UNIVERSIDAD AUTÓNOMA DE CIUDAD JUÁREZ

Instituto de Ingeniería y Tecnología

Departamento de Ingeniería Eléctrica y Computación

PRONOSTICAR EL PRECIO DEL BITCOIN CON REDES NEURONALES

Reporte Técnico de Investigación presentado por:

Jesús Arenas Martínez 148585

Requisito para la obtención del título de:

INGENIERO EN SISTEMAS COMPUTACIONALES

Dr. Javitt Higmar Nahitt Padilla Franco

Ciudad Juárez, Chihuahua 27 de noviembre de 2019

Ciudad Juárez, Chihuahua, a 7 de Noviembre de 2019

Asunto: Liberación de Asesoría

Mtro. Ismael Canales Valdiviezo
Jefe del Departamento de Ingeniería
Eléctrica y Computación
Presente.-

Por medio de la presente me permito comunicarle que, después de haber realizado las asesorías correspondientes al reporte técnico Pronosticar el precio del Bitcoin con redes neuronales, del alumno Jesús Arenas Martínez, de la Licenciatura en Ingeniería en Sistemas Computacionales, considero que lo ha concluido satisfactoriamente, por lo que puede continuar con los trámites de titulación intracurricular.

Sin más por el momento, reciba un cordial saludo.

Atentamente

Dr. Javitt Higmar Nahitt Padilla Franco
Profesor Investigador IIT

UACJ
DEPARTAMENTO DE INGENIERÍA ELÉCTRICA Y COMPUTACIÓN

Ccp:
Coordinador del Programa de Sistemas Computacionales
Jesús Arenas Martínez
Archivo

Ciudad Juárez, Chihuahua, a 7 de Noviembre de 2019

Asunto: Autorización de Publicación

C. Jesús Arenas Martínez

Presente.-

En virtud de que cumple satisfactoriamente los requisitos solicitados, informo a usted que se autoriza la publicación del documento de Pronosticar el precio del Bitcoin con redes neuronales, para presentar los resultados del proyecto de titulación con el propósito de obtener el título de Licenciado en Ingeniería en Sistemas Computacionales.

Sin otro particular, reciba un cordial saludo.

Mtra. Ivonne Haydee Robledo Portillo

Profesor Titular de Seminario de Titulación II

Declaración de Originalidad

Yo, Jesús Arenas Martínez, declaro que el material contenido en esta publicación fue elaborado con la revisión de los documentos que se mencionan en el capítulo de Bibliografía, y que la solución obtenida es original y no ha sido copiada de ninguna otra fuente, ni ha sido usada para obtener otro título o reconocimiento en otra institución de educación superior.

Jesús Arenas Martínez

Agradecimientos

Agradezco el apoyo del Dr. Nahitt Padilla por haberme ayudado y encaminado en el transcurso del proyecto. En contribuir con el contenido del documento y en la aportación de ideas para la realización del mismo. En sí porque tuvo la disposición de ayudarme desde el principio que le pedí que fuera mi asesor.

Al profesor Rene Noriega por impulsarme, por haber ayudado en la redacción del documento. En darme como alumno la motivación de que las cosas que haga, las realice con pasión y de ser constantes. Que leyendo podemos fomentar nuestra creatividad y esos consejos han servido para este proyecto.

A la maestra Ivonne Robledo por estar en constante acompañamiento en mi elaboración del documento, en darme consejos de como transmitir de mejor manera mis ideas a los que leerán este trabajo. También por creer en la capacidad que poseo como estudiante.

Y por último a todos los profesores de esta carrera en la universidad, que, con sus clases y esfuerzo propio, pude obtener los conocimientos necesarios y mostrar que he adquirido las habilidades suficientes para aplicarlas en el desarrollo del proyecto.

Dedicatoria

Este trabajo es dedicado para mi madre, que me ha enseñado a través de las etapas de mi vida varias lecciones de perseverancia. Como aquella vez que tuve que vivir junto con ella una experiencia a principios de cuando ingrese a la universidad, le dio cáncer de mama y las veces que la acompañaba cuando recibía quimioterapias, pude notar en ella que, hasta cuando parece que todo se ve en contra hay que tener fe, pensar de la mejor manera y seguir adelante.

A mi padre le dedico esto, por la fortaleza que me ha mostrado. En diferentes caídas que ha tenido mi madre por enfermedades en la vida, me enseñaba la fortaleza en cada momento y también por la dedicación que me mostraba a lo hora de hacer una actividad o trabajo.

A mi hermano porque me enseñó el valor del trabajo duro. Que el valor del trabajo duro no significa trabajar muchas horas o realizar muchas actividades, sino que significa que hay que trabajar inteligente en la vida. Que hay momentos en que uno debe pararse analizar el panorama de las cosas y resolverlas.

A Ana Quijano que me acompañó durante todo este tiempo en mi formación como un hombre, el cual me ha ayudado a perseguir mis metas y el poder ser parte de su vida.

Por último, a Dios, porque ha podido influir en mi a través de las personas anteriores que mencioné para mi desarrollo en la vida.

Índice general

Índice de figuras

Índice de tablas

Índice de listas

Resumen

El proyecto analiza el desarrollo de un modelo con redes neuronales recurrentes de tipo LSTM que ayude al pronóstico del precio de Bitcoin. El objetivo es pronosticar el precio en el lapso de tiempo de una semana a futuro. La hipótesis dice qué mediante el análisis y modelado de las fluctuaciones pasadas en el precio del Bitcoin es factible pronosticar su valor futuro con un 50 % de nivel de confianza. Se utilizaron datos históricos de la criptomoneda desde el 2009 hasta 2019 en temporalidad medidas en días. Los parámetros que se usaron de entrada para el desarrollo, son los mismos datos extraídos de la *Blockchain* y el precio. Se construyeron dos modelos, el primero que contiene todos los parámetros y el segundo solo el precio. En el primer modelo solo se llegó hasta el entrenamiento de la red neuronal y el segundo se utilizó para realizar el pronóstico de una semana. Los resultados que se obtuvieron con el primer modelo son deficientes en la exactitud del precio conforme al tiempo, sin embargo, los parámetros de la *Blockchain* están ligadas con la fluctuación del precio de Bitcoin. El segundo modelo de igual manera tuvo una precisión deficiente precio, pero aprendió de los rangos en que oscila el precio y las posibles bajadas o subidas que presenta. Se cumplió el objetivo con el segundo modelo y la hipótesis este proyecto no fue comprobada ya que, le fue difícil al modelo pronosticar el precio del Bitcoin y el porcentaje no tuvo el nivel de confianza factible. Las recomendaciones conforme al proyecto es que deben usarse tramas de tiempo más cortos por ejemplo de una hora para tener mejores resultados en el pronóstico.

Palabras clave: Bitcoin, Pronóstico, LSTM, Series de tiempo, Red Neuronal recurrente.

Abstract

The project analyzes the development of a model with recurrent neural networks of the LSTM type that helps the Bitcoin price forecast. The objective is to forecast the price in the span of one week to the future. The hypothesis says that by analyzing and modeling past fluctuations in the price of Bitcoin it is feasible to forecast its future value with a 50 % confidence level. Historical data from the cryptocurrency from 2009 to 2019 were used in time periods in days. The parameters that were used as input for development are the same data extracted from the Blockchain and the price. Two models were built, the first one containing all the parameters and the second only the price. In the first model only the training of the neural network was reached and the second was used to make the forecast of one week. The results obtained with the first model are deficient in the accuracy of the price according to time, however, the parameters of the Blockchain are linked to the fluctuation of the Bitcoin price. The second model also had poor price accuracy, but learned from the ranges in which the price oscillates and the possible decreases or increases it presents. The objective was met with the second model and the hypothesis this project was not proven since, it was difficult for the model to predict the price of Bitcoin and the percentage did not have the feasible level of confidence. The recommendations under the project is that shorter time frames should be used, for example, one hour to have better results in the forecast.

Keywords: Bitcoin, Forecast, LSTM, Time Series, Recurring Neural Network.

Introducción

La inteligencia artificial (IA) ha estado en constante crecimiento gracias a la evolución de la capacidad computacional en diferentes dispositivos. Muchas de las empresas hoy en día en tecnología, como Google, desarrollan productos que benefician a la población mundial por los servicios que ofrece e integran esta tecnología, como en este caso lo usa su buscador. Las redes neuronales son modelos de la IA, donde los sistemas aprenden con análisis de datos.

Desde el 2009 aproximadamente surgió un nuevo mercado financiero donde los activos son códigos digitales que funcionan como dinero a los que se les llamó criptomonedas o monedas digitales. El Bitcoin es la pionera de este mercado, es la que más volatilidad llega a presentar y la que mayor dominancia tiene en el mercado de estos activos digitales. Se volvió muy confiable gracias a la tecnología *Blockchain*, es la red que usa para hacer todas las transacciones.

Las personas que invierten en el Bitcoin han visto que sube y baja su precio de manera impresionante y las ganancias que pueden tener gracias a su volatilidad. Sin embargo, un análisis antes de operar en el mercado es fundamental para tener mejor controlado los riesgos de inversión.

Se podrá ver más adelante como es que las redes neuronales pueden ayudar a la realización de pronósticos de series de tiempo conforme al precio del Bitcoin y que resultados se pueden obtener con el análisis de diferentes datos para la construcción de un modelo que ayude al usuario en la toma de decisiones.

En el capitulo 1 se describen las criptomonedas y porque están teniendo un mayor crecimiento en cuanto a su uso por parte de los usuarios. Se enfocará en la parte de inversión en el mercado y las problemáticas que se pueden encontrar los inversionistas en el análisis de sus operaciones de compraventa.

En el capitulo 2 se introduce la teoría que sustenta el proyecto para la comprensión del tema. Se expondrá la tecnología *Blockchain* aplicado a Bitcoin, después lo que es el *trading* en el mercado de las criptomonedas y se concluye con las redes neuronales aplicadas en las series de tiempo.

En el capitulo 3 se describe la metodología que se utilizó y los pasos a seguir por etapa, mostrando el desarrollo de los modelos para pronósticos del precio del Bitcoin por medio de diferentes datos.

En el capitulo 4 se explican los resultados de los modelos desarrollados y la efectividad que presentaron. Por último en el capitulo 5 se mostraran las conclusiones del proyecto si se cumplió el objetivo y las recomendaciones que se pueden aplicar a este proyecto.

Capítulo 1
Planteamiento del Problema

En el mundo existen diferentes mercados donde se puede invertir, los más conocidos son la bolsa de valores y Forex que es el mercado internacional de divisas. Empezó a surgir un nuevo mercado donde el concepto era utilizar monedas virtuales que pudieran ser usadas como cualquier otra divisa internacional sin que estuviera regulada por alguna institución financiera o gubernamental. Y así es como empezaron a surgir las criptomonedas empezando con la pionera que es Bitcoin y tiempo después existirían más hasta conformarse como un nuevo mercado.

Bitcoin es una criptomoneda que genera mucha liquidez en todo el mundo. Inversionistas multimillonarios de diferentes lugares del mundo han mantenido sus fondos de inversión a salvo en Bitcoins. Incluso ha sido la base para crear otras criptomonedas que comúnmente se les denomina *altcoins* o criptomonedas alternas.

1.1. Antecedentes

Una criptomoneda es un activo o moneda digital que se comercian solo por Internet. El Bitcoin es una criptomoneda, se observa que ocupa el primer lugar del mercado dominando el 51.06 %, según los datos registrados del sitio web Coinmarketcap [1].

El Bitcoin se utiliza en la compraventa para sacar ganancias a través de la variación de su precio y también se puede utilizar como medio de pago. A lo largo de los años el Bitcoin ha presentado altas y bajas de su valor de manera significativa, puesto que, a muchos usuarios les interesa invertir en este activo debido a que hoy puede valer un precio bajo y mañana uno mayor.

La manera en que funciona la tecnología que utiliza el Bitcoin generó confianza en los usuarios para que empezaran a operar con esta criptomoneda. Su valor no se encuentra vinculado a un comportamiento de una economía en concreto. El precio depende de la cantidad de usuarios que operen con ella.

Bitcoin es dinero digital para los usuarios, el concepto por el cual se rige esta criptomoneda es que todas las transacciones funcionen de una manera descentralizada. Esto quiere decir que no está regulada por una institución financiera. Lo que se pretende es mantener el anonimato de los usuarios, ese fue el concepto en un principio.

Cuando una persona tiene una tarjeta de débito o crédito, la pueden utilizar para pagar un producto en una tienda departamental, en un sitio de comercio electrónico o para pagar servicios en Internet, por mencionar varios ejemplos. Entonces, el banco brinda una tarjeta a los usuarios y está ligada al número de cuenta donde se encuentra el dinero. El banco es un intermediario y se encarga de realizar las transacciones ya sea , entre las mismas cuentas de la institución hasta cuentas de otros bancos al momento que se utiliza la tarjeta. Por lo tanto, el banco sabe qué compró esta persona, cuánto dinero transfirió o cuánto dinero le han depositado si es que otra persona u organización le transfirió dinero.

Existen plataformas en línea ya sea una casa de cambio (*Exchange*) o carteras (*Wallets*), que permiten las transacciones de Bitcoins. Si se quiere transferir esta criptomoneda a un Exchange para poder cambiarlos a dinero real o transferirlos para guardarlos en una *Wallet*, sólo se necesita saber las direcciones de ambas carteras y el monto. Esto se puede realizar por medio de la tecnología *Blockchain* que utiliza Bitcoin.

La primera especificación del protocolo Bitcoin y la prueba del concepto la publicó Satoshi Nakamoto en el 2009 en una lista de correo electrónico. Satoshi abandonó el proyecto a finales de 2010 sin revelar mucho sobre su persona [2].

El precio del Bitcoin se determina por la oferta y demanda. Cuando hay mucha demanda de usuarios que quieren comprar esta criptomoneda el precio sube y cuando es muy baja cae. En el transcurso de los años el precio del Bitcoin ha tenido altas y bajas, esto quiere decir que varía constantemente, también se ha mantenido en rangos estables donde no hay muchos cambios.

El precio del Bitcoin se puede observar en diferentes sitios web. El sitio buybitcoinworldwide.com, muestra en sus inicios valía menos de un dólar, después en febrero del 2011, sobrepaso $1.00 dólar quedando $1.09. Se estará hablando el precio en dólares del Bitcoin a lo largo del documento. Mas tarde en el mismo año 2011, entre mediados de mayo y junio, el precio tuvo un alza de hasta $29.60, como se muestra en la Figura 1.1.

Figura 1.1: Precio del Bitcoin en junio 2011.
Fuente: buybitcoinworldwide.com.

Después en noviembre de ese mismo año llegó de nuevo a los $2.05 aproximadamente. En el 2012, el precio estuvo oscilando entre un rango de los $5.00 a $13.00. De enero del 2013 los precios empezaron a subir dentro de los $15.00 llegando a su punto máximo el 9 de abril a los $230.00. Para el 16 de abril tuvo una baja drástica hasta los $68.36, véase la Figura 1.2.

Figura 1.2: Precio del Bitcoin en junio 2013.
Fuente: buybitcoinworldwide.com.

A finales de diciembre del 2013 se llegó a elevar hasta los $1,147.25 y en finales del enero 2014 se elevó hasta los $990. Después de eso tuvo una tendencia a la baja en su precio, véase la Figura 1.3.

Figura 1.3: Precio del Bitcoin en junio 2013.
Fuente: buybitcoinworldwide.com.

En el 2017, hubo un gran crecimiento en el precio ya que aumentaron más los *Exchanges* para la compra y venta de las criptomonedas. En este año se pudo observar que el precio se elevó de una manera significativa de lo que se podía verse en años anteriores como se mostraron en las gráficas pasadas, véase la Figura 1.4.

Figura 1.4: Precio del Bitcoin en junio 2017.
Fuente: buybitcoinworldwide.com.

Es importante que los usuarios tengan confianza en el Bitcoin y que sea más conocida por los usuarios. El precio está basado en la oferta y la demanda. Mientras haya más personas que compren esta criptomoneda, el precio sube y mientras menos adquieran esta cae como se mencionó anteriormente. Hay otros factores que influyen mucho, por ejemplo: las noticias o eventos que generen desconfianza o incertidumbre.

El valor del Bitcoin puede bajar debido a que si hay muchos usuarios que empiezan a vender, puede generar miedo en las personas porque empieza a tener una tendencia a la baja, entonces las personas venden las criptomonedas lo antes posible para que no pierdan todavía más valor del que están perdiendo en ese momento.

Bitcoin fue la pionera en este nuevo mercado, al igual que esta existen otras que también son importantes, sin embargo, utilizan otro tipo de *Blockchain*. La *Blockchain* del Bitcoin se puede utilizar para crear otras criptomonedas por parte de organizaciones o comunidades de usuarios. Esto se puede hacer ya que es de código abierto y por esto se pueden crear diferentes.

Durante el paso de los años han salido nuevas criptomonedas, pero también han desaparecido por la oferta y demanda, también se debe a problemas que tienen las organizaciones. Entonces el Bitcoin no está absuelto de esto. Aun es un mercado joven y por eso existe mucha

variación en su precio o volatilidad.

Gracias a la variación de precio del Bitcoin existen usuarios que se dedican a hacer *trading*, esto se refiere a comprar y vender estos activos en un *Exchange*. Estos usuarios al comprar esta criptomoneda primero hacen un análisis técnico y uno fundamental. El técnico se refiere a ver los precios en periodos de tiempo del pasado ya sea, minutos, horas, días, semanas, meses y años; a través de gráficas e indicadores técnicos que permitan al usuario estimar el valor de esta criptomoneda. El análisis fundamental, se trata acerca de la investigación de noticias o eventos que puedan alterar el precio. Dependiendo de las conclusiones que hayan sacado deciden comprar o vender sus criptomonedas.

El mercado de las criptomonedas es parecido a Forex, donde se puede hacer *trading* con las divisas de todo el mundo, y se comercia casi de la misma manera, la diferencia radica en la variación de precios, por ejemplo: el dólar a peso mexicano (USD/MXN). Un dólar está en pesos a $19.1023, después de un rato sube a $19.2510, la diferencia fue que incrementó unos centavos más y eso se puede ver en casi todas las divisas. Con grandes cantidades de dinero se ve la diferencia en la ganancia. Pero en el mercado de las criptomonedas la diferencia puede ser de centavos hasta dólares en el caso del Bitcoin.

El precio del Bitcoin suele ser muy volátil en ciertas ocasiones, si no se hace un buen análisis puede que esta criptomoneda este en un precio alto que para uno mismo se supondría que es el precio bajo según la conclusión del análisis, pero, en el mercado empieza a bajar la criptomoneda y al parecer ya no rebasa el valor de la inversión inicial, entonces ahí es cuando se ve que se está obteniendo una pérdida. También que no se sepa hacer un estudio el usuario es más propenso que las obtenga. Aunque tanto en los dos casos puede haber probabilidades de que haya ganancias.

Los usuarios que invierten en Bitcoin tratan de comprarlo a un precio bajo en el mercado y venderlo cuando su valor suba de manera considerable para sacar una ganancia. Claro, habiendo hecho el análisis técnico y fundamental con anterioridad. O también si no se ha

hecho el análisis, todo depende que técnicas use el usuario. Además, lo hacen en tramas de tiempo ya sea en minutos o en horas ya que, al hacer varios movimientos de este tipo, incrementa más la ganancia.

1.1.1. Proyectos relacionados

Existen proyectos que integran diferentes métodos para obtener un aproximado del valor del Bitcoin a futuro. En [3] basa su investigación recabando datos haciendo un análisis de sentimiento a través de los *tweets* y sacando información a través de otras redes sociales.

En [4] se analizan las series de tiempo del precio de Bitcoin con una red neuronal bayesiana (BNN por sus siglas en inglés) que utiliza información de la *Blockchain* además de las variables macroeconómicas y abordan los precios de los Bitcoins altamente volátiles. En sus conclusiones se menciona que la BNN logra una predicción de dirección relativamente precisa a diferencia de otros modelos que se habla en el artículo.

En este otro estudio [5] se utilizaron los modelos *Long-Short Term Memory* (LSTM), *Autoregressive Integrated Moving Average* (ARIMA) y *Recurrent Neural Networks* (RNN) , donde LSTM fue capaz de reconocer las dependencias a largo plazo. ARIMA no tuvo buenos rendimientos en cuanto a la predicción. LSTM es más tardado en entrenar el modelo que utilizó para hacer la predicción. LSTM superó a RNN en las pruebas, pero no significativamente.

Por último, en [6] utiliza modelos de regresión para predecir el precio del Bitcoin a una hora en el futuro. Además, usan una red neuronal y observan la precisión que tienen. En sus conclusiones mencionan que para complementar la predicción se debía analizar al usuario en la adopción, el comportamiento y las características de flujo financiero dentro de los intercambios de esta criptomoneda.

1.2. Definición del problema

Hacer un análisis antes de realizar una operación en el mercado, es recomendable que el usuario pueda calcular sus riesgos al momento de que quiera comprar y vender un Bitcoin. Ya que, sin hacer un estudio previo se estaría operando al azar con el dinero que invierte.

Por lo tanto, existen usuarios principiantes que quieren invertir en Bitcoin y no tienen idea que información buscar para operar de una mejor manera en el mercado o en que datos pueden basar su decisión para que realicen sus análisis y que puedan comprar o vender en momentos oportunos.

1.3. Objetivos

1.3.1. Objetivo general

Desarrollar un prototipo que permita pronosticar el precio del Bitcoin en el periodo de tiempo de una semana, ayudando al *trader* en la toma de decisiones.

1.3.2. Objetivos específicos

- Investigar artículos o proyectos relacionados sobre el proyecto.
- Determinar los factores que alteran el precio del Bitcoin conforme al análisis técnico y fundamental.
- Obtener el historial de precio del Bitcoin para hacer gráficas en el análisis técnico.
- Obtener noticias, eventos e información relacionada con el Bitcoin que pueda alterar su precio para el análisis fundamental.

- Diseñar un modelo que integre los factores que alteran el precio para el pronóstico del Bitcoin.
- Hacer pruebas para observar la probabilidad de que se cumpla el pronóstico en un periodo de tiempo determinado.

1.4. Hipótesis

Mediante el análisis y modelado de las fluctuaciones pasadas en el precio del Bitcoin es factible pronosticar su valor futuro con un 50 % de nivel de confianza.

1.5. Justificación

Este proyecto pretende ayudar al usuario a reducir la pérdida de su dinero y aumentar las ganancias en la compraventa del Bitcoin. Aportará beneficios económicos al usuario ayudándolo en el análisis para una mejor toma de decisiones.

Hay usuarios que hacen los análisis para saber en qué momento es mejor entrar en el mercado, pero hay otros que no saben por donde empezar a informarse o en que basar sus decisiones de compra y venta para operar con Bitcoin. Entonces todo radica en el estudio antes de operar para que se puedan calcular esos riesgos que se enfrentan los usuarios.

Este proyecto se basa en diferentes fuentes para recabar información relacionada con el Bitcoin para ayudar a predecir el precio. Además de las técnicas y análisis tradicionales que puede llegar a utilizar el usuario para operar en el mercado. Este proyecto pretende ser una herramienta para ayudar a que se tenga un mejor control de riesgos.

Capítulo 2
Marco teórico

La *Blockchain* es una tecnología o conjunto de tecnologías que sigue en auge en diferentes áreas, no solo en criptomonedas, si no en otros ámbitos por ejemplo, la transferencia segura de archivos. En el campo de las criptomonedas, las transacciones que se realizan en ella son de manera transparente y sin intermediarios.

Las redes neuronales artificiales pretenden modelar matemáticamente, las neuronas que se encuentran en el cerebro para la realización de sistemas de aprendizaje automático. Es decir que un programa pueda aprender como lo hace una persona. Sin embargo se verá a lo largo del capítulo que algunas redes neuronales artificiales no sacan el mejor rendimiento, como lo haría una red neuronal recurrente en específico las LSTM enfocándose en este proyecto.

El análisis de estas dos tecnologías servirán de construcción para los pronósticos.

2.1. Blockchain

En español significa cadena de bloques. En la aplicación de las criptomonedas, sería como un libro contable muy grande donde se quedan registradas las transacciones que hacen los usuarios a través de su red. En [7] se menciona que es una base de datos distribuida pública de registros. En la red de la *Blockchain*, se realizan las transacciones sin usar intermediarios

porque lo hace de manera descentralizada.

En ella se registra bloques de información, en este caso son las transacciones que hacen los usuarios y los enlaza con el bloque anterior para verificar que esta no haya sido modificada. Todo esto se hace a través de la red P2P que maneja la *Blockchain*.

La *Blockchain* tiene un conjunto de tecnologías que hace posible las transacciones en su red. Para que se pueda realizar una transacción se necesitan, los bloques de información, carteras digitales, nodos y los mineros. Todo esto funciona dentro de la red del *Blockchain*.

2.1.1. Tipos de Blockchain

Existen dos tipos de *Blockchain*:

Las *Blockchains* privadas son usadas por una o varias organizaciones, donde todo esta centralizado y solo miembros autorizados pueden estar en ella. Además, los detalles que suceden en la red no están disponibles para todos. Un ejemplo de esto sería el envío de documentos confidenciales en la organización o que dispongan de su propia criptomoneda y quieran tener el control de su activo.

En los *Blockchains* públicos cualquiera puede leer los datos que almacena. Todas las transacciones son públicas y los usuarios pueden mantenerse anónimos. Bitcoin y Ethereum son ejemplos destacados de este tipo.

2.1.2. Bloques

Los registros de una transacción de esta base de datos distribuida son bloques de información. Estos bloques están enlazados y son cifrados. Por lo que ayudan a saber el camino que llevó a través de la red.

Cada bloque que forma parte de la cadena (excepto el bloque que inicia la cadena), en

[8] hablan de que están formados por:

- Un código alfanumérico que está enlazado con el bloque anterior.
- El paquete de transacciones que incluye (cuyo número viene determinado por diferentes factores).
- Otro código alfanumérico que estará enlazado con el siguiente bloque, esto se puede ver en la Figura 2.1.

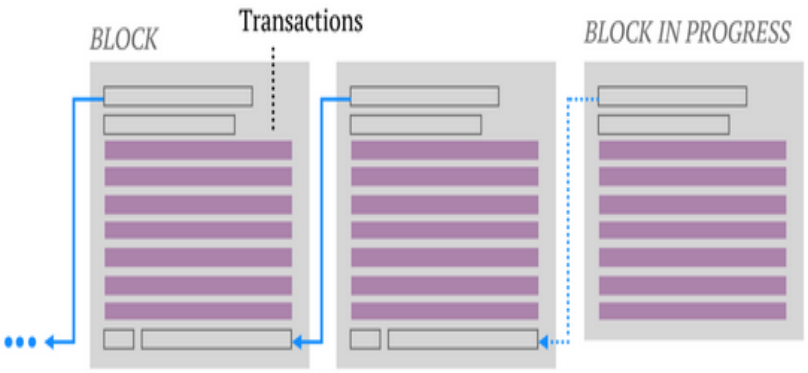

Figura 2.1: Cadena de bloques.
Fuente: https://academy.bit2me.com/que-es-cadena-de-bloques-blockchain.

2.1.3. Carteras digitales

Las carteras digitales pueden ser sitios web que ofrecen el servicio de almacenar las criptomonedas, en este caso el Bitcoin. Además, las casas de cambio también ofrecen este servicio. Un hardware también puede ser una cartera digital como un dispositivo de almacenamiento USB especializada para guardar criptomonedas, pero utiliza una interfaz gráfica para la interacción con el usuario.

2.1.4. Nodos

En los nodos se encuentra distribuido la base de datos, como se mencionó anteriormente, es la base de datos pública. Por eso se dice que está distribuida porque cada nodo en la red tiene una copia de las transacciones. Por lo que un intento de modificación en la transacción se verifica por otros subconjuntos de nodos llamados mineros.

Los mineros son equipos de hardware que pueden ser de potencia baja o alta, por ejemplo, laptops, computadoras de escritorio hasta clústeres de tarjetas gráficas. En [8] menciona que estos se dedican a resolver cálculos matemáticos de los bloques para poder verificar la transacción. Son recompensados con criptomonedas, en este caso es el Bitcoin, por brindar sus servicios y mantener la red activa.

LA *Blockchain* tiene una red P2P, donde los mineros reciben notificaciones de nuevas transacciones y las agrupan en un bloque nuevo. Compiten con otros mineros ya que, el primero que logra crear un bloque válido y lo sella recibe Bitcoins por ese servicio.

Con la *Blockchain*, se sincroniza entre los nodos y esto evita que se de una alteracion de datos en las transacciones, lo que permite que nadie haga cambios al sistema o realice fraudes para beneficiarse, modificando el libro de cuentas para desviar dinero (Bitcoins) de un lado a otro sin que otros se enteren.

2.1.5. Transacciones

Las transacciones en la *Blockchain* son anónimas, los usuarios no dan datos personales, pero la red necesita una dirección para poder transferir. En el siguiente ejemplo, la cartera A es un usuario llamado Alonso y la cartera B es un usuario llamado Pedro. Lo único que necesita la cartera A es la dirección de la cartera B, véase la Figura 2.2.

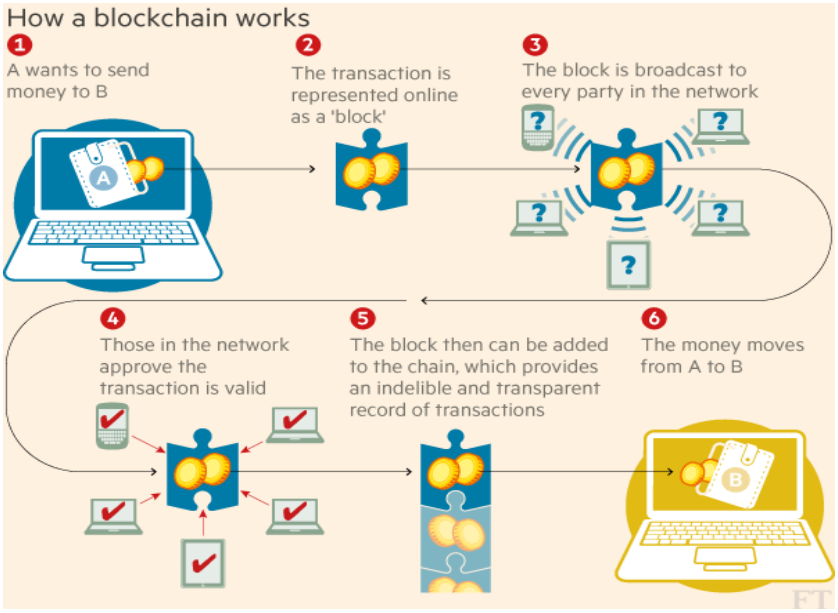

Figura 2.2: Proceso de transacción en la Blockchain.
Fuente: https://academy.bit2me.com/que-es-cadena-de-bloques-blockchain/

Cuando la cartera A ingresa la dirección de la cartera B con la cantidad en Bitcoins a transferir, entonces la transacción se representa en un bloque, como se puede apreciar en el paso 2. En el paso 3, los mineros reciben el bloque y tratan de validarlo. Una vez que se valida, se añade a la cadena de bloques, este es el paso 5. Por ultimo el monto en Bitcoins llega a la cartera B.

2.2. Trading

El *trading* consiste en la compraventa de activos cotizados con mucha liquidez de mercado (sobre todo, acciones, divisas y futuros) en un mercado financiero electrónico y regulado [9]. La liquidez es la facilidad con la que un activo puede convertirse en dinero en efectivo.

En el mercado de las criptomonedas se puede hacer *trading* ya que son consideradas como un activo. Y de las criptomonedas, se puede obtener un beneficio económico cuando aumenta

su valor, que en este caso hablamos del Bitcoin, de su compraventa.

Los usuarios que se dedican hacer *trading* son llamados *traders*. Hay en [8] diferentes tipos de *trading* dependiendo del tiempo que permanecen las operaciones abiertas de compra o venta por parte del *trader*:

- *Day trading*: El *trader* abre y cierra sus operaciones en el mismo día ya sea con un beneficio o una pérdida en lo económico.
- *Scalping*: Este tipo se opera en periodos muy cortos en un día. Pueden durar segundos.
- *Swing trading*: Pueden durar las operaciones días.
- *Trading* tendencial: No tienen un lapso temporal y consiste en tomar posiciones a favor de una tendencia al alza o la baja.

  Para hacer *trading* se necesita de una computadora, conexión a Internet y estar registrados en una *Exchange* en la web. Esta última es una plataforma de intercambio de criptomonedas a dinero o viceversa.

2.2.1. Análisis en trading

Para un *trader* es muy importante realizar un análisis antes de hacer una operación en el mercado ya que, le permite calcular sus riesgos de una forma controlada. Hay dos tipos de análisis que combinados pueden ayudar a una mejor toma de decisiones: el fundamental y técnico.

El análisis fundamental conforme al mercado de las criptomonedas. Es el análisis que trata de recopilar información como son las noticias o cualquier evento que pueda hacer que el precio del Bitcoin cambie y que, además, no se pueda medir con datos cuantitativos para estimar su tendencia.

Este análisis se basa en el estudio de datos cuantitativos como es el precio y el volumen que tienen las criptomonedas. El análisis técnico se basa en la consideración del mercado proporcionando el estudio del comportamiento de la acción, analizando el pasado y proyectándose hacia el futuro [10]. En el Bitcoin se tendrá que ver el comportamiento que tiene analizando su precio pasado y el volumen para hacer una proyección de una tendencia al alza o a la baja en un futuro.

También ayuda a anticipar los cambios de tendencia que tendrá el precio del Bitcoin. Con ayuda de gráficos, el que se usa comúnmente es el de velas japonesas. La forma en que están graficado es conforme a los precios y el volumen que hay en el mercado.

2.3. Series de tiempo

Por serie de tiempo, se refiere a datos estadísticos que se recopilan, observan o registran en intervalos de tiempo regulares (diario, semanal, semestral, anual, entre otros) [11]. Los análisis en series de tiempo tienen los siguientes objetivos:

- Determinar si se observan ciertos patrones o movimientos no aleatorios.
- Estudiar los patrones que se han dado en el tiempo para ver posibles movimientos futuros.
- Hace posible pronosticar los movimientos futuros, así como otros aspectos que estén sincronizados.

Para realizar un análisis de este tipo, primero se identifican los componentes de la serie de tiempo, después aplicar las técnicas estadísticas para su análisis y, finalmente, hacer las proyecciones o pronósticos de eventos futuros.

Es común usar las series de tiempo en Bitcoin ya que, al operar en el mercado se analiza el precio en movimientos pasados para pronosticar un movimiento futuro.

El método clásico identifica cuatro influencias o componentes:

- Tendencia.
- Fluctuaciones cíclicas.
- Variaciones estacionales.
- Variaciones irregulares.

2.4. Redes neuronales artificiales

Las redes neuronales es un modelo simplificado que emula el modo en que el cerebro procesa la información [12]. Tienen como objetivo dar lugar a un sistema inteligente que haga tareas complejas.

Existen unidades de procesamiento que están interconectadas y funcionan de manera simultánea, por esto mismo es que asemeja al comportamiento biológico de las neuronas. En estas unidades de procesamiento se organizan en capas. Hay tres partes en una red neuronal, la capa de entrada, capas ocultas y la capa de salida, véase la Figura 2.3.

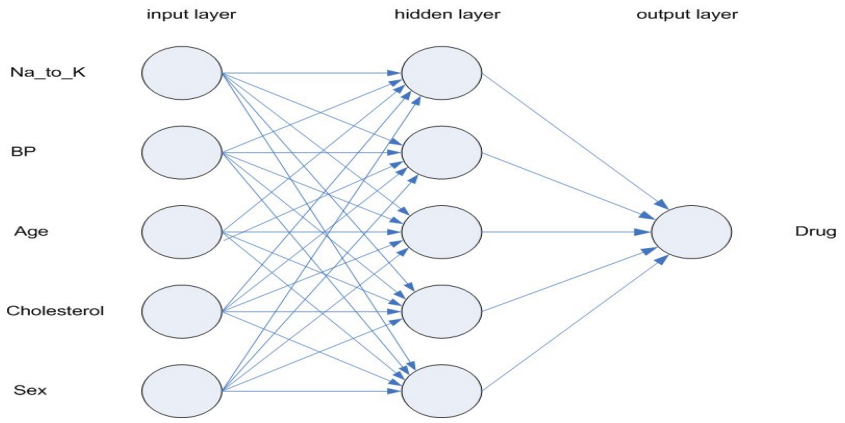

Figura 2.3: Estructura de una red neuronal.
Fuente: ibm.com.

Los datos se ingresan en la primera capa, los valores se propagan hasta las siguientes neuronas de cada capa y al final arroja un resultado. La red aprende examinando los registros que se hayan ingresado, luego realiza una predicción para cada registro. Si no se obtiene el resultado adecuado entonces se repite el mismo procedimiento para mejorar sus predicciones, de manera que se va entrenando la red para que de mejores resultados.

Al final del entrenamiento se comparan con los resultados que son conocidos para evaluar la efectividad que tiene la red. Conforme se progresa en el entrenamiento, será más precisa en los resultados que son conocidos para cuando esté entrenada la red, se pueda utilizar para hacer predicciones a futuro donde se desconocen los resultados.

En este caso los datos que utilizará para entrenar la red, son los factores que alteran el precio del Bitcoin. Se entrenará la red para que pueda predecir los resultados que se conocen, ya sea de una fecha a otra que previamente se conoce cuál fue el precio.

Esto para evaluar la efectividad que presenta la red y cuando se obtengan resultados más precisos, se aplicará para hacer predicciones a futuro del precio. Aquí en esta parte es donde se deberá enfocar. Hay diferentes tipos de redes neuronales y existen varias aplicaciones, una de ellas es en las series de tiempo.

2.5. Interfaz de programación de aplicaciones

Una interfaz de programación de aplicaciones (API por sus siglas en inglés), es un conjunto de herramientas, definiciones o protocolos que se usa para diseñar e integrar al software de las aplicaciones [13]. Permite que cuando se esté desarrollando un *software* se pueda comunicar con otro software o servicio sin la necesidad de saber cómo está programado o estructurado.

Se utilizará otra API del sitio web *Blockchain* para obtener datos cuantitativos para el análisis técnico.

Capítulo 3
Desarrollo del Proyecto

Se llevó a cabo un proyecto de desarrollo tecnológico, donde se realizó el pronóstico del precio de Bitcoin en el periodo de tiempo de una semana. Se tuvo que hacer un estudio experimental donde se sacaron datos históricos de la *Blockchain* y así obtener los precios a futuro a través de redes neuronales recurrentes específicamente las LSTM. Las librerías de *Tensorflow* y *Keras* ayudan a construir fácilmente modelos de redes neuronales. El poder computacional es de gran importancia para el entrenamiento ya que, se necesita buen rendimiento para la construcción de RNR y así se pueda obtener los resultados esperados.

3.1. Producto propuesto

El prototipo que se desarrolló, es un programa que extrae datos de la *Blockchain*, los almacena en archivos con extensión CSV, después estos sirven como datos de entrada anteriormente preprocesados, es decir, se ajustan para la red neuronal. Los datos que se tienen almacenados, funcionaron como entrenamiento, ya que se utilizaron los datos anteriores de 2018 como datos de entrada y los datos del 2019 hasta junio del mismo año como evaluación del modelo de la red neuronal. Después de este entrenamiento, sirvió de aprendizaje y mostró los precios de la siguiente semana en una gráfica.

Como se describió en los antecedentes, el prototipo va dirigido a usuarios que invierten

en el Bitcoin, ayudando en sus análisis antes de operar, tomando en cuenta los datos de la *Blockchain*.

3.2. Descripción de la metodología

La metodología que se usó fue la de diseño experimental. Se eligió este tipo de metodología, ya que es la que mejor se adoptó al problema. Como se puede ver en la Figura 3.1.

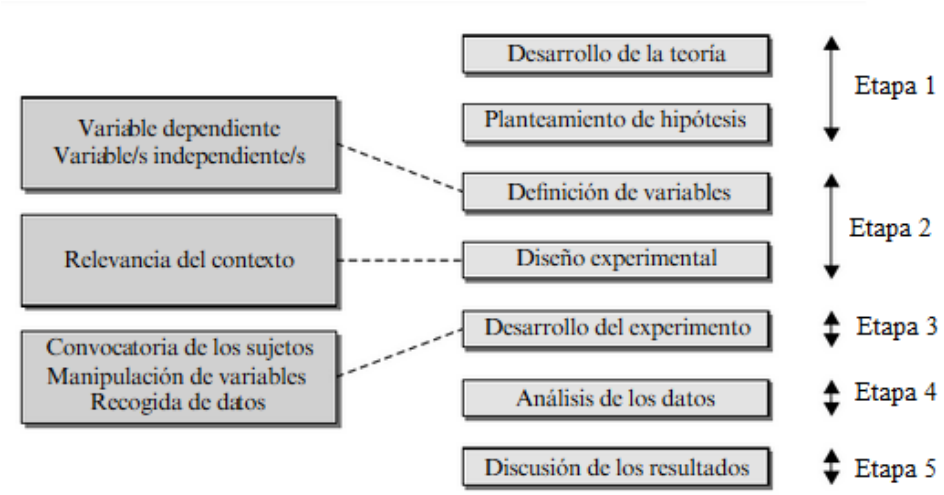

Figura 3.1: Metodología experimental.
Fuente: researchgate.net.

1. En la primera etapa se tuvo que tener un planteamiento del problema, puesto que contribuye a una mejor visualización del panorama y de la solución final del problema. Luego se sigue con el planteamiento de la hipótesis, que son especulaciones sobre el comportamiento del fenómeno a estudio. Como se vio anteriormente en el capítulo 1.

2. En la siguiente etapa consistió en realizar la operación de la hipótesis, esto quiere decir que los conceptos que se explicaron en este, se representaron en un conjunto de variables para la determinación de un diseño experimental apropiado.

3. En la tercera etapa se llevó a cabo el experimento, aquí se hizo el programa conforme al diseño del experimento.

4. En esta etapa se emplearon métodos estadísticos para analizar los datos, de modo que los resultados y conclusiones son objetivos más que apreciativos.

5. La última etapa se obtienen las conclusiones del proyecto. Se comparan los resultados con la hipótesis planteada confirmando las mismas o buscando explicaciones adicionales para el caso de que los resultados sean contrarios a las hipótesis.

En esta metodología, ya se lleva a cabo desde el capítulo 1, donde se plantea el problema y la hipótesis. En las próximas etapas se desarrollan en este capítulo.

3.3. Etapas

3.3.1. Definición de variables

La determinación de las variables que tienen mayor influencia en la variable de respuesta se sacaron del sitio blockchain.com. Es un sitio web que funge como un intermediario que proporciona servicios de explorador de datos sacados de la misma *Blockchain*. Esta pagina ofrece información de diferentes criptomonedas, sin embargo, los datos que se sacaron fueron del Bitcoin.

A continuación, se encuentran listados las variables que se utilizaron en el proyecto.

Variables independientes:

- Total de Bitcoins en circulación: El número total de Bitcoins que ya se han extraído; en otras palabras, el suministro actual de Bitcoins en la red.

- Capitalización del mercado: El valor total en USD del suministro de Bitcoins en circulación, calculado por el precio promedio diario del mercado en las principales bolsas.
- Volumen de intercambio de USD: El valor total en USD del volumen de negociación en los principales intercambios de Bitcoins.
- Tamaño de la Cadena de Bloques (Blockchain): El tamaño total de todos los encabezados de bloque y transacciones. No incluye índices de bases de datos.
- Tamaño del bloque promedio: El tamaño promedio de bloque *Megabytes*.
- Transacciones por bloque: El número promedio de transacciones por bloque.
- La mediana de tiempo de confirmación (de pago): El tiempo medio para que una transacción se acepte en un bloque minado y se agregue al libro mayor público (nota: solo incluye transacciones con tarifas de minero).
- Tarifa de *hash*: La cantidad estimada de tera *hash* por segundo (billones de *hashes* por segundo) que la red Bitcoin está realizando.
- Dificultad de encontrar un nuevo bloque: Una medida relativa de lo difícil que es encontrar un nuevo bloque. La dificultad se ajusta periódicamente en función de la cantidad de poder de *hashing* que ha desplegado la red de mineros.
- Ingresos de explotación: Valor total de las recompensas en bloque de *Coinbase* y tarifas de transacción pagadas a los mineros.
- Tasa total de la transacción (BTC): El valor total de todas las tarifas de transacción pagadas a los mineros (sin incluir el valor de la base de monedas de las recompensas en bloque).

- Tarifa de transacción total (USD): El valor total de todas las tarifas de transacción pagadas a los mineros (sin incluir el valor de la base de monedas de las recompensas en bloque).
- Porcentaje de costo del volumen de transacciones: Los ingresos de los mineros como porcentaje del volumen de la transacción.
- Costo por transacción: Los ingresos de los mineros divididos por el número de transacciones.
- Direcciones únicas: El número total de direcciones únicas utilizadas en la cadena de bloques de Bitcoin.
- Número total de transacciones por día: El número de transacciones diarias confirmadas de Bitcoin.
- Número total de transacciones.
- Tasa de transacciones: El número de transacciones de Bitcoin agregadas al *mempool* por segundo.
- Número de transacciones *mempool*: El número de transacciones que esperan ser confirmadas.
- *Mempool* tamaño de crecimiento: La velocidad a la que crece el *mempool* por segundo.
- Tamaño *mempool*: El tamaño agregado de las transacciones que esperan ser confirmadas.
- Número de salidas de transacciones no utilizadas: El número de salidas de transacciones de Bitcoin no gastadas, también conocido como el tamaño del conjunto UTXO.

- Volumen de transacciones excluyendo direcciones populares: El número total de transacciones de Bitcoin, excluyendo aquellas que involucran cualquiera de las 100 direcciones más populares de la red.
- Volumen de transacción excluyendo las cadenas largas: El número total de transacciones de Bitcoin por día, excluyendo aquellas que forman parte de largas cadenas de transacciones.
- Volumen total de salida: El valor total de todas las salidas de transacciones por día (incluye las monedas devueltas al remitente como cambio).
- Volumen estimado de transacciones: El valor total estimado de las transacciones en la cadena de bloques de Bitcoin (no incluye las monedas devueltas al remitente como cambio).
- Volumen estimado de las transacciones en USD.

Variable dependientes:

- Precio de Mercado del Bitcoin (USD).

Con estas variables, sirven para representar la hipótesis y ponerlas en operación para las siguientes etapas. Entonces, el precio del Bitcoin se ve afectado por estas variables independientes.

3.3.2. Diseño experimental

Anteriormente se habló acerca de las variables que se usaron en el proyecto. Entonces para la obtención, se tuvo que buscar en diferentes sitios web que ofrecieran este servicio como explorador de datos de la *Blockchain* y que tuvieran una API. Entonces el sitio blockchain.com

es el mas completo que se encontró. La API permitió en este caso, acceder al servidor y extraer los datos relacionados al Bitcoin.

El sitio proporciona diferentes gráficos del Bitcoin, de estos, es donde se extrajeron los datos. La API tiene la siguiente nomenclatura, véase la Figura 3.2.

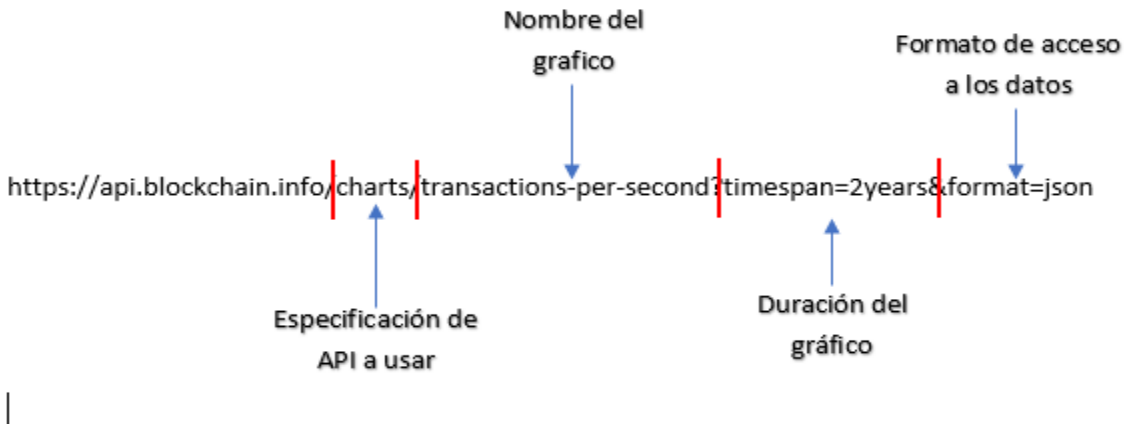

Figura 3.2: Metodología experimental.
Fuente: Elaboración propia.

Los parámetros de fecha se representan como AAAA-MM-DD hh:mm:ss o AAAA-MM-DD. La zona horaria es UTC. Las duraciones se representan concatenando el número de unidades de tiempo y la unidad de tiempo que representa (por ejemplo, '1 año', '3 meses', etc.). Las unidades de tiempo que están disponibles son: minuto, hora, día, semana y año.

La duración del tiempo en los gráficos, el valor predeterminado es de 1 año para la mayoría, una semana para los gráficos *mempool*. En este caso se tuvieron que sacar datos de 30 días, de 2 años y datos desde el año 2009.

La comunicación funciona de la siguiente manera, se hace una solicitud de los datos que necesitan extraer al servidor a través de la API y devuelve una respuesta. Actúa como intermediario entre el cliente y el servidor para la comunicación, véase la Figura 3.3.

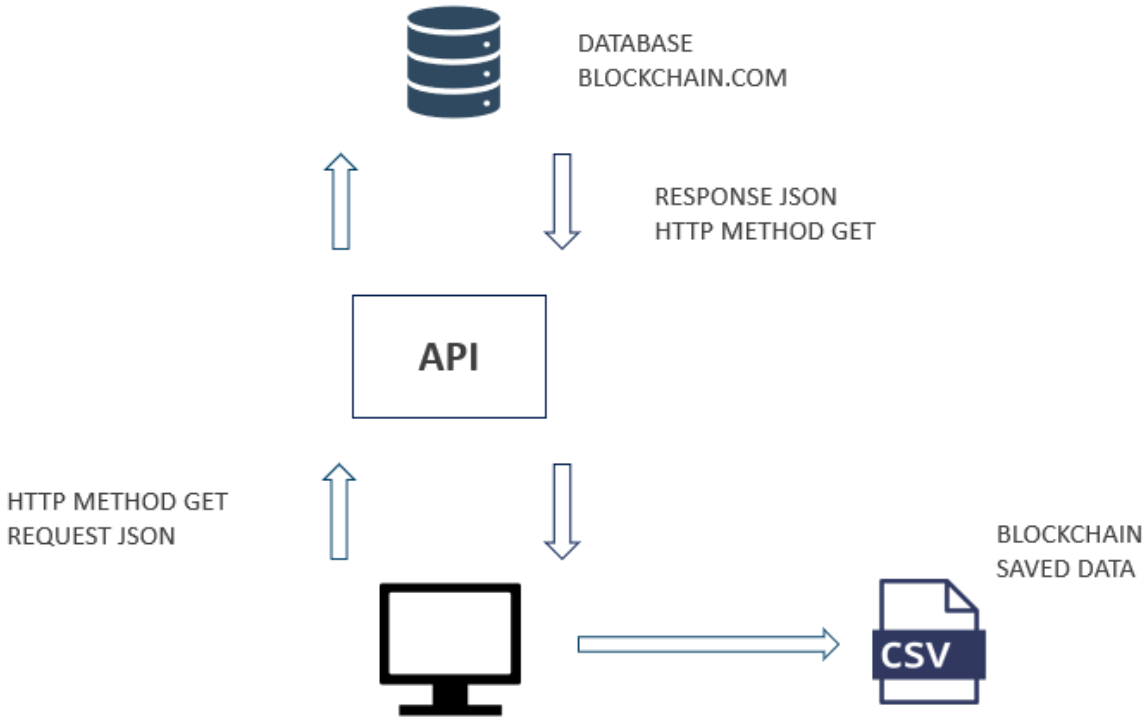

Figura 3.3: Solicitud de datos con API.
Fuente: Elaboración propia.

En la construcción del modelo se realizaron los siguientes procesos. Para ver el marco general creado, véase la Figura 3.4.

Figura 3.4: Marco general de las etapas del modelo.
Fuente: Elaboración propia.

3.3.3. Desarrollo del proyecto

Programa de recolección de datos

Los siguientes programas fueron desarrollados en Python 3.7. Las especificaciones de la computadora donde se realizó el proyecto son las siguientes:

- Tipo: Laptop Dell 5580.
- Procesador: i5-7440.
- Unidad de almacenamiento: Solid State Drive.
- RAM: 8GB DDR4.

Para llevar a cabo lo que anteriormente se describió en el diseño, se realizó en orden todo este proceso. Lo que primeramente se hizo fue la extracción de datos. Comenzando con el desarrollo se hizo una función llamada *blockData* donde abre los archivos de texto que contiene todas las *url*. En total son tres archivos ya que cada uno son para sacar datos históricos de diferentes lapsos de tiempo. A continuación tienen los siguientes nombres:

- *BlockAllData*
- *BlockData2Y*
- *BlockData30d*

Esta función lo que hace es leer cada línea del archivo y después le envía la *url* a la función *blockAPI*.

En la función *blockAPI* se recibe la *url* que es el acceso de cada gráfico, donde se sacó la información. Con la librería *request* permite la comunicación con los servidores a través de la API, entonces se hizo una llamada al servidor por medio del método *GET*.

Al momento de hacer esta petición en el sitio Blockchain.com en la misma *url*, se esta especificando el formato al que se va a acceder, en este caso es de tipo JSON. A continuación, en la Lista 1 se observa la estructura.

```
{
  "status": "ok",
  "name": "Confirmed Transactions Per Day",
  "unit": "Transactions",
  "period": "day",
  "description": "The number of daily confirmed Bitcoin transactions.",
  "values": [
    {
      "x": 1442534400, // Unix timestamp (2015-09-18T00:00:00+00:00)
      "y": 188330.0
    },
    ...
}
```

Fuente: Elaboración propia.

Lista 1: Estructura JSON de los datos.

El archivo JSON en sí es un diccionario, donde se utilizaron las siguientes variables para posteriormente manejar datos con más facilidad:

- *Name*
- *Unit*
- *Period*
- *Values*

En la variable *Values*, se dividen en dos variables, en X y Y. Entonces a X, se tomó como la fecha, en este caso está en formato *Unix* por lo que se tuvo que convertir. Después Y son los valores que representan cada gráfico.

Entonces para separar en este caso las dos variables que se encuentran en *Values*, se procede a normalizar el JSON. El código quedaría como se muestra en la Lista 2.

```
response  = requests.get(url)
response_json = json.loads(response.text)
values = json_normalize(response_json['values'])
```

Fuente: Elaboración propia.

Lista 2: Petición a los datos a través de API.

Luego de esto, el servidor manda una respuesta con los datos que se le han pedido. Para poder manejar y crear una estructura personalizada, la librería *Pandas* permite realizar un marco de datos, definiendo en este caso el nombre de las columnas para la organización.

Como se dijo anteriormente, la variable X se tomó como la fecha, entonces, *Pandas* también ayuda a cambiar el formato de la fecha. Se quedó con la estructura: AAAA:MM:DD hh:mm:ss. En la Lista 3 se observa como esta construido el marco de datos conforme a los valores de los archivos CSV que usan los gráficos del sitio web.

```
df = pd.DataFrame({"Date":values['x'], "Values":values['y'],
                   "Name":response_json['name'],
                   "Period":response_json['period'],
                   "Unit":response_json['unit']})

df['Date'] = pd.to_datetime(df['Date'], unit='s')
```

Fuente: Elaboración propia.

Lista 3: Construcción de los archivos CSV a guardar.

Anteriormente en el diseño del experimento, se dijo que las *URL* son de datos históricos desde 2009 y de 2 años atrás. La diferencia es que los datos de 2 años de antigüedad que están accesibles, aparecen seguidos los días, mientras que los del 2009 en adelante se muestran cada tres días.

Así maneja la información el servidor al momento de que el programa se conectó con la API. Entonces, lo que se realizo fue unir los datos históricos de 2 años y los que son desde el 2009 en un solo marco de datos.

Cuando los datos ya estuvieron preparados se guardan en un archivo CSV. Si es la primera vez que se van a guardar, el programa crea una carpeta llamada *CSV*. En caso de que ya este la carpeta creada y los archivos, al momento de que se mande a llamar esta función nuevamente, actualizará los datos que faltan en los archivos, véase la Figura 3.5.

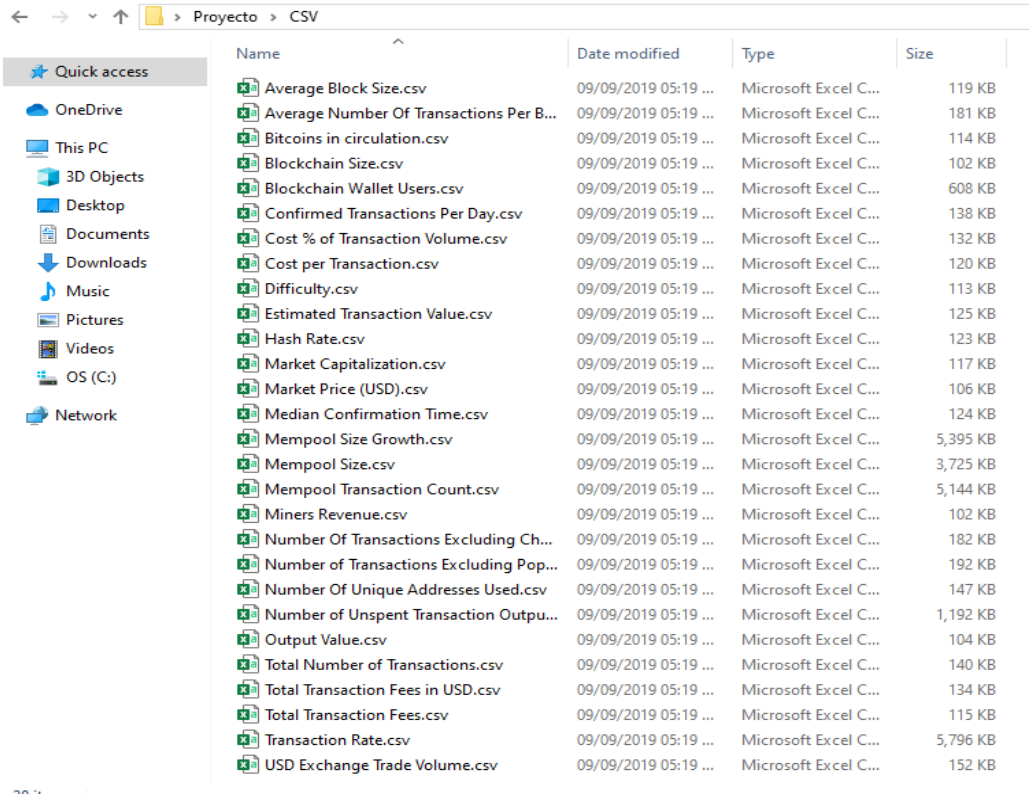

Figura 3.5: Archivos extraídos de las API.
Fuente: Elaboración propia

Al momento de la actualización, los datos históricos con lapsos de tiempo desde el 2009, de dos años y los de 30 días, se descubrió que en ciertas APIs, por ejemplo, en los gráficos *mempool*, se añaden nuevos registros y que cambian al momento de hacer nuevas peticiones al servidor. Esto se puede observar puesto que, en algunos registros, un día puede tener muchos valores, dependiendo la trama de tiempo que se maneje. Entonces al momento de llamar a la API se encuentran nuevos registros desde hace cuatro años atrás.

Entonces, llamar de nuevo a la API para buscar nuevos datos, los registros se duplicaron y se arregló implementando una función a la que se llamó *dropDuplicateData()*. Lo que hace es simplemente abrir cada uno de los archivos, los mete en un marco de datos y con *Pandas* elimina las tuplas duplicadas guiándose por la fecha, vease la Lista 4.

```
def dropDuplicateData():
        path = 'CSV/'
        csv_files = glob.glob(path +'*.csv')

        for filename in csv_files:
                print(filename)
                data = pd.read_csv(filename)
                df = pd.DataFrame(data)
                df = df.sort_values('Date')
                df.drop_duplicates('Date', inplace=True, keep='last')
                df.to_csv("{}".format(filename, header=0),
                            encoding='utf-8', index=False)
```

Lista 4: Función para eliminar datos duplicados.

Programa de modelo RNR LSTM con múltiples variables

Primero se importan las siguientes librerías para la creación de la red neuronal. Se utilizaron *Tensorflow* version 2.0 y su API de alto nivel *Keras*. Después la librería *Pandas* se usó para trabajar con los datos y *numpy* para la creación de matrices. *Sklearn* sirvió para el preprocesamiento de los datos, de modo que el modelo de red neuronal recurrente la interprete. *Keras* está a un nivel de abstracción mayor que *Tensorflow*, pero juntos son fáciles para la creación de modelos de aprendizaje automático, véase la Lista 5.

```
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
from sklearn.preprocessing import MinMaxScaler
import tensorflow as tf
from tensorflow.keras import layers
```

Fuente: Elaboración propia.

Lista 5: Librerías utilizadas para la creación de la RNR.

Previamente en la recolección de los datos, estos se guardaron en archivos CSV. Por lo que realizó un marco de datos con todos los archivos. De cada archivo se utilizó como índice la fecha y los valores de cada uno. Después en el marco de datos se hizo que todos los valores *Nan* sean sustituidos por ceros, véase la Tabla 3.1.

Tabla 3.1: Valores de todos los archivos CSV en un marco de datos.

Date	Values1	Values2	Values3	Values4	Values5	Values6
2009-01-03 00:00:00	0.000285	1.0	50.0	0.0	0.0	1.0
2009-01-06 00:00:00	0.000000	1.0	50.0	0.0	0.0	0.0
2009-01-09 00:00:00	0.000215	1.0	750.0	0.0	0.0	14.0
2009-01-12 00:00:00	0.000232	1.0	12050.0	0.0	0.0	95.0
2009-01-15 00:00:00	0.000242	1.0	30450.0	0.0	0.0	136.0

Para poder empezar con el entrenamiento se utilizaron los datos del 2018 hacia atrás. Estos mismos sirvieron para realizar la validación o pronóstico del precio, en este caso los datos del año 2019 se usaron para esto, véase en la Figura 3.6.

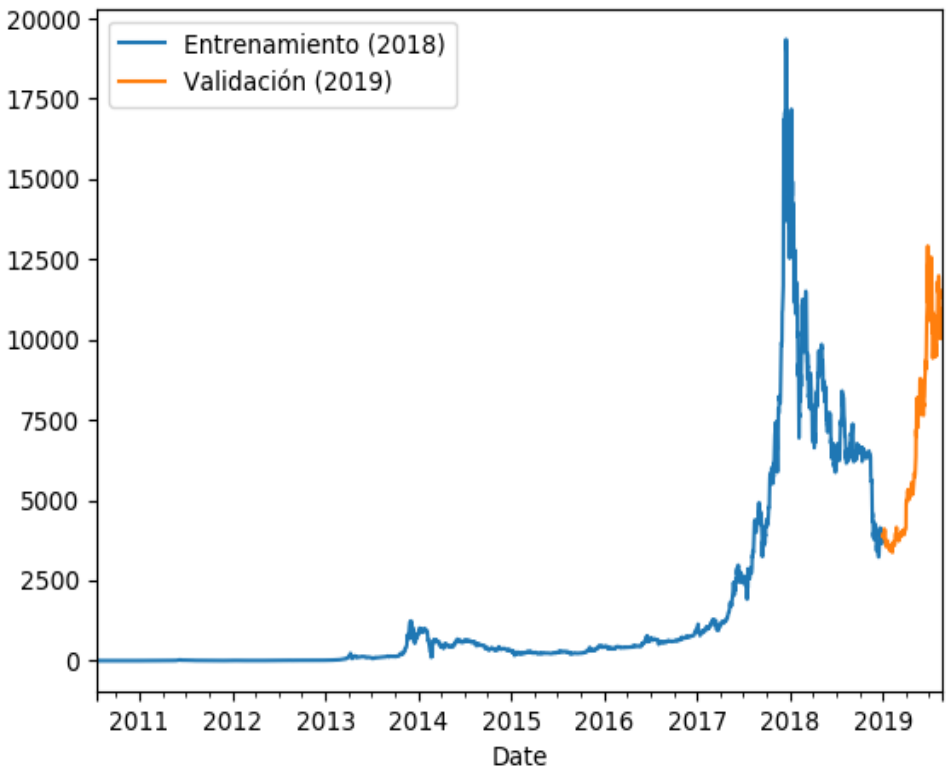

Figura 3.6: Visualización de datos para entrenamiento y validación.
Fuente: Elaboración propia.

Luego se procede a la normalización de los datos, el cual se utilizó *MinMaxScaler* de la librería *sklearn*. Estos estarán transformados en un rango entre -1 y 1. Después se definió el número de pasos de entrenamiento, es decir, se agarraron 60 datos de todas las columnas del marco de datos para la predicción de los 7 días a futuro, a continuación se muestra en la Tabla 3.2.

Tabla 3.2: Rango de datos de entrenamiento y evaluación del precio del Bitcoin.

0.00026311	0	0.00019849	0.0002144	0.00022318	0.00019936
0.00021006	0.00020838	0.0002239	0.00021011	0.00019926	0.00019915
0.00020082	0.0002079	0.00019928	0.00022629	0.00021275	0.00019936
0.00019939	0.00020674	0.00019915	0.00019938	0.00019923	0.00021167
0.00023852	0.00021207	0.00021232	0.00019922	0.00021808	0.00022951
0.00019937	0.00021564	0.00019931	0.00019934	0.00019916	0.00020395
0.00026125	0.00019926	0.00019935	0.00021561	0.0002132	0.00020737
0.00019902	0.00019936	0.00020224	0.00019935	0.00019935	0.00019931
0.00019914	0.00019911	0.00019932	0.00019931	0.00019924	0.00019936
0.00019916	0.00019901	0.00019917	0.00019918	0.00022319	0.00019921

Estos datos se ajustan con la función *reshape* para el modelo de la red neuronal.

La red neuronal como se había planteado en el diseño del experimento, sería una de tipo recurrente, en especial las LSTM. Entonces para empezar se definirán las capas.

- La primera capa es de tipo LSTM de 120 neuronas, se declara la entrada con (60,20) y activación tangente hiperbólica.
- La segunda es de tipo LSTM y consta 60 neuronas y activación tangente hiperbólica.
- La capa de salida es una capa Dense con 1 unidad y activación tangente hiperbólica.

El siguiente código representa lo descrito anteriormente, véase la Lista 6.

```
single_step_model = tf.keras.models.Sequential()
single_step_model.add(tf.keras.layers.LSTM(120,
                        return_sequences=True,
                        activation='tanh',
                        input_shape=x_train.shape[-2:]))
single_step_model.add(tf.keras.layers.LSTM(60, activation='tanh'))
single_step_model.add(tf.keras.layers.Dense(1, activation='tanh'))

single_step_model.compile(optimizer=tf.keras.optimizers.RMSprop(), loss='mse')


single_step_history = single_step_model.fit(train_data, epochs=EPOCHS,
                                             steps_per_epoch=EVALUATION_INTERVAL,
                                             validation_data=val_data,
                                             validation_steps=50)
```

Fuente: Elaboración propia.

Lista 6: Modelo RNR LSTM.

Programa de modelo RNR con una variable

En el siguiente modelo se hace el pronóstico de una semana a futuro con una sola variable,, el cual es el precio de cierre. Se usó el mismo rango de datos con el modelo anterior sin embargo, esta vez se utilizaron datos extraídos de *Yahoo Finance*. El nombre del archivo se llama *yahoo-BTC* y contiene 3396 registros desde el año 2014, véase la Tabla 3.3.

Tabla 3.3: Datos del precio de cierre de Bitcoin.

Date	Close
2014-01-01	815.94
2014-01-02	856.91
2014-01-03	884.26
2014-01-04	924.69
2014-01-05	1014.74

Se usó el mismo rango de datos que el modelo anterior y se normalizó con la escala min-max. Esta forma de normalización ya viene integrada con la librería *scikit-learn*.

La red LSTM tendrá como entrada 5 datos consecutivos, y como salida 1 dato. La predicción a partir de esos pasos consecutivos, se conformará de esta forma el set de entrenamiento. Después se crea el modelo con redes neuronales usando las siguientes capas:

- La primera capa es de tipo LSTM de 50 unidades y se aplicó regularización de deserción (*Dropout*) de 20 %.
- La capa de salida es una capa Dense con 1 unidad.

El siguiente código representa lo descrito anteriormente, véase la Lista 7.

```
modelo = tf.keras.models.Sequential()

modelo.add(layers.LSTM(50, activation='tanh', input_shape=dim_entrada))
modelo.add(layers.Dropout(0.2))
modelo.add(layers.Dense(dim_salida, activation="tanh"))

modelo.summary()
modelo.compile(optimizer='adam',
               loss='mean_squared_error',
               metrics=['accuracy'])

callback = tf.keras.callbacks.EarlyStopping(monitor='val_loss', patience=5)

history = modelo.fit(X_train,Y_train,epochs=50,batch_size=512,
                         validation_data=(X_test, Y_test),
                         shuffle=True,
                         callbacks=[callback])
```

Fuente: Elaboración propia.

Lista 7: Modelo RNR.

Se compila el modelo, se utilizó el optimizador *Adam* y la función de perdida de error medio al cuadrado. Cuando se encajó el modelo en el *set* de entrenamiento, fueron 50 épocas que se utilizaron y un tamaño de lote de 512. La función *EarlyStopping*, se encarga de encontrar la mejor época con mejores resultados y terminar el entrenamiento.

Una vez que se acabo el entrenamiento se puede guardar el modelo de nuestra red neuronal si se llegaron a buenos resultados, en este caso se guardaron varios modelos como *model.h5*. También se puede cargar si se desea.

Para el pronóstico se usaron los precios de cierre de 20 días antes del ultimo registrado. Estos datos son los que sirvieron como datos de entrada, véase en la siguiente Tabla 3.4

Tabla 3.4: Rango de datos para pronóstico.

Date	Close
2019-09-20	10181.64
2019-09-21	10019.72
2019-09-22	10070.39
2019-09-23	9729.32
2019-09-24	8620.57
2019-09-25	8486.99
2019-09-26	8118.97
2019-09-27	8251.85
2019-09-28	8245.92
2019-09-29	8104.19
2019-09-30	8293.87
2019-10-01	8343.28
2019-10-02	8393.04
2019-10-03	8259.99
2019-10-04	8205.94
2019-10-05	8151.50
2019-10-06	7988.16

Después de seleccionar el set de entrenamiento se procedió a realizar el pronostico ya sea con el modelo que se haya cargado o con el modelo que se ejecutó en el programa pero no se haya querido guardar.

Capítulo 4
Resultados y Discusiones

En este capítulo se presentarán los resultados del pronóstico del precio del Bitcoin obtenido conforme al modelo de red neuronal construido, comparándolos con los resultados del mercado Bitcoin contra el dólar americano (BTC-USD). También se podrán ver la exactitud del modelo en su fase de entrenamiento conforme a los pronosticado. Se verán las diferentes métricas que se utilizaron para ver el funcionamiento de la red.

4.1. Resultados del entrenamiento con múltiples variables

Los resultados que se obtuvieron con los datos de la *Blockchain*, tuvieron una buena respuesta. El pronóstico que se hizo fue del lapso de tiempo de una semana. Esto solo fue un entrenamiento para la red neuronal y observar la precisión, véase la Figura 4.1

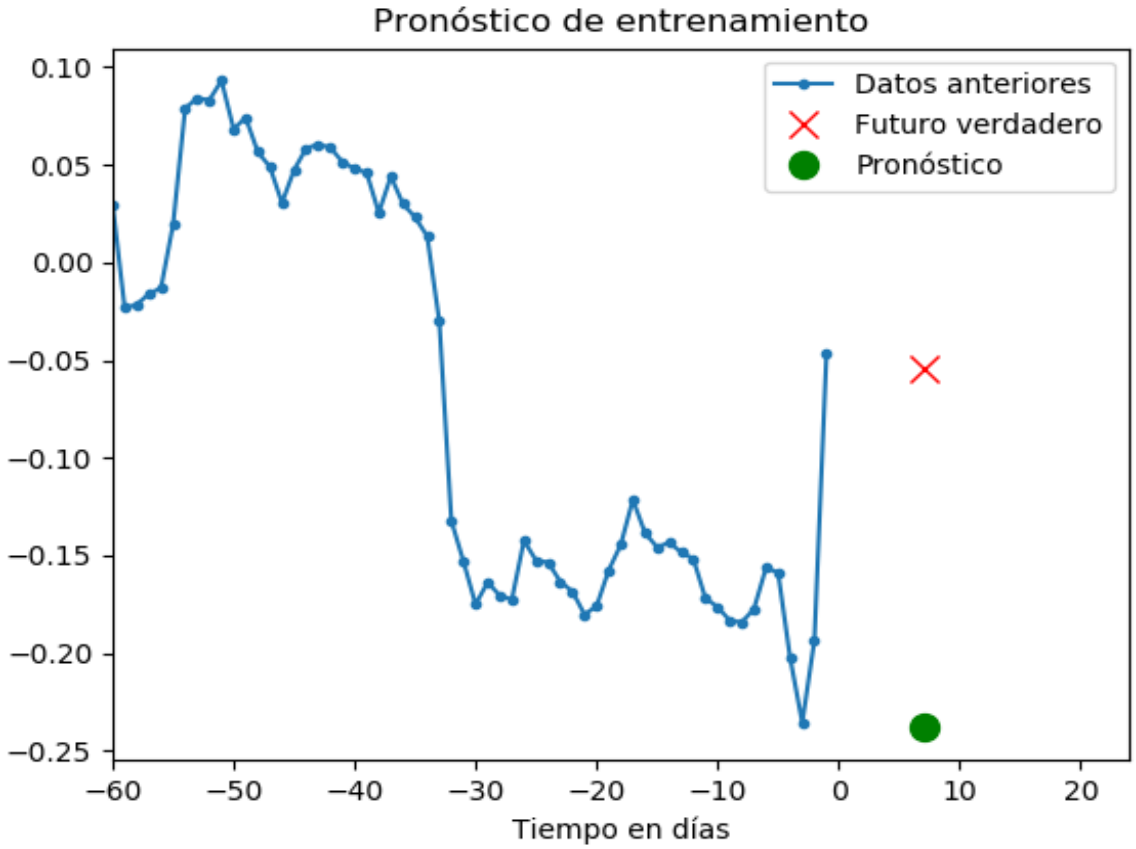

Figura 4.1: Resultado del entrenamiento con múltiples variables.
Fuente: Elaboración propia.

El tiempo que se selecciono de entrenamiento fue de 60 días anteriores. En este modelo las fechas que sirvieron para comparar el pronóstico es de 13 al 19 de agosto del 2019. Con este resultado nos muestra que los datos de la *Blockchain* están relacionados con el precio del Bitcoin, sin embargo, presentan una precisión deficiente.

4.2. Resultados del entrenamiento con una variable

Una vez que se construyó el modelo de redes neuronales se hizo su entrenamiento. En los resultados se puede apreciar que se acerca al precio real. Sin embargo, en el pronóstico, se

observa que en ciertos puntos existen picos donde el precio sube o baja con fuerza y es ahí donde el modelo presenta esa deficiencia debido a la alta volatilidad en tiempos cortos, véase la Figura 4.2.

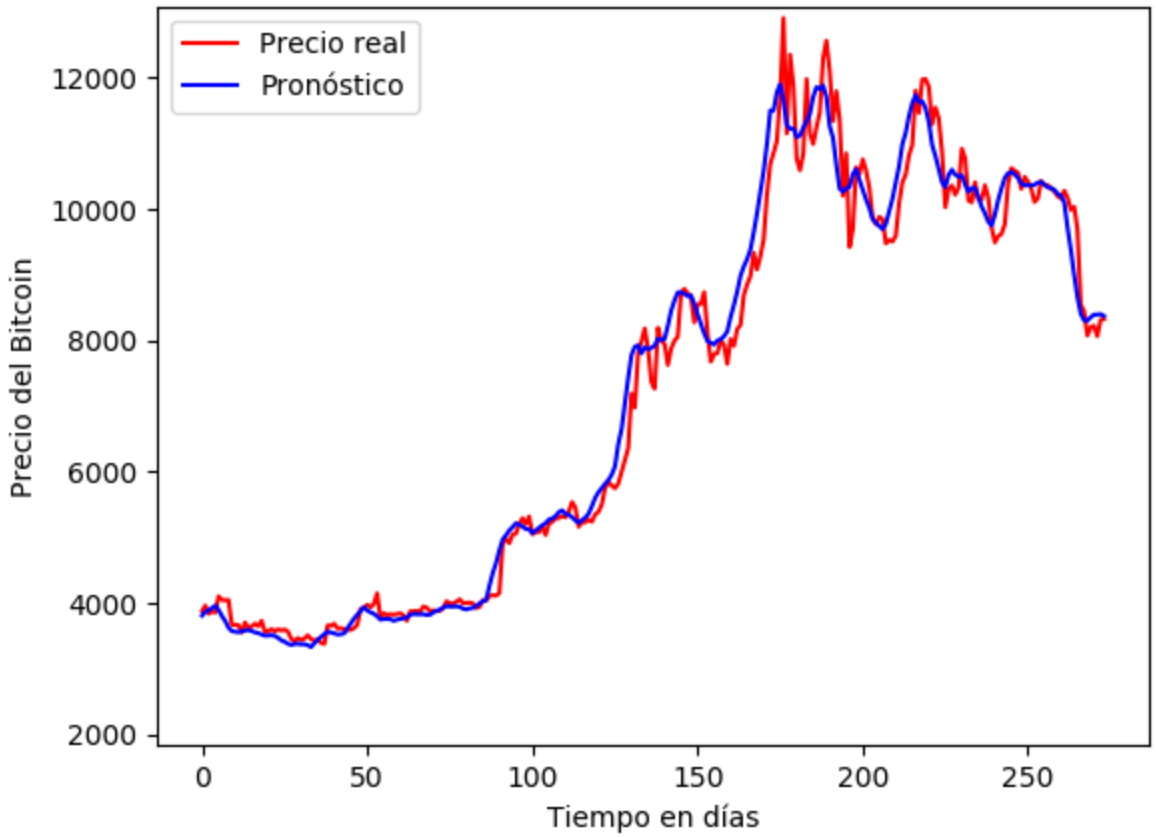

Figura 4.2: Resultado del entrenamiento con una variable.
Fuente: Elaboración propia.

Como se había mencionado en el capítulo 3, los datos de validación son del año 2019. Al observar estas comparaciones la diferencia mínima oscila entre los $100 a $300 USD. En los picos donde se observa que hay mucha volatilidad, va desde los $800 a $2000 USD.

A partir del domingo 6 de octubre del 2019 se hizo el siguiente pronóstico de las fechas del 07 al 13 de octubre del mismo año. Se obtuvo el precio del Bitcoin de una semana a futuro conforme a lo que se indico en el objetivo del capítulo 1. Se puede observar que el día lunes,

el precio cerrará hasta los $8430 USD y de ahí empezará a tener una caída llegando a la zona de los $8200 USD aproximadamente. Después tendrá una recuperación de los $8325 USD y en el último día tendrá ese retroceso en el precio, véase la Figura 4.3.

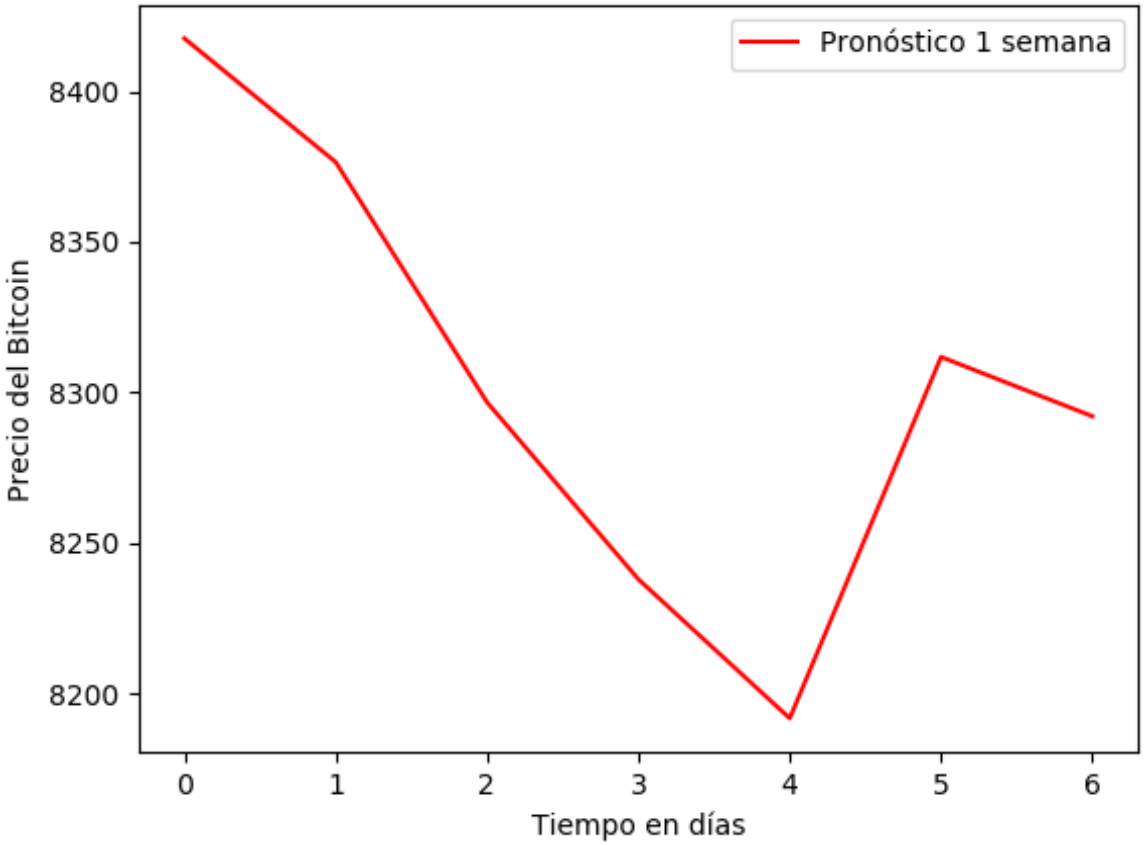

Figura 4.3: Pronostico de una semana.
Fuente: Elaboración propia.

A continuación se muestra el precio real de la semana que se pronosticó. Se recabo los datos del mercado y como resultado, se observa que los días lunes y martes están en rango de precio entre los $225 y $250 USD. Para los días miércoles y jueves, subió el precio considerablemente hasta llegar a un rango entre $8570 a $8600 USD. A partir del viernes el precio retrocedió hasta llegar a la zona de los $300 USD y siguió manteniéndose en esa zona, véase la Figura 4.4.

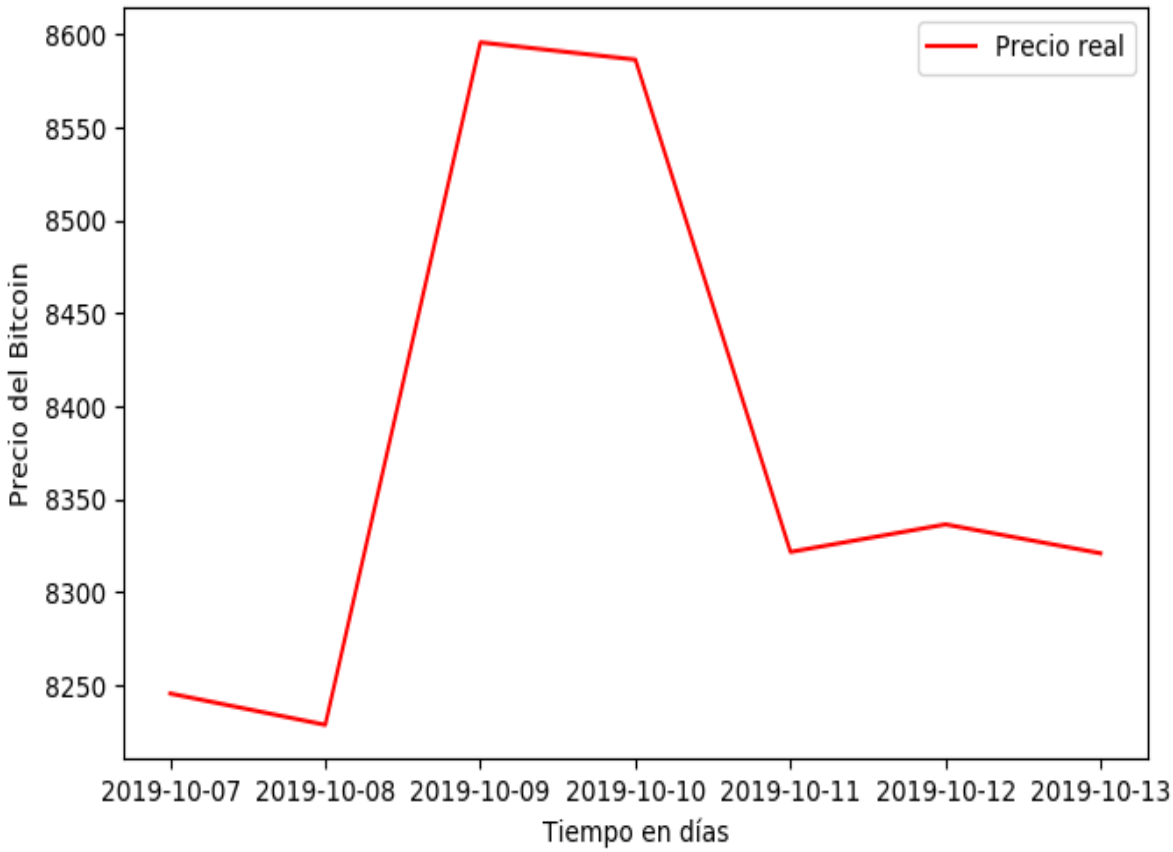

Figura 4.4: Precio real del Bitcoin del 07 al 09 de octubre.
Fuente: Elaboración propia.

Comparando los resultados del pronóstico y el precio real del mercado, el modelo acertó en que el precio subiría drásticamente debido a la volatilidad, aunque el modelo no fue preciso en los días. Sin embargo, en esta subida que detecto la red neuronal, el valor del Bitcoin tuvo una diferencia entre los $160 USD del real. También acertó en la caída que tuvo en esa semana, pero como se dijo antes no fue preciso en los días, pero tuvo una diferencia más abajo del precio real de $120 USD aproximadamente. En porcentaje de acertacion obtuvo un 30 % el pronóstico.

Capítulo 5
Conclusiones

En este capítulo se tratará sobre los aprendizajes que se obtuvieron durante el desarrollo del proyecto. La importancia de comprender los conocimientos adquiridos amplía el panorama en que se encuentra el proyecto en la resolución del problema. Además, las diferentes problemáticas que se pueden encontrar en el desarrollo del modelo. Se mostrarán las comparaciones que se mencionaron en el objetivo y si se cumplieron con lo especificado. También se darán recomendaciones a futuro que podrían ayudar a mejorar los pronósticos del Bitcoin.

5.1. Con respecto al objetivo de la investigación

Se cumplió con el objetivo del proyecto que se había pronosticado durante una semana. El modelo fue capaz de establecer rangos de precios congruentes al precio real, sin embargo, le fue difícil aprender en periodos cortos de alta volatilidad.

La *Blokchain* tiene variables importantes que pueden indicar la tendencia del precio en el Bitcoin y la extracción de estos datos sirven para entrenar a la red neuronal. De esto es lo que se indicó en la hipótesis, que los datos que se registran en la *Blokchain* están relacionados con el precio.

Fue necesario investigar en diferentes artículos posibles problemáticas semejantes y poder

ver la resolución que aplicaron y así poder delimitar y descartar para este proyecto.

Algo que se tuvo que tomar en cuenta en este proyecto fue qué tipo de variables son las que afectan al precio del Bitcoin. Como se mencionó en antecedentes, el mercado de las criptomonedas y al igual que otros mercados de fondos de inversión, se pueden hacer el análisis técnico y fundamental.

Las redes neuronales existen en muchas aplicaciones para la resolución de diferentes problemáticas, sin embargo, existen diferentes tipos y en este caso las redes neuronales recurrentes sirvieron específicamente las LSTM para el desarrollo del modelo. Es muy común que este tipo se usen en series de tiempo ya sea para la predicción en la bolsa de valores.

En el modelo que se construyó se tienen que considerar varios aspectos como el *underfitting* y *overfitting*, los cuales presentan problemas al momento de construir el modelo de redes neuronales. Estos pueden ocasionar que den resultados no tan correctos debido a la cantidad de datos que se poseen de entrenamiento o que la red no pueda aprender nuevos datos de entrada.

El modelo tuvo una buena disminución del error, sin embargo, en cuestión de la exactitud fue deficiente.

5.2. Con respecto a la hipótesis de la investigación

La hipótesis en este proyecto no fue comprobada ya que, el programa le fue difícil pronosticar el precio del Bitcoin y el porcentaje no tuvo el nivel de confianza factible. Sin embargo, el modelo tuvo una buena acertación sobre en qué rango del precio estaría oscilando.

Los datos que se utilizaron fuero de tramas de tiempo de un día pero es mejor que se utilicen lapsos de tiempo mas pequeños, como horas o minutos.

5.3. Recomendaciones para futuras investigaciones

Es necesario poder contar con buenas cantidades de datos para el entrenamiento y un buen manejo de los datos al momento de la normalización para evitar *underfitting* y *overfitting*.

Se puede considerar juntar un análisis de sentimiento del Bitcoin en redes sociales para complementarlo con análisis técnico y fundamental del mercado. Además, se pueden hacer otras implementaciones considerando diferentes indicadores técnicos que se utilizan en mercados tradicionales. Por ejemplo, los retrocesos de fibonacci que indican zonas del precio importantes donde puede haber un cambio de tendencia alcista o bajista.

Durante la investigación del proyecto hubo otros modelos construidos con redes neuronales bayesianas, donde se obtienen mejores resultados que las LSTM.

Bibliografía

[1] "Cryptocurrency Market Capitalizations | CoinMarketCap." [Online].

Available: https://coinmarketcap.com/. [Accessed: 07-Apr-2019].

[2] "Preguntas más frecuentes - Bitcoin." [Online]. Available: https://bitcoin.org/es/faq#que-es-bitcoin. [Accessed: 07-Apr-2019].

[3] I. Bagdonas, "Predicting Bitcoin Price using Machine Learning," no. July, pp. 0–99, 2018.

[4] H. Jang and J. Lee, "An Empirical Study on Modeling and Prediction of Bitcoin Prices with Bayesian Neural Networks Based on Blockchain Information," IEEE Access, vol. 6, pp. 5427–5437, 2017.

[5] S. McNally, J. Roche, and S. Caton, "Predicting the Price of Bitcoin Using Machine Learning," in 2018 26th Euromicro International Conference on Parallel, Distributed and Network-based Processing (PDP), 2018, pp. 339–343.

[6] A. Greaves and B. Au, "Using the Bitcoin Transaction Graph to Predict the Price of Bitcoin," 2015.

[7] A. Narayanan, J. Bonneau, E. Felten, A. Miller, and S. Goldfeder, "Bitcoin and Cryptocurrency Technologies," 2015.

[8] M. Swan, Blockchain: blueprint for a new economy, First. Sebastopol, 2015.

[9] "¿Qué es el 'trading'?" [Online] Available: https://www.bbva.com/es/obtener-carta-deducibilidad-intereses-credito-hipotecario/. [Accessed: 14-May-2019].

[10] A. C. Alfaro and R. A. Santos, "El análisis técnico y fundamental en un contexto de globalización: Bancolombia The technical and fundamental analysis in a context of globalization: Bancolombia," vol. 6, no. 73601, pp. 1–39, 2015.

[11] G. Ríos, "Series de Tiempo," Univ. Chile. Fac. Ciencias Fis. y Mat., p. 52, 2008

[12] "El modelo de redes neuronales." [Online]. Available: ´

https://www.ibm.com/support/knowledgecenter/es/SS3RA7_sub

/modeler_mainhelp_client_ddita/components/neuralnet/neuralnet_model.html. [Accessed: 07-Apr-2019].

[13] "¿Qué son las API?" [Online]. Available: https://www.redhat.com/es/topics/api/what-are-application-programming-interfaces. [Accessed: 07-Apr-2019].

Apéndice A
Código de funciones API

```
import datetime
import requests
import yfinance as yf
import json
from pandas.io.json import json_normalize
import pandas as pd
import time
import os.path as path
import os
import errno
import glob
```

Figura A.1: Librerías del programa API.

```
def blockAPI(url):

    print(url+"---------Listo...")

    response  = requests.get(url)
    response_json = json.loads(response.text)
    values = json_normalize(response_json['values'])

    df = pd.DataFrame({"Date":values['x'], "Values":values['y'], "Name":response_json['name'],
                       "Period":response_json['period'], "Unit":response_json['unit']})

    df['Date'] = pd.to_datetime(df['Date'], unit='s')

    if path.exists("CSV/{}.csv".format(response_json['name'])):
        file = pd.read_csv("CSV/{}.csv".format(response_json['name']))
        df2 = pd.DataFrame(file)
        df2 = df2.append(df)
        df2.to_csv("CSV/{}.csv".format(response_json['name']), encoding='utf-8', index=False)
    else:
        try:
            os.mkdir('CSV')
        except OSError as e:
            if e.errno != errno.EEXIST:
                raise
        df.to_csv("CSV/{}.csv".format(response_json['name']), encoding='utf-8', index=False)
```

Figura A.2: Función de descarga de datos de la Blockchain.

```
def fPrice():
    market_info = pd.read_html("https://coinmarketcap.com/currencies/bitcoin/historical-data/?start=20130428&end="
                               +time.strftime("%Y%m%d"))[0]
    df = pd.DataFrame(market_info)
    df = df.sort_values('Date')
    df.to_csv("CSV/Coinmarketcap-Historical-Data-Bitcoin.csv", encoding='utf-8', index=False)


def yahoo_BTC():
    btc = yf.Ticker("BTC-USD")
    #get historical market data
    hist = btc.history(period="max")
    df = pd.DataFrame(hist)
    df.to_csv("CSV/yahoo-BTC.csv", encoding='utf-8')

def dropDuplicateData():
    path = 'CSV/'
    csv_files = glob.glob(path +'*.csv')

    for filename in csv_files:
        print(filename)
        data = pd.read_csv(filename)
        df = pd.DataFrame(data)
        df = df.sort_values('Date')
        df.drop_duplicates('Date', inplace=True, keep='last')
        df.to_csv("{}".format(filename, header=0), encoding='utf-8', index=False)
```

Figura A.3: Funciones de descarga del historial del precio y eliminación de datos duplicados.

```
def blockData2Y():
    with open('links/BlockData2Y.txt','r') as lineas:
        for linea in lineas:
            link = linea.strip()
            blockAPI(link)

def blockAllData():
    with open('links/BlockAllData.txt','r') as lineas:
        for linea in lineas:
            link = linea.strip()
            blockAPI(link)
```

Figura A.4: Funciones de carga de URL de API's.

Apéndice B
Código del pronóstico del precio con múltiples variables

```
import numpy as np
import glob
import matplotlib.pyplot as plt
import pandas as pd
from sklearn.preprocessing import MinMaxScaler
import tensorflow as tf
from tensorflow.keras import layers
```

Figura B.1: Librerías.

```
def create_time_steps(length):
  time_steps = []
  for i in range(-length, 0, 1):
    time_steps.append(i)
  return time_steps

def show_plot(plot_data, delta, title):
  labels = ['Datos anteriores', 'Futuro verdadero', 'Pronóstico']
  marker = ['.-', 'rx', 'go']
  time_steps = create_time_steps(plot_data[0].shape[0])
  if delta:
    future = delta
  else:
    future = 0

  plt.title(title)
  for i, x in enumerate(plot_data):
    if i:
      plt.plot(future, plot_data[i], marker[i], markersize=10,
               label=labels[i])
    else:
      plt.plot(time_steps, plot_data[i].flatten(), marker[i], label=labels[i])
  plt.legend()
  plt.xlim([time_steps[0], (future+5)*2])
  plt.xlabel('Tiempo en días')
  return plt
```

Figura B.2: Funciones para los pasos en el tiempo y de graficación del resultado.

```python
def multivariate_data(dataset, target, start_index, end_index, history_size,
                      target_size, step, single_step=False):
  data = []
  labels = []

  start_index = start_index + history_size
  if end_index is None:
    end_index = len(dataset) - target_size

  for i in range(start_index, end_index):
    indices = range(i-history_size, i, step)
    data.append(dataset[indices])

    if single_step:
      labels.append(target[i+target_size])
    else:
      labels.append(target[i:i+target_size])

  return np.array(data), np.array(labels)
```

Figura B.3: Función de recorrido en la serie de tiempo.

```python
# get data file names
path = 'FILES/'
csv_files = glob.glob(path +'*.csv')

frame = pd.DataFrame()
list_ = []
for file_ in csv_files:
    df = pd.read_csv(file_,index_col='Date')
    list_.append(df['Values'])
frame = pd.concat(list_, sort=True, axis=1)
frame.sort_index(axis=1)
frame = frame.dropna()
frame.index.name='Date'
frame.to_csv("joinFrames.csv", encoding='utf-8')

df = pd.read_csv('joinFrames.csv', index_col='Date', parse_dates=['Date'])
features = df.iloc[0:]
print(features.head())
```

Figura B.4: Preparación del marco de datos.

```
df = pd.read_csv('joinFrames.csv', index_col='Date', parse_dates=['Date'])
features = df.iloc[0:]
print(features.head())


features.plot(subplots=True)

dataset = features.values
print("dataset[:, 10]",dataset[:, 10])

set_entrenamiento = 1727

# Normalización del set de entrenamiento
sc = MinMaxScaler(feature_range=(-1,1))
dataset = sc.fit_transform(features.values)
```

Figura B.5: Normalización de los datos.

```
past_history = 60
future_target = 7
STEP = 1

x_train, y_train = multivariate_data(dataset, dataset[:, 10], 0,
                                     set_entrenamiento, past_history,
                                     future_target, STEP,
                                     single_step=True)
x_val, y_val = multivariate_data(dataset, dataset[:, 10],
                                 set_entrenamiento, None, past_history,
                                 future_target, STEP,
                                 single_step=True)

"""Let's look at a single data-point."""

print ('Single window of past history : {}'.format(x_train[0].shape))
```

Figura B.6: Función de recorrido llamada para preparar datos en la serie de tiempo.

```
BATCH_SIZE = 7
BUFFER_SIZE = 100
EVALUATION_INTERVAL = 1000
epocas = 10

set_entrenamiento_escalado = tf.data.Dataset.from_tensor_slices((x_train, y_train))
set_entrenamiento_escalado = set_entrenamiento_escalado.cache().shuffle(BUFFER_SIZE).batch(BATCH_SIZE).repeat()

set_validacion_escalado = tf.data.Dataset.from_tensor_slices((x_val, y_val))
set_validacion_escalado = set_validacion_escalado.batch(BATCH_SIZE).repeat()

modelo_multival = tf.keras.models.Sequential()
modelo_multival.add(tf.keras.layers.LSTM(120, return_sequences=True, activation='tanh',
                                         input_shape=x_train.shape[-2:]))
modelo_multival.add(tf.keras.layers.LSTM(60, activation='tanh'))
modelo_multival.add(tf.keras.layers.Dense(1, activation='tanh'))

modelo_multival.compile(optimizer=tf.keras.optimizers.RMSprop(), loss='mse')

"""Let's check out a sample prediction."""

for x, y in set_validacion_escalado.take(1):
  print(modelo_multival.predict(x).shape)

history = modelo_multival.fit(set_entrenamiento_escalado, epochs=epocas,
                              steps_per_epoch=EVALUATION_INTERVAL,
                              validation_data=set_validacion_escalado,
                              validation_steps=50)
```

Figura B.7: Modelo red neuronal recurrente LSTM.

```
def plot_train_history(history, title):
  loss = history.history['loss']
  val_loss = history.history['val_loss']

  epochs = range(len(loss))

  plt.figure()

  plt.plot(epochs, loss, 'b', label='Training loss')
  plt.plot(epochs, val_loss, 'r', label='Validation loss')
  plt.title(title)
  plt.legend()

  plt.show()

plot_train_history(history,
                   'Single Step Training and validation loss')


for x, y in set_validacion_escalado.take(3):
  plot = show_plot([x[0][:, 10].numpy(), y[0].numpy(),
                    modelo_multival.predict(x)[0]], 7,
                   'Pronóstico de entrenamiento')
  plot.show()
```

Figura B.8: Función de muestreo de perdida de error y graficación del resultado.

Apéndice C
Código del pronóstico del precio con una variable

```
import numpy as np
np.random.seed(4)
import matplotlib.pyplot as plt
import pandas as pd
from sklearn.preprocessing import MinMaxScaler
import tensorflow as tf
from tensorflow.keras import layers
```

Figura C.1: Librerías.

```
def show_plot(real, prediccion):
    plt.plot(real[0:len(prediccion)],color='red', label='Precio real')
    plt.plot(prediccion, color='blue', label='Pronóstico')
    plt.ylim(1.1 * np.min(prediccion)/2, 1.1 * np.max(prediccion))
    plt.xlabel('Tiempo en días')
    plt.ylabel('Precio del Bitcoin')
    plt.legend()
    plt.show()

#
# Lectura de los datos
#
data = pd.read_csv('yahoo-BTC.csv', index_col='Date', parse_dates=['Date'])
dataset = pd.DataFrame(data)
dataset.head()
```

Figura C.2: Función de muestreo de resultados del pronóstico.

```
set_entrenamiento = dataset['2014':'2017'].iloc[:,3:4]
set_validacion = dataset['2018':].iloc[:,3:4]

print(set_entrenamiento)
print(set_validacion)

set_entrenamiento['Close'].plot(legend=True)
set_validacion['Close'].plot(legend=True)
plt.legend(['Entrenamiento (2018)', 'Validación (2019)'])
plt.show()

# Normalización del set de entrenamiento
sc = MinMaxScaler(feature_range=(0,1))
set_entrenamiento_escalado = sc.fit_transform(set_entrenamiento)
#print("set_entrenamiento_escalado: ",set_entrenamiento_escalado)
```

Figura C.3: Datos de entrenamiento y normalización.

```
for i in range(time_step,m):
    # X: bloques de "time_step" datos: 0-time_step, 1-time_step+1, 2-time_step+2, etc
    X_train.append(set_entrenamiento_escalado[i-time_step:i, :])

    # Y: el siguiente dato
    Y_train.append(set_entrenamiento_escalado[i,:])
X_train, Y_train = np.array(X_train), np.array(Y_train)

# Reshape X_train para que se ajuste al modelo en Keras
X_train = np.reshape(X_train, (X_train.shape[0], X_train.shape[2],X_train.shape[1]))
print("X_train",X_train)

x_test = set_validacion.values
#print(set_validacion.values)
x_test = sc.fit_transform(x_test)
#rint(x_test)
#print(len(x_test))

X_test = []
Y_test = []
for i in range(time_step,len(x_test)):
    X_test.append(x_test[i-time_step:i, :])
    Y_test.append(x_test[i,:])
X_test, Y_test = np.array(X_test), np.array(Y_test)
X_test = np.reshape(X_test, (X_test.shape[0],X_test.shape[2],X_test.shape[1]))
```

Figura C.4: Preparación de pasos en el tiempo en datos de entrenamiento y validación.

```
# Red LSTM
#
dim_entrada = (X_train.shape[1],X_train.shape[2])
print(dim_entrada)
dim_salida = 1

modelo = tf.keras.models.Sequential()

modelo.add(layers.LSTM(50, activation="tanh", input_shape=dim_entrada))
modelo.add(layers.Dropout(0.2))
modelo.add(layers.Dense(dim_salida, activation="tanh"))

modelo.summary()
modelo.compile(optimizer='adam',
               loss='mean_squared_error',
               metrics=['accuracy'])

callback = tf.keras.callbacks.EarlyStopping(monitor='val_loss', patience=5)

history = modelo.fit(X_train,Y_train,epochs=50,batch_size=512,
                     validation_data=(X_test, Y_test), shuffle=True,
                     callbacks=[callback])
# Evaluate the model on the test data using `evaluate`
print('\n# Evaluate on test data')
results = modelo.evaluate(X_test, Y_test, verbose=0)
print("%s: %.2f%%" % (modelo.metrics_names[1], results[1]*100))
```

Figura C.5: Red neuronal recurrente LSTM.

```
prediccion = modelo.predict(X_test)
print(prediccion)
prediccion = sc.inverse_transform(prediccion)

print(prediccion)
# Graficar resultados
show_plot(set_validacion.values,prediccion)

print("Guardar modelo: Y or N: ", end="")
resp = input()

if((resp == 'Y') or (resp=='y') or (resp=='yes') or (resp=='Yes')):
    modelo.save("model.h5")
    print("Saved model to disk")
```

Figura C.6: Resultados de entrenamiento y guardado del modelo.

```
set_entrenamiento2 = dataset['2019-10-27':].iloc[:,3:4]
sc2 = MinMaxScaler(feature_range=(-1,1))
set_entrenamiento_escalado2 = sc2.fit_transform(set_entrenamiento2)

print(set_entrenamiento2)


x_test2 = set_entrenamiento_escalado2
x_test2 = sc2.transform(x_test2)

X_test2 = []
Y_test2 = []
for i in range(time_step,len(x_test2)):
    X_test2.append(x_test2[i-time_step:i, :])
    Y_test2.append(x_test2[i,:])
X_test2, Y_test2= np.array(X_test2), np.array(Y_test2)
X_test2 = np.reshape(X_test2, (X_test2.shape[0],X_test2.shape[2],X_test2.shape[1]))
```

Figura C.7: Preparación de datos para pronóstico de una semana.

```
def agregarNuevoValor(x_test,nuevoValor):
    for i in range(x_test.shape[2]-1):
        x_test[0][0][i] = x_test[0][0][i+1]
        print(x_test[0][0][i])
    x_test[0][0][x_test.shape[2]-1]=nuevoValor
    return x_test
```

Figura C.8: Función para el manejo de datos del pronóstico.

```
if((resp1 == 'Y') or (resp1=='y') or (resp1=='yes') or (resp1=='Yes')):
    model_cargado = tf.keras.models.load_model('model.h5')
    print("Cargado")

    results=[]
    for i in range(7):
        parcial=model_cargado.predict(X_test2)
        results.append(parcial[0])
        print(X_test2)
        X_test2=agregarNuevoValor(X_test2,parcial[0])

else:

    results=[]
    for i in range(7):
        parcial=modelo.predict(X_test2)
        results.append(parcial[0])
        print(X_test2)
        X_test2=agregarNuevoValor(X_test2,parcial[0])

adimen = [x for x in results]
inverted = sc2.inverse_transform(adimen)
print(inverted)
```

Figura C.9: Realización del pronóstico.

```
adimen = [x for x in results]
inverted = sc2.inverse_transform(adimen)
print(inverted)


prediccion1SemanaDespues = pd.DataFrame(inverted)
prediccion1SemanaDespues.columns = ['Pronostico']
prediccion1SemanaDespues.to_csv('pronostico1Semana.csv')

plt.plot(prediccion1SemanaDespues,color='red', label='Pronóstico 1 semana')
plt.xlabel('Tiempo en días')
plt.ylabel('Precio del Bitcoin')
plt.legend()
plt.show()
```

Figura C.10: Muestreo del pronóstico de una semana.