, https://www.iartificial.net/regularizacion-lasso-l1-ridge-l2-y-elasticnet/#Regularizacion_Ridge_L2, [Última visita: 01/09/2022].

[28] Microsoft Corporation, “Welcome to LightGBM’s documentation!”, https://lightgbm.readthedocs.io/en/v3.3.2/, [Última visita: 21/09/2022].

[29] Skforecast Docs, “Direct multi-step forecaster”, https://joaquinamatrodrigo.github.io/skforecast/0.4.3/notebooks/direct-multi-step-forecasting.html, [Última visita: 21/09/2022].

[30] Binance, “API de Binance”, https://www.binance.com/es/binance-api, [Última visita: 05/08/2022].

[31] Aprende Machine Learning, “Sets de Entrenamiento, Test y Validación”, https://www.aprendemachinelearning.com/sets-de-entrenamiento-test-validacion-cruzada/, [Última visita: 20/09/2022].

[32] Microsoft Corporation, “lightgbm.LGBMRegressor”, https://lightgbm.readthedocs.io/en/latest/pythonapi/lightgbm.LGBMRegressor.html#lightgbm.LGBMRegressor, [Última visita: 21/09/2022].

[33] CoinMarketCap, “Cuenta regresiva al Halving de Bitcoin”, https://coinmarketcap.com/es/halving/bitcoin/ , [Última visita: 21/09/2022].