INSTITUTO
UNIVERSITÁRIO
DE LISBOA

Cryptocurrency price prediction using LSTM neural networks

José Luís Almeida Pereira

Master in Data Science

Supervisor:
Dr. Sancho Oliveira, Professor,
Iscte - University Institute of Lisbon

Co - Supervisor:
Dr. Diana Mendes, Professor,
Iscte - University Institute of Lisbon

October, 2022

BUSINESS SCHOOL

TECNOLOGIAS E ARQUITETURA

Cryptocurrency price prediction using LSTM neural networks

José Luís Almeida Pereira

Master in Data Science

Supervisor:
Dr. Sancho Oliveira, Professor,
Iscte - University Institute of Lisbon

Co - Supervisor:
Dr. Diana Mendes, Professor,
Iscte - University Institute of Lisbon

October, 2022

Acknowledgements

I would like to thank my professors Dr. Diana Mendes and Dr. Sancho Oliveira for the support, motivation, and opportunity to guide me in the completion of this dissertation.

To my partner Cidália Eusébio, for all the support, understanding and shared knowledge.

To my parents, José Pereira and Maria Pereira for all the encouragement and motivation provided.

To my sister Sandra Pereira, for all the motivation.

To Dr. José Neves, for his guidance and advice in choosing this course.

Resumo

O interesse em moedas digitais tem aumentado por parte de indivíduos e investidores. A bitcoin é a moeda digital com maior capitalização de mercado, no entanto, a sua alta volatilidade alinhada à incerteza política, torna muito difícil prever seu valor. Portanto, existe a necessidade de criar modelos avançados que utilizem métodos matemáticos e estatísticos para reduzir o risco de investimento. Este estudo tem como objetivo verificar se as redes neurais artificiais de memória longo curto prazo (LSTM) e redes bidirecionais de memória longo curto prazo (BiLSTM) podem ser usadas juntamente com o filtro Savitzky-Golay para prever os preços de fecho do dia seguinte da bitcoin. Os resultados mostraram que existe evidência que ambas as redes podem ser usadas de forma efetiva. LSTM obteve um erro percentual absoluto médio (MAPE) de 4.49 e BiLSTM um MAPE de 4,44. Também o uso do filtro Savitzky-Golay e regularização, melhora significativamente o desempenho de previsão dos modelos.

Palavras-chave: previsão; cripto moeda; Savitzky–Golay; LSTM; BiLSTM; redes neurais

Abstract

The interest in cryptocurrencies is increasing among individuals and investors. Bitcoin is the leading existing cryptocurrency with the highest market capitalization. However, its high volatility aligns with political uncertainty making it very difficult to predict its value. Therefore, there is a need to create advanced models that use mathematical and statistical methods to reduce investment risk. This research aims to verify if long short-term memory (LSTM), and bidirectional long short-term memory (BiLSTM) neural networks, can be used with Savitzky–Golay filter to predict next-day bitcoin closing prices. We found evidence both networks can be used effectively to predict bitcoin prices. LSTM performed 4.49 mean absolute percentage error (MAPE) and BiLSTM 4.44 MAPE. We also found that using Savitzky–Golay filter and dropout regularization significantly improved the model's prediction performance.

Keywords: forecasting; cryptocurrency; Savitzky–Golay; LSTM; BiLSTM; neural networks

Contents

List of Figures

List of tables

CHAPTER 1

Introduction

Thousands of digital currencies and hundreds of exchangers have been created since the introduction of bitcoin in 2008 (Chiu & Keister, 2022). Digital currencies have attracted great attention from investors, regulators, and the public (Giudici et al., 2020). The number of markets and activities developed around digital currencies has increased exponentially, including online trading platforms, crypto-based derivatives trading, and crypto lending platforms. In addition, central banks are now investigating the possibility of creating a central bank digital currency (CBDC), with some, already in the process of its creation. Although, digital currencies also bring concerns to the market.

The high volatility of digital currencies and the lack of intrinsic value have generated public, scientific and political discussions (Giudici et al., 2020). Concerns that this type of currency is a bubble without any kind of fundamental value and the possibility of allowing tax evasion, could lead governments to increase laws to regularize them. All this speculation makes cryptocurrency price prediction very difficult, therefore, cryptocurrency price prediction has been an important research topic addressed by many researchers worldwide.

Bitcoin is the leading existing cryptocurrency with a market capitalization of over $440,091,646,995.21. Its value allows leveraging all blockchain technology for a wide digital circulation (Coinmarketcap, 2022). Over the last few years, many researchers have used classical, statistical, and financial methods such as - autoregressive integrated moving average (ARIMA) or generalized autoregressive conditional heteroscedasticity (GARCH) - to predict bitcoin prices (Gradojevic et al., 2021).

The increase in computing power and the development of deep and machine learning algorithms, allowed the creation of new models to predict bitcoin prices. Artificial neural network (ANN), convolutional neural network (CNN), recurrent neural network (RNN), and long short-term memory (LSTM) are some of the algorithms that have been used to create deep learning systems to predict the price of bitcoin.

For the improvement of bitcoin price prediction, many external variables have been proposed; for example, Panagiotidis et al. (2018), performed a Least Absolute Shrinkage and Selection Operator (LASSO) approach to find the best determinates of bitcoin returns. They concluded that gold returns and policy uncertainty were the most important predictors. However, interest rates, NASDAQ (National Association of Securities Dealers Automated Quotations), oil prices, and exchange rates were also considered important determinates.

Although many methods and models have already been studied, no paper using Savitzky–Golay filter to predict bitcoin prices were found. The only study to use the filter was Klein et al. (2018); however, they used the filter to smooth the correlation plots between gold and bitcoin. They performed a comparison of correlation, volatility, and portfolio performance between bitcoin, gold, and S&P500.

This study aims to verify if LSTM neural networks can be efficiently used to predict bitcoin prices. We propose to use the Savitzky–Golay filter to smooth the high volatility of bitcoin, and LSTM and BiLSTM models to predict the next day's bitcoin closing price. To the best of our knowledge, this is the first time that these two methods are used together to predict bitcoin prices. We experienced Savitzky–Golay filter to eliminate the random noise of bitcoin prices while preserving the accurate spectral signal. To avoid using less significant or irrelevant features that would lead to the creation of noise, higher prediction errors and an increase in complexity and execution times, we performed a wrapper forward features selection method. We also run several robust experiments analyses to observe the impact of the number of hidden layers, the number of units per layer, batch size, optimizer, learning rate and dropout regularization on the model's prediction performance. Lastly, we also compare both models regarding training times and prediction errors.

In the next Chapter, we present a systematic literature review on cryptocurrency forecasting. Chapter 3 presents the adopted methodologies to forecast bitcoin closing prices and performance measures. Chapter 4 presents the descriptive statistics of data, findings, and results of several experiments and analyses. Lastly, in Chapter 5, the analyses and experimental results are discussed.

CHAPTER 2
Literature Review

This section presents a summary of the performed literature review. We aimed to understand the evidence on some current research topics developed to obtain financial forecasts, particularly those directed to cryptocurrencies. We cover tools and algorithms from statistics and time series, deep learning, recurrent neural networks, and ensemble machine learning.

We used the Prisma tool methodology to perform a systematic literature review. The search strategy, inclusion and exclusion criteria, and information extraction are presented in Appendix A.

2.1. Traditional time series Methods

This section describes the methods and algorithms in the selected research papers based on traditional time series forecasting.

Septiarini et al. (2020) presented a study that aimed to build a model based on traditional statistical and artificial intelligence methods to predict the price of the bitcoin cryptocurrency. The authors used time series models, such as Autoregressive Integrated Moving Average (ARIMA) and Exponential Smoothing (ES). They also used artificial intelligence methods like fuzzy time series and Adaptive Neuro-Fuzzy Inference System (ANFIS). Their results showed that the statistical methods performed better than the artificial intelligence methods. The forecasting results show that the exponential smoothing classical method had the best performance with the smallest root mean squared error (RMSE) and mean squared error (MSE).

Tan and Kashef (2019) made a comparative study between machine learning, deep learning and ARIMA statistical methods to predict the price of the bitcoin cryptocurrency. In the study, they used five features to describe each transaction defined as the open, high, low, and closed price and transaction volume of cryptocurrency. The models used in the comparison were Bayesian Regression (BR), Auto Regression (AR), Long Short-Term Memory (LSTM) and Support Vector Machines (SVM). The results showed that the LSTM algorithm had better performance than the others, followed by SVM and ARIMA, respectively.

Munim et al. (2019) presented a study to predict next-day bitcoin price using two univariate models, ARIMA and NNAR (neural network autoregression), performed with and without model forecast re-estimation for each step. The paper used two training-samples and a two-test samples to perform cross-validation. The results showed that ARIMA models perform better than NNAR models in both test-samples forecasts. They suggest that the reason that may have influenced the best performance of the

ARIMA models may be related to the fact that they apply a feed-forward NNAR model, however, the result could have been improved using a back-propagation algorithm. They also found that forecasts with ARIMA models are similar with or without model re-estimation.

This review of research using traditional time series models indicates that forecasts based on techniques like ARIMA have low prediction errors and can outperform deep learning models. Although, this could be related to the data characteristics since the correct collection and preparation of data is a fundamental process in the excellent performance of algorithms. Sometimes algorithms that use more traditional techniques outperform complex deep learning algorithms that use more sophisticated techniques. Septiarini et al. (2020) consider that modern models cannot guarantee better forecasting results because each case study's data characteristic is unique.

2.2. Neural Networks Methods

This section describes the methods and algorithms in the selected research papers based on neural network forecasting.

Radityo et al. (2018) conducted a comparative study with various artificial neural network (ANN) methods to predict the next day's Bitcoin closing price. They used backpropagation neural network (BPNN), genetic algorithm neural network (GANN), genetic algorithm backpropagation neural network (GABPNN), and neuroevolution of augmenting topologies (NEAT) for this task. According to the study results, GABPNN obtained the best mean absolute percentage error (MAPE), only 1.88%. However, the training time of the algorithm is not realistic for applications where the volume of the data is much higher. BPNN was three times faster, obtaining a 1.98% MAPE. This study was a significant contribution to the scientific community and people working with machine learning since it compared a set of ANN methods and, at the same time, raised awareness of the importance of training times. Obtaining a slightly higher prediction error can be a better solution if it allows an algorithm with substantially lower complexity and execution time.

The correct parameterization of models is essential for their good performance, mainly when we use networks with a simpler architecture. Jay et al. (2020) proposed a stochastic neural network model based on the random walk theory to predict cryptocurrency prices. They simulated market volatility with a multi-layer perceptron model that induces layer-wise randomness into the observed activation neural network features. The results found that the proposed model was effective in decoding market volatility. Almost all models that used stochastic versions - performed better than those - that used deterministic versions. They draw attention to the importance of optimization techniques to tune the hyperparameters, and they consider that it was essential for the results of their study, but it could be improved further with perfect solution adjustment of hyperparameters.

2.3. Single deep learning methods

This section describes the methods and algorithms in the selected research papers based on single deep learning forecasting.

Recurrent neural networks (RNNs), Convolutional neural networks (CNN) and LSTM are artificial deep neural networks; they can analyse past time sequences of arbitrary lengths to make predictions.

Deep learning is one of the main methods used for cryptocurrency forecasting. Ferdiansyah et al. (2019) presented a study to create a model to predict Bitcoin prices using LSTM neural networks. The proposed model used four years of historical bitcoin data to train the model and one year to test its performance. They conclude that the proposed model was successful in predicting next-day Bitcoin prices; however, given the obtained RMSE value, they consider that the model was not good enough to make Bitcoin investment decisions.

Data preparation and quality are crucial when using Deep Learning algorithms. Rizwan et al. (2019) developed a multivariate Deep learning model using LSTM and Gated recurrent unit (GRU). They collected the Bitcoin exchange rate, the volume of trades, total transaction fees, the number of transactions, cost per transaction, and average hash rate. However, they also consider that external information can affect the price of cryptocurrencies, such as international economic indicators. The study results have revealed that the algorithms parameterization and the data quality used in the modelling process are essential to obtain good predictions.

The correct selection of variables from a set of available data is also essential for the good performance of deep learning algorithms. Lamothe-Fernandez et al. (2020) conducted a comparative study between price prediction methods for bitcoin. The study verified that the choice of a new set of significant variables improved the algorithms' performance, offering good stability on models developed for one- and two-year timeframes. The algorithm with the best performance was Dynamic Convolutional Neural Network (DRCNN).

LSTM deep learning neural nets are effective methods when leading with time series data. Lahmiri and Bekiros (2019) carried out a study using LSTM to learn chaotic and self-similar patterns for the top three cryptocurrencies (Bitcoin, Digital Cash and Ripple). They consider that deep learning using LSTM is efficient for both short and long terms temporal information simultaneously, allowing them to extract hidden patterns from temporal sequences with non-linear and chaotic data.

Real-time cryptocurrency price prediction is currently an important research topic. Zoumpekas et al. (2020) presented a prototype implementation of a web-based system appropriate for real-time prediction of Ethereum closing prices. The system uses a deep learning LSTM model to generate one prediction every half hour and the past 30 minutes to generate predictions for the future 5 minutes. They found that LSTM and GRU neural network models based on the performance of the study can be used for real-time prediction of the Ethereum price.

GRU is a simplified version of LSTM that requires less training time due to the improvement of network performance. Like LSTM networks, GRU are also very robust when dealing with time series data. Phaladisailoed and Numnonda (2018) developed a comparative study between deep learning models to predict the bitcoin price, where they tested Huber Regression, LSTM and GRU. They concluded that from all the models tested, GRU showed the best accuracy and convergence time performance. However, they consider that the results can be further improved by using a collection of variables with greater explanatory power in the variation of the cryptocurrency price.

Public attention and the macroeconomic environment are foremost aspects of predicting cryptocurrency prices. Liu et al. (2021) built a Stacked Denoising Autoencoders (SDAE) model that uses a feature system with 40 bitcoin price determinants, taking into consideration variables of the public attention, cryptocurrency market, and macroeconomic environment. The results showed that SDAE better predicted the bitcoin prices when compared with support vector regression (SVR) and back propagation neural network (BPNN). They consider that the factor with a high contribution to the good performance of the algorithm was the inclusion of a variable system that uses not only cryptocurrency market factors, but also public attention and the macroeconomic environment.

Almost all researchers propose the analysis of sentiment and public opinion as essential factors to improve the performance of cryptocurrency forecasting models. Wang and Chen (2020) conducted a study where they found that adding social media comments features can significantly improve the accuracy of cryptocurrency price forecasts. They presented a variety of machine and deep learning models and the result showed that LSTM had the best prediction result, but the main discovery was that adding social sentiment variables can significantly improve the accuracy of all models tested.

Throughout the literature review of this section, we could verify that deep learning methods like LSTM and GRU are robust in predicting cryptocurrency prices. However, the main point to retain is that the good quality of data collected and the choice of variables that add explanatory power to the models are essential for a good performance. The use of economic variables and social sentiment are also important factors in the good performance of the algorithms.

2.4. Ensemble and Machine learning methods

This section describes the methods and algorithms in the selected research papers based on ensemble and machine learning forecasting.

When we aggregate the predictions of a group of models, we get a better prediction than with the best individual prediction; we can use the prediction that gets the most votes; this is called Ensemble Machine learning. Derbentsev et al. (2021) developed a comparative performance study of machine learning ensemble algorithms to predict cryptocurrency prices. They performed Random Forests (RF) and Stochastic Gradient Boosting Machine (SGBM) to predict Bitcoin, Ethereum, and Ripple prices. The study revealed efficiency using ensemble learning methods; the out-of-sample prices forecast

obtained for SGBM and RF revealed a MAPE for the three crypto currencies within 0.92%-2.61%. SGBM had better prediction performance for Bitcoin and Ripple, and RF had better prediction performance for Ethereum.

Mallqui and Fernandes (2019) analysed the behaviour of ANN and Support Vector Machines (SVM) in predicting bitcoin prices. The experiments revelated SVM algorithm obtained the best results for all predictions with 1.58% MAPE.

Saad et al. (2020) conducted a study where analysed cryptocurrency prices through a variable correlation analysis. They performed the correlation between features such as transaction rate, hash rate, number of users, total bitcoins, and price. They have mapped the change in features and network activities to understand the dynamics of the cryptocurrencies and used their findings to perform machine learning models such as Linear Regression (LR), Random Forest (RF), and Gradient Boosting (GB). We mention that their approach had better performance than previous studies that predict bitcoin prices based on previous prices.

Like the previous section, it was also possible to verify the importance of the quality of the variables used in the prediction models. It is starting to become evident that it is familiar to almost all researchers that the quality of information collected is one of the main points to be considered when forecasting cryptocurrency prices.

2.5. Hybrid machine and deep learning Methods

This section describes the methods and algorithms in the selected research papers based on hybrid machine and deep learning forecasting.

Hybrid-based models can significantly improve cryptocurrencies prices forecasting. Patel et al. (2020) proposed an LSTM-GRU hybrid model to predict Litecoin and Monero in different scenarios: one, three, and seven-day price prediction. The results have shown that the hybrid proposed model is considerably better when compared with LSTM applied alone. The best model had a 2.06% MAPE and was performed for the Litecoin 3-days prediction window.

Livieris et al. (2021) presented a CNN-LSTM model that used Bitcoin, Ethereum, and Ripple data as input features to process them independently to find helpful information from each cryptocurrency. The results showed the proposed model had efficiently analysed the data layers individually, reducing the overfitting of the model while ensuring relatively lower computational costs compared to single CNN neural networks.

Kristjanpoller & Minutolo (2018) proposed a group of hybrid Artificial Neural Network-Generalized Auto Regressive Conditional Heteroskedasticity (ANN-GARCH) models to forecast the Bitcoin price. They have tested twelve models, considering different combinations of models and inputs. The model with the best performance was an Exponential Generalized Autoregressive Conditional Heteroskedasticity (EGRACH) with 1.64% MAPE.

Altan et al. (2019) presented a hybrid model based on LSTM neural network and empirical wavelet transformer (EWT) decomposition along with cuckoo search (CS) algorithm. The proposed model was compared with LSTM and EWT-LSTM. The EWT-LSTM-CS had the best performance when tested in Bitcoin, Litecoin, Digital Cash, and Ripple. The results also showed that the proposed model could successfully capture non-linear characteristics for digital currencies forecast.

This section presented several hybrid models that significantly improved cryptocurrency price forecasting. There is an enormous potential for research development in hybrid models since the combinations of models and variables are huge.

2.6. Literature review summary

Model performance can be affected by different factors to consider when developing models. Data quality is a critical factor for model performances; using only robust methods and techniques for predicting cryptocurrency does not guarantee good results. Collecting data and variables that influence the variation of cryptocurrency prices is essential. In the literature review analysis, it was evident that many researchers consider including macroeconomic and public sentiment variables as key factors to increase the performance of the models. Wang and Chen (2020) concluded that adding variables that measure public sentiment and opinion greatly improves the models' performance. However, it was visible that many researchers do not use public sentiment and opinion analysis techniques, showing that there is space for improvement in the research on this topic.

Uncertainty regarding the legislation that regulates the cryptocurrency market is also a factor that may strongly interfere with its volatility. The analysis of sentiment and opinion regarding this factor can help to explain the variation in cryptocurrency prices. However, it is essential to emphasize that each country has different legislation. The legislation change in a country with strong international economic influence could affect its price. The United States has many investors, and a change in its legislation could significantly change cryptocurrency prices. Ferdiansyah et al. (2019) consider that the stock market, including cryptocurrencies, is influenced by many uncertainties and political issues.

Among all, the hybrid models based on LSTM networks have proven a good performance in all the studies carried out on this systematic review, as shown in table 6.4 of appendix A. Due to the immense possibilities of hybrid model combinations, we consider this approach offers space for improvement. Patel et al. (2020) proposed an LSTM-GRU hybrid model, and the results have shown that the proposed model performed better in different scenarios when compared with LSTM applied alone.

Studies can be improved using hybrid models that use macroeconomic variables, sentiment analysis, and public opinion — paying particular attention to the sentiment and public opinion of countries with international economic influences.

CHAPTER 3

Methodology

3.1. Software and hardware

Data understanding, preparation, and modelling were conducted in Python 3.8, a high-level, interpreted, general-purpose programming language. Three Python libraries were used: Pandas for data manipulation and understanding, NumPy to create three-dimensional arrays to feed deep learning algorithms, and Keras, which acts as an interface for TensorFlow to develop deep learning LSTM neural networks.

All experiments were implemented on a personal computer device with AMD Ryzen 7 5800x, 8 cores, 4.7 GHz, and 32 GB RAM.

3.2. Methodology strategy

Our methodology follows the deep learning system architecture presented in figure 3.1. Deep learning is a subfield of machine learning and aims to mimic how humans gain a specific type of knowledge through experiences. The word 'Deep' represents using a neural network with more than three layers of depth (Chollet, 2021). The network depth creates a deep hierarchical representation learning, where layers are stacked on top of each other. It is a multistage information distillation process where the information is purified by passing through several filters. The network learns data representations through the multistage sequence process.

As shown in figure 3.1, the methodology was structured following a sequence of processes. First, economic variables and bitcoin prices were collected. Second, feature selection was performed with the wrapper forward selection method. Third, the volatility and the noise of the bitcoin closing prices were removed using a Savitzky–Golay filter. Fourth, data pre-processing was performed to prepare the data for deep learning algorithms. The data set was divided into three chunks: 65% for training, 15% for validation, and 20% for testing. The normalization of the three sets was performed to have the data on the same scale. In order to have the data ready for the deep learning algorithms, a rank-3 tensor was used to create sequences of 16-time steps as input and 1-time step as the label. Savitzky–Golay filter was only applied to training labels.

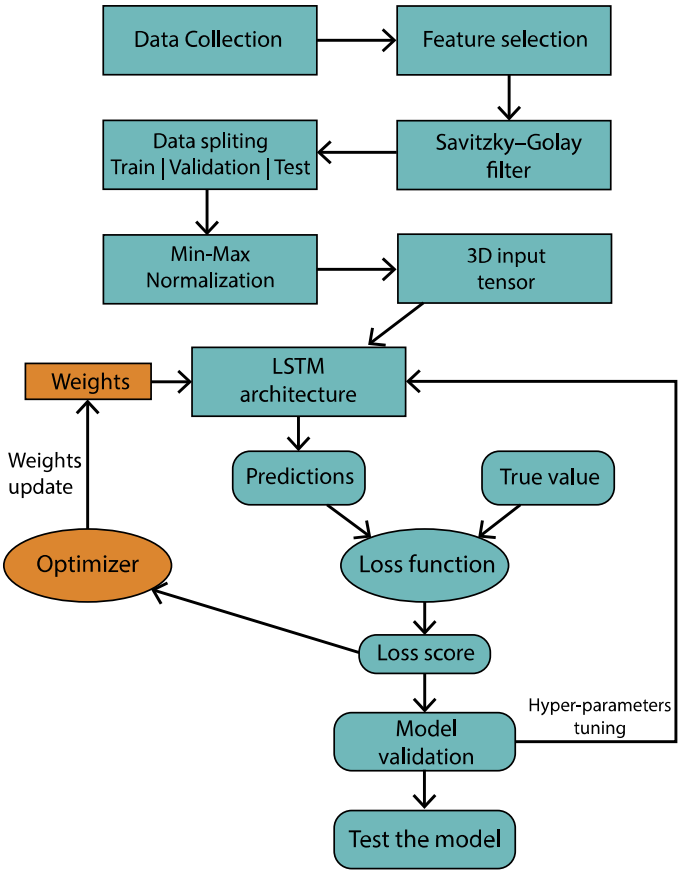

Figure 3.1: Deep learning System architecture

Selecting the right architecture for deep learning systems is very important. In this research study, two types of networks were used, Long Short-term Memory neural networks (LSTM) and Bi-Directional Long Short-term Memory neural networks (Bi-LSTM), variants of Recurrent Neural Networks (RNN), a type of neural network well-suited to process time series step-by-step.

The network learning process was accomplished by observing and mapping a significant number of inputs and labels through a deep sequence of data transformations (layers) (Chollet, 2021). The transformation done on the inputs was performed by the layer weights, which are also called layer parameters (illustrated in figure 3.1). The learning process consisted of finding the layers weights values that allowed the network to map the inputs and their associated labels correctly. A network can contain many layers; therefore, finding the correct value for all the weights is a complex task.

To control the output of the LSTM network – the algorithm first had to observe and measure how far the output (prediction) was from the actual value (Chollet, 2021). The measurement was performed using the MSE loss function, which compares the distance (loss score) between the forecasts and the true value. The loss score was used as a response signal to adjust the values of the weights in a direction that allowed the algorithm to minimize the loss score. The adjustment was made by the optimizer, using a gradient descendent algorithm. The gradient of the loss regarding the model's parameters is computed to find the downhill direction, and the weights (parameters) are moved in small steps (equation 3.1) in the opposite direction from the gradient (equation 3.2), allowing to reduce the loss a little each iteration.

Figure 3.2 shows how the optimizer works. The weights (w) are randomly initiated and repetitively adjusted with small steps until the algorithm converges to a value close to the global minimum. This is achieved using the learning rate hyperparameter and the loss gradient. The learning rate controls the speed of the gradient descendent; therefore, it is crucial to choose a reasonable value for this hyperparameter. If the learning rate is too low, the algorithm will have to go through many iterations, and the loss value may get stuck in a local minimum. If the learning rate is too large, the loss value may exceed the global minimum and jump between completely random locations on the loss curve.

$$step = learning\ rate * gradient(loss, W) \tag{3.1}$$

$$W = W - learning\ rate * gradient(loss, W) \tag{3.2}$$

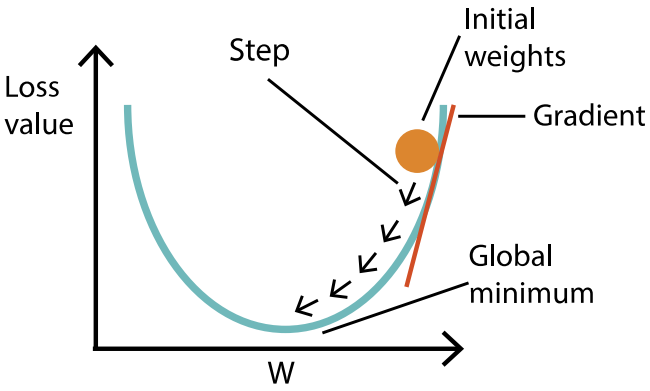

Figure 3.2: Weights optimizer

The gradient descent algorithm measures the local gradient of the loss value concerning the weights (w), and follows the direction that allows obtaining a greater gradient descent (Géron, 2019). When the gradient is zero, the minimum has been reached. The gradient calculates how much the loss value changes when the weights are slightly tweaked. This process is performed iteratively until the minimum is found. Equation 3.3 shows how to calculate the gradient; the aim is to find the set of weights that minimizes the loss value.

$$\Delta_W MSE(W) = \frac{2}{n} X^T (X.W - Y) \tag{3.3}$$

Where W are the weights, n is the number of observations used in the batch where MSE is measured, X is the matrix containing all the features values of the batch (excluding labels), T corresponds to the transpose matrix, and Y is the matrix containing all the labels values of the batch.

Once the gradient vector ($\Delta_W MSE(W)$) is obtained, it is subtracted from W, to go in the downhill direction. The gradient is multiplied by the learning rate to determine the size of the downhill step (equation 3.2).

The LSTM network contains several hyper-parameters that were adjusted to improve the performance of the algorithm's predictions. The model validation error was used as a response signal to adjust the LSTM hyper-parameters in the direction that allowed the algorithm to minimize the validation error using manual fine-tuning strategy.

3.3. Data Collection

Data collection was performed based on the analysis of the most used variables by the articles included in the literature review.

Yfinance was the python application program interface used to download the data from the web site https://finance.yahoo.com. For our study, we collected data from bitcoin, Nasdaq, SP500, gold, oil, volatility index, treasury yield 10 years, British Pound, and Euro prices expressed as US dollars from September 17th, 2014, to April 9th, 2022. Financial daily times stock exchange 100 Index prices in British pound sterling, and US Dollar prices in Japanese Yen were also collected for the same time interval.

3.4. Features description

Data quality is an important factor for the good performance of deep learning prediction algorithms. The choice of the features was motivated by the literature review, where it was possible to verify that using macroeconomic variables improves the performance of the prediction models. Table 3.1 presents the description of the features used in this research study.

Table 3.1: Features description

Features	Description
Close	Bitcoin closing price in USD
Open	Bitcoin opening price in USD
High	Bitcoin highest price of the day in USD
Low	Bitcoin lowest price of the day in USD
Volume	Bitcoin total transactions volume of the day
Nasdaq	National Association of Securities Dealers Automated Quotations closing price in USD
SP500	Standard and Poor's 500 closing price in USD. Index of 500 large listed limited liability companies traded in the United States
Gold	Gold closing price in USD
Oil	Oil closing price in USD

Vix	Volatility Index closing price in USD. Measure of stock market expectations of volatility based on S&P 500 index
Ftse100	Financial Times Stock Exchange 100 Index closing price in GBP. Share index of the 100 companies listed on the London Stock Exchange with the highest market capitalisation.
Tnx	Treasury Yield 10 Years
Gbp_usd	British pound sterling closing price in USD
Eur_usd	Euro closing price in USD
Usd_jpy	US Dollar closing price in Japanese yen

3.5. Feature selection

In total, 15 features were collected to be used as input in the deep learning algorithms. Feature selection was performed with the wrapper forward method to avoid using less significant or irrelevant features that would create noise, higher prediction errors, and increase complexity and execution times. This method consists of selecting one feature and iteratively adding a new feature that improves the model's performance until the point that adding a new feature does not improve the model.

After applying the wrapper forward selection method, it was observed that using only Bitcoin's closing price as an input feature allowed predictions with less noise, lower computational costs, and lower forecast errors. Therefore, for the final models, it was decided to use only the bitcoin closing price as an input feature.

3.6. Train/validation/test data

The dataset was split into training, validation, and test subsets with a ratio of 65%, 15%, and 20%, respectively. Data related to the period from September 17th, 2014, to August 16th, 2019, was used for training, data related to the period from August 17th, 2019, to October 10th, 2020, was used to validate the model, and data related to the period from October 4th, 2020, to April 9th, 2022, was used to test the models (see figure 3.3). This process allowed us to train, validate, test, and tune the parameters of the models, ensuring they can perform well on unseen data. The train and validation split strategy was found after several experiments carried out with different split percentages. We chose the split percentage that obtained the lowest prediction error in the test data.

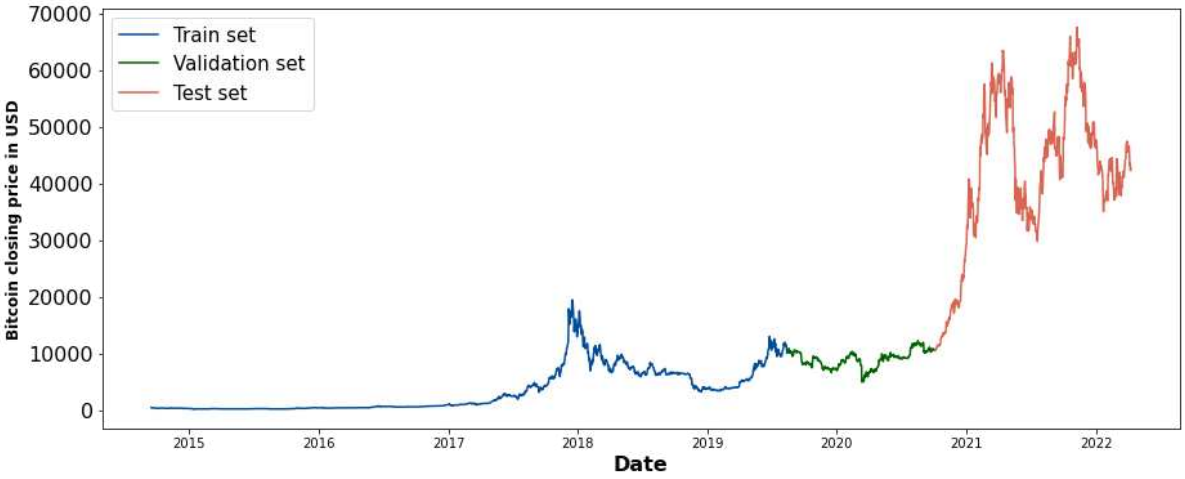

Figure 3.3: Train and test bitcoin closing price (USD per bitcoin)

3.7. Data Pre-processing

This section presents all the steps performed in the data pre-processing phase. Savitzky–Golay filter, data normalization, and data representation for the neural network are explained in detail.

3.7.1. Savitzky–Golay filter

Bitcoin experiences considerable fluctuations in its valuation, making it difficult to predict the trend of its value, (Ferdiansyah et al., 2019). Bitcoin's price volatility is influenced by supply and demand, public sentiment, and government legislation, factors that work together to create price volatility.

Filter operations are important data preparation techniques to be applied before data processing. Savitzky–Golay filter smoothing is a digital filter presented by Savitzky and Golay (1964) that can be applied to a time series to reduce the high-frequency noise in a signal and get a smoother sequence of points. Savitzky–Golay filter uses a windows filter with an equally spaced number of points and fits a polynomial of order *N* to them. The window is moved point by point along the signal, and the fitting polynomial process is carried out at each step.

In order to choose the optimal parameters for the Savitzky–Golay filter, we tried all possible combinations between 0 and 50 for the rolling window size and the polynomial order. The combination of parameters with the lowest error on the validation set was used for the final models. We performed a Savitzky–Golay filter using 29 points (days) rolling window and fitted a 9th-order degree polynomial step by step along the signal.

3.7.2. Feature Scaling

One of the most critical transformations performed on data is feature scaling. Deep learning algorithms typically do not perform well when features are on different scales. Min-Max normalization was the

technique used to normalize the data. This practice subtracts the minimum value from the observed value and divides it by the difference between the maximum and minimum, and consequently, the data ranges between 0 and 1.

3.7.3. Data representation for neural networks

To prepare the data for the deep learning algorithms, we used a rank-3 tensor, which visually can be interpreted as a cube with compartments (see figure 3.4), where the first axis represents the samples, the second the number of time steps, and the third the features. Tensors are the data structure used by deep learning systems. They are a generalization of matrices (rank-2 tensors) that can be used with an arbitrary number of dimensions. They can be defined as a container where the data will be stored and used in our system.

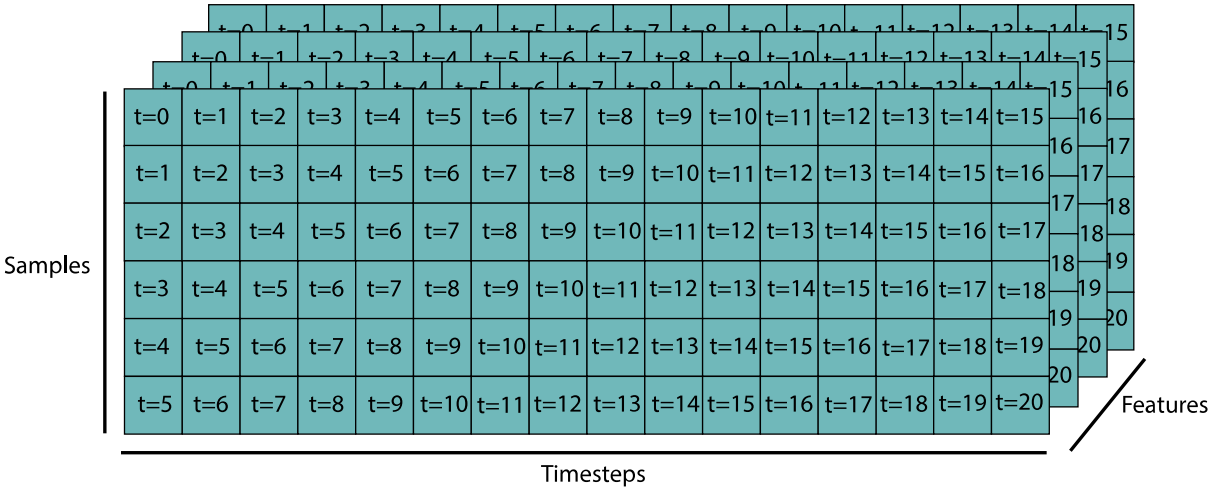

Figure 3.4: Rank-3 timeseries data tensor

We considered 16-time steps, where every step corresponded to one-day bitcoin closing price. This data structuring allowed the creation of a rolling window to represent the dataset as a supervised learning problem, with inputs and labels, where bitcoin prices from the past 16 days were used as input to predict the next day's bitcoin closing price (label) (see figure 3.5).

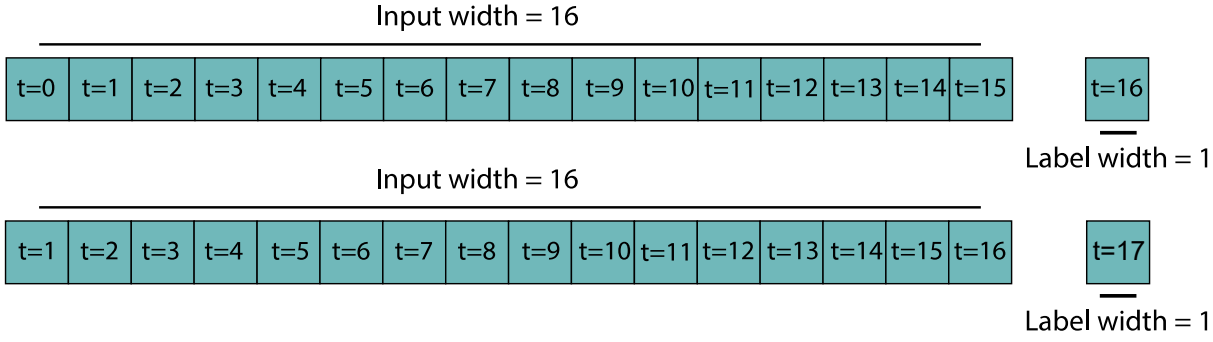

Figure 3.5: Supervised learning rolling window

3.8. A common-sense non-deep learning baseline

Before developing complex deep learning models, we performed bitcoin closing prices prediction with a simple non-deep learning approach. This procedure allows to define a baseline model that must be beaten to demonstrate the usefulness of more advanced deep learning models. We used a 20-day moving average (MA) to predict bitcoin closing prices. Moving average is a widely used indicator in technical analysis; it helps to smooth time series prices and constantly updates the average price over the time series.

3.9. Modelling

This section presents the LSTM and BiLSTM algorithms and the architecture of the final model's implementation.

3.9.1. LSTM network

Most deep learning neural networks do not have memory. The process of mapping inputs to labels uses the entire input time steps sequence at once, turning time steps into just one data point, and causing the inputs to be mapped to labels without memorizing the pattern of the sequences. Recurrent Neural Networks (RNN) were created to solve the time dependency problem; however, despite having good performance learning short-term dependencies, they have difficulties learning Long-Term dependencies. Hochreiter & Schmidhuber (1997) created a type of recurrent neural network (RNN) called Long Short-Term Memory (LSTM) that aimed to solve this problem. LSTM is capable of learning short and long-term dependencies, and they are now regularly used because they work tremendously well in sequential data, including time series (Korstanje, 2021).

The LSTM network is organized in repeated chain modules (figure 3.6), where each module represents a time step in the sequence.

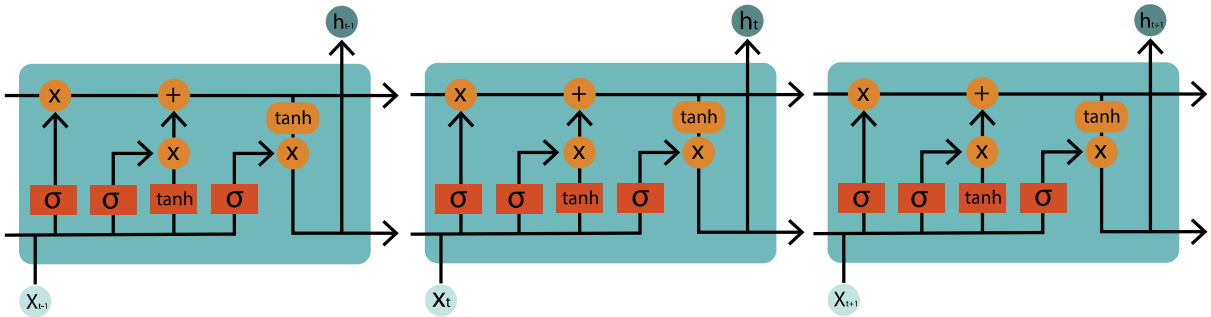

Figure 3.6: LSTM chain modules

One of the main points in the LSTM network is the cell state (C_t), the vector located at the top of the cell modules that crosses the entire chain (see figure 3.7). This cell state is responsible for storing

the long-term information dependencies and patterns with only a few linear interactions, as explained in detail below.

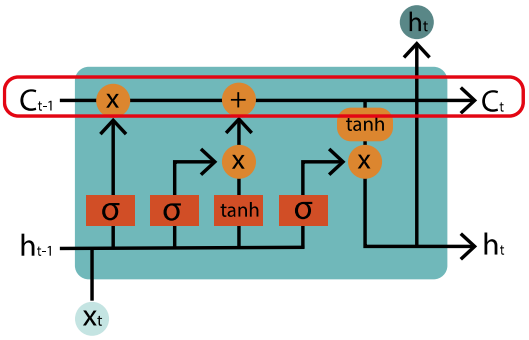

Figure 3.7: LSTM cell state

LSTM network module has three gates composed of a sigmoid neural network layer that act as filters and decide what information is removed or added to the cell state (C_t) (see figure 3.8).

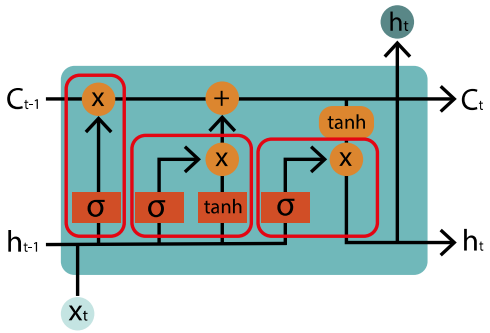

Figure 3.8: LSTM three gates

The first step performed by an LSTM cell is to decide which information will be forgotten from the last time step, cell state (C_{t-1}). This process is performed by a Sigmoid layer called "forget gate" (f_t), which is responsible for concatenating the input in the current time step (x_t) with the hidden vector of the previous time step (h_{t-1}), multiplied by the weight matrix (W_f). To this operation is added the Bias term (b_f), and afterward, the sigmoid function is applied to the entire expression (illustrated in figure 3.9) (equation 3.4). The result of the operation is a number between 0 and 1 for each element of the cell state (C_{t-1}), where 1 represents "remember all information", and 0 represents "forget all information".

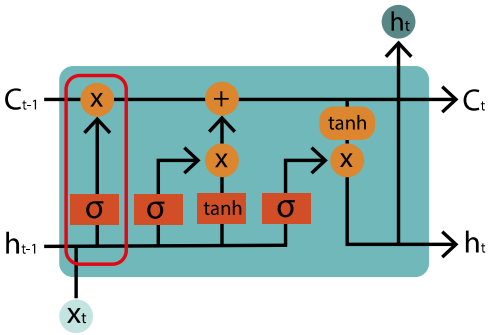

Figure 3.9: LSTM forget gate

$$f_t = \sigma(W_f.[h_{t-1}, x_t] + b_f) \tag{3.4}$$

The second step performed by an LSTM decides what new information is added to the cell state(C_t). This process is carried out in two parts. The first consists of a sigmoid layer, called "input gate", which will act as a filter and decide which values to update. This "input gate" (i_t) is processed by applying the sigmoid function to the concatenation between the input data of the current time step (x_t) with the hidden vector of the previous time step (h_{t-1}), multiplied by the matrix of weights (W_i) and added to the term bias (b_i) (equation 3.5). In the second part, a hyperbolic tangent layer (*tanh*) creates a vector of candidate values (C'_t) to the cell state (C_t). This process is performed again by concatenating the input data from the current time step (x_t) with the hidden vector from the previous time step (h_{t-1}), multiplied by the weight matrix (W_c) and added to the bias term (b_c), but instead of the sigmoid function, the hyperbolic tangent function (*tanh*) is applied to the expression (illustrated in figure 3.10) (equation 3.6).

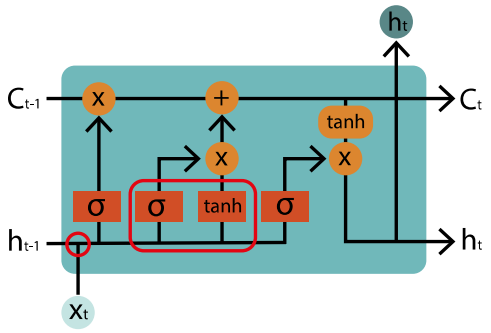

Figure 3.10: LSTM Input gate

$$i_t = \sigma(W_i.[h_{t-1}, x_t] + b_i) \tag{3.5}$$

$$C'_t = \tanh(W_c.[h_{t-1}, x_t] + b_c) \tag{3.6}$$

The third step combines the two parts from the previous step to decide which information should be passed to the current state (C_t). Multiplication is performed between the vector of candidate values (C'_t) to the cell state (C_t) and the input gate (i_t), which acts as a filter of information and decides which input data from the current time step (x_t) and the hidden vector from the previous time step (h_{t-1}) is important to keep and should be passed to the new cell state (C_t) (illustrated in figure 3.11).

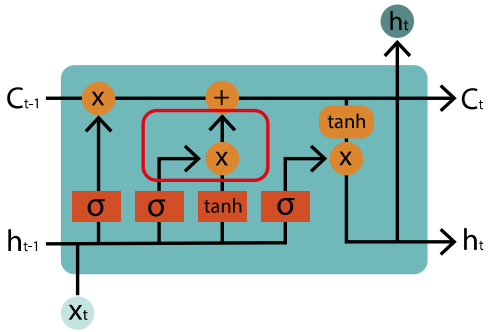

Figure 3.11: LSTM Input gate

The fourth step uses the forget gate (f_t) and the input gate (i_t) to make an update in the cell state (C_t). A sum is made between what should be forgotten about the previous time step (C_t^f) and what is important to add as new information (C_t^i), the result is the update of the old cell state (C_{t-1}) into the new cell state (C_t) (illustrated in figure 3.12) (equation 3.7 and 3.8).

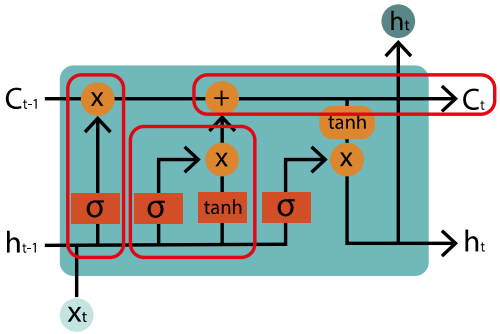

Figure 3.12: Update LSTM cell state

$$C_t = C_t^f + C_t^i \tag{3.7}$$

$$C_t = f_t * C_{t-1} + i_t * C'_t \tag{3.8}$$

In the fifth and last step, the output to be performed by the cell in the current time step is decided. This process is performed by a sigmoid layer called output gate (o_t), that will act as a filter to obtain the output and the hidden state of the current time step (h_t). Once again, this output gate (o_t) is obtained

by applying the sigmoid function to the concatenation between the input data of the current time step (x_t) with the hidden vector of the previous time step (h_{t-1}), multiplied by the matrix of weights (W_o) and added to the term bias (b_o)(equation 3.9). Then, the cell state at the current time step (C_t) is passed through a hyperbolic tangent (*tanh*) and is multiplied by the output gate (o_t). The result is the hidden vector in the current time step (h_t) and the output to be passed to the dense layer, which will be used as a prediction of the LSTM network (illustrated in figure 3.13) (equation 3.10).

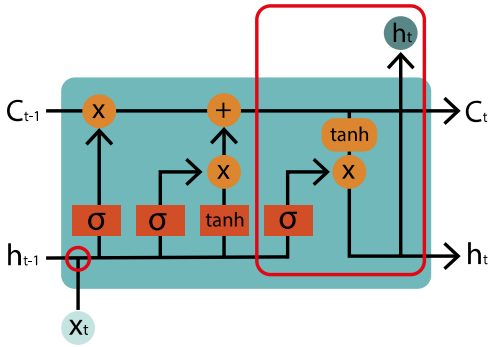

Figure 3.13: LSTM output gate

$$o_t = \sigma(W_o[h_{t-1}, x_t] + b_o) \tag{3.9}$$

$$h_t = o_t * \tanh(C_t) \tag{3.10}$$

3.9.2. BiLSTM network

A bidirectional long-short-term memory network (BiLSTM) is a variant of the LSTM network and is known for its good performance in sequential data predictions. Graves & Schmidhuber (2005) were the first to propose this new network. They applied BiLSTM to phoneme classification, and since then, BiLSTM has been used regularly, demonstrating excellent performance in speech recognition and natural language processing tasks.

LSTM models only consider the input data regarding the past; BiLSTM was created to solve this problem. They consider sequential dependencies regarding the past and the future. Its architecture consists of applying two LSTMs. The first is applied to the input data in a sequential chronological direction (forward layer), and the second is applied to the input data in an anti-chronological sequential direction (backward layer). Then the final outputs are the concatenation between the forward and backward layers. Using LSTM network in both directions helps to improve the learning of long-term dependencies and, consecutively, the prediction errors.

In figure 3.14 it is possible to visualize the operation of the BiLSTM network; the cell architecture is the same as explained in the previous point; the only difference is its application in both chronological and anti-chronological directions of the sequential input data.

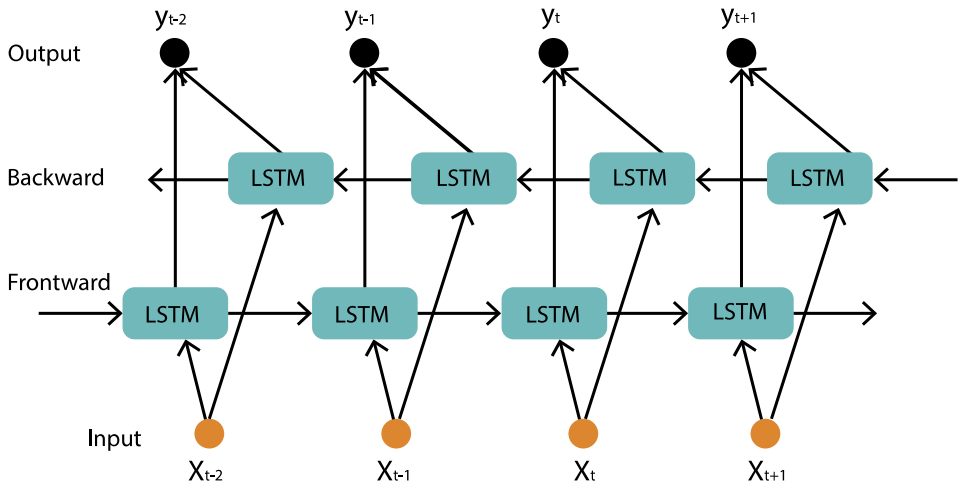

Figure 3.14: BiLSTM neural network

3.9.3. Final models Implementation

This section presents the implementation of the final developed LSTM and BiLSTM models in detail.

3.9.3.1. LSTM architecture

We implemented an LSTM model using Python's Keras library (illustrated in figure 3.15). The first layer took as input the rank 3 tensor that was prepared in the data pre-processing phase. The tensor had the shape of:

[batch size=32, time steps=16, number of features=1].

The 32 batches allowed us to calculate the prediction error at each iteration and adjust the weights of the LSTM architecture in the direction that allowed us to reduce the error. As seen before, 16-time steps (days) were used as input to predict the next day's Bitcoin closing price.

The second layer used in the LSTM architecture consisted of 128 output units and a sigmoid activation function. The third layer was a dropout that was used to regularize the model. This practice was essential because it impeded the model's ability to fit the training data perfectly, allowing it to obtain a model with better performance during validation. Regularization allowed us to obtain a more regular, simple, generic model with a smoother prediction curve and better performance on the validation and test data. Dropout was set to randomly exclude 60% of the layers output features during training. To predict the next day's bitcoin closing price, a dense layer with 1 unit was used in the output layer, corresponding to a 1-day forecast.

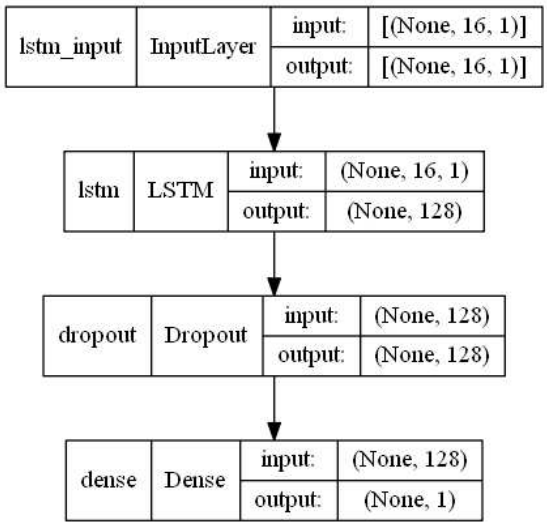

Figure 3.15: Stacked LSTM architecture model in Keras

The model was trained using 65% of the data, validated on 15%, and tested on the remaining 20%. The validation set was used to evaluate the loss at the end of each epoch, but the model was not trained on it. The validation was essential; it allowed us to validate the model's performance during the training process. Also, the information given through the validation error helped to tune the hyper parameters and settings and identify if the model learning process was moving in the right direction.

It is important to note that the test data defined initially is only used to test the model after the training and tuning process is complete. This allows an unbiased evaluation of the final model performance. Adjusting the model weights and hyperparameters based on their performance on the test data would be a mistake and lead the model to overfit the test data. The model would perform artificially well on the test data because it was optimized for it; however, it would not have the same performance in data never seen before.

The model was defined to be trained over 200 epochs; however, an early stopping was defined to interrupt the training process as soon as the validation loss has stopped improving for more than 30 epochs, and of course, the best model obtained during the training phase was saved. This process allowed to stop the training process as soon as the model started to overfitting.

The mean squared error (mse) loss function was used as the response signal to adjust the value of the weights in a direction that allows the algorithm to minimize the loss score. This adjustment was made by the adaptive moment estimation (Adam) optimizer, which modified the weights in the direction that minimises the prediction error. Adam algorithm was presented by Kingma & Ba (2014), and it is a stochastic gradient descendent method based.

In order to monitor the train and validation loss during the training process, we used learning curves. In figure 3.16 it is possible to see that the loss in validation and training decreased to the point

of stability and maintained a minimum distance between the two until the end of the training process, which means a good fit of the model to the training and validation data.

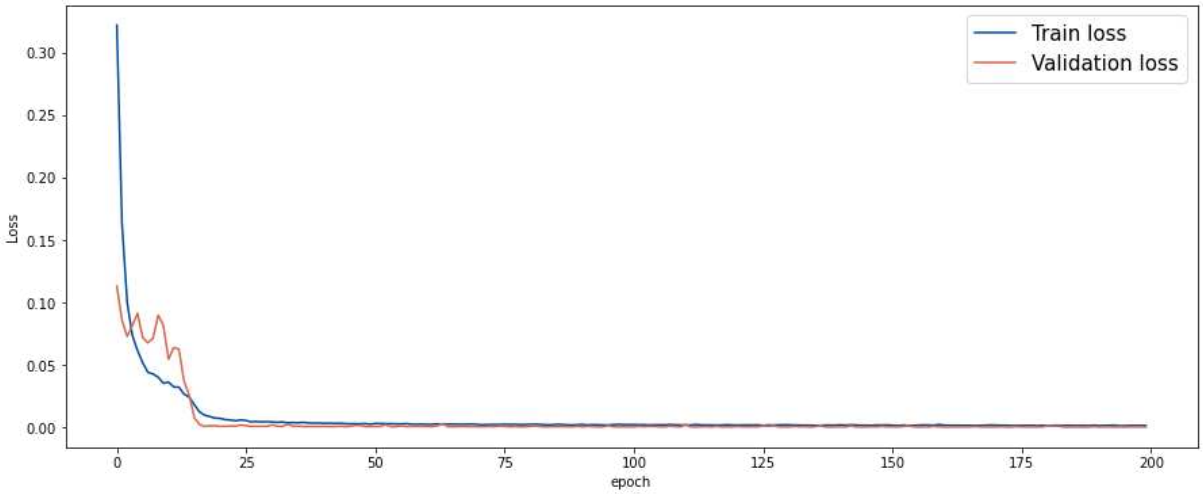

Figure 3.16: LSTM train and validation loss

3.9.3.2. BiLSTM architecture

We implemented a BiLSTM model also using Python's Keras library (illustrated in figure 3.17). The first layer of the model took as input a rank 3 tensor with the format of [batch size=32, time steps=16, number of features=1]. The second layer was a BiLSTM which consisted of 128 output units and a sigmoid activation function. Like the previous model, BiLSTM also was defined with a dropout layer set to randomly exclude 60% of the layers output features during training. The output layer was formed by a dense layer with 1 unit to predict next day bitcoin closing price.

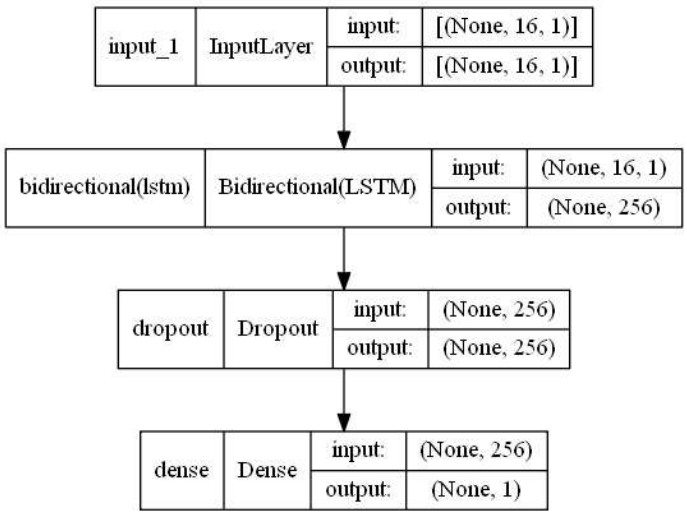

Figure 3.17: Stacked BiLSTM architecture model in Keras

The model was trained using 65% of the data, validated on 15%, and tested on the remaining 20%. The validation set was used to evaluate the loss at the end of each epoch. The model was defined to be trained over 200 epochs, however an early stopping was set to interrupt the training process as soon as the validation loss has stopped improving for more than 20 epochs. The best model obtained during the training process was saved.

From the learning curves in figure 3.18, it is possible to see that the loss in validation and training decreased to the point of stability and maintained a minimum distance between the two until the end of the training process, which again means a good fit of the model to the training and validation data.

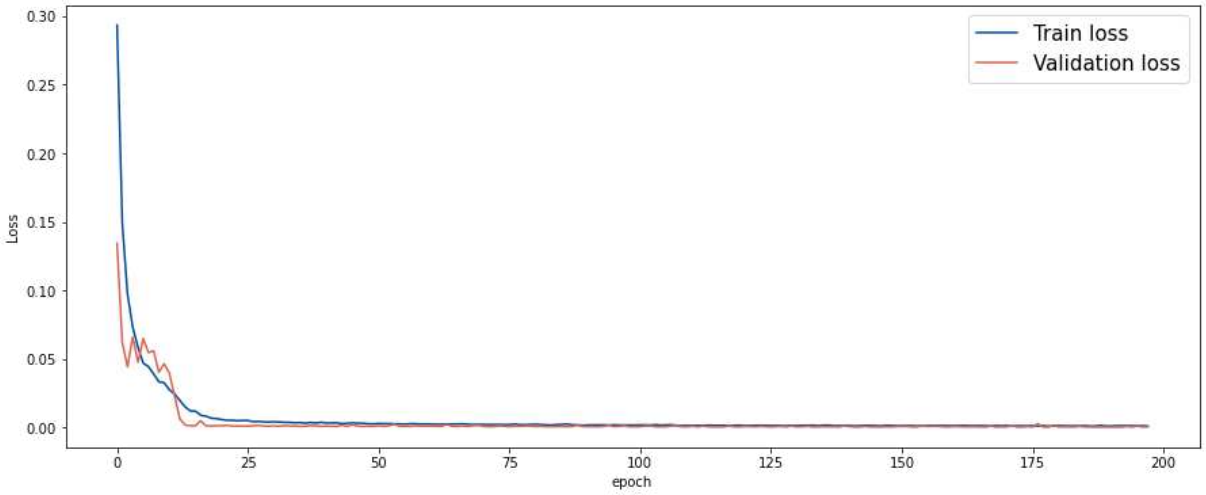

Figure 3.18: BiLSTM train and validation loss

3.9.3.3. Hyper-parameters tuning and regularization

The configuration of the presented models results from hyper-parameters tuning, an important and sophisticated step to obtain a model with good prediction results. The aim was to find a model that maximized generalization performance; therefore, we repeatedly trained and evaluated different hyperparameters settings until good results were achieved on the validation data. Different configurations were tested with the following hyper-parameters:

- Number of layers
- Number of units per layer
- Batch Size
- Optimizer
- Learning rate.

To regularize and tune the hyper-parameters, we increased the number of layers, made them bigger, and trained the model for more epochs until statistical power was achieved, the point right at the border

between underfitting and overfitting. In order to maximize generalization performance, we used regularization dropout to randomly exclude 60% of the layers output features during the training process. To finalize and get the best possible model, we repeatedly train and evaluate the model in validation to adjust the number of units, learning rate, optimizer, and dropout percentage.

3.10. Evaluating deep learning models

To obtain a model with good performance on the validation and test data, it was necessary to measure and compare its performance with the 20-day moving average baseline model and the other studies included in the review. To accomplish this task, predictions were made using the validation and test data predictors (time steps). Then, predictions were converted back to the real scale in dollars. Once the conversion was done, it was necessary to compare the predictions with the observed values; for this purpose, four performance measures were selected, which are presented in the following points.

3.10.1. Root Mean Squared Error

Root mean squared error (RMSE) is a standard performance measure used in regression problems; it gives an idea of the error made by the prediction system and penalizes larger errors. The mathematical formula for its calculation is presented in equation 3.11:

$$RMSE = \sqrt{\frac{1}{n}\sum_{t=1}^{n}(y_t - \hat{y}_t)^2} \tag{3.11}$$

Where n is the number of observations used for testing, y is the observed value, $\hat{y}$ is the predicted value and *t* is the time step.

3.10.2. Mean Absolute Error

Mean absolute error (MAE) is also a widely used measure in regression problems, however, unlike RMSE it is more resistant to anomalies, to large errors. Calculating the mean of absolute errors is a way to ensure that summing the errors won't make cancel each other out. The interpretation of RMSE and MAE are similar; a lower measurement value indicates a better model. MAE mathematical formula is presented in equation 3.12:

$$MAE = \frac{1}{n}\sum_{t=1}^{n}|y_t - \hat{y}_t| \tag{3.12}$$

Where n is the number of observations used for testing, y is the observed value, $\hat{y}$ is the predicted value, and *t* is the time step.

3.10.3. Mean Absolute Percentage Error

Mean absolute percentage error (MAPE) is calculated by taking the error of each prediction divided by the observed value; it is a standardized percentage value on a scale between 0 and 1, with 0 meaning bad and 1 good performance. Compared to the previous errors, MAPE allows us to communicate and compare the performance of the model in a more understood way since it is a standardized error on the same scale. The mathematical formula is presented in equation 3.13:

$$MAPE = \frac{100}{n}\sum_{t=1}^{n}\left|\frac{y_t - \hat{y}_t}{y_t}\right| \quad (3.13)$$

Where n is the number of observations used for testing, y is the observed value, ŷ is the predicted value and *t* is the time step.

3.10.4. R Squared

R squared (R^2) is a performance measure that calculates the ratio between the sum of squared errors and the total sum of squares; it normally ranges from 0 to 1, with 0 meaning bad and 1 good performance. However, there are cases where R^2 can have negative values, in situations where predictions are worse than the average. R^2 can easily be used as a percentage, just being multiplied by 100. The mathematical formula is presented in equation 3.14:

$$R^2 = 1 - \frac{\sum_{t=1}^{n}(y_t - \hat{y}_t)^2}{\sum_{t=1}^{n}(y_t - \bar{y})^2} \quad (3.14)$$

Where n is the number of observations used for testing, y is the observed value, ŷ is the predicted value, $\bar{y}$ is the mean and *t* is the time step.

CHAPTER 4

Data and Results

This chapter presents the data analysis, basic descriptive statistics, and all findings and results of the several experiments carried out for feature selection, Savitzky–Golay filter, hyper-parameters tuning, and dropout regularization. All experiments were trained on the training dataset and validated on the validation dataset, except the final models tested on the test dataset. LSTM and BiLSTM architecture configurations were used as described in the methodology.

4.1. Descriptive statistics

For the range of collected data, the median bitcoin closing price was 6401.27 USD per bitcoin. This value is considerably below the mean because the bitcoin price was relatively low in the first years of analysis compared to the last two years.

The collected data showed a large dispersion of the bitcoin closing prices. As shown in table 4.1, the difference between the minimum and maximum is high, and the standard deviation is greater than the mean.

Table 4.1: Descriptive statistics

	count	mean	standard deviation	minimum	25% percentile	median	75% percentil	maximum
Open	2762.000	11852.373	16468.605	176.897	610.001	6397.815	10820.764	67549.734
High	2762.000	12167.165	16903.202	211.731	613.090	6519.593	11094.353	68789.625
Low	2762.000	11507.241	15973.295	171.510	607.351	6308.550	10525.772	66382.062
Close	2762.000	11866.016	16473.764	178.103	610.262	6401.270	10838.912	67566.828
Volume	2762.000	14925547131.417	19927929598.061	5914570.000	82080178.000	5283425000.000	25281067552.750	350967941479.000
nasdaq	2762.000	91.352	44.166	34.802	59.537	80.355	112.111	210.748
gold	2762.000	1422.242	264.785	1050.800	1223.825	1298.100	1716.275	2051.500
oil	2762.000	55.569	14.930	-37.630	46.063	53.500	63.667	123.700
sp500	2762.000	2849.042	787.660	1829.080	2139.165	2711.020	3223.852	4796.560
vix	2762.000	17.891	7.625	9.140	12.890	15.910	20.810	82.690
ftse100	2762.000	6913.250	544.034	4993.900	6546.500	7031.700	7347.975	7877.500
tnx	2762.000	1.968	0.637	0.499	1.579	2.048	2.387	3.234
gbp_usd	2762.000	1.358	0.102	1.149	1.289	1.326	1.411	1.644
eur_usd	2762.000	1.143	0.049	1.039	1.108	1.133	1.178	1.296
usd_jpy	2762.000	111.443	5.291	99.906	107.996	110.499	113.894	125.629

In figure 4.1 it is possible to visualize the evolution of the bitcoin closing price over the years. We can see that the price in September 2014 was approximately 400 USD per bitcoin and remained relatively low for two years. The first considerable increase in bitcoin price was verified between May and December 2017, when it reached 19497 USD per bitcoin. The price went down slowly until

December 2018, when it reached the value of 3236 USD per bitcoin. Then it started an upward trend until March 2021, when it reached the value of 61243 USD per bitcoin. In March 2021, it started a downward trend until July of the same year, when it reached the value of 31533 USD. In July 2021, it was followed by an upward trend until November 2021, reaching the maximum value of 67566 USD per bitcoin. From that day until today, it has registered a downward trend.

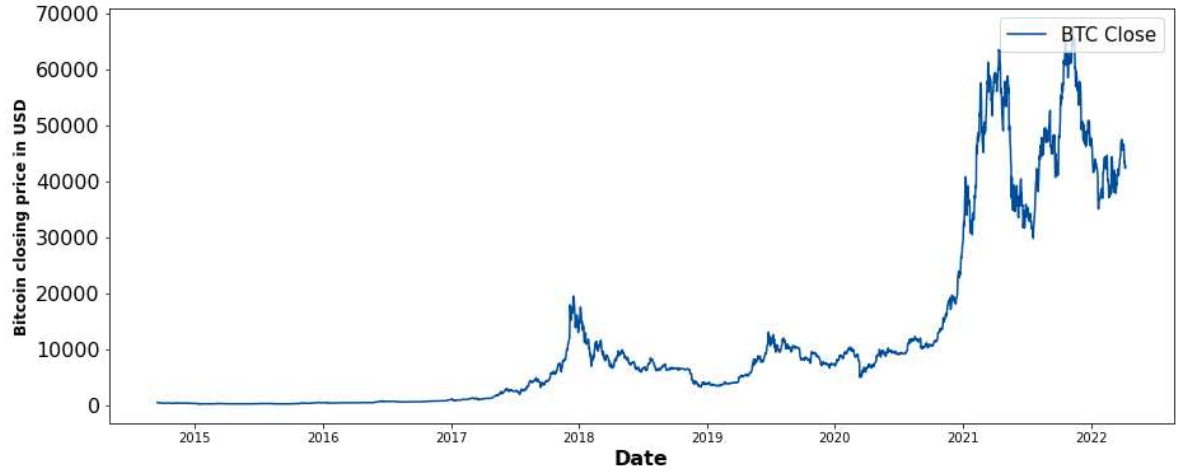

Figure 4.1: Evolution of bitcoin closing price (USD per bitcoin)

4.2. Savitzky–Golay filter

Using Savitzky–Golay filter to smooth the volatility of bitcoin closing prices improved LSTM and BiLSTM bitcoin prediction performance. Experiments performed without and with Savitzky–Golay filter - for all possible combinations between 1 and 50 for the rolling window size (W) and order of the polynomial (N) - were tested in validation data. Table 4.2 shows the best prediction combinations. It has been verified that all best-performing experiments that used Savitzky–Golay filter had better performance predicting the bitcoin closing prices compared to the experiment that didn't use the filter. The filter reduced the noise to obtain a more precise signal in the data; it was essential for deep learning algorithms to achieve better results.

Table 4.2: LSTM and BiLSTM models performance with different Savitzky–Golay filter parameters

		LSTM				BiLSTM			
W	N	RMSE	MAE	MAPE	R^2	RMSE	MAE	MAPE	R^2
No filter		3342.73	2210.08	6.97	0.917	3497.36	2329.09	7.24	0.909
9	5	3144.79	2091.09	6.64	0.926	3116.62	2070.36	6.49	0.926
19	7	3172.63	2067.08	6.61	0.925	3074.59	2051.42	6.50	0.930
21	6	3152.69	2090.26	6.64	0.926	3115.17	**2029.70**	6.50	0.928
25	6	3143.92	2108.06	6.60	0.926	3084.61	2049.10	6.44	0.929
29	9	**3130.30**	**2058.47**	**6.53**	**0.927**	**3061.37**	2035.04	**6.33**	**0.930**

4.3. Feature Selection

Using only bitcoin closing price as input feature allowed us to obtain forecasting models with less noise, lower forecast errors, and less complexity and execution times. Table 4.3 and 4.4 show experiments performed with the wrapper forward selection method, where different sets of input features, feeding the deep learning models, were tested.

Experiments carried out with the LSTM networks revealed that the use of closing, opening and highest bitcoin price of the day as input features allowed us to obtain models with lower forecast errors (see table 4.3). However, using only the closing price allowed us to obtain a model with less noise, less complexity, and less execution times (see figure 4.2).

Table 4.3: LSTM models performance with different set of features

Features	LSTM			
	RMSE	MAE	MAPE	R^2
Close	3283.11	2218.79	6.95	0.920
Close / Open	3235.18	2168.42	6.82	0.920
Close / Open / High	**3107.13**	**2011.65**	**6.45**	**0.928**
Close / Open / High / Low	3315.34	2283.80	7.08	0.918
Close / Open / High / Volume	3129.46	2048.94	6.48	0.927
Close / Open / High / Nasdaq	3191.31	2160.88	6.72	0.924
Close / Open / High / Sp500	3183.46	2142.23	6.72	0.920
Close / Open / High / Gold	3174.89	2136.21	6.62	0.925
Close / Open / High / Oil	3129.20	2032.78	6.48	0.927
Close / Open / High / Vix	3183.41	2139.98	6.67	0.925
Close / Open / High / Ftse100	3165.45	2078.15	6.57	0.925
Close / Open / High / Tnx	3182.57	2059.44	6.53	0.925
Close / Open / High / Gdp_usd	3127.72	2061.94	6.47	0.927
Close / Open / High / Eur_usd	3185.95	2142.59	6.72	0.924
Close / Open / High / Usd_jpy	3171.71	2157.17	6.65	0.925

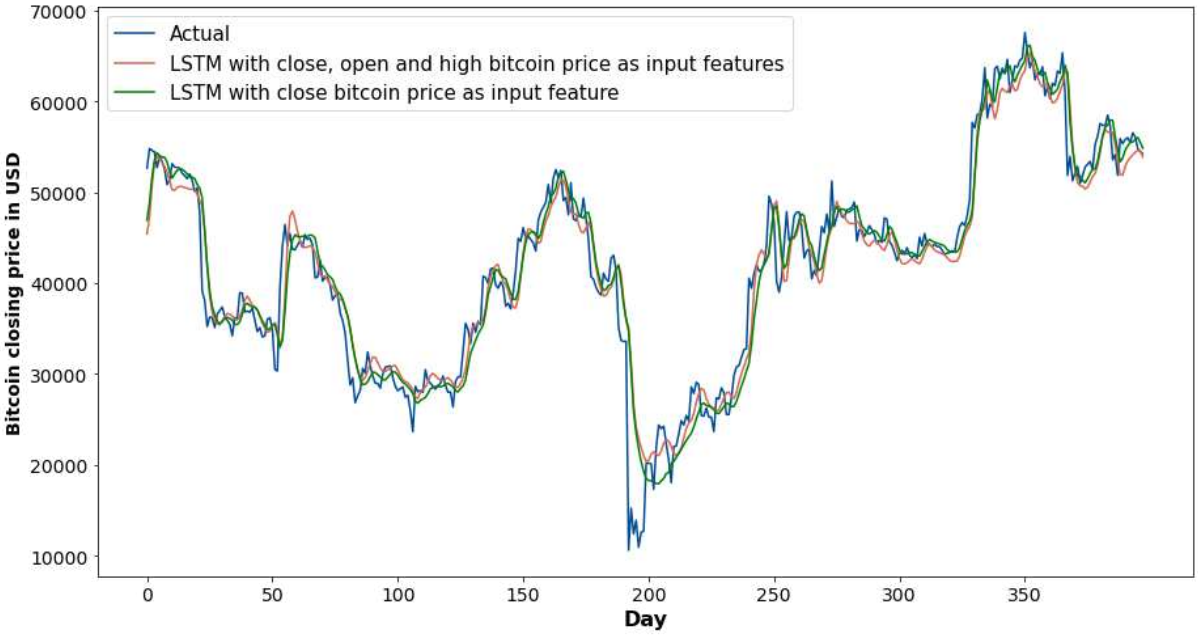

Figure 4.2: Actual and LSTM predicted bitcoin closing Price with different set of input features

Regarding the BiLSTM network, it was verified that using only bitcoin closing price as input feature allowed us to obtain models with lower forecast errors, less noise, less complexity, and less execution times (see table 4.4 and figure 4.3).

Table 4.4: BiLSTM models performance with different set of features

	BiLSTM			
Features	RMSE	MAE	MAPE	R^2
Close	**3238.28**	**2150.95**	**6.69**	**0.922**
Close / Open	3585.27	2383.20	7.46	0.904
Close / Open / High	3316.55	2179.54	6.93	0.918
Close / Open / High / Low	3505.78	2317.42	7.25	0.909
Close / Open / High / Volume	3359.71	2165.48	6.94	0.916
Close / Open / High / Nasdaq	3434.88	2355.29	7.26	0.912
Close / Open / High / Sp500	3590.82	2396.17	7.57	0.904
Close / Open / High / Gold	3396.71	2274.77	7.03	0.914
Close / Open / High / Oil	3271.60	2152.19	6.70	0.920
Close / Open / High / Vix	3298.65	2288.37	6.95	0.919
Close / Open / High / Ftse100	3318.92	3318.92	7.00	0.918
Close / Open / High / Tnx	3470.02	2262.89	7.13	0.910
Close / Open / High / Gdp_usd	3368.71	2241.26	7.05	0.916
Close / Open / High / Eur_usd	3289.14	2203.24	6.92	0.919
Close / Open / High / Usd_jpy	3559.16	2410.04	7.44	0.906

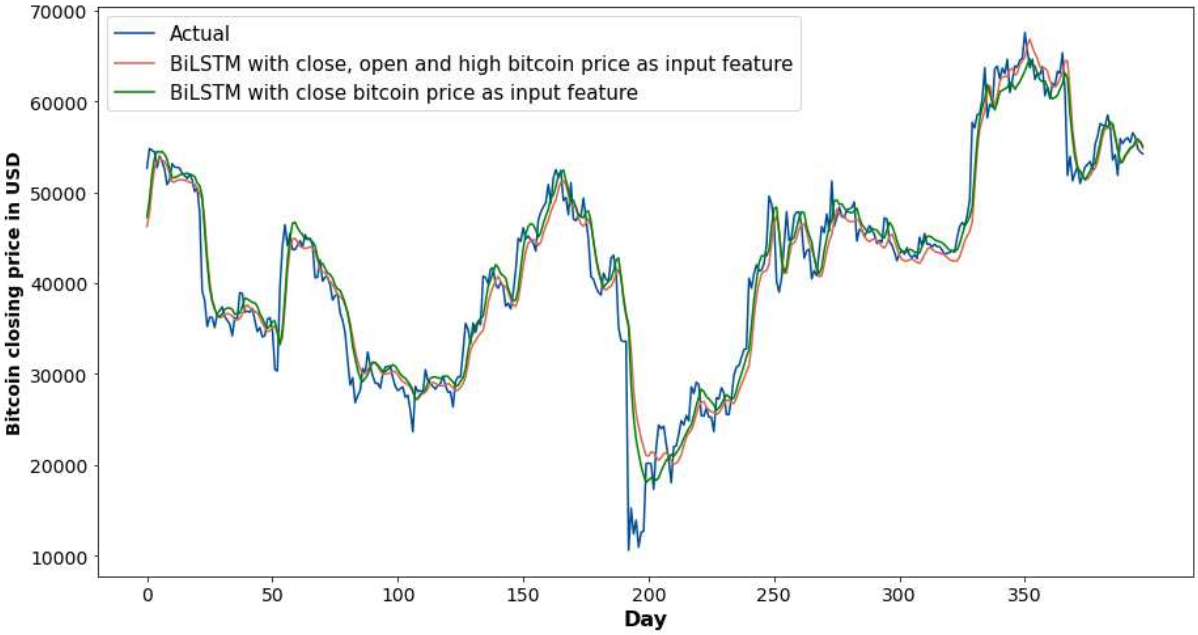

Figure 4.3: Actual and BiLSTM predicted bitcoin closing Price with different set of input features

4.4. Hyper-parameters tuning

This section presents all the findings regarding the LSTM and BiLSTM hyper-parameters tuning process. The results presented can help other researchers to select the best set of hyper-parameters to predict bitcoin closing prices. The prediction errors are the outcome of the hyperparameter experiments and analysis performed in LSTM and BiLSTM models as described in the methodology.

4.4.1. Number of hidden layers

Using only one LSTM and BiLSTM layer showed the best performance in predicting the bitcoin closing price. As shown in table 4.5, experiments performed with different numbers of hidden layers demonstrated that - the use of 3 layers - had the lowest prediction error in LSTM and BiLSTM models. However, using just 1 layer showed a greater ability of the model to generalize to new data, as it has greater performance predicting bitcoin closing prices, with a smoother, more generic, and less noisy forecast line, as shown in figure 4.4 and 4.5.

Table 4.5: LSTM and BilSTM models performance with different number of layers

Number of layers	LSTM				BiLSTM			
	RMSE	MAE	MAPE	R^2	RMSE	MAE	MAPE	R^2
1	3246.93	**2188.49**	6.84	0.922	3247.16	**2134.94**	6.70	0.922
2	1762.59	2699.54	5.49	0.946	1867.56	2804.09	5.75	0.941
3	**1712.57**	2611.85	**5.25**	**0.949**	**1613.32**	2608.14	**5.21**	**0.949**

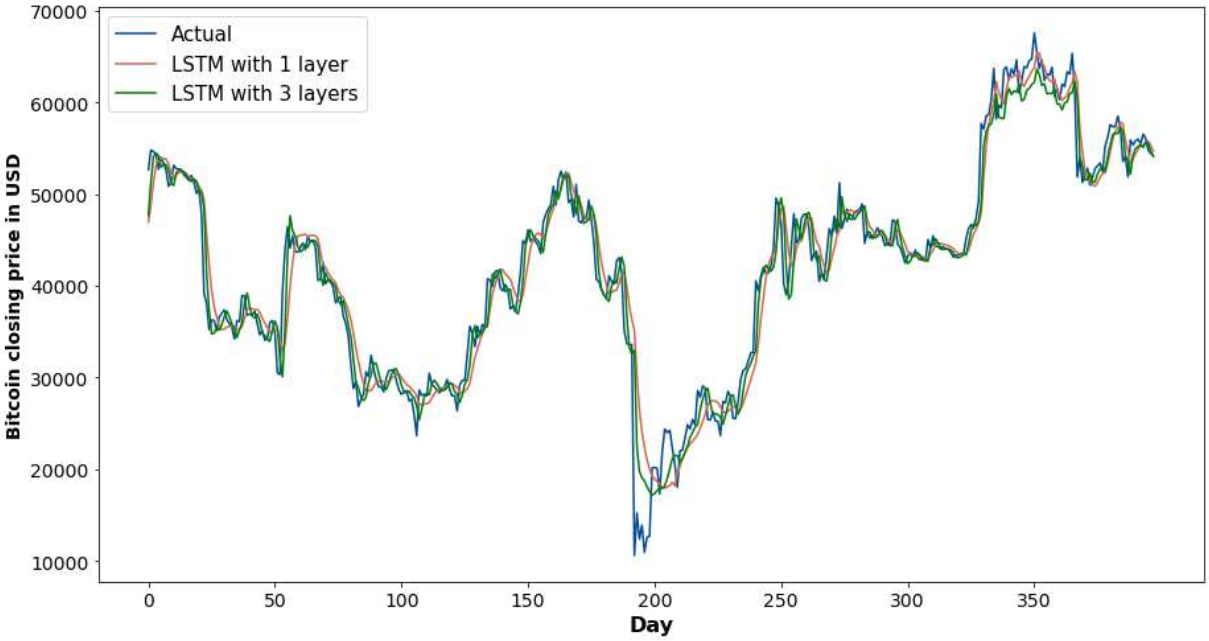

Figure 4.4: Actual and LSTM Predicted bitcoin closing price with 1 and 3 layers

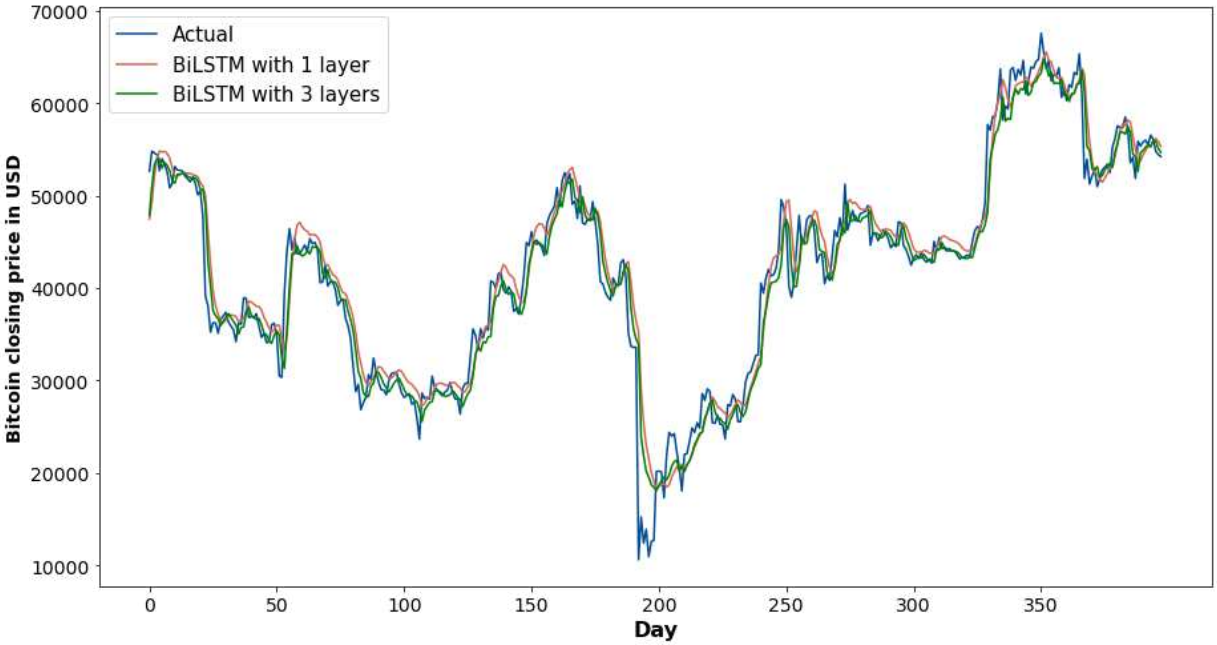

Figure 4.5: Actual and BiLSTM Predicted bitcoin closing price with 1 and 3 layers

4.4.2. Number of units in the LSTM and BiLSTM hidden layer

The number of units was essential in obtaining a good bitcoin closing price model. Table 4.6 shows experiments performed with different numbers of units in LSTM and BiLSTM hidden layers. The analysis of the results allowed us to conclude that models with less than 128 units performed worse than models with more than 128 units.

Table 4.6: LSTM and BilSTM models performance with different number of units

Number of units	LSTM				BiLSTM			
	RMSE	MAE	MAPE	R^2	RMSE	MAE	MAPE	R^2
8	6867.34	5550.13	13.06	0.651	4928.082	3660.91	10.32	0.820
16	4328.68	3302.36	8.83	0.861	4371.28	3108.03	9.38	0.858
32	3669.35	2527.12	7.92	0.900	3632.78	2430.64	7.68	0.902
64	3304.33	2222.36	7.01	0.919	3286.50	2169.41	6.84	0.920
128	3246.93	2188.49	6.84	0.922	3247.16	2134.94	6.70	0.922
256	3104.74	2078.35	6.53	0.928	**3078.94**	**2097.30**	**6.52**	**0.929**
512	**3095.31**	**2056.67**	**6.46**	**0.929**	3174.27	2154.87	6.60	0.925

4.4.3. Batch size

The batch size also significantly impacted the performance of bitcoin closing price prediction models. As presented in table 4.7, experiments performed with different batch sizes revealed that using a smaller batch size - from 1 to 32 - improved LSTM and BiLSTM performance, while using a larger batch size - from 64 to 128 - revealed to show worse results. Batch sizes bigger than 32 led to instabilities at the beginning of the training process; the models did not generalize as well as those trained with smaller batch sizes.

Table 4.7: LSTM and BilSTM models performance with different batches sizes

Batch size	LSTM				BiLSTM			
	RMSE	MAE	MAPE	R^2	RMSE	MAE	MAPE	R^2
1	3274.96	2207.95	6.82	0.921	**2939.96**	**1950.40**	**6.01**	**0.936**
2	3125.02	2080.04	6.49	0.927	3229.85	2116.24	6.55	0.922
4	2971.66	1966.20	6.01	0.934	3200.26	2092.39	6.54	0.924
8	**2905.80**	**1936.88**	**5.96**	**0.937**	3182.01	2099.67	6.65	0.925
16	2954.15	1956.51	6.01	0.935	3332.42	2205.90	6.90	0.917
32	3246.93	2188.49	6.84	0.922	3247.16	2134.94	6.70	0.922
64	3505.20	2312.82	7.40	0.909	3399.81	2297.60	7.21	0.914
128	5282.08	3884.17	10.85	0.793	5402.10	3979.06	11.54	0.784

4.4.4. Optimizer

Adam was the best performing optimizer predicting bitcoin closing prices. Table 4.8 shows the performance of the LSTM and BiLSTM models using different optimizers with a learning rate of 0.01. The results show that Adam had the lowest prediction error, followed by the Root Mean Squared Propagation (RMSprop) and Nesterov-accelerated Adaptive Moment Estimation (Nadam), which also

obtained considerably good results. It was also possible to verify that the use of stochastic gradient descent (SGD), adaptive delta (Adadelta), adaptive gradient algorithm (Adagrad), and follow the regularized leader (FTRL), has led LSTM and BiLSTM to obtain a negative R^2, which means the prediction tends to be less accurate than the average value of bitcoin price over the time.

Table 4.8: LSTM and BilSTM models performance with different optimizers

Optimizer	LSTM				BiLSTM			
	RMSE	MAE	MAPE	R^2	RMSE	MAE	MAPE	R^2
Adam	**3104.74**	**2078.35**	**6.53**	**0.928**	**3078.94**	**2097.30**	**6.52**	**0.929**
SGD	20094.66	17231.65	37.98	-1.982	18957.69	16025.88	35.42	-1.658
RMSprop	3115.26	2078.85	6.57	0.928	3304.65	2137.46	6.80	0.919
Adadelta	24488.96	21926.32	48.84	-3.430	23948.09	21298.05	47.32	-3.240
Adagrad	22772.79	20068.77	44.44	-2.830	23368.41	20687.62	45.88	-3.030
Adamax	4302.81	3084.40	9.23	0.863	5497.24	4141.19	11.32	0.776
Nadam	3235.27	2147.86	6.78	0.922	3128.99	2098.33	6.53	0.927
Ftrl	24052.05	21390.73	47.51	-3.275	21189.74	21189.74	47.04	-3.210

4.4.5. Learning Rate

The learning rate was an important parameter in obtaining good forecasting performance. Table 4.9 shows the LSTM and BiLSTM performance on validation data using different learning rates with Adam optimizer. It was found that the use of a 0.01 learning rate had the best prediction performance. It was also verified that 0.0001 and 0.1 learning rates had considerably poor results in the LSTM model. The 0.0001 leaning rate was too low; the loss value got stuck in a local minimum and could not reach the global minimum. The 0.1 learning rate was too large, and the loss value exceeded and jumped between random locations near the global minimum.

Table 4.9: LSTM and BilSTM models performance with different learning rates

Learning Rate	LSTM				BiLSTM			
	RMSE	MAE	MAPE	R^2	RMSE	MAE	MAPE	R^2
0.1	5184.61	3908.55	10.56	0.801	3117.77	2069.02	6.44	0.928
0.01	**2724.27**	**1782.43**	**5.36**	**0.945**	**2976.49**	**2000.62**	**6.25**	**0.934**
0.001	3209.38	2167.31	6.71	0.923	3165.10	2112.80	6.61	0.925
0.0001	4322.67	3093.52	9.29	0.861	3645.32	2839.01	6.69	0.919

4.5. Dropout regularization

Dropout regularization allowed us to obtain a more regular, simple, and generic model with a smoother prediction curve, and with better performance on validation data. Table 4.10, figures 4.6 and 4.7 show experiments performed without (0% dropout) and with dropout defined to exclude between 10% and 60% of the layers output features during training. It is possible to visualize that not using dropout allows a lower prediction error; however, setting dropout to 60% allows a more regular, simple, smoother, and generic prediction of the bitcoin closing price.

Table 4.10: LSTM and BilSTM models performance with different dropout percentage

Dropout	LSTM				BiLSTM			
Percentage	RMSE	MAE	MAPE	R^2	RMSE	MAE	MAPE	R^2
0%	**2448.71**	**1521.08**	**4.48**	**0.955**	**2408.90**	**1486.53**	**4.36**	**0.957**
10%	3093.80	2164.90	6.63	0.929	2799.43	1850.94	5.71	0.942
60%	3206.12	3206.12	6.82	0.923	3202.53	2169.83	6.66	0.924

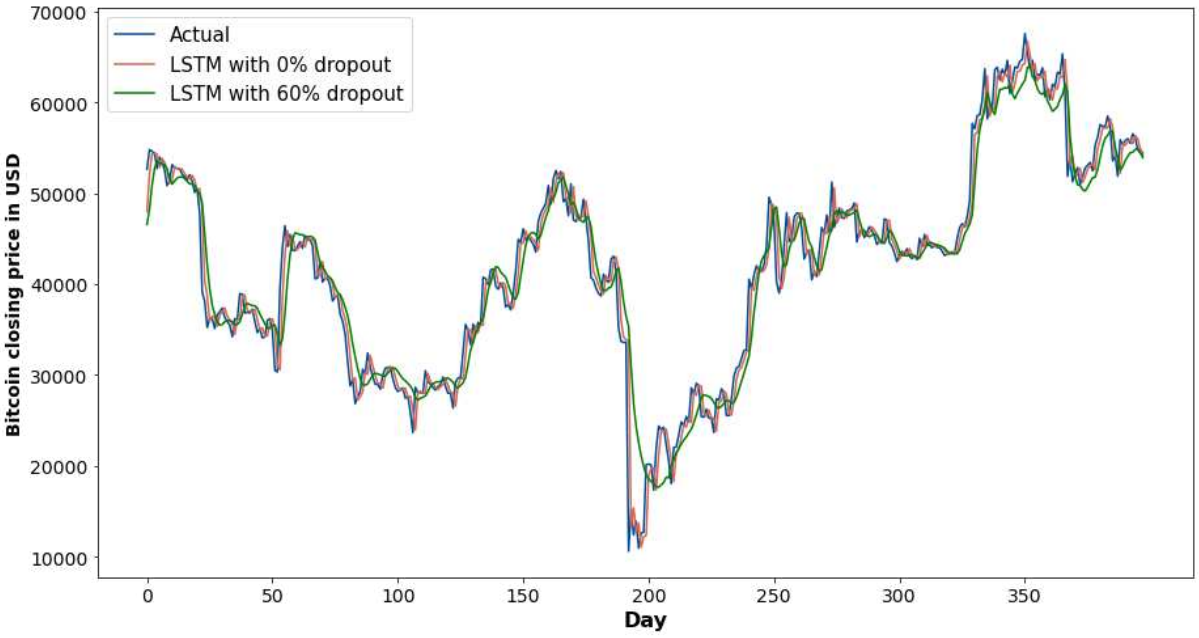

Figure 4.6: Actual and LSTM Predicted bitcoin closing price with 0% and 60% dropout

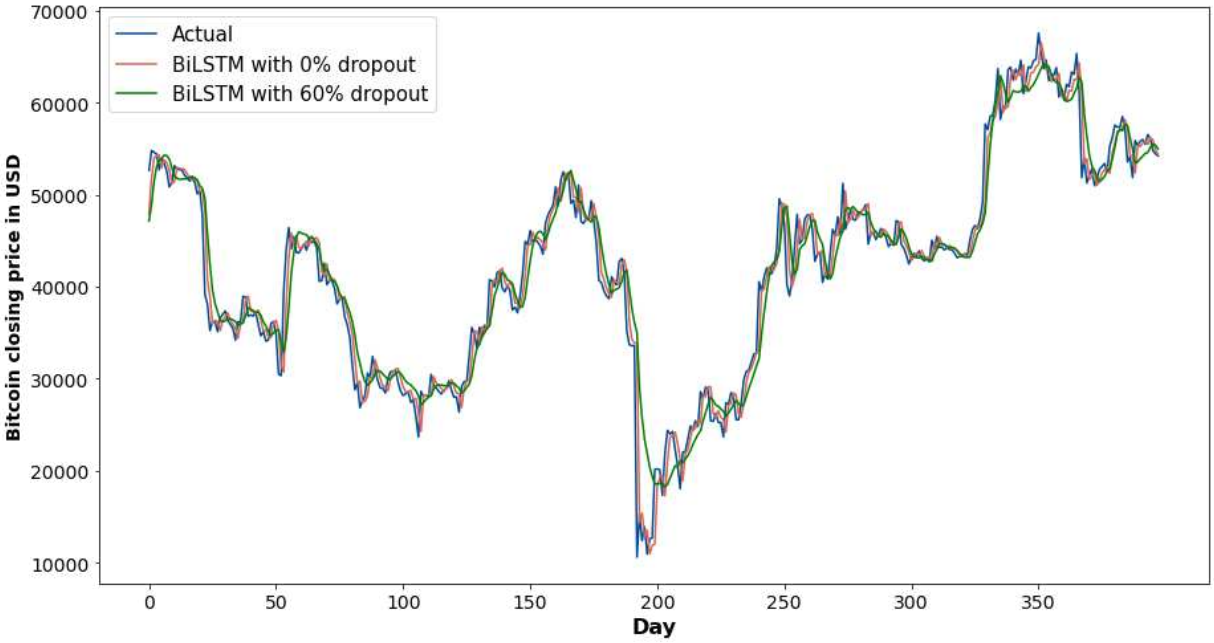

Figure 4.7: Actual and BiLSTM Predicted bitcoin closing price with 0% and 60% dropout

4.6. Final LSTM and BiLSTM models evaluation

This section presents the final LSTM and BiLSTM evaluation results - on the test dataset. Each model was trained and tested 100 times to obtain statistically significant values for each performance metric. It is important to note that this evaluation was only performed after the models were evaluated and tuned in the validation data. The prediction errors presented in this section are the outcome of the final LSTM and BiLSTM model configuration described in the methodology.

LSTM and BiLSTM networks had almost the same performance predicting the Bitcoin closing price. LSTM obtained 4.49% MAPE and BiLSTM 4.44% MAPE (see table 4.11); however, LSTM was 32,28% faster in the model training process (see table 4.12). Both deep learning models performed considerably better than the 20-day moving average baseline.

Table 4.11: Final LSTM and BiLSTM models performance

Model	RMSE	MAE	MAPE	R^2
20-Day MA	4291.67	3327.81	8.14	0.887
LSTM	2223.60	1762.16	4.49	0.970
BiLSTM	**2220.88**	**1744.49**	**4.44**	**0.970**

Table 4.12: Training times for LSTM and BiLSTM in seconds

Model	Time (sec)
LSTM	**57.81**
BiLSTM	76.47

In figure 4.8 it is plotted the actual and predicted bitcoin closing prices for LSTM and BiLSTM. The red and green lines are the result of the bitcoin closing price predictions, and the blue line is the actual bitcoin closing prices from the test data.

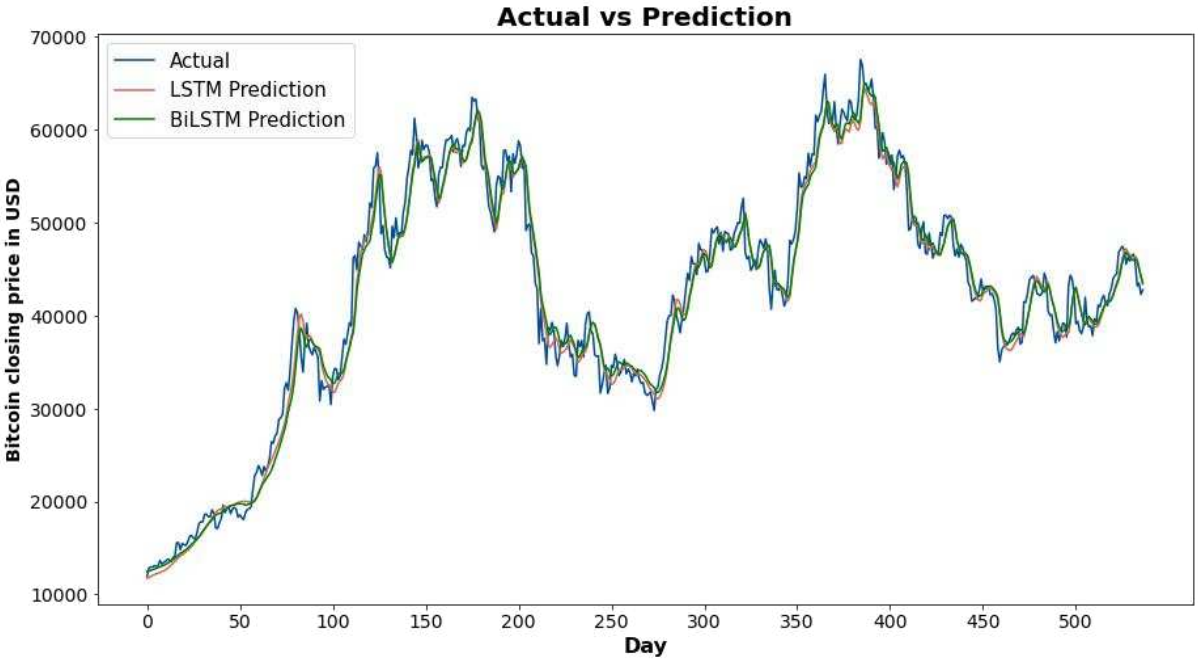

Figure 4.8: Actual, BiLSTM and LSTM predicted bitcoin closing Price

CHAPTER 5

Discussion

5.1. Key findings

With the work developed in this dissertation, we found evidence that LSTM and BiLSTM models could be used effectively in predicting the bitcoin closing prices. In addition, it was also confirmed that using Savitzky–Golay filter and dropout regularization significantly improved the performance of the models. Lastly, we did not find evidence that using micro and macroeconomic variables improves the performance of the models.

5.2. Study limitations

As with all research studies, this work also has some limitations. First, the economic variables collected for the study showed to be little significant or almost irrelevant, leading to the creation of noise, higher prediction errors and an increase in complexity and execution times. Therefore, the proposed final models were built only using bitcoin closing prices. Second, due to the time and scope limitations of this master dissertation, we did not take into consideration the analysis of the public sentiment toward cryptocurrency investing.

5.3. Limitations from studies

Several limitations also arose from the existing studies included in the review. One is that most studies focus on achieving better performance by exploring only more sophisticated models and techniques, ignoring gathering information that can lead the models to obtain better results. It was also verified that most studies did not consider the complexity and non-stationarity of the cryptocurrency time series. Most studies did not use any differentiation or filter to smooth the high volatility of cryptocurrency prices. Statistical and deep learning methods could benefit from the clearer signal in the data. Lastly, public sentiment, policies, and laws toward digital currencies were not taken into consideration by the majority of the studies.

5.4. Strengths

Notwithstanding the limitations, this study has key strengths. First, this is the earliest study to use Savitzky–Golay filter with LSTM models to predict bitcoin prices. The filter was used to eliminate the

random noise of bitcoin prices while preserving the true spectral signal. The clearer signal was essential to help deep learning algorithms to achieve better predicting results. Second, models presented in this study had relatively low prediction errors on test data compared to the studies included in the review. The models effectively predict the price and the trend of bitcoin, with a smooth forecast line and low noise. Third, this study describes in detail the methodology strategy and deep learning algorithms architectures used.

5.5. Incidental observation

Although not the focus of this research, we noticed some incidental observations. First, we found evidence that the good parameterization of the models is essential to obtain a model with good performance. The number of hidden layers, number of units per layer, batch size, optimizer, and learning rate were factors that strongly interfered with the performance of the prediction models. Second, scaling up the models to the point right at the border between underfitting and overfitting and then using dropout to regularize - proved to be the most effective way to develop a model that predicts bitcoin closing prices. The time spent in the hyper-parameter tuning process was high; however, it was the key point to obtaining models with good results.

5.6. Comparison with deep learning models included in the review

Table 5.1 compares the models developed in this study and the next day's bitcoin deep learning models included in the review.

Among all the studies included in the review, the models developed in this study obtained the second-best MAPE for LSTM single models and the fifth-best MAPE among all deep learning studies.

Rather than knowing the exact future price, bitcoin investors are more interested in the future price trend. Most studies in the review predicted the bitcoin price very close to the last day, obtaining a prediction line with noise. This research study took this problem into consideration and employed dropout regularization and a suitable parameterization to prevent overfitting and obtain a smooth prediction line with little noise and low prediction error. Savitzky–Golay filter was also essential to obtain good predictions.

Radityo et al. (2018) developed the two deep learning models included in the review with the lowest MAPE. The success factor in obtaining these results was determined by using the genetic algorithm backpropagation neural network (GABPNN), and backpropagation neural network (BPNN), leading to a 1.88% and a 1.998% MAPE, respectively. Feature generation was also an essential factor. They used exponential moving average (EMA), 12-day rolling window along volume, high, low, and close prices as variables to predict bitcoin prices.

The ANN and RNN models developed by Mallqui and Fernandes, (2019) had a 3.06% and 3.36% MAPE, respectively, considerably good results compared to other deep learning models in the review. They used a technique called Correlation-based Feature Subset selection (CFS) to evaluate the value of a subset of variables, considering each attribute's predictive capacity and the level of redundancy between them.

Table 5.1 shows that the algorithm used can make the difference; however, as verified in this study and the literature review, factors such as data processing, feature selection, feature generation, and hyper-parameters tuning have tremendous importance in the algorithm's prediction performance

Table 5.1: Errors of the deep learning models included in the review

Nº	Author (year)	Cryptocurrency	Interval data	Method	Results		
					RMSE	MAPE	R^2
This study		Bitcoin	1-day	LSTM	2223.60	4.47	0.971
				BiLSTM	2220.88	4.44	0.970
4	Ferdiansyah et al. (2019)	Bitcoin	1-day	LSTM	288.60	-	-
5	Livieris et al. (2021)	Bitcoin	1-day	LSTM	256.68	-	0.953
6	Rizwan et al. (2019)	Bitcoin	1-day	GRU	-	-	0.992
				LSTM	-	-	0.992
8	Lahmiri and Bekiros (2019)	Bitcoin	1-daty	DLNN	2750.00	-	-
				GRNN	8800.00	-	-
11	Altan et al. (2019)	Bitcoin	1-day	LSTM	1474.20	9.59	-
				EWT-LSTM	776.74	6.14	-
				EWT-LSTM-CS	623.41	3.55	-
15	Liu et al. (2021)	Bitcoin	1-day	BPNN	390.07	37.36	-
				SDAE	160.63	10.19	-
16	Tan and Kashef (2019)	Bitcoin	1-day	LSTM	33.70	-	-
17	Mallqui and Fernandes (2019)	Bitcoin	1-day	ANN	41.62	3.06	-
				RNN	42.34	3.36	-
18	Radityo et al. (2018)	Bitcoin	1-day	BPNN	-	1.998	-
				GANN	-	4.461	-
				GABPNN	-	1.883	-
19	Jay et al. (2020)	Bitcoin	1-day	MLP	-	3.06	-
				LSTM	-	3.20	-

5.7. Future research

This study points to several promising directions for future research. First, studies can be improved using data smoothing techniques like the Savitzky-Golay filter, combined with hybrid models that use cryptocurrency prices, sentiment analysis, and public opinion. Second, future studies should search for economic variables with high explanatory value over cryptocurrency prices. Third, transfer learning is a rising research problem in machine learning that should be considered to predict newer cryptocurrency prices with less temporal data available. Data from older cryptocurrencys should be used to predict the prices of the newer cryptocurrencys. Fourth, it may be worth exploring automated hyper-parameter tuning techniques to find the best set of hyper-parameters. Lastly, future studies should also focus on studying the volatility and returns of bitcoin prices.

CHAPTER 6

Conclusion

This study aimed to verify whether LSTM and BiLSTM neural networks can be used to predict bitcoin closing prices. Data was collected from daily bitcoin prices in USD and economic variables from September 17th, 2014, to April 9th, 2022. The dataset was split into 65% for training, 15% for validation and 20% to test the model. In order to reduce the high-frequency noise in the signal, Savitzky–Golay filter was used with 29 points (days) rolling window and a 9th-order degree polynomial fitted step by step along the signal. The transformation of the features in the same scale was employed by Min-Max normalization. Rank-3 tensor was used to prepare the data for the deep learning algorithms and to create a rolling window to represent the dataset as a supervised learning problem; the past 16 days' bitcoin prices were used as input to predict the next day's bitcoin closing price.

We applied the BiLSTM and LSTM algorithms to make the predictions. The LSTM model had as input a rank 3 tensor with 32 batch sizes, 16-time steps, and 1 feature. The second layer of the model was a LSTM with 128 output units and a sigmoid activation function. The third layer was a dropout, set to randomly exclude 60% of the layers output features during training. The output model was set with 1-unit dense layer, which was used to forecast 1 day bitcoin closing price. The model was trained over 200 epochs, with an early stopping defined to interrupt the training process when the validation loss has stopped improving for more than 30 epochs. The BiLSTM model was defined with the same configuration as LSTM; the only difference was the early stopping that was defined to interrupt the training process when the validation loss had stopped improving for more than 20 epochs. To obtain a model with good performance on the validation and test dataset - RMSE, MAE, MAPE, and R^2 - were used to measure the model's performance.

The empirical results showed that both LSTM and BiLSTM could be used effectively in predicting the bitcoin closing prices, with almost the same prediction error, 4.49% and 4.44% MAPE, respectively. We also found evidence that the Savitzky-Golay filter and dropout regularization significantly improved the model's performance. However, we did not find evidence that using economic variables improves the performance of the models. Experiments performed with the wrapper forward selection method, where different sets of input features to feed the deep learning models were tested showed that using only Bitcoin's closing price as input feature allowed forecasting models with less noise, lower forecast errors and less complexity and execution times. We also observed some incidental observations, the good parameterization of the models was essential to obtain a model with good performance. The number of hidden layers, number of units per layer, batch size, optimizer and learning rate revealed to be factors that strongly interfere in the performance of the prediction models.

This study has several contributions to science and society in general. First, bitcoin investors are more interested in the future price trend rather than knowing the exact future price; for that reason, the models developed in this study give information about both: the exact future price and price trend. This study provides valuable information that can be used by fund managers, investment portfolio managers, governments, and investors in general to support investment decision-making. Second, this was the first study to use Savitzky–Golay filter with LSTM models to predict bitcoin prices, the filter was used to eliminate the random noise of bitcoin prices while preserving the true spectral signal. Third, this study presents the results of several experiments performed in the manual hyperparameter tuning process, allowing future researchers to use this study as support to choose the set of optimal hyperparameters for the learning algorithms.

This study also has some limitations. First, final models were built only using bitcoin closing prices, the economic variables collected for the study showed to be little significant or almost irrelevant, leading to the creation of noise and higher prediction errors. Second, due to the time and scope limitations of this master thesis, we did not take into consideration the analysis of the public sentiment toward cryptocurrency investing. Third, the discussion of the deep learning models included in the review were measured in different time periods, which could lead to limited analysis. Models based on minute and hourly data frequency were not included in the comparison, because they tended to obtain lower forecast errors due to the higher forecasting frequency.

Bitcoin price prediction research can be further improved. First, studies should use digital filters to smooth data volatility, and technical analysis, combined with hybrid models that use cryptocurrency prices, sentiment analysis, and public opinion. Second, future studies should search for economic variables with high explanatory value over cryptocurrency prices. Third, transfer learning is a rising research problem in machine learning that should be considered to predict newer cryptocurrency prices with less temporal data available. Fourth, automated hyper-parameter tuning should be considered to find the best set of hyper-parameters. Lastly, future studies should contemplate the volatility and returns of bitcoin prices.

References

Altan, A., Karasu, S., & Bekiros, S. (2019). Digital currency forecasting with chaotic meta-heuristic bio-inspired signal processing techniques. *Chaos, Solitons and Fractals*, *126*, 325–336. https://doi.org/10.1016/j.chaos.2019.07.011

Chiu, J., & Keister, T. (2022). The Economics of Digital Currencies: Progress and Open Questions. *Journal of Economic Dynamics and Control*, 104496. https://doi.org/10.1016/j.jedc.2022.104496

Chollet, F. (2021). *Deep learning with python* (Second). Manning Publications.

Coinmarketcap. (2022). *Cryptocurrency Prices, Charts And Market Capitalizations*. CoinMarketCap. https://coinmarketcap.com/

De Oliveira Monteiro, A. H., De Souza, A. D., Batista, B. G., & Zaparoli, M. (2019). *Market prediction in criptocurrency: A systematic literature mapping*. PervasiveHealth: Pervasive Computing Technologies for Healthcare. Scopus. https://doi.org/10.1145/3330204.3330272

Derbentsev, V., Babenko, V., Khrustalev, K., Obruch, H., & Khrustalova, S. (2021). Comparative Performance of Machine Learning Ensemble Algorithms for Forecasting Cryptocurrency Prices. *International journal of engineering*, *34*(1), 140–148. https://doi.org/10.5829/ije.2021.34.01a.16

Ferdiansyah, Othman, S. H., Radzi, R. Z. R. M., Stiawan, D., Sazaki, Y., & Ependi, U. (2019). A LSTM-Method for Bitcoin Price Prediction: A Case Study Yahoo Finance Stock Market. *ICECOS 2019 - 3rd International Conference on Electrical Engineering and Computer Science, Proceeding*, 206–210. https://doi.org/10.1109/ICECOS47637.2019.8984499

Géron, A. (2019). *Hands-On Machine Learning with Scikit-Learn, Keras and TensorFlow* (Second). O'Reilly Media.

Giudici, G., Milne, A., & Vinogradov, D. (2020). Cryptocurrencies: Market analysis and perspectives. *Journal of Industrial and Business Economics*, *47*(1), 1–18. https://doi.org/10.1007/s40812-019-00138-6

Gradojevic, N., Kukolj, D., Adcock, R., & Djakovic, V. (2021). Forecasting Bitcoin with technical analysis: A not-so-random forest? *International Journal of Forecasting*. https://doi.org/10.1016/j.ijforecast.2021.08.001

Graves, A., & Schmidhuber, J. (2005). Framewise phoneme classification with bidirectional LSTM and other neural network architectures. *International Joint Conference on Neural Networks 2005*, *18*(5), 602–610. https://doi.org/10.1016/j.neunet.2005.06.042

Hochreiter, S., & Schmidhuber, J. (1997). Long Short-term Memory. *Neural computation*, *9*, 1735–1780. https://doi.org/10.1162/neco.1997.9.8.1735

Jay, P., Kalariya, V., Parmar, P., Tanwar, S., Kumar, N., & Alazab, M. (2020). Stochastic Neural Networks for Cryptocurrency Price Prediction. *IEEE Access*, *8*(1), 82804–82818.

Kingma, D. P., & Ba, J. (2014). Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Klein, T., Pham Thu, H., & Walther, T. (2018). Bitcoin is not the New Gold – A comparison of volatility, correlation, and portfolio performance. *International Review of Financial Analysis*, *59*, 105–116. https://doi.org/10.1016/j.irfa.2018.07.010

Korstanje, J. (2021). *Advanced Forecasting with Python* (First). Apress.

Kristjanpoller, W., & Minutolo, M. C. (2018). A hybrid volatility forecasting framework integrating GARCH, artificial neural network, technical analysis and principal components analysis. *Expert Systems with Applications*, *109*, 1–11.

Lahmiri, S., & Bekiros, S. (2019). Cryptocurrency forecasting with deep learning chaotic neural networks. *Chaos, Solitons & Fractals*, *118*, 35–40.

Lamothe-Fernandez, P., Alaminos, D., Lamothe-Lopez, P., & Fernandez-Gamez, M. A. (2020). Deep Learning Methods for Modeling Bitcoin Price. *Mathematics*, *8*(8). https://doi.org/10.3390/math8081245

Liu, M., Li, G., Li, J., Zhu, X., & Yao, Y. (2021). Forecasting the price of Bitcoin using deep learning. *Finance research letters*, *40*. https://doi.org/10.1016/j.frl.2020.101755

Livieris, I. E., Kiriakidou, N., Stavroyiannis, S., & Pintelas, P. (2021). An Advanced CNN-LSTM Model for Cryptocurrency Forecasting. *Electronics*, *10*(3). https://doi.org/10.3390/electronics10030287

Mallqui, D. C. A., & Fernandes, R. A. S. (2019). Predicting the direction, maximum, minimum and closing prices of daily Bitcoin exchange rate using machine learning techniques. *Applied Soft Computing Journal*, *75*, 596–606. https://doi.org/10.1016/j.asoc.2018.11.038

Munim, Z. H., Shakil, M. H., & Alon, I. (2019). Next-Day Bitcoin Price Forecast. *Journal of Risk and Financial Management*, *12*(2), 103. http://dx.doi.org/10.3390/jrfm12020103

Panagiotidis, T., Stengos, T., & Vravosinos, O. (2018). On the determinants of bitcoin returns: A LASSO approach. *Finance Research Letters*, *27*, 235–240. https://doi.org/10.1016/j.frl.2018.03.016

Patel, M. M., Tanwar, S., Gupta, R., & Kumar, N. (2020). A Deep Learning-based Cryptocurrency Price Prediction Scheme for Financial Institutions. *Journal of Information Security and Applications*, *55*. https://doi.org/10.1016/j.jisa.2020.102583

Phaladisailoed, T., & Numnonda, T. (2018). Machine learning models comparison for bitcoin price prediction. *Proceedings of 2018 10th International Conference on Information Technology and Electrical Engineering: Smart Technology for Better Society, ICITEE 2018*, 506–511. https://doi.org/10.1109/ICITEED.2018.8534911

Radityo, A., Munajat, Q., & Budi, I. (2018). Prediction of Bitcoin exchange rate to American dollar using artificial neural network methods. *2017 International Conference on Advanced Computer*

Science and Information Systems, ICACSIS 2017, *2018-Janua*, 433–437. https://doi.org/10.1109/ICACSIS.2017.8355070

Rizwan, M., Narejo, S., & Javed, M. (2019). Bitcoin price prediction using Deep Learning Algorithm. *2019 13th international conference on mathematics, actuarlal science, computer science and statistics (MACS-13)*.

Saad, M., Choi, J., Nyang, D., Kim, J., & Mohaisen, A. (2020). Toward characterizing blockchain-based cryptocurrencies for highly accurate predictions. *IEEE Systems Journal*, *14*(1), 321–332. https://doi.org/10.1109/JSYST.2019.2927707

Savitzky, A., & Golay, M. J. E. (1964). Smoothing and Differentiation of Data by Simplified Least Squares Procedures. *Analytical Chemistry*, *36*(8), 1627–1639. https://doi.org/10.1021/ac60214a047

Septiarini, T. W., Taufik, M. R., Afif, M., & Masyrifah, A. R. (2020). A comparative study for Bitcoin cryptocurrency forecasting in period 2017-2019. *Journal of Physics: Conference Series*, *1511*(1), 12056.

Tan, X., & Kashef, R. (2019). Predicting the closing price of cryptocurrencies: A comparative study. *ACM International Conference Proceeding Series*. https://doi.org/10.1145/3368691.3368728

Wang, Y., & Chen, R. (2020). Cryptocurrency price prediction based on multiple market sentiment. *Proceedings of the Annual Hawaii International Conference on System Sciences*, *2020-Janua*, 1092–1100.

Zoumpekas, T., Houstis, E., & Vavalis, M. (2020). ETH analysis and predictions utilizing deep learning. *Expert Systems with Applications*, *162*, N.PAG-N.PAG.

Appendices

Appendix A - Systematic literature review methodology

Search strategy

We used Prisma tool methodology to facilitate a systematic literature search. We searched the following databases: Scopus, EBSCO, Web of Science and ProQuest Central covering the period since January 2015 until October 2021.

The search terms were defined based on an iterative analysis of previous studies. And the search strategy used the following keywords:

("Forecasting") AND ("Cryptocurrency") AND ("LSTM" OR "Long-short-term memory" OR "Neural networks" OR "Deep Learning" OR "Machine learning").

The search resulted in 246 published articles (scopus:140, EBSCO:41, web of science:42, proquest:23). Publications from all databases were combined and duplicates removed, resulting in 165 articles. Figure 6.1 shows the PRISMA flow diagram of the literature search.

Inclusion and exclusion criteria

An initial inclusion and exclusion were applied. All articles written in a language other than English, Portuguese or Spanish were removed, resulting in 158 articles. In the next step, abstract conferences, and articles unrelated with the research topic were removed. In addition, 18 articles were removed due to unavailability resulting in a total of 116 publications.

Due to the high number of articles, a further inclusion and exclusion criteria was applied based on De Oliveira Monteiro et al. (2019) publication described in detail in Table 6.1 and 6.2, respectively, leaving 72 publications. Finally, in a last step, We applied additional selection criteria due to the high volume of articles available. Only publications that specified the data source, data type, algorithms, validation criteria and results were selected. Publications that only use classification methods and focus on predicting the upward or downward trend in the price of cryptocurrencies were excluded.

The literature search resulted in 20 publications, which are shown in table 6.3.

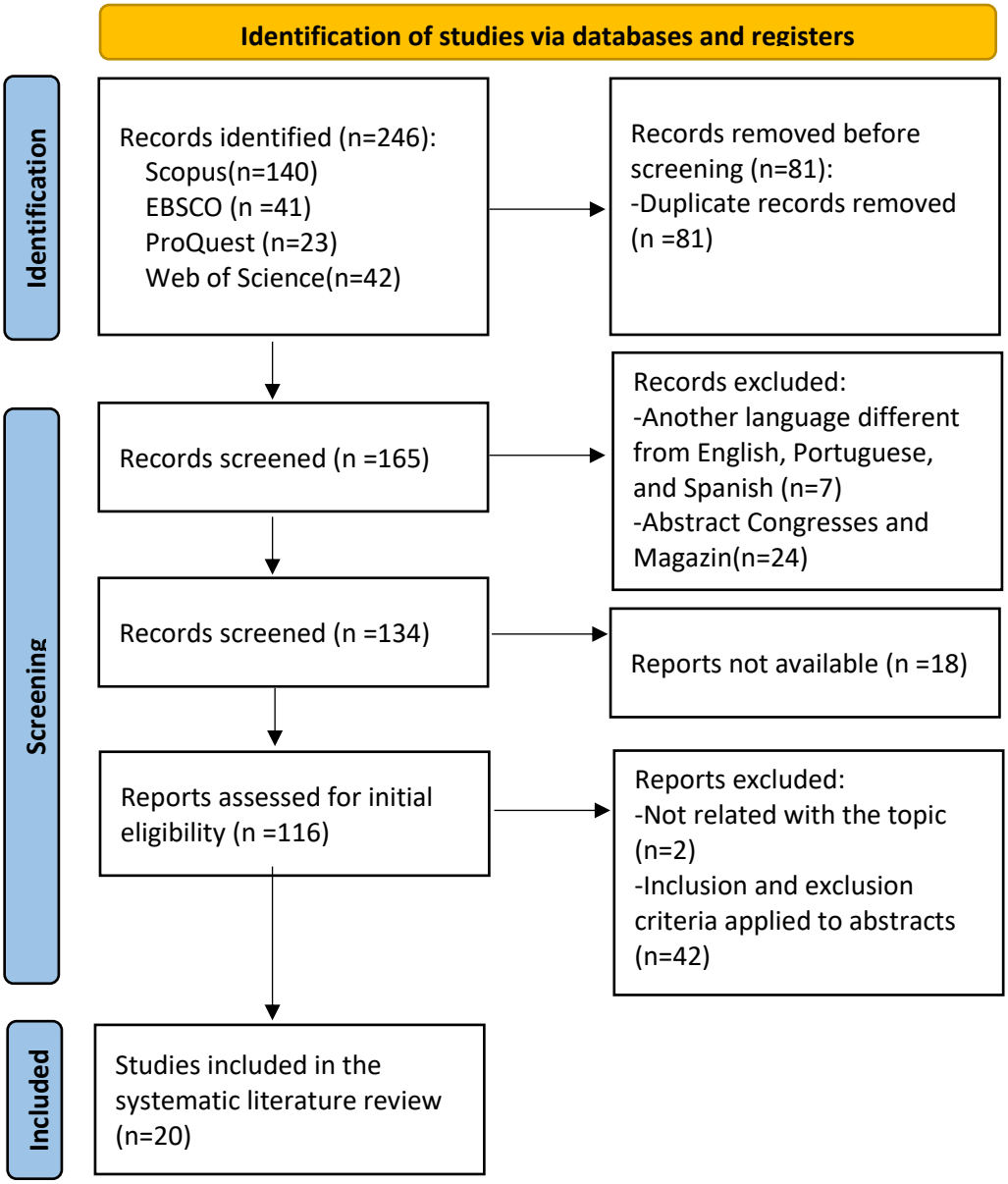

Figure 6.1:PRISMA Flow Diagram (2020): Diagram of the Search and screening Process

Table 6.1: Inclusion criteria

Identifier	Description
CI-01	Publications mentioning cryptocurrencies price analysis and forecasting can be selected
CI-02	Publications mentioning cryptocurrency price forecasting algorithms, techniques and methods can be selected.
CI-03	Publications that mention any type of analysis or factor that might influence the price of cryptocurrencies can be selected.

Table 6.2: Exclusion Criteria

Identifier	Description
CE-01	Publications where keywords are not included in the title, abstract, keywords or body text cannot be selected.
CE-02	Publications that only mention the terms used in the search cannot be selected.
CE-03	Publications where cryptocurrencies are applied to areas other than price forecasts cannot be selected.
CE-04	Publications that only present or describe cryptocurrencies, blockchain or derived technologies cannot be selected.

Table 6.3: Articles included in the systematic literature review

Nº	Author (year)	Title	Criteria
1	Septiarini et al. (2020)	A comparative study for Bitcoin cryptocurrency forecasting in period 2017-2019	CI1
2	Patel et al. (2020)	A Deep Learning-based Cryptocurrency Price Prediction Scheme for Financial Institutions	CI2
3	Kristjanpoller and Minutolo (2018)	A hybrid volatility forecasting framework integrating GARCH, artificial neural network, technical analysis and principal components analysis.	CI2
4	Ferdiansyah et al. (2019)	A LSTM-Method for Bitcoin Price Prediction: A Case Study Yahoo Finance Stock Market	CI2
5	Livieris et al. (2021)	An Advanced CNN-LSTM Model for Cryptocurrency Forecasting	CI2
6	Rizwan et al. (2019)	Bitcoin price prediction using Deep Learning Algorithm	CI2
7	Derbentsev et al. (2021)	Comparative Performance of Machine Learning Ensemble Algorithms for Forecasting Cryptocurrency Prices	CI2
8	Lahmiri and Bekiros (2019)	Cryptocurrency forecasting with deep learning chaotic neural networks.	CI2
9	Wang and Chen (2020)	Cryptocurrency price prediction based on multiple market sentiment	CI1
10	Lamothe-Fernandez et al. (2020)	Deep Learning Methods for Modelling Bitcoin Price	CI2
11	Altan et al. (2019)	Digital currency forecasting with chaotic meta-heuristic bio-inspired signal processing techniques	CI2
12	Zoumpekas et al. (2020)	ETH analysis and predictions utilizing deep learning	CI2
13	Phaladisailoed and Numnonda (2018)	Machine learning models comparison for bitcoin price prediction	CI2
14	Munim et al. (2019)	Next-Day Bitcoin Price Forecast	CI1
15	Liu et al. (2021)	Forecasting the price of Bitcoin using deep learning	CI3
16	Tan and Kashef (2019)	Predicting the closing price of cryptocurrencies: A comparative study	CI2
17	Mallqui and Fernandes (2019)	Predicting the direction, maximum, minimum, and closing prices of daily Bitcoin exchange rate using machine learning techniques	CI2
18	Radityo et al. (2018)	Prediction of Bitcoin exchange rate to American dollar using artificial neural network methods	CI2

19	Jay et al. (2020)	Stochastic Neural Networks for Cryptocurrency Price Prediction	CI2
20	Saad et al. (2020)	Toward characterizing blockchain-based cryptocurrencies for highly accurate predictions	CI3

Information extraction

After the selection of final publications to be included in the literature review, a complete reading of each article was performed and the following information was extracted: title, publication year, authors, data source, data period, training and test split strategy, modelling methods, model quality metrics and results. Table 6.4 shows the summarised information of the publications selected for systematic literature review.

Table 6.4: Information summary of the studies included in the review

№	Author (year)	Data Source	Data	Data Period	Train test split strategy	Modelling Methods	Model quality metrics	Results (Best model)
1	(Septiarini et al., 2020)	coinmarketcap.com	Bitcoin	5/1/2017-1/10/2019	Train-75% Test-25%	ANFIS, FTS, ES, ARIMA	RMSE, MSE	RMSE-98.74(ES)
2	(Patel et al., 2020)	Investing.com	Litecoin, Monero	Litecoi- 24/04/2016 - 23/02/2020 Monero- 30/01/2015 - 23/02/2020	Unknown	LSTM, LSTM - GRU	MSE, RMSE, MAE, MAPE	MAPE Litcoin-2.0581%(LSTM-GRU) Monero-4.0727%(LSTM-GRU)
3	(Kristjanpoller & Minutolo, 2018)	coinmarketcap.com	Bitcoin	13/09/2011 - 26/08/2017	Unknown	GARCH, EGARCH, APGARCH	MSE	MAPE-1.64% (EGARCH)
4	(Ferdiansyah et al., 2019)	finance.yahoo.com	Bitcoin	27/06/2014 - 27/06/2019	Train-80% Test-20%	LSTM	RMSE	RMSE-288.59866(LSTM)
5	(Livieris et al., 2021)	finance.yahoo.com	Bitcoin, Etherium, Ripple	01/01/2017 - 31/10/2020	Train-82% Validation- 7% Test- 11%	CNN-LSTM	MAE, RMSE, R2	R2 Bitcoin- 0.953(CNN-LSTM) Ethereum- 0.964(CNN-LSTM) Ripple-0.962(CNN-LSTM)
6	(Rizwan et al., 2019)	Unknown	Bitcoin	19/08/2013 - 19/07/2016	Train-70% Test-30%	LSTM, GRU	R2, MSE	R2-0.992(GRU)
7	(Derbentsev et al., 2021)	finance.yahoo.com	Bitcoin, Ethereum, Ripple	01/01/2015 - 31/12/2019	Train-80% Test-20%	RF, SGBM	RMSE, MAPE	MAPE - BTC -2.31% (SGBM) ETH- 2.26%(RF) XRP -0.92%(SGBM)
8	(Lahmiri & Bekiros, 2019)	Unknown	Bitcoin, Digital Cash, Ripple	Bitcoin-16/07/2010 to 01/10/2018 Digital Cash-21/01/2015 to 01/10/2018 Ripple-08/02/2010 to 01/10/2018	Train-90% Test-10%	GRNNs, DLNN	RMSE	RMSE Bitcoin-2750 (DLNN) Digital Cash-19.2926 (DLNN) Ripple- 0.0499(DLNN)
9	(Wang & Chen, 2020)	binance.com huobi.com Forums	Bitcoin, Ethereum, ether, Ripple, litecoin	User reviews- 1/10/2018 - 31/12/218 Bitcoin/Ethereum/Tether/Ripple/Litecoin-1/1/2019-31/3/2019	Train- Fiat Validation- contract transaction Test- contract transaction	LSTM,CNN, SVM, BPNN, RBF	MAPE, MAE, RMSE	MAE- Bitcoin- (LSTM + sentiment) Ethereum - (LSTM + sentiment) Tether - (LSTM + sentiment) Ripple - (LSTM + sentiment) litecoin - (LSTM + sentiment)
10	(Lamothe-Fernandez et al., 2020)	lockchain.info International Financial Statistics, World Bank, FRED Sant Louis, Google Trends and Quandl	Bitcoin	2011-2019	Train-70% Validation- 10% Test- 20%	DRCNN, DNDT, DSVR	RMSE	MAPE - Bitcoin- 0.52%(DRCNN)
11	(Altan et al., 2019)	Unknown	Bitcoin, Ripple, Dash, Litcoin	Bitcoin- 18/07/2010-28/03/2019 Ripple-22/01/2015-28/03/2019 Dash-14/02/2014-28/03/2019 Litcoin-24/08/2016-28/03/2019	Train-85% Test-15%	LSTM, EWT-LSTM, EWT-LSTM-CS	MAPE, MAE, RMSE	MAPE- Bitcoin-3.55% (EWT-LSTM-CS) Ripple-1.72% (EWT-LSTM-CS) Dash-1.47% (EWT-LSTM-CS) Litcoin-2.77% (EWT-LSTM-CS)
12	(Zoumpekas et al., 2020)	poloniex.com coinmarketcap.com	Ethereum	08/08/2015 - 28/05/2018	Train-53% Validation-27% Test-20%	CNN, LSTM, sLSTM, BiLSTM, GRU	RMSE, MAE	RMSE-0.92(LSTM)
13	(Phaladisailoed & Numnonda, 2018)	bitstamp.net	Bitcoin	01/01/2012 - 01/01/2018	Train-70% Test-30%	Theil-Sen Regression, Huber Regression, LSTM, GRU	MSE, R2	R2-0.992(LSTM / GRU)
14	(Ziaul Haque Munim et al., 2019)	data.nasdaq.com	Bitcoin	01/01/2012 - 04/10/2018	Train-20% Test-80%	ARIMA, NNAR	RMSE, MAPE, MASE	MAPE- 3.65% (ARIMA)
15	(Liu et al., 2021)	coindesk.com	Bitcoin	01/07/2013 - 31/12/2019	Train-80% Test-20%	BPNN, PCA-SVR, SVR, SDAE	MAPE, RMSE, DA	MAPE-0.1019(SDAE)
16	(Tan & Kashef, 2019)	coinmarketcap.com	Bitcoin	28/04/2013 - 11/05/2018	Train-100% Test- out of sample	BR, AR, ARIMA, LSTM, SVM	ME, RMSE, MAE, MPE, MAPE	RMSE- 33.70 (LSTM)
17	(Mallqui & Fernandes, 2019)	bitcoincharts.com quandl.com investing.com	Bitcoin	1/04/2013-01/04/2017	Train-75% Test-25%	ANN, SVM, RNN	MAE, MAPE, RMSE	MAPE-1.81%(SVM)
18	(Radityo et al., 2018)	cryptocompare.com	Bitcoin	01/01/2014-02/04/2017	Train-80% Test-20%	BPNN, GANN, GABPNN, NEAT	MAPE	MAPE-1.998 ± 0.038 %(BPNN)
19	(Jay et al., 2020)	bitinfocharts.com	Bitcoin Litecoin Etherium	2017-2019	Train-75% Test-25%	MLP, LSTM	MAPE, MAE, RMSE, MSE	MAPE- Bitcoin - 2.5589%(MLP) Litecoin -2.3886%(MLP) Etherium -2.3405%(MLP)
20	(Saad et al., 2020)	blockchain.com/api etherscan.io	Bitcoin Etherium	April 2016-May2018	Train-85% Test-15%	LR, GB, RF, LSTM	RMSE, MAE	RMSE- Bitcoin-0.0175(LSTM) Etherium-0.0718(LSTM)