Università
Ca'Foscari
Venezia

Master's Degree

In Economics and Finance

Curriculum Finance

Final Thesis

Cryptocurrencies Price Prediction using LSTM Neural Network model

Supervisor
Ch. Prof. Marco Corazza

Graduand
Giacomo Minotti
Matriculation Number 882161

Academic Year
2021 / 2022

Alla mia famiglia ed in particolare ai miei genitori,

i quali mi hanno sostenuto e consigliato in ogni mia scelta.

CONTENTS

ABSTRACT

Cryptocurrencies are rising popularity day by day and attracting more and more attention from both academia and financial industry. Nowadays, digital currencies provide an alternative exchange currency method to the well-known traditional one. Since their use in financial applications is constantly growing, investors are becoming increasingly attracted to this promising type of investment. However, due to the complexity of the temporal dynamics of digital assets, predictions remain difficult to perform. The cryptocurrency market is characterized by high volatility and sharp price swings over a short period of time; hence the development of effective and reliable price forecasting models turns out to be extremely important for financial investors to take accurate decisions. These problems can be overcome by predicting cryptocurrency prices through a machine learning technique. As contribution, this thesis provides an empirical study on applying Long Short-Term Memory (LSTM) model to predict five major cryptocurrencies that are: Bitcoin (BTC), Ethereum (ETH), Cardano (ADA), Solana (SOL) and Polkadot (DOT). The study starts from the data collection, needed for the data analysis process, to the LSTM model evaluation. The accuracy of the model performance is evaluated in terms of Root Mean Square Error (RMSE), Mean Square Error (MSE), Mean Absolute Error and R-Squared (R2). These parameters are computed for all five cryptocurrencies to determine in which of these, LSTM model is the best fit to predict accurate prices for the future.

INTRODUCTION

The rapid development of cryptocurrencies during the last decade is one of the most disruptive and controversial innovations in the modern global economy. The digital asset's market has expanded at an unprecedented rate in its short lifetime, passing from the introduction of Bitcoin, in January 2009, to more than 20,000 cryptocurrencies until today. However, this constantly increasing financial market is characterized by high volatility and significant price fluctuations over time. These aspects with the addition of a lack of legislative regulation of their transaction in most countries led to a large increase of significant risks involved with investing in digital assets. This has generated contentious debates regarding their position and function in the modern economy. Hence, the challenge of creating suitable techniques and models for forecasting prices of cryptocurrencies is relevant for the scientific community as well as for investors and traders. Nowadays, digital assets forecasting is considered as one of the most arduous time-series prediction problems because of the numerous unpredictable factors involved and the high volatility of cryptocurrencies' prices, which leads to complex temporal dependencies[1].

Several research teams have been working to develop adequate models which can forecast asset and cryptocurrency prices. In the last decade, machine learning models have been used in this field since several ML techniques have had considerable empirical support for problems concerning non-linearity and noisy environments. Due to the complexity of the task, deep learning presents an intriguing technical solution based on its performance in related fields. In recent year, DL approaches were used on time-series predictions, with an emphasis on real-world application areas like digital assets. Most of these techniques make use of advanced DL models and particular architectural layouts based on Recurrent neural network (RNN) and Long Short-Term

[1] V. DERBENTSEV, A. MATVIYCHUK, V.N. SOLOVIEV, *Forecasting of Cryptocurrency Prices Using Machine Learning,* In Advanced Studies of Financial Technologies and Cryptocurrency Markets, Springer, Berlin/Heidelberg, Germany, pp. 211–231, 2020.

Memory (LSTM) methods, which were been preferred over others ML approaches because of the temporal nature of cryptocurrency data[2]. Although many related studies have been conducted in more traditional fields, such as stock market, there are still few research on deep learning in cryptocurrency market predictions.

The general object of this thesis is to explore this research gap and to determine how well LSTM models are suited in prediction tasks in the domain of cryptocurrencies. The sample of five digital assets is primarily selected on the total market capitalization, per September 01, 2022 (https://coinmarketcap.com/), to obtain an adequate overview of the whole cryptocurrency market and further investigate any potential variations in forecast accuracy. More in detail, the purpose of this research is two-fold. First, to build a LSTM model to predict the cryptocurrency price of the five digital assets selected and further investigate the forecast accuracy through different performance methods. Secondly, to determine for which digital asset LSTM model has the best fit.

The structure of the thesis is the following. The first chapter is focused on understanding the phenomenon of digital assets, starting from describing the historical path that led cryptocurrencies to the achievement of the notoriety they enjoy today to their structural characteristics; subsequently it is examined the technology that produced these assets, the blockchain. Furthermore, it provides an overview of the multiple factors which affect the cryptocurrencies' price and the different methodological approaches to forecast it. In the second chapter, it is conducted a top-down description of the theory needed for the comprehension of the empirical part of the study, laying the background and fundamentals for the reader. Starting from the Machine Learning technology, the research continues with the illustration of several Depp Learning methods, such as Artificial Neural Network and Recurrent Neural Network, up to the deep dive into the Long Short-Term model. The third chapter outlines the research's architecture which is used in this thesis. On the one hand, an in-depth description of the cryptocurrencies selected for the study is provided. Then, it is explained in detail how the various input data is collected and how it is cleaned,

[2] W. LU, J. LI, J. WANG, L. QUIN, *A CNN-BiLSTM-AM method for stock price prediction*, Neural Comput. Appl, pp.1-13, 2020.

pre-processed, and splitted into training and testing parts to be useable by the LSTM model. Additionally, it is presented a section of data analysis, where it is shown and explored the behaviour and distribution on data and the relationship between digital assets. Chapter 4 starts with an illustration of the different languages, libraries, and tools which are used to develop the system. It continues presenting the LSTM model built and describing hyperparameters of the neural networks. Furthermore, the fourth section outlines the evaluation results of the LSTM model using several performance parameters, such as Root Mean Squared Error (RMSE), Mean Squared Error (MSE), Mean Absolute Error (MAE) and R Square Score (R^2), and discusses broadly about the observations. In order to have a fair comprehension of the goodness in fit of the LSTM approach, an ARIMA model is built respecting the same parameters.

Chapter 1
CRYPTOCURRENCIES AND BLOCKCHAIN

1.1 What are cryptocurrencies and how they are born

"*I think that the Internet is going to be one of the major forces for reducing the role of government. The one thing that's missing, but that will soon be developed, is a reliable e-cash, a method whereby on the Internet you can transfer funds from A to B without A knowing B or B knowing A."*[3]

— Milton Friedman, 1999.

It is not so late when in 1999, the recipient of the Nobel Memorial Prize in Economic Sciences in 1976, Milton Friedman, presents the principles behind what came to life ten years later: the cryptocurrencies. In that period, the unknown identity Satoshi Nakamoto created Bitcoin, what became the first type of a new class of assets called cryptocurrencies.

In general, it can be said that a cryptocurrency is "a digital or virtual currency used for the exchange and transfer of goods and services digitally, based on cryptography and peer to peer technologies"[4]. It uses cryptography to make sure the transfer and exchange of digital assets over the network is secure, to control and regulate the addition of cryptocurrency in circulation[5]. Therefore, the term cryptocurrency.

However, the success of Bitcoin, before becoming the first example of a digital asset, hides numerous failed attempts. Conceptually, the idea of cryptocurrencies dates

---

[3] M. FRIEDMAN, *Milton Friedman Full Interview on Anti-Trust and Tech*, National Taxpayers Union, 1999. Available at: https://www.youtube.com/watch?v=mlwxdyLnMXM.

[4] G. COMANDINI, *Da Zero alla Luna, quando come e perché la Blockchain sta cambiando il mondo,* Flaccovio, Palermo, p. 75, 2020.

[5] D. GARCIA, C. J. TESSONE, P. MAVRODIEV, N. PERONY, *The digital traces of bubbles: feedback cycles between socio-economic signals in the bitcoin economy*, Journal of the Royal Society Interface, 2014.

back to 1981-1982, when David Chaum[6] published the "*Untraceable Electronic Mail, Return Addresses, and Digital Pseudonyms*" and "*Blind Signatures for Untraceable Payments*"[7] papers[8].

In his works, first, he describes a method to hide the content of the email texts and the identity of the users through cryptography. This before that the message is signed, so that the signer cannot determine the content. A second correspondent can answer using an untraceable return address while a first correspondent can maintain their anonymity. No single universally trusted authority is necessary for the procedure. Then, he introduces a new cryptographic system, based on blind signatures, which can be publicly verified just like a regular digital signature. This digital approach for cash transfer is proposed in a way that makes impossible for third parties to influence payments, time and amounts of transactions held by an individual. It also aimed to attribute to users the ability to prove payments or to determine the beneficiary's identity in particular circumstances[9].

The idea of creating a suitable mechanism for exchange resources in a digital form and in full autonomy has been accepted and implemented by Cypherpunks, a community mainly made up of young programmers dating back to the late eighties. It is identified as a movement of "crypto-anarchists", libertarian activists, operative in the construction of anonymous computer systems thanks to the use of cryptography, so that the transfer of information and money remains confidential, and it doesn't need any intervention of a financial institution that acts as an intermediary[10]. This new concept of forming a digital payment system based on cryptography is the main reason why so many programmers decided to join this movement. This push the form of an

---

[6] *David Chaum is a computer scientist and cryptographer born in 1955 in the United States. He is considered a pioneer in cryptography and privacy-preserving technologies. Since he has an extensive career as a professor, researcher and developer of cryptographic protocols, he is titled as the "father of digital money".*

[7] D. CHAUM, *Untraceable Electronic Mail, Return Addresses, and Digital Pseudonyms*, Department of Computer Science, University of California, Santa Barbara, 1981. Available at: http://www.lix.polyt-echnique.fr/~tomc/P2P/Papers/Theory/MIXes.pdf.

[8] D. CHAUM, *Blind Signatures for Untraceable Payments*, Department of Computer Science, University of California, Santa Barbara, 1982. Available at: https://sceweb.sce.uhcl.edu/yang/teaching/csci52-34WebSecurityFall2011/Chaum-blind-signatures.PDF.

[9] A. JUDMAYER, N. STIFTER, K. KROMBHOLZ, E. WEIPPL, *Blocks and Chains: Introduction to Bitcoin, Cryptocurrencies, and Their Consensus Mechanisms*, Synth. Lect. Inf. Secur. Privacy, Trust, 2017.

[10] E. HUGHES, *A Cypherpunk's Manifesto*, March 1993. Available at: https://www.activism.net/cyphe-rpunk/manifesto.html.

ever-wider community, in which is possible to communicate anonymously and not in a traceable way. Cypherpunks are the main source of incentive for the birth of the first innovative form of payment.

In the following years, a trial-and-error process gives birth to one of the greatest technological inventions of the 21st century: Bitcoin. It can be said, the creation of the first cryptocurrency is not the result of the genius of a single person, but of that process, which lasted more than a decade. The intention of being able to exchange resources, without the intervention of any intermediary, is developed so radically to prompt many Cypherpunkers to experiment multiple technological processes until reaching a solution that is able to balance efficiency, safety and decentralization.

Hence, in October 2008, Satoshi Nakamoto released the Bitcoin whitepaper, a document entitled "Bitcoin A peer-to-peer Electronic Cash System"[11]. In January 2009, he introduced the blockchain technology, through the creation of the first block of the bitcoin blockchain, the so-called Genesis Block, within the latter there is the inscription "The Times 03 / Jan / 2009 Chancellor on brink of second bailout for banks"[12], which represents the title of the article in "The Times" on the date of the first transaction through the Bitcoin protocol.

Nowadays, bitcoin is the most successful cryptocurrency in terms of market capitalisation above the 20,000 altcoins[13] that circulated in the world. Some are similar to Bitcoin, while others are based on different technologies, or have new features that allow them to do more than transfer value.

The foundation technology behind the functioning of cryptocurrencies is blockchain, often defined as a “distributed, decentralized, public ledger”[14] where to store the full transaction history of each user. These digital currencies are framed

---

[11] G. COMANDINI, *Da Zero alla Luna, quando come e perché la Blockchain sta cambiando il mondo*, Flaccovio, Palermo, p. 3, 2020.

[12] F. ELLIOT, G. DUNCAN, *Chancellor on brink of second bailout for banks*, The Times, January 2009.

[13] *Altcoin is the term used to describe a coin or token that is not Bitcoin, hence alternative digital assets. This terminology is based on the notion that Bitcoin is the original cryptocurrency and that all others are afterwards referred to as “alternate” or “alternative” coins.* Available at: https://academy.binance.com/en/glossary/altcoin.

[14] A. HAYES, *Blockchain Facts: What Is It, How It Works, and How It Can Be Used*, June 24, 2022. Retrieved from Investopedia: https://www.investopedia.com/terms/b/blockchain.asp.

within a decentralized, peer-to-peer[15]system that organizes and records transactions through a node distribution. Each node in the blockchain network has the possibility to take part in the transaction process. However, there are few exceptions of cryptocurrencies which work on complete anonymity. But, while their systems hide the transaction history, they still prove the existence of every transaction through a cryptographic proof.

Cryptocurrencies inherit their characteristics directly from the blockchain on which they are developed, but they all share some basic peculiarities. A good digital currency is established on the principle of decentralization. So, the rules of the cryptocurrency cannot be changed by any central bank or subset of users without reaching consensus.

It's managed by peer-to-peer networks participants running free, open-source software which connect them to other members. Generally, anyone who wants to take part in the network is free to join and share information between other nodes[16]. Hence, thanks to decentralization, cryptocurrencies are extremely resistant to censorship or shutdown.

Since each node has a copy of the database, it affectively acts as its own server. When a new set of transactions, a block, is added to the blockchain, it gets broadcast from node to node so that each one can update its own database in the same way. Even if a node goes offline, its peers will still be able to obtain information from other nodes. This feature allows cryptocurrencies to be used 24 hours a day, 365 days a year, allowing the transfer of value globally and without the need of intermediaries.

Being permissionless, without the presence of a "*middleman*" or the control by any government or central authority, permits anyone with an Internet connection to transfer funds whenever wants, nearly-instantly and for low fees[17].

---

[15] *Peer-to-peer (P2P) is a distributed networking or computing architecture that divides tasks or workloads across several computer systems (each one acting as an individual peer). P2P networks can be used to share any kind of digital data, as cryptocurrencies.* Available at: https://academy.binance.com/en/glossary/peer-to-peer.

[16] *Taking Bitcoin as an example, each computer running the Bitcoin client is referred to as a node. Hence the Bitcoin network is made up of thousands of computer nodes spread around the world, and this is what makes Bitcoin a peer-to-peer, distributed economic system.* Available at: https://academy.binance.com/en/glossary/node

[17] A. NARAYANAN, J. BONNEAU, E. FELTEN, A. MILLER, S. GOLDFEDER, *Bitcoin and cryptocurrency technologies: a comprehensive introduction*, Princeton University Press, 2016.

Another feature of cryptocurrencies, that is linked to those already mentioned, is the trustlessness, for which it is unnecessary to establish any relationship of trust with third parties or central institutions to carry out transactions or, again, trust the entity of a good or service[18].

All these properties, that have been previously described, are obtained through the underlying blockchain technology. Moreover, the blockchain of a specific cryptocurrency is similar to a bank's balance sheet, hence it acts as a digital ledger which store the full transaction history of every user. This results in a high grade of transparency, since every transaction is published publicly, without exception. This feature does not undermine the level of privacy, which is in any case maintained through pseudonymity, for which the transactions are required by indicating the addresses of the parties involved, who become identifiable.

The use of blockchain also permits an elevated grade of security, which constantly checked and verified by a huge amount of computing power, called *hash power*.

1.2 Blockchain technology

At the base of all cryptocurrencies there is a technology able to improve the world known, to destabilize what are considered the social economic pillars of modern society, just like did electricity in its applications before, and electronics, with the computer and the internet, then. A technology capable of leading humanity towards what will be remembered as the Fourth Industrial Revolution, together with Artificial Intelligence (AI), cloud computing and the Internet of Things (IoT): the blockchain[19].

Having a basic understanding of this technology can help to figure out why it is considered so revolutionary. For this purpose, it is essential to introduce the blockchain from its definition, a necessary step to undertake this study[20].

---

[18] F. TSCHORSCH, B. SCHEUERMANN, *Bitcoin and beyond: A technical survey on decentralized digital currencies*, IEEE Communications Surveys & Tutorials, vol. 18, no. 3, pp. 2084–2123, 2016.

[19] K. SCHWAB, *The Fourth Industrial Revolution*, World Economic Forum, Cologny, p. 143, 2016. Available at: https://law.unimelb.edu.au/__data/assets/pdf_file/0005/3385454/SchwabThe_Fourth-_Industrial_Re-volution_Klaus_S.pdf

[20] A. ROSIC, *What is Blockchain Technology? A Step-by-Step Guide For Beginners*, sez. Guides, Blockgeeks, 2017. Available at: https://blockgeeks.com/guides/what-is-blockchain-technology/

“*The blockchain is a distributed ledger according to a peer-to-peer system that is cryptographically secure, immutable (or in any case extremely difficult to modify) and upgradable only by consent or agreement between peers*”[21].

This ledger is digital and distributed across the entire network of computer systems on the blockchain and it is structured as a chain of registers[22], called blocks[23]. The chain can be enriched with new blocks containing data only at the end of the database, while the previous blocks are not editable or removable[24].

Indeed, at the base of the security and immutability of the system, there are the consensus protocol and the cryptography. These guarantee that the only way to alter the data of each block is to invalidate all the blocks after the altered data. The system provides that for each block generated, the hash value[25] of the previous one is entered in the input, which is useful to produce the hash of the new block.

The hash is done through the use of mathematical formulas known as cryptographic hash functions, which is one way to generate a fixed-size output (the hash value) from an input of variable size. All this means that any change of the input will result in a changed hash value of the block and all those subsequent to it. Figure 1.1 illustrates the structure of a typical blockchain.

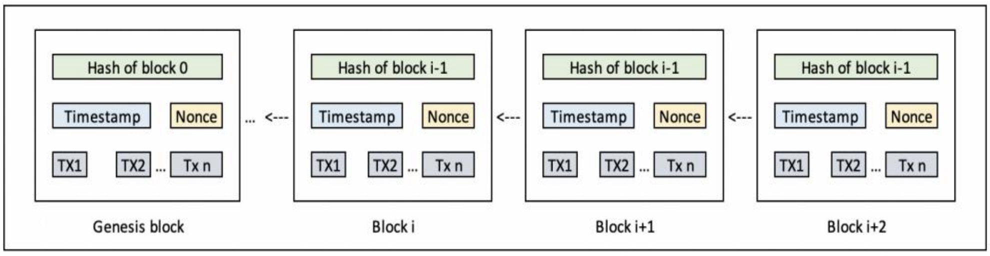

Figure 1.1: Blockchain Technology.

[21] I. BASHIR, *Mastering Blockchain: A deep dive into distributed ledgers, consensus protocols, smart contracts, DApps, cryptocurrencies, Ethereum, and more*, Packt, Birmingham – Mumbai, p. 16, August 2020.

[22] K. GAUTAM, N. SHARMA, P. KUMAR, *Empirical Analysis of Current Cryptocurrencies in Different Aspects*, Noida, India, pp. 344–348, 4–5 June, 2020.

[23] R. ADAMS, B. KEWELL, G. PARRY, *Blockchain for Good? Digital Ledger Technology and Sustainable Development Goals*, World Sustainability Series, Switzerland, 2018.

[24] A. NARAYANAN, J. BONNEAU, E. FELTEN, A. MILLER, S. GOLDFEDER, *Bitcoin and cryptocurrency technologies: a comprehensive introduction*, Princeton University Press, 2016.

[25] *The “hash value” or “checksum” is the output created by the hash function when it turns an input, such as a text, into a string of bytes with a fixed length and structure.* Available at: https://www.bitpanda.com/academy/en/lessons/what-is-a-hash-function-in-a-blockchain-transaction/

Each block is a data collection that records: multiple transactions, that are the actions triggered by the participants, a timestamp, which is the hash value of the previous block, and a nonce[26], which is a random number for hash validation[27]. This system ensures the integrity of the entire blockchain up to the first block, called genesis block.[28] Blocks and transactions are validated in a totally decentralized way, i.e. on each node of the blockchain. If most of the nodes in the network agree on the validity of transactions on a block and the validity of the block itself, then, the latter can be added to the chain.

The agreement between the nodes is evaluated through a consensus mechanism, that is the process in which the majority of the validators come to an accordance on the status of the ledger. The distributed ledger is sent to each user of the network updated on the last transaction.

At this mechanism of processing transactions can be attributed three main features.

- *Safety*. Thanks the use of cryptography and digital signature, the sender of each transaction is verified and the content of each one is secured. In addition to these precautions for users, through the instant distribution of information within each node of the network, it is almost impossible for a hacker to commit a fraud in the system. Indeed, to corrupt a certain information, it is necessary to modify all the instances of the database saved on each node. This kind of attack on the blockchain as well as being very difficult to do, it is not convenient because extremely expensive. Due to this impossibility, the blockchain turns out to be a particularly secure network.
- *Transparency*. Each copy of the ledger, replicated on the nodes, contains all the transactions, which can be freely consulted by all users. This ensures

---

[26] *A nonce refers to an arbitrary number or value that can only be used once. Nonces are often used on authentication protocols and cryptographic hash functions. In the blockchain technology, a nonce refers to a pseudo-random number that is utilized as a counter during the process of mining and it works in combination with hash as a control element to avoid manipulation of the block information.* Available at: https://academy.binance.com/en/glossary/nonce

[27] *"Validation rules" or "Consensus rules" define what is permitted to be included in a block. These rules are absolute and any block or transaction that would violate any of the strict criteria governing block-level validation must be rejected.* Available at: https://reference.cash/protocol/blockchain-/transaction-validation/block-level-validation-rules

[28] R. GARAVAGLIA, *Tutto su Blockchain, Capire la tecnologia e le nuove opportunità*, Hoepli, Milano, 2019, p. 19.

that whoever wish to inquire about the history of a certain datum has full access to its origin and to all evolutions it has undergone up to the present moment. In this context, it is not needed the presence of a guarantor to ensure that such data is true, as it is the blockchain which guarantees its validity. This high level of transparency encourages the degree of trust that participants place in the system, since users do not also have to evaluate the reliability of the intermediary or other participants in the network.

- *Irreversibility*. Once a transaction is completed, added to the chain and the accounts are updated, the blockchain records cannot be edited or deleted. This is because they are linked to every transaction record that came before them. Various computational algorithms and approaches secure that the recording on the database is permanent, chronologically ordered, and available to all others of the network. The only way to correct an operation is to create another one that is the reverse to the first.

The value of the blockchain goes far beyond the already well-known identification as payment infrastructure, monitoring system or digital identity management tool. It can renovate the concept of trust at the application level, giving rise to a paradigm shift within the implementation of the applications themselves and allowing the system to be free to innovate.

1.3 Cryptocurrency price prediction

In the last decade, the rapid development of the blockchain technology and its applications in multiple fields, such as cryptography, computer science and economics, has led to the birth of numerous digital currencies and the creation of cryptocurrency markets. Due to the unique nature of the combination of the encryption technology[29] and currency units[30], cryptocurrencies have been established and widely recognized as a new electronic alternative exchange currency method. In a relative short time, the

---

[29] *The process of converting information into a secret code that conceals its real meaning is known as encryption. Cryptography is the study of encrypting and decrypting information.* Available at: https://www.techtarget.com/searchsecurity/definition/encryption#:~:text=Encryption%20is%20the%20method%20by,encrypted%20data%20is%20called%20ciphertext.

[30] U. Rajput, F. Abbas, H. Oh, *A solution towards eliminating transaction malleability in bitcoin*, Journal of Information Processing Systems, vol. 14, no. 4, pp. 837-850, 2018.

global cryptocurrency market rose to a peak of over $3 trillions on November 10, 2021[31]. On the same day, a single bitcoin, the most capitalized cryptocurrency, has registered a maximum value of $69,275, measuring a growth of over eight million percentage since its birth. As well as Bitcoin's rapid rise in value, it declined in a short time from its peak to the current level of $19,000, perfectly displaying its volatile and high-risk nature. According to a Grayscale's research, long-term crypto capital market assumptions imply that the total crypto market value and global stock market share could reach ~$10 trillion (~7.5% share) over 7 years[32].

Therefore, in these years, the dynamics of digital currency and cryptocurrency market prices have attracted the attention of more and more people. The astonishing rise in price over the years made cryptocurrencies one of the most popular and promising type for investments and trading speculations. They have recently become large enough to be included within an institutional portfolio. Indeed, investors are increasingly evolving their portfolios adding this alternative asset class, as new financial products on the cryptocurrency market are created with the apparition of options and future on Bitcoin and Ethereum[33].

Hence, experts are constantly studying the market to better understand, model and predict the trends within the price fluctuations[34]. But the generation of an accurate prediction model for such an intricate problem is very challenging[35]. It is difficult to determine the exact causes that drive the price of cryptocurrencies over time. Due to their young history, the price prediction problem is still in its nascent stages and takes further research efforts to better understand the dynamics in this new field. To find a

[31] Data available at: https://coinmarketcap.com/

[32] D. GRIDER, M. MAXIMO, M. ZHAO, *The Postmodern Portfolio Crypto Allocation Thesis*, Grayscale Investments LLC, Stamford, p. 19, 2020. Available at: https://grayscale.com/learn/postmodern-portfolio-theory/

[33] D. KUMAR, S.K. RATH, *Predicting the trends of price for Ethereum using deep learning techniques*, In Artificial Intelligence and Evolutionary Computations in Engineering Systems, Springer, pp. 103–114, 2020.

[34]].R.K. ALKHODHAIRI, S.R. ALJALHAMI, N.K. RUSAYNI, J.F. ALSHOBAILI, A.A. AL-SHARGABI, A. ALABDULATIF, *Bitcoin candlestick prediction with deep neural networks based on real time data*, CMC Computers Materials & Continua, 2021.

[35] J. CHEVALLIER, D. GUÉGAN, S. GOUTTE, *Is it possible to forecast the price of bitcoin? Forecasting*, pp. 377–420, 2021.

method able to accurately predict the price behaviour, traders are focusing on statistical and Machine Learning (ML) approaches.

The complexity and difficulty of cryptocurrency price forecasting and analysis is caused by the large number of unpredictable factors[36], which are involved in this field, and the significant fluctuations and volatility[37], resulting in complicated temporal dependencies [38]. Indeed, among professional and retail investors, cryptocurrencies are well known for their high volatility, noisy price fluctuations and presence of speculative bubbles[39]. Hence the features of digital currency time series make it arduous to predict their future fluctuations using historical price values[40].

As regard the multiple factors which may affect cryptocurrency prices, they are divided into internal and external factors. The latter are those connected to the crypto market aspects, as the market trend, the attractiveness and speculators; the issue is some cryptocurrencies that are not highly ranked, are market-driven by the total market capitalization. Other external factors are macro-economic elements like stock markets, interest rate, exchange rate and regulations. As concern the intern factors, the main one is the supply and demand, where issues are related to coins circulations, transaction cost, reward system[41], fork rules[42], mining difficulty[43] (hash rate)[44].

---

[36] I. E. LIVIERIS, S. STAVROVIANNIS, E. PINTELAS, P. PINTELAS, *A novel validation framework to enhance deep learning models in time-series forecasting,* Neural Comput., Appl., 2020.

[37] V. DERBENTSEV, A. MATVIYCHUK, V. N. SOLOVIEV, *Forecasting of Cryptocurrency Prices Using Machine Learning,* In Advanced Studies of Financial Technologies and Cryptocurrency Markets, Springer, Berlin/Heidelberg, Germany, pp. 211–231, 2020.

[38] R. CHOWDHURY, M. A. RAHMAN, M. S. RAHMAN, M. MAHDY, *Predicting and Forecasting the Price of Constituents and Index of Cryptocurrency Using Machine Learning*, 2019. Available at: arXiv:1905.08444.

[39] S. CORBET, B. LUCEY, L. YAROVAYA, *Date stamping the bitcoin and ethereum bubbles,* Finance Research Letters, vol. 26, pp. 81 – 88, 2018.

[40] G. S. ATSALAKIS, I. G. ATSALAKI, F. PASIOURAS, C. ZOPOUNIDIS, *Bitcoin price forecasting with neuro-fuzzy techniques,* European J. Oper. Res. 276 (2), pp. 770–780, 2019. Available at: https://-www.sciencedirect.com/science/article/abs/pii/S037722171930075X

[41] *A block reward is unit of digital token gained by the so-called validators, users who help to verify transactions on a blockchain protocol.* Available at: https://www.coindesk.com/learn/what-is-a-block-reward/

[42] *In the cryptocurrency sector, the term "fork" refers to a software update aimed at introducing changes, new features or new rules.* Available at: https://www.coinbase.com/it/learn/crypto-basics/-what-is-a-fork.

[43] *A cryptocurrency's mining difficulty indicates how challenging and time-consuming it is to get the correct hash for each block.* Available at: https://www.bitpanda.com/academy/en/lessons/what-does-mining-difficulty-mean/

[44] L. KRISTOUFEK, *What are the main drivers of the bitcoin price? evidence from wavelet coherence analysis*, PloS one, 2015.

Despite the knowledge of all these factors, cryptocurrency price behaviour remains still largely unexplored. Therefore, this sector presences lots of new opportunities for researchers and analysts to compare digital asset prices with standard financial ones.

It is up to the analyst to understand what are the causal relationships that determine the price, and to work using the best methodological approaches to forecast prices of the different financial assets. Forecasting models can be specified through different approaches:

- *Demand and Supply models*: taking into account the interaction of market players who take economic decisions on some indicators or regularities.
- *Econometric and balance models*: based on economic principles or behavioural financial laws.
- *Time Series models and Autoregressive models*: focused on the past dynamics.
- *Stochastic models*: given random events or factors such as external shocks.

It should be underlined that predicting cryptocurrencies' prices significantly differs from forecasting other financial assets, which have been subjected to a large amount of theoretical and empirical research. Therefore, the classic models listed above may not be the most suitable for this goal.

Nowadays, several attempts and scientific research have been made in relation to stock price prediction[45], while there are still few studies on digital currency price prediction. This because the history of the cryptocurrency market is very recent compared to the stock market. Hence, it might be useful to focus on new prediction techniques appropriate for the cryptocurrency market in addition to the existing stock market prediction models.

The cryptocurrency market differs from the traditional stock market, and it has lot of new characteristics[46]. Structurally, both cryptocurrency and stock price data have arbitrary properties such as time series data, but the former frequently exhibits

[45] E. KITA, M. HARADA, T. MIZUNO, *Application of Bayesian network to stock price prediction*, Artificial Intelligence Research, vol. 1, no. 2, pp. 171-184, 2012.

[46] H. JANG, J. LEE, *An empirical study on modelling and prediction of bitcoin prices with Bayesian neural networks based on blockchain information*, IEEE Access, vol. 6, pp. 5427-5437, 2017.

considerable volatility, with significant price fluctuation. Cryptocurrency exchanges are available on a 24/7 global market, while traditional stock exchanges have their own trading session (e.g., New York Stock Exchange from 09:30 am to 04:00 pm) with no trading during weekends and state holidays. Compared to the stock market, the cryptocurrency one is less liquid, so much easier to manipulate. This can result in unexpected and rapid increase or decrease in price, as so-called “pump and dumb” schemes[47]. An additional aspect regards the custody; cryptocurrency trading generally needs the investors to store the coins themselves and this can embarrass users who are not inclined to do so. All these features make the cryptocurrency price prediction complex, and they lead to the investigation of new analytical techniques suitable for the digital currency market.

Financial and economic studies have long placed a strong emphasis on price forecasting since it may provide a significant competitive edge for investors. Additionally, it can assist decision-makers in taking the appropriate actions at the right time to maximize profits and reduce losses. As a result, several research studies investigated various options and proposed a variety of time series prediction methods to address the problem of cryptocurrency price forecasts.

Time series predictions is not a new approach, given that is one of the most used data science techniques in economics, finance, statistics, and econometrics[48]. It is a technique to predict the future value. It involves creating models based on previous tendencies, presuming that historical behaviours will continue to present in the future, and using them to predict future events. This type of information is called *pattern* and by analysing these data, it is possible to trace it back to information, like trends, seasonality, cyclicity, and irregularities.

In order to make comprehensive the differences between these terms, they need to be defined more carefully (Figure 1.2):

---

[47] *A pump and dump scheme regards the inflation of an asset’s price using false information. The perpetrators sell their cheaply bought positions at a much higher price once it has significantly gone up. Pump and dump patterns are increasingly found in the cryptocurrency industry.* Available at: https://www.investopedia.com/terms/p/pumpanddump.asp.

[48] J. CAO, Z. LI, J. LI, *Financial time series forecasting model based on ceemdan and lstm*, Physica A: Statistical Mechanics and its Applications, vol. 519, pp. 127–139, 2019.

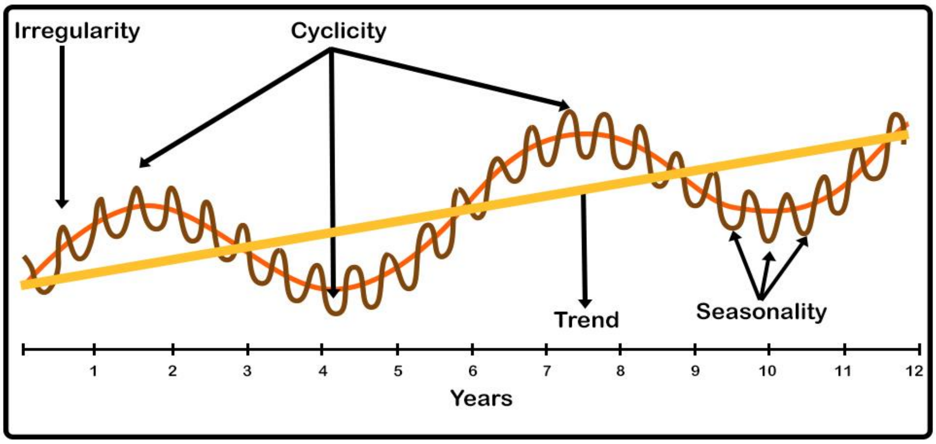

Figure 1.2: Illustration of Trend, Seasonality, Cyclicity and Irregularity

- *Trend*: a trend is defined as increase or decrease in a value over a certain period of time. It does not have to be linear. An increasing or positive trend represents a long-term growth, while a decreasing or negative trend indicates a decline. A trend is said to change its direction when it goes from an increasing trend to a decreasing trend or vice versa.
- *Seasonality*: it can be described as a regularly repeating pattern that happens on known fixed intervals (such as the time of the calendar year). Seasonality does not include patterns that repeat on an irregular schedule.
- *Cyclicity*: a cycle occurs when data exhibit rises and falls, which are not of a fixed frequency. It refers to oscillations around the trend, excluding irregular components, and showing a series of stages of expansion and contraction.
- *Irregularity*: it exists when data has no trend, seasonality or cyclicity.

This kind of technique is more suitable for task where trends, seasonal patterns, cyclic effects are present. ARIMA models are likely one of the most used methods for time series forecasting over a short-term condition. It works when displays a steady or constant pattern across time with the fewest number of possible outliers. However, this possibly does not always work, especially when data swing drastically and it is extremely volatile, like in cryptocurrency scenarios[49]. Because of the lack of

[49] C. SCHEIER, W. TSCHACHER, *Appropriate algorithms for nonlinear time series analysis in psychology*, in Nonlinear dynamics in human behavior, World Scientific, pp. 27–43, 1996.

seasonality and the high volatility in cryptocurrency market, these traditional methods are not particularly successful for price prediction.

Financial time series forecasting has seen steady advancements over the past few years[50]. As result of the recent increase in computational power of computers, more machine learning methods have been used to forecast time series data in the cryptocurrency market.

Despite the most common ML approaches, researches are constantly developing more efficient and sophisticated algorithms, like Deep Neural Networks (DNN) models[51]. Over the last decade, deep learning has significantly enhanced the efficiency of its applications becoming the main stage for price forecasting[52].

Digital currency market produces a wide amount of highly non-linear transaction data. To properly analyse these dynamic data, it is needed a model, which is capable of exploring and extracting hidden information and internal patterns in the digital currency raw data[53]. Several studies have demonstrated that Deep neural network models fit good for learning hidden features within financial time series[54] [55]. They are able to outperform traditional statistical models showing a much more reliable prediction[56] [57]. With the high accuracy and low error of the crypto forecasting prices, DNN techniques are getting increasingly implemented in the field of cryptocurrency

---

[50] A. A. ARIYO, A. O. ADEWUMI, C. K. AYO, *Stock price prediction using the arima model*, UKSim-AMSS 16th International Conference on Computer Modelling and Simulation. IEEE, pp. 106–112, 2014.

[51] *Deep learning neural networks attempts to mimic the human brain through a combination of data inputs, weights, and bias. These elements work together to accurately recognize, classify, and describe objects within the data.* Available at: https://www.ibm.com/cloud/learn/deep-learning

[52] M. NIKOU, G. MANSOURFAR, J. BAGHERZADEH, *Stock price prediction using deep learning algorithm and its comparison with machine learning algorithms*, Intelligent Systems in Accounting, Finance and Management, pp. 164–174, 2019.

[53] S. LAHMIRI, S. BEKIROS, *Cryptocurrency forecasting with deep learning chaotic neural networks*, Chaos, Solitons & Fractals, vol. 118, pp. 35–40, 2019.

[54] R. ADCOCK, N. GRADOJEVIC, *Non-fundamental, non-parametric bitcoin forecasting*, Physica A: Statistical Mechanics and its Applications, vol. 531, p. 121, 2019.

[55] M. NAKANO, A. TAKAHASHI, S. TAKAHASHI, *Bitcoin technical trading with artificial neural network*, Physica A: Statistical Mechanics and its Applications, vol. 510, pp. 587 – 609, 2018.

[56] D. KUMAR, SK RATH, *Predicting the trends of price for ethereum using deep learning techniques*, In Artificial Intelligence and Evolutionary Computations in Engineering Systems, pp. 103–114, 2020.

[57] L. YANG, X. LIU, X. LI, Y. LI, *Price prediction of cryptocurrency: An empirical study*, In International Conference on Smart Blockchain, pp. 130–139, Springer, 2019.

price prediction. However, these methods are in turn limited since they have not taken into account all the variables affecting the cryptocurrency market.

Some researchers (see Pintelas et al.[58] and Livieris et al.[59]) have pointed out that there are two primary factors that contribute to the difficulty of forecasting. First, digital currencies time-series are very similar to random walk process, and second, the presence of autocorrelation in the errors and the lack of stationarity[60], which make deep learning models not so extremely efficient. In general, non-stationary series, which are characterized by high volatility, frequently exhibit heteroskedasticity, and changing over time in properties, such as mean, variance and kurtosis.

The technique which has likely showed a better-quality price prediction between all the different DNN approaches is the Long Short-Term Memory (LSTM) model, primarily thanks the temporal nature of its more advanced algorithms[61]. Latest studies concluded that using the right LSTM architecture allows to capture long-term sequence patterns and both long and short-term dependencies[62].

---

[58] E. PINTELAS, I.E LIVIERIS, S. Stavroyiannis, T. Kotsilieris, P. Pintelas, *Investigating the Problem of Cryptocurrency Price Prediction: A Deep Learning Approach*, In IFIP International Conference on Artificial Intelligence Applications and Innovations, Springer, Berlin/Heidelberg, Germany, pp. 99–110, 2020.

[59] I.E LIVIERIS, E. PINTELAS, S. STAVROYIANNIS, P. PINTELAS, *Ensemble Deep Learning Models for Forecasting Cryptocurrency Time-Series*, Algorithms, pp. 13-121, 2020.

[60] I.E LIVIERIS, S. STAVROYIANNIS, E. PINTELAS, P. PINTELAS, *A novel validation framework to enhance deep learning models in time-series forecasting*, Neural Comput. Appl., pp. 17149–17167, 2020.

[61] S. MCNALLY, J. ROCHE, S. CATON, *Predicting the Price of Bitcoin Using Machine Learning*, in 2018 26th Euromicro International Conference on Parallel, Distributed and Network-based Processing (PDP), pp. 339-343, 2018.

[62] I.E LIVIERIS, E. PINTELAS, P. PINTELAS, *A CNN–LSTM model for gold price time-series forecasting*, Neural Comput. Appl., pp. 17351–17360, 2020.

Chapter 2
MACHINE LEARNING AND ITS SUB-BRANCHES

Machine Learning (ML) has largely developed during the last decades becoming responsible for some of the most significant advancement in technology. In the financial sector, ML is being applied in a wide range of areas including for example, algorithmic trading, portfolio management, fraud detection and prevention. At the same time, it is being commonly used by people in their daily life even without knowing it, such as Goggle Maps and Alexa. In the last period, more attention has gone to Deep Learning methods which, thanks to the recent development in neural network architecture and training algorithms, were capable to outperform any other machine learning technique in terms of market growth. According to Forbes, the global machine learning market and the deep learning market will growth at a CAGR of 42.8% and 52.1% respectively by 2024[63]. Among these models, in this chapter it will be presented one type of Recurrent Neural Network, the Long Short - Term Memory model, which has showed to be particularly suitable for sequential data such as time series and price prediction. In order to provide a general framework of the topic that allows a better understanding of the model, it will be illustrated historical notes of machine learning, the definition of the term and its different types of learning techniques.

2.1 Machine Learning

2.1.1 Origins and evolution of Machine Learning

Although it is a sub-branch of artificial intelligence, even today the term machine learning is frequently abused to allude to AI and vice versa, with the risk to create confusion to who is not expert on the subject. This interchange is due to

[63] L. COLUMBUS, *Roundup of Machine Learning Forecasts And Market Estimates, 2020*, Forbes, January 19, 2020. Available at: https://www.forbes.com/sites/louiscolumbus/2020/01/19/roundup-of-machine-learning-forecasts-and-market-estimates-2020/?sh=7f7f599e5c02

the ML learning and decision-making capacities and its scope of using technologies around AI. The erroneous juxtaposition is also pushed to the fact that, until the late 1970s, ML was only considered as part of a wider AI system. From this time on, it separated to evolve on its own, shifting the focus from training for AI to solving practical problems in terms of providing services.

Before this decade, there are several dates that deserve to be mentioned. The first expression of ML dates back to 1943, when the logician Walter Pitts and the neuroscientist Warren McCulloch published a paper where they endeavoured to mathematically outline out the processes and choice making in human cognition and neural networks[64]. They were able to highlight the correspondence that exist between the brain and the computer machine, where the neuron stands for the digital processor and the brain is the computing device.

In 1950, the mathematician and computer scientist Alan Turing proposed the Turing test, through which it was possible to put an end to the dilemma for which machines can be considered "intelligent" or less, and, therefore, if it can be deemed an artificial intelligence. The criterion to evaluate the level of intelligence of the machine stands for the ability of the computer to reproduce the human behaviour. The test required an examiner to have a conversation with both a human being and with a machine and be able to identify their respectively identities.

The 1950s was when pioneering researches contributed to the definition of ML in different ways, both through formal methods, probabilistic approaches and the application of the so-called neural networks[65]. The term machine learning was coined for the first time, in 1952, by Arthur Samuel, an IBM developer in the field of computer gaming and artificial intelligence[66]. He wrote the first computer program with the scope to play the first "intelligent" game of checkers, in which the algorithm is able to evaluate both the possible moves at its disposal and the opponent's counter ones.

---

[64] W.S McCulloch, W. Pitts, *A logical calculus of the ideas immanent in nervous activity*, The Bulletin of Mathematical Biophysics, Volume 5, Issue 4, pp 115-133, 1943.

[65] *Neural networks, also known as artificial neural networks (ANNs) or simulated neural networks (SNNs), reflect the functioning of the human brain, allowing computer programs to identify patterns and address common problems in the fields of AI, machine learning, and deep learning.* Available at: https://www.ibm.com/cloud/learn/neural-networks

[66] IBM, Deep Blue. Available at: https://www.ibm.com/ibm/history/ibm100/us/en/icons/deepblue/

Moreover, the algorithm not only considers the opponent's possible counter move in the turn immediately following, but also the other possible moves in subsequent turns.

The first neural network model was created by Frank Rosenblatt[67] at the Cornell Aeronautical Laboratory in 1957. It was an electronic device capable of showing learning ability, called “Perceptron”. It was made through the combination of Donald Hebb’s model of brain cell interaction[68] with Arthur Samuel’s machine learning efforts. This model was a network with an input and an output state, and an intermediate learning rule based on the error backpropagation algorithm[69]. Basically, based on the evaluation of the output data, compared to an input data, it alters the weights of the connections causing a difference between the actual output and the desired one[70]. Although the perceptron seemed promising, it could not recognize many kinds of visual patterns (such as faces), causing frustration and stalling neural network research.

In the 1960s was discovered a new direction in neural network research, which no longer consists in the use of only one layer in the perceptron[71], but in multiple ones allowing a significance increase in the processing power. The spread of multilayers approach led to the study of new algorithms such as the backpropagation, which is actually used to train deep neural networks[72].

---

[67] *Frank Rosenblatt (1928 – 1971) was a research psychologist and project engineer at the Cornell Aeronautical Laboratory in Buffalo, New York, who stood out in in the field of deep learning.*

[68] *Donald Olding Hebb (1904 – 1985) was a Canadian psychologist considered as the father of neuropsychology and neural networks. He was one of the first scientists to investigate the link between the nervous system and behaviour, and to understand how the function of neurons contributed to psychological processes such as learning. He gave its name to one of the fundamental learning algorithms in the field of neural networks, the Hebbian learning, which he introduced in his work The Organization of Behaviour: A Neuropsychological Theory in 1949.* Available at: https://can-acn.org/donald-olding-hebb/

[69] *In machine learning, backpropagation algorithms are a set of methods used to efficiently train artificial neural networks following an optimization method, such as a gradient descent approach which exploits the chain rule.*

[70] F. ROSENBLATT, *The perceptron: A probabilistic model for information storage and organization in the brain*, Psychological Review, 65(6), pp. 386-408, 1958. Available at: https://citeseerx.ist.psu.-edu/viewdoc/download?doi=10.1.1.335.3398&rep=rep1&type=pdf

[71] *Perceptron is a linear Machine Learning algorithm used for supervised learning for various binary classifiers. It was invented by Mr. Frank Rosenblatt for performing certain calculations to detect input data capabilities or business intelligence.* Available at: https://www.javatpoint.com/perceptro-n-in-machine-learning

[72] W. LIU, Z. WANG, X. LIU, N. ZENG, Y. LIU, F. E. ALSAADI, *A survey of deep neural network architectures and their applications*, Neurocomputing, vol. 234, pp. 11-26, 2017.

Despite these first successes, a discrepancy between the analysis of AI and its subfield ML was continuing to persist. In the 1970s and 1980s, while the study of artificial intelligence was focused on logical knowledge-based approach rather than algorithms, the research in machine learning followed a probabilistic one. Therefore, the ML continued to encounter several application problems. One of these, the acquisition and representation of large amounts of data, that science was not able to deal with yet. The increasing computational complexity was not driven sufficiently by an infrastructural growth since there was no hardware capable of supporting the weight of such operations. Hence, ML developed as a distinct discipline from classical artificial intelligence, began to take hold seriously only in the 90s. The turning point occurred with the transition from a knowledge-based to a data-based approach. The goal was no longer to create a form of artificial intelligence, but rather to solve specific practical problems. This different approach, based on methods and tactics used in statistics and probability theory, has allowed a strong development of the discipline of ML. The industry goal shifted from training for artificial intelligence to solve practical problems in terms of providing services.

The first economic problem resolved applying a ML methodology dates back to 1988, when Halbert White, Professor of Economics at the University of California, published a paper in which forecasted the IBM daily stock return using a Neural Network approach[73]. From this event, the adoption of ML in Finance raised. A large increase of studies on the subject occurred with the spread of internet, thanks to the fact that it allowed a significant acceleration of digital applications, as well as an availability and distribution of information.

In the last twenty years, parallel to the growth of Internet, the transition to web 2.0[74] and the spread of social media have granted an exponentially production of data. This

[73] H. WHITE, *Economic prediction using Neural Networks: the case of the IBM daily stock returns*, Department of Economics University of California, San Diego, 1988. Available at: http://machine-learning.martinsewell.com/ann/White1988.pdf

[74] *Web 2.0 is the current version of the web and it refers to the second generation of internet services, which focused on enabling users to interact with content on the web. Web 2.0 has been driven by key innovations such as mobile internet access and social networks, that involved the growth of user-generated content alongside interoperability and usability for end users.* Available at: https://www.investopedia.com/web-20-web-30-5208698

has kicked off the generation of what are now called "Big Data"[75] and the development of the activity of data mining: the set of techniques that allow the extraction of useful information precisely from these large quantities of data. In ML, data plays a fundamental role, as they represent the source of knowledge that the computer can use to learn. This is the main reason why ML has seen its diffusion only in the last two decades. The latest innovations which are enhancing even more the application power of ML are 5G connection[76] and quantum computers[77].

Hence, ML is now responsible for some of the most significant advancements in technology. One of these is the creation of the so-called Web 3[78]. The fast evolution of ML research has translated into an overwhelming number of ML platforms, frameworks and APIs than can be used to add intelligent capabilities to Web 3 solutions.

2.1.2 Definition of Machine Learning

The first definition "*Machine Learning is the field of study that gives computers the ability to learn without being explicitly programmed*"79 belongs to Arthur Lee Samuel, an American computer scientist, who was the first to coin the term in 195980. This explanation of the term frames Machine Learning as an approach able to create systems that learn from initial data, entered as input, with the aim to manage other data, which were not initially considered by the programmer.

[75] *"Big data" are large computer data generally differently structured, that cannot be analysed and archived with traditional tools. Big data analysis requires specific skills and advanced technologies, such as super computers and algorithms capable of supporting the processing of such large files, in order to extract useful, hidden information.*

[76] *5G is the fifth generation of cellular technology. It is designed to increase speed, reduce latency, and improve flexibility of wireless services.* Available at: https://www.cisco.com/c/en/us/solutions-/what-is-5g.html

[77] *Quantum computers are machines that store data and perform computations using the properties of quantum physics.* Available at: https://www.technologyreview.com/2019/01/29/66141/what-is-quantum-computing/

[78] *The term "Web 3" was coined in 2014 by Gavin Wood, founder of Polkadot and co-founder of Ethereum. It is used to indicate the latest generation of digital innovations, such as machine learning, artificial intelligence (AI) and blockchain technology and their applications on the internet.* Available at: https://academy.binance.com/en/articles/the-evolution-of-the-internet-web-3-0-explained

[79] A.L. SAMUEL, *Some studies in machine learning using the game of checkers*, IBM Journal of research and development, vol. 3, no. 3, pp. 210–229, 1959.

[80] K.D FOOTE, *A Brief History of Machine Learning*, March 26, 2019. Available at: https://www.dataversity.net/a-brief-history-of-machine-learning/

Despite this initial explanation of the subject, the most accredit definition by the scientific community is that provided by another American, Tom Michael Mitchell, director of the Carnegie Mellon University for the Machine Learning Department:

"A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P if its performance at tasks in T, as measured by P, improves with experience E"[81].

This description of the task in which ML is concerned, offers a fundamentally operational definition rather than explaining the field in cognitive terms.

Hence, in simple words, ML algorithms allow computers to learn independently thanks the data provided through experiences. There is a learning process, in a sense of experience acquisition, when the performance of the machine improves after the carrying out of a task or the execution of an action, which could be even wrong, starting from the assumption that "making mistakes in learning " principle applies and works also for the human. In Mitchell's definition, "learning" is the mean to achieve the ability to perform the "task". The latter describes how machine learning should process a set of features that have been quantitatively measured.

Looking at ML from an IT perspective, instead of coding the program through which the machine is "told" what to do step by step, the program is provided with only data sets that are processed through algorithms that develop their own logic to perform the task required. While ML models have very different learning strategies and goals, they all share these common characteristics of training phase, which allow the algorithm to learn how to put the assigned input in correspondence with a certain output.

At the base of machine learning there is a series of different algorithms that, starting from native notions, are able to take a specific decision rather than another or carry out actions learned over time.

The machine learning process could be divided into three macro phases:

- Learning from data, in the different forms in which they can occur.

[81] T.M MITCHELL, *Machine Learning*, McGraw-Hill, 1997.

- Data evaluation, in which the system hypothesizes statistical models that describe the observed reality.
- Optimization of the estimated models and formulation of a response / action based on the feedback collected with the experience.

2.1.3 Types of Machine Learning

As previously seen, trough the learning and improvement upon past experiences, ML teaches computer programs to reason as human tends to do. To do this, data and labels are provided to the computer and the machine tries to learn from patterns that recur frequently; after this, the computer can recognize these patterns even on data it has never seen before. Depending on the type of data provided as input or on what are the tasks required, the process of ML can be implemented in different ways. The ML algorithms can be classified info three learning systems: supervised learning, unsupervised learning and reinforcement learning. Parallel to these approaches, more complex ad sophisticated ones have been developed over time. One of the best known, which in the last period has had the opportunity to spread thanks to its great applications, is the Deep Learning. It is a branch of Machine Learning algorithms that uses Deep Neural Networks to discover patterns from data and learn correlation from them. The main characteristics of these techniques and some of their most important algorithms are presented in the following paragraphs.

2.1.3.1 Supervised Learning

From a logical point of view, supervised learning starts from the assumption that, if the system has an adequate number of information, it will have an experience E such as to allow it to determine a function capable of approximating the value of the target. Given the parallel between the estimated function and the objective one, when input data, which are not contained in E, are proposed, then the inductive hypothesis should still be able to provide satisfying answers[82].

[82] IEEE Electron Devices Society, Institute of Electrical and Electronics Engineers, and Vaigai College of Engineering, *Proceeding of the 2018 International Conference on Intelligent Computing and Control Systems (ICICCS)*, IEEE, Piscataway, New Jersey, June 14- 15, 2018.

In Supervised Machine Learning, the model is provided with both input datasets and information about its outputs, with the aim of identifying patterns and general rules that associate initial information x with final result y. In this way, when faced with a new task, all is needed is to draw on experiences included in the system, analyse them, and decide which answer to give based on already codified experiences, even for new inputs not present in the training dataset. Hence, the goal of supervised learning is to provide a function that is able to learn from the input-output pairs provided in the training phase, and then obtain the desired results in a new test dataset. Each input data x_i is characterized by values, typically numerical, called *features*, which quantitatively describe the characteristics of a given example. The output y_i, called *target value*, represents the result and it can be precisely assessed through performance measures, adapted to the type of problem being faced.

Supervised learning problems in turn can be divided into two macro-categories: *Classification* problems, where the aim is to predict discrete values representing the classes to which the input data belongs (for example the analysis of a positive or negative feeling), and *Regression* problems, in which are predicted continuous values (for example the forecasting of stock prices).

Some of the most diffuse Classification techniques are Logistic Regression, Support Vector Machines (SVMs) or K-nearest Neighbours (KNN), while for the Regression techniques, the most relevant are Linear Regression, Decision Trees or Artificial Neural Networks (ANNs).

2.1.3.2 Unsupervised Learning

Another type of machine learning is Unsupervised Learning. It has extremely different characteristics and methods respect supervised learning; it does not aim to classify samples, but, instead, tries to group samples according to a similarity criterion. In an opposite manner, the system receives only the input dataset, and the algorithm must learn to recognize hidden patterns without being given any specific indication of the output values. The reason of this difference lies in the type of data, which are provided to the system in the form of datasets. These are unlabelled datasets without any indication regarding the output result; hence, the previously analysed classification and regression methods are not applicable.

Thanks to the use of statistical and probabilistic techniques, these algorithms are highly efficient in cases where problems insist on numerical data, unlike the analysis of non-numerical data. In this case, a set of input data and a variant cost function are provided depending on the end goal. The output of such an algorithm may be: the reduction of the dimensional space representation of "*features*", the discovery of examples that deviate from the distribution or the clustering of data with similar features.

The unsupervised learning algorithm can be further classified into two types of problems: Clustering and Association. In the first one, clustering algorithms aggregate data with similar characteristics into groups called "*clusters*". On the other hand, association techniques are used to find relationships between variables in large amount of data.

2.1.3.3 Reinforcement Learning

Third and last branch of ML is Reinforcement Learning. Through the study of decision-making processes, it seeks to understand which is the best way to take decisions in a given context. In the two previous models of ML, the data provided to the system turned out to be of fundamental importance. However, reinforcement learning uses a totally different approach. The model does not need an initial dataset to start from, but in order to achieve a given goal, it can simply make its first choices randomly using information about the surrounding environment. To do this, the algorithm uses an "agent" which explore the environment where it is located and perform actions.

The agent is not instructed what actions to take, but it has to discover by itself which ones are deemed correct through obtaining the rewards associated. The agent learns what are the best actions to take by gaining experience through a "trial and error" process. At a certain instant t, the agent interacts with the environment. For each t, the agent receives a reward related to the action taken in $t - 1$ and a status as input. The status is defined as the interaction between the agent and the environment. Based on these, the agent determines the action to be performed. The latter is received by the environment which process it and elaborates a new state and reward signal corresponding to the next input of the agent in the time period $t + 1$. The process, repeated several times, creates the learning algorithm of the Reinforcement Learning agent.

The objective of the agent is to gain as much as possible in terms of the expected cumulative reward. To achieve this goal, the agent has the ability to understand which moves are most successful, simply evaluating the reward gained and consequently adapting the policy, which is the function that maps from observations of the environment to actions. On the contrary, if the behaviour of the agent is considered incorrect, it will be applied a punishment, which will decrease the probability of the repetition of the error.

2.2 Deep Learning

Despite ML-based models have achieved some success in cryptocurrency price prediction[83], various researchers have pushed forward to study applications of the DL techniques to provide even more accurate price forecasts. This is due both the highly complex nature of cryptocurrencies and to the technical side of deep learning. The latter requires a large amount of labelled data and a considerable processing power. Hence, with the spread of Big Data science and the increase of computational abilities of computers through the use of high-performance Graphical Processing Units (GPUs)[84], the usage of Deep Learning has grown more and more in recent years, especially in fields like Time Series Processing, Natural Language Processing[85] and Computer Vision[86].

---

[83] *Price prediction is the process of estimating a good's, or service's price based on several variables, including its characteristics, demand, seasonal trends, the cost of other products, offers from other suppliers, etc.* Available at: https://www.altexsoft.com/blog/business/price-forecasting-machine-learning-based-approaches-applied-to-electricity-flights-hotels-real-estate-and-stock-pricing/

[84] *Graphics processing unit, or GPU, is a specialized processor originally designed to accelerate graphics rendering. GPUs can process many pieces of data simultaneously, making them useful for machine learning, video editing, and gaming applications.* Available at: https://www.intel.com/content/www/us/en/products/docs/processors/what-is-a-gpu.html

[85] *Natural language processing (NLP) refers to the branch of artificial intelligence or AI, concerned with giving computers the capacity to comprehend text and react to text or voice data—and respond with text or speech of their own—in much the same way humans do.* Available at: https://www.ibm.com/cloud/learn/natural-language-processing

[86] *Computer vision is a branch of artificial intelligence (AI) that enables computers and systems to extract useful information from digital images, videos and other visual inputs — and take actions or make suggestions based on that information.* Available at: https://www.ibm.com/topics/computer-vision

2.2.1 Artificial Intelligence, Machine Learning and Deep Learning

Artificial intelligence lays the foundation for both Machine Learning and Deep Learning. But, while ML represents a sub-branch of AI, Deep Learning is in turn a specialized form of Machine Learning. Often ML and DL terms are interchanged in meaning, but, in reality, they indicate two fundamentally different methods that have distinct fields of application. To better understand the difference between these branches, it helps to imagine them as sets and subsets as shown in Figure 2.1.

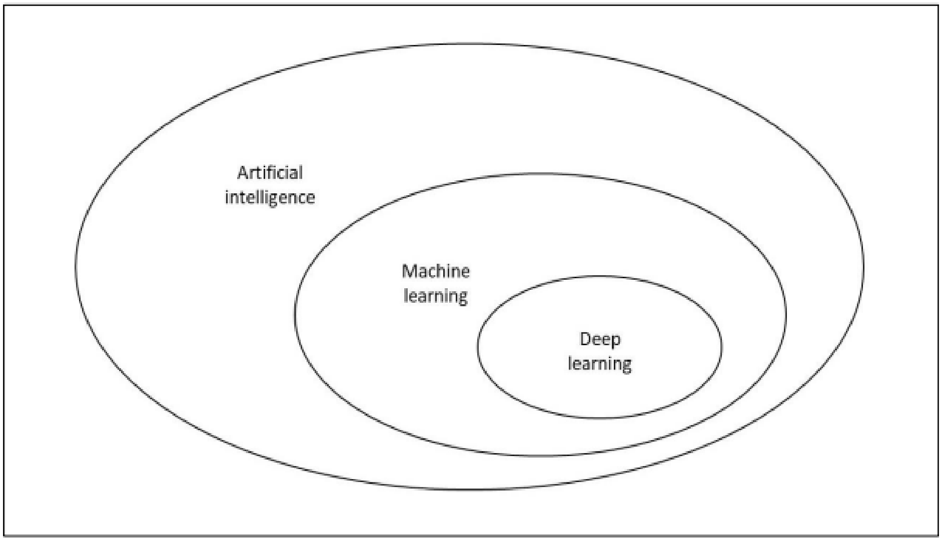

Figure 2.1: Artificial intelligence, machine learning and deep learning

The outermost circle represents the AI, which stands for the goal to achieve: a method capable of automate intelligent tasks normally performed by humans. Within this set, there is a smaller one, that of Machine Learning. It is a data analysis technique that gives computers the ability to learn from experience. Machine Learning algorithms use computational methods to learn information directly from data without relying on a predetermined equation as a model generating the data.

Finally, Deep Learning is a particular type of machine learning that learns to perform classification tasks directly from the processing of unstructured data such as images text and sound, and by automating feature extraction, it reduces some reliance on human specialists. It is a novel approach of learning representations from data that emphasizes learning successive layers[87] of increasingly meaningful representations. The "deep" in deep learning stands for the idea of learning successive layers of increasingly meaningful representations from data to get more precise details. The number of processing layers through which the data transformation takes place is called the "depth" of the model.

2.2.2 Neural Networks

In deep learning, these layered representations are learnt via mathematical models called Neural Networks, which are constructed as layers piled on top of each other. They are conceived and structured on the human brain, each part of them is attributable to an element belonging to it. Just as the human brain combines a series of inputs deriving from the sensory organs and provides outputs that can range from the execution of a movement to the formulation of a though, that's how Neural Networks combine a series of inputs to provide one or more outputs[88].

NN is an information processing paradigm which is used to study the behaviour of a complex system by computer simulation. It is inspired by the biological way of processing information by human brain.

Neural Networks are interesting for their ability to process data, which are often noisy or very complex, and to derive a series of patterns that even humans are not able to extract. Thanks to the presence of intermediate levels[89] in the Neural Network, it is possible to extrapolate from each single feature in input a representation of the same at a higher level of abstraction. Combining it with the other features, important information is obtained in view of solving the problem. Hence, the network gains an excellent ability of adaptation, that allows to change its parameters in progress, in order to solve any type of solution in the best possible way.

---

[87] *A layer is the higher fundamental unit in deep learning. A layer is a container that typically receives weighted input, transform it with a set of mostly non-linear functions and then passes these values as output to the following layer.* Available at: https://developer.nvidia.com/blog/deep-learning-nutshell-core-concepts/

[88] A. GÉRON, *Hands-on machine learning with Scikit-Learn, Keras, and TensorFlow: Concepts, tools, and techniques to build intelligent systems*, O'Reilly Media, 2019.

[89] *Intermediate layers (levels) are the layers between input layer and the output layer. It is also referred to as the "hidden layer" since it is where all of the neural network calculations take place.* Available at: https://towardsdatascience.com/comprehensive-introduction-to-neural-network-architecture-c08c6d8e5d98

To better understand how Neural Network solves any specific problem in the same way as a human brain would, let first describe the structure of a biological neuron. Its main components are three: the dendrites, the axon and the cell body as illustrated in Figure 2.2. The dendrites are similar to fibers branched in various directions and are connected to several cells in that cluster. Dendrites receive the signals from neighboring neurons and then, the axon transmits the signal to the other neurons. At the terminal part of the axon, a synapse is used to make the connection with the dendrite. The output signal is carried down the length of the axon as electric impulses. Each neuron has one axon. Axons send impulses from one neuron to another like a domino effect.

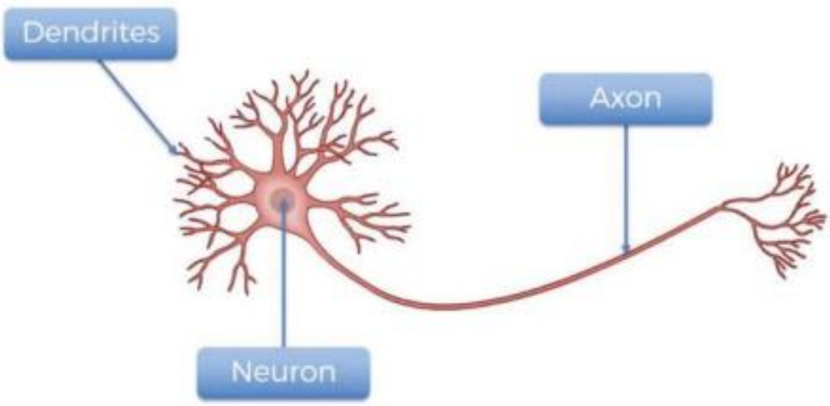

Figure 2.2: Basic illustration of a neuron

A "computer neuron" is built in a similar manner. In deep learning models, a neuron is a node through which data and computations flow. The component which represents the neuron is called perceptron[90]. One or more input signals are received by it. These input signals may originate from neurons located in a previous layer of the neural network or from the raw data set. After performing certain calculations, it uses a synapse to transmit some output signals to neurons further inside the neural net. The input layer resembles the dendrites while the output represents the axon. Each input is assigned a weight or coefficient, W_i, which affect the relevance of the previous neuron in the overall neural network. After a neuron receives its inputs from the neurons in the layer before of the model, it adds up each signal multiplied by its associated weight.

[90] *A perceptron is a single-layer neural network that consists of input values, weights, bias, net sum followed by an activation function.* Available at: https://www.javatpoint.com/perceptron-in-machine-learning

The neuron stores the weighted sum of all the input variables adding to the result a possible value called bias, whose effect is to translate the value of the activation function and to help better fit the data. These weights are computed in the training phase of the neural network through techniques called gradient descent[91] and backpropagation[92]. The neural network training procedure is used to find and set the values of the individual weights and biases of each neuron. Once that the inputs are multiplied by their linked weight, they are then sent on to a non-linear function, known as an activation function. A neuron uses it to add non-linear properties in the network. A non -linear relationship suggests that a change in the first variable doesn't necessarily result with a constant change in the second. The introduction of non-linearity allows to better identify patterns in data. The activation function is used to determine whether a neuron should be activated or not. The input signals are generated by other neurons i.e. the output of other neurons, and the network is built to make predictions/computations in this manner. Therefore, at the end of the process, by applying a known input to the network, an output should be obtained that is as close as possible to the real one. Then, this output value is transmitted on the subsequent layer through another synapse. This is the basic idea of a neural network (Figure 2.3).

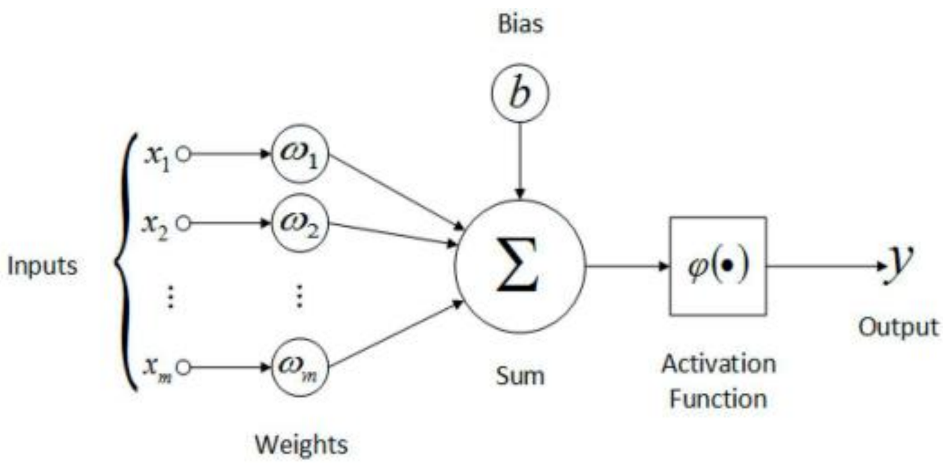

Figure 2.3: Stereotypical neuron in deep learning models

[91] *Gradient descent (GD) is an iterative first-order optimisation algorithm used to find a local minimum/maximum of a given function. This method is frequently used in machine learning and deep learning to minimise a cost/loss function.* Available at: https://developer.nvidia.com/blog/a-data-scientists-guide-to-gradient-descent-and-backpropagation-algorithms/

[92] *Backpropagation is the method by which components that affect the output of a neuron (bias, weights, activations) are iteratively modified to reduce the cost function.* Available at: https://developer.-nvidia.com/blog/a-data-scientists-guide-to-gradient-descent-and-backpropagation-algorithms/

2.3 Artificial Neural Networks

The set of multiple perceptrons forms a layer having as its input, n input, and as its output, a number of outputs equal to the number of neurons that compose the final layer. Finally, intermediate layers of nodes can be combined to build an Artificial Neural Network (ANN). The latter represents a complete neural network, which contain at least three different levels: an input one, one or more hidden layers formed by a variable number of neurons, and an output level consisting of a series of outputs equal to the number of neurons which form the layer itself, as shown in Figure 2.4[93]. Any network with more than one hidden layer is referenced as a Deep Neural Network (DNN).

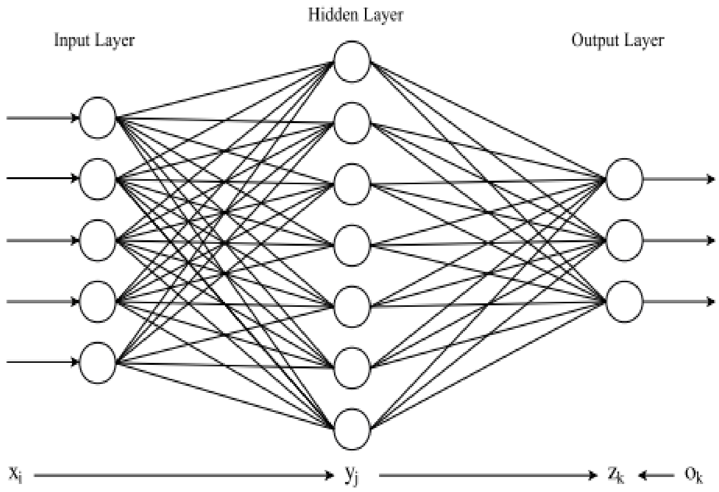

Figure 2.4: Simple artificial neural network

Each layer is fully connected to the next layer in the network, i.e. each unit belonging to a layer is linked to all the units belonging to the two adjacent layers. As they are connected through links, they interact by taking the data and performing operations on it and then, passing the results of these operations over to the other connected node. Each link between the nodes is associated with a certain weight and threshold.

The weight is a real value which controls the signal between two nodes and interestingly, if a node's output is above the specified threshold value settled by the

[93] A. GÉRON, *Hands-on machine learning with Scikit-Learn, Keras, and TensorFlow: Concepts, tools, and techniques to build intelligent systems*, O'Reilly Media, pp. 286-289, 2019.

threshold function, it triggers that node sending data to the next level of the network. This mean that the output of one node becomes the input of the next node. If a network generates a “good or desired” output, there is no need to adjust the weights. However, if the network generates a “poor or undesired” output, then the system alters the weights in order to improve subsequent results. In short, links between the nodes have the power to alter the weight according to examples and previous experiences.

The learning process of an Artificial neural network is performed during the backpropagation phase in combination with an optimization algorithm, with the aim of adjusting all the weights of the network after having calculated the cost function. A cost function is an essential parameter in determining how well the neural network model performs for a given dataset. Since it computes the difference between the expected value and predicted value and represents it as a single real number, it is used to calculate the accuracy in a network's prediction. Hence, the goal of the neural network is to reduce this function as much as possible. To perform this operation, a gradient descendent rule or other relevant optimization algorithms can be used to update the weights. Gradient descent involves analysing the slope of the curve of the cost function. Based on the slope, the weights are adjusted to minimise the cost of function in steps rather than computing the values for all possible combinations.

Initially, the weights and biases of the network are assigned randomly, so in the first stage, where the data is inserted at the input level and transmitted to the output level, the network simply implements a series of more or less random transformations. Consequently, the network response will be far from the expected value, and naturally, at the same time, the loss score will be very high. Then, repeating the training loop a sufficient number of times, the neurons will have the ideal weight values, which minimize the loss function[94].

Over the years, different forms of Artificial Neural Network have been developed. Among these, the ones that have found more success thanks their use case are:

- *Feed-forward Neural Networks* (FNNs): are the first and the simplest form of ANNs, where input data travel in one direction, passing from the input

[94] E. GROSSI, M. BUSCEMA, *Introduction to artificial neural networks*, January 2008. Available at: https://www.researchgate.net/publication/5847739_Introduction_to_artificial_neural_networks

layer directly through hidden layers to the output nodes without forming cycles or loops in the graph representing the network. Based on the presence or not of the hidden layers, they can be further classified into single-layered or multi-layered feed-forward neural network. Some of the applications of Feed-forward neural networks are found in face recognition, computer vision and speech recognition.

- *Convolutional Neural Networks* (CNNs): are a type of feed-forward neural network, which take advantage of the principles of linear algebra to recognize patterns. Being suitable for processing visual and two-dimensional data allows CNNs to distinguish by superior performance with image, speech, or audio signal inputs. Hence, most of the applications of CNNs are focused on image processing, computer vision, speech recognition and machine translation. Instead of a standard two-dimensional array, convolutional neural network contains a three-dimensional arrangement of neurons.
- *Recurrent Neural Networks* (RNN): are a class of Artificial Neural network with hidden states based on recurrent computation. Planned to save the output of a layer, RNN is fed back to the input to help in predicting the outcome of the layer. The first layer is formed similarly to a feed forward neural network followed by recurrent neural network layer, where some information it had in the previous time-step is remembered by a memory function. Contrary to the CNNs, where it is imposed fix-length input, in the RNNs it is permitted to overcome these types of limitations. In fact, it is possible to use inputs of arbitrary length. In contrast with FNNs, where data directly moves from input to output nodes, the neurons in RNNs are connected in a loop through cyclic or recurrent connections among nodes of distinct levels. Applications of Recurrent Neural Networks can be found in text to speech processing, image tagger, sentiment analysis or translation.

2.4 Recurrent Neural Networks

Since the empirical projected proposed in this research is conducted using a type of Recurrent Neural Network, before going through the explanation of the LSTM model, an overview of the RNN approach is provided below. Recurrent Neural Networks are an evolution of feedforward neural network with the feature of having an internal memory, which gives the faculty to be one of the most promising algorithms. Many researchers have showed that lots of challenging problems were resolved with promising results by RNNs outperforming other artificial neural network architectures. All this mainly thanks to the increase in computing power and in the amount of available data, which has only recently allowed to demonstrate the RNN true potential[95].

Recurrent Neural Networks' primary purpose is to handle with dataset, which have a sequential relationship between inputs and outputs, like time series data[96]. Therefore, the output from the previous step is fed to the current step as an input. Such characteristic confers to RNNs the power to evolve machine learning from static to dynamic models predicting the outcome at the same current time-step, through the history of sequences and not only using data in a given time instant. Hence, thanks to their internal memory, RNNs can have a much deeper understanding of the input they receive, allowing to have a greater precision in predicting what will happen in the future.

Given that RNNs were inspired by Convolution Neural Networks and Feed-Forward Neural Networks, there are some similarities in the way they operate as visible in Figure 2.5.

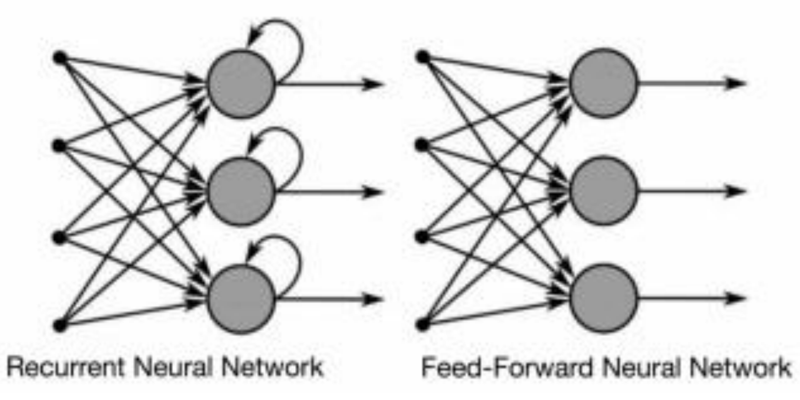

Figure 2.5: Comparison between Recurrent Neural Network and Feed-forward Neural Network.

[95] R.M. SCHMIDT, *Recurrent Neural Networks (RNNs): A gentle Introduction and Overview*, November 23, 2019. Available at: https://arxiv.org/pdf/1912.05911.pdf

[96] Y. SUGOMORI, *Java Deep Learning Essentials*, Packt Publishing Ltd, pp. 229-234, 2016.

They all use data to train and learn relations and patterns. However, one of the main distinctions is, while in FNNs data directly moves from input to output nodes, the neurons in RNNs are connected in a loop, through cyclic or recurrent connections among nodes of distinct level. These cycles allow the network's hidden units to see its own previous output, so they give the network memory and introduce the notion of time into the model. The structure of a simple recurrent neural network is shown in Figure 2.6.

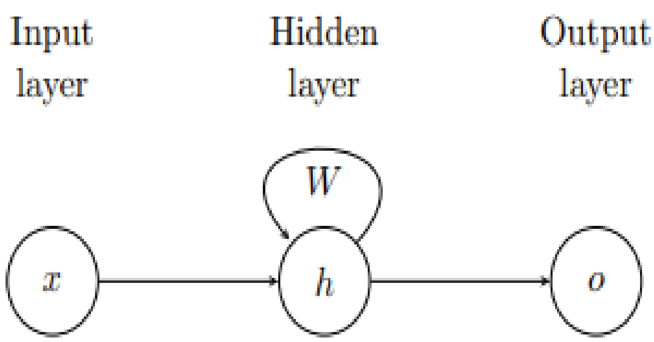

Figure 2.6: Simple Recurrent Neural Network with one hidden layer

Standard feed-forward neural networks only consider the current input, so they remember only information learnt along each single interaction in training. Differently, recurrent networks take as input, not just the current input example while training but, also what they have perceived previously in time. In fact, recurrent networks preserve sequential information based on prior inputs and outputs through a hidden state vector, which allows them to store information about the past efficiently (as in Figure 2.7).

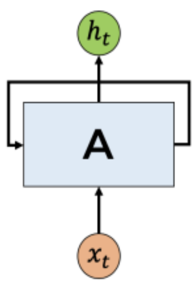

Figure 2.7: Illustration of a Recurrent neural network

This means that the same input at time t could produce a different output depending on prior inputs and outputs. The fact that a recurrent neural network could be represented by a cell unit with a finite state loop shows how they are not so different from standard feed-forward neural networks. Hence, RNN can be imagined as multiple copies of the same network, each of them with the same weights and biases. This feature made it possible to explain how to train the network through the "unfolding" concept, which simply means setting in advance the number of timesteps on which to perform the analysis. It consists in a transformation of a network from a RNN type to a feed-forward type (as in Figure 2.8).

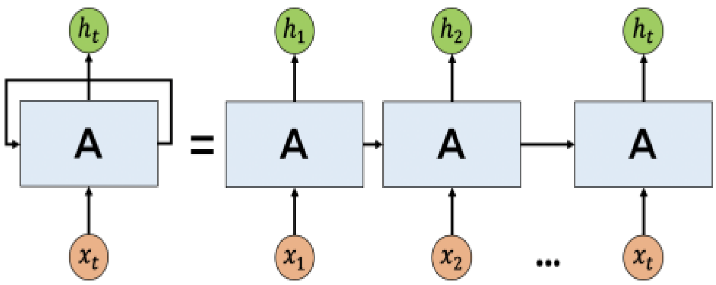

Figure 2.8: Recurrent neural network "unrolled"

Information moves through the network for a data sequence where inputs are acted on by the hidden state of the cell to produce the output and the hidden state is passed to the next time step. In this way, recurrent networks share parameters across each layer of the network. But, while in FNNs the weights across each node are different, in RNNs they are the same within each layer of the network.

To facilitate the learning process, these weights are adjusted through the backpropagation and gradient descent methods. The most used algorithm to determine the weights derivatives for RNN is the Backpropagation Through Time (BPTT) algorithm, which is slightly different from traditional backpropagation, as it is specific to sequence data such as time series. The principles of BPTT are the same as traditional backpropagation, where the model trains itself by calculating errors from its output layer to its input layer. These calculations allow to adjust and fit the parameters of the model appropriately. BPTT differs from the traditional approach as BPTT sums errors at each time step, while feedforward networks do not shape parameters across each layer. The way BPTT operates is by unfolding the RNN in time through the unrolling of all input timesteps. For each timestep there is one input timestep, one copy of the network, and one output. After that the errors are computed and accumulated for each timestep, the network is rolled back up and the weights are updated. Given that the problem is order dependent and that the internal state from the previous timestep is used as an input for the next, each timestep of the unrolled RNN may be viewed as an extra layer. A weakness of BPTT to consider is it can be computationally expensive when using a large number of timesteps.

In addition, a more general problem associated with basic RNNs is the weakness of long-term dependency, which consists in their inability to recall old data, which worsens as time passes. While RNNs are very useful to model a current input through previous information, they are not always able to explain this relation. In other words, this network cannot work appropriately in long data series. This kind of issue is also known as the *Exploding gradient* or *Vanishing gradient* problems[97]. The occurrence of one problem or another depend on the magnitude of the gradient. Exploding gradient occurs when the gradient is too big, creating a very unstable model. In this case the weights of the model become too large. On the other hand, Vanishing gradient occurs when the gradient is too small. Its value is reduced more and more, going to update the values of the weights until they become insignificant[98]. When this happens, the algorithm can no longer learn.

Different methods can be used to reduce the effect of Exploding and Vanishing gradient. One solution proposed to cope with these problems is the change of the structure of the model. A common substitution in type is the Long Short-Term Memory (LSTM), a neural network capable to use long term memory[99].

---

[97] *The problem of the vanishing gradient was first discovered by the computer scientist Sepp Hochreiter in 1991 and a bit later by Yoshua Bengio, a professor at the University of Montreal in 1994. Both contributed significantly to the research in Deep learning and RNNs field.*

[98] S. HOCHREITER, *The vanishing gradient problem during learning recurrent neural nets and problem solutions*, International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems, vol. 6, no. 02, pp. 107–116, 1998.

[99] S. HOCHREITER, J. SCHMIDHUBER, *Long short-term memory*, Neural computation, vol. 9, no. 8, pp. 1735–1780, 1997.

2.5 Long Short – Term Memory Networks

As mentioned above, RNNs are capable to learn and consider trend and context when training. This thanks to their feedback loops, that allow information to persist with the use of both current and past inputs. However, RNNs have one major limitation, they used to lose memory in the long run. To overcome the backpropagation error and the vanished gradient problem, Hochreiter and Schmidhuber introduced the LSTM network in 1997[100]. From this date on, it was widely applied in time series prediction areas[101], recently including cryptocurrencies price prediction[102].

2.5.1 LSTM Architecture

The LSTM term is linked to the ability of the architecture to identify patterns in both short and long term[103]. With this new type of structure, it was possible to reach a longer horizon of lags with faster learning time. This advancement has allowed to capture longer potentially unknown dependencies in time-series data[104]. With the implementation of additional feedback connections respect to the feedforward ones, the LSTM structure guarantees to predict future data while maintaining significant memory of the previous series. In other words, preserving prior information, the LSTM is able to self-improve its capacity to learn signal sequences and inherent patterns.

Since it is composed of an input layer, one or more hidden layers and an output layer, the LSTM has a structure that is very similar to an RNN-style architecture. The difference is in hidden layers, which consist of special blocks called *memory blocks*. In these resides the "engine" of the LSTM networks in effectively facing long-term dependencies. Memory blocks contain memory cells with recurrent connections, that

[100] S. HOCHREITER, J. SCHMIDHUBER, *Long short-term memory*, Neural Computation, pp. 1735–1780, 1997.

[101] H.Y. KIM, C.H. WON, *Forecasting the volatility of stock price index: A hybrid model integrating LSTM with multiple GARCH-type models*, Expert Systems with Applications, pp. 25-37, 2018.

[102] T. PHALADISAILOED, T. NUMNONDA, *Machine Learning Models Comparison for Bitcoin Price Prediction*, 10th International Conference on Information Technology and Electrical Engineering (ICITEE), Kuta, pp. 506-511, 2018.

[103] S. HOCHREITER, J. SCHMIDHUBER, *Long short-term memory*, Neural Computation, pp. 1735–1780, 1997.

[104] P. SRIVASTAVA, *Essentials of Deep Learning: Introduction to Long Short-Term Memory*, December 10, 2017. Available at: https://www.analyticsvidhya.com/ blog/2017/12/fundamentals-of-deep-learning-introduction-to-lstm/

allow to preserve the temporal state of the process, effectively acting as a “memory” of the network. Figure 2.9 illustrates the structure of a LSTM memory cell[105], where:

1. X_t: is the current input at time t
2. C_t: is the new update memory, hence it represents the long-term memory of the neuron. C_{t-1} contains the real information available at instant $t-1$ and together with the state h_{t-1}, it is used at time t to determine the new information C_t.
3. h_t: is the current output and it acts as a short-term memory. Similarly to that of RNNs, it represents the hidden state of the neuron. h_{t-1} is the output of the last LSTM unit and it is combined within the various gates[106] to determine useful information to be taken from long-term memory.

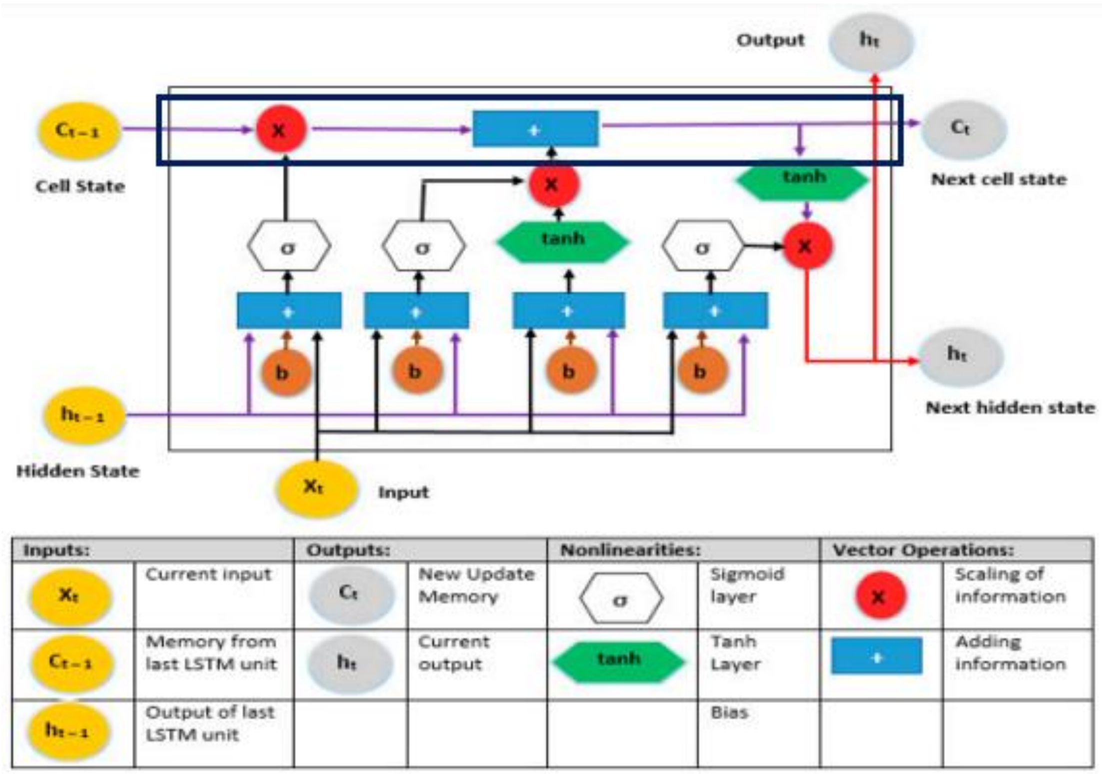

Figure 2.9: The structure of a long short-term memory algorithm.

---

[105] X.H. LE, H.V. HO, G. LEE, S. JUNG, *Application of Long Short-Term Memory (LSTM) Neural Network for Flood Forecasting*, Water, pp. 1387, 2019.

[106] *Gates are comparable to neural network nodes that determine which information is allowed to enter the cell state. The gates have their weights, and with the use of a recurrent neural network learning process, it will decide which information is important to maintain or forget during training. They consist of a sigmoid neural net layer and a point-wise multiplication operation.* Available at: https://www.ssla.co.uk/long-short-term-memory/

The heart of the LSTM operation resides in the cell state[107] marked by the upper blue rectangle, that goes from the "cell state" on the left to the "next cell state" on the right. It allows to preserve the information along the entire LSTM neuron, despite the interaction of the sequence processing. As the cell sate proceeds on its path within the neuron, it encounters various inputs and outputs, that permit to remove or add information on its state. The function of removing or adding information is controlled by the so-called gates. Their task is to decide which information must be saved or deleted from the state during the training phase. At every time step t, a transformation function decides what information to remove from the cell sate. The Sigmoid layers, represented by the white hexagons, set the amount of each component in the cell state C_{t-1} that must pass. This thanks to the normalization of the values between 0 and 1. An output with a value equal to 1 means that all information is let through, while a value equal to 0 means that no information is passed[108][109]. The other activation function is present in the $tanh$ layers marked by the green hexagons. The hyperbolic tangent function is responsible to product values between -1 and $+1$ used to regulate the network. In fact, in order to prevent the explosion of the magnitude of the data, mathematical operations are concerned with transform the values that are introduced into the network. The graph of both these functions can be seen in Figure 2.10.

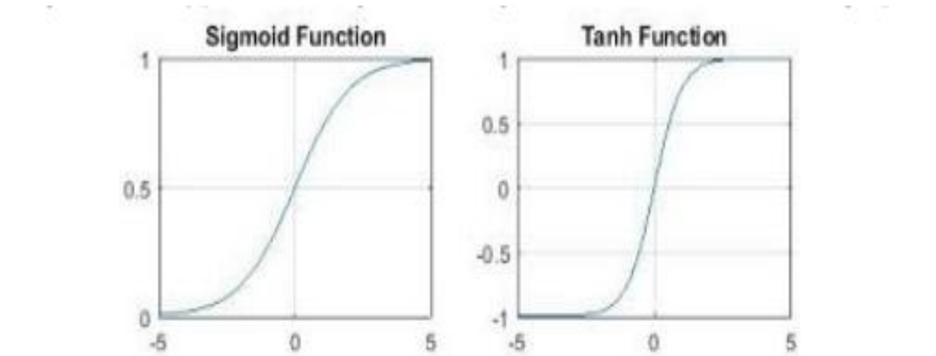

Figure 2.10: Hyperbolic tangent and Sigmoid activation functions graphs

[107] *Cell state is a memory of the LSTM cell and it essentially encodes the information of the inputs (relevant info.) that have been observed up to that step (at every step).* Available at: https://medium.com/@humble_bee/rnn-recurrent-neural-networks-lstm-842ba7205bbf#:~:text=C-ell%20state%20is%20a%20memory,state%20and%20no%20cell%20state.

[108] C. OLAH, *Understanding LSTM Networks*, 2015. Available at: http://colah.github.io/posts/2015-08-Understanding-LSTMs/

[109] T. FISCHER, C. KRAUSS, *Deep learning with long short-term memory networks for financial market predictions*, European Journal of Operational Research, vol. 270, no. 2, pp. 654-669, October 16, 2018.

In order to protect and control the cell state, an LSTM has three types of gates: the forget gate, the input gate and the output gate[110]. They are essential to manage the cell state and to control the flow of information into and out of the cell. The three gates allocation is illustrated in Figure 2.11[111].

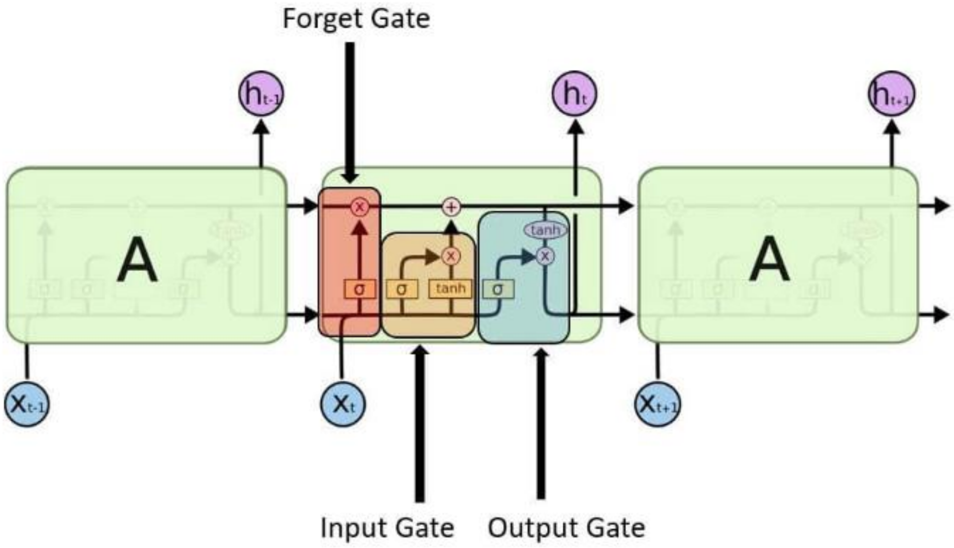

Figure 2.11: Illustration of Forget Gate, Input Gate and Output Gate

2.5.2 Forget Gate

After that the input parameters have entered the network, the first gate they encountered is the Forget layer. The parameters into account are two, x_t , the input at time t and h_{t-1}, the output at time $t-1$. The forget gate determines which information, coming from instant t (C_{t-1}) to use at instant $t-1$. The latter is then combined with the current input and next pass through the sigmoid function. The Sigmoid activation function generates an output with values between 0 and 1 which multiply element by element in the C_{t-1} vector. f_t, a vector with the same length as C_{t-1}, determines each element of C_{t-1} should be multiply by which number between 0 to 1. Hence, it allows to establish which information, originating from long-term

110 H.H. GOH, R. HE, D. ZHANG, H. LIU et al., *Short-term wind power prediction based on pre-processing and improved secondary decomposition*, Journal of Renewable and Sustainable Energy, 2021.

111 T. FISCHER, C. KRAUSS, *Deep learning with long short-term memory networks for financial market predictions*, European Journal of Operational Research, vol. 270, no. 2, pp. 654-669, 2018.

memory at the instant $t-1$, must be reused at the current time. The following equation shows the formula which describes how the forget gate inputs are combined.

$$f_t = \sigma(Wf \cdot [h_{t-1}, x_t] + bf)$$

where f_t denotes the activation function of the forget gate, W denotes the weight matrix, b denotes the bias vector and σ denotes the sigmoid function[112].

2.5.3 Input Gate

The Input Gate is used to update the state of the cell determining which of the input values should be used to change the memory. As for the Forget Gate, the information of the previous output and that of the current input are passed through the sigmoid function. It determines whether to allow 0 or 1 values through, thereby deciding which values must be updated. The input data are also transformed through the tanh function, which assigns weights to the data provided, determining their importance on a scale of -1 to 1. $tanh$ layer creates a vector of new candidate values that could be potentially added to the state. Then the combination of these two outputs generates the update to the state cell.

$$i_t = \sigma\left(W_f \cdot [h_{t-1}, x_t] + b_i\right)$$
$$\tilde{C}_t = tanh(W_c \cdot [h_{t-1}, x_t] + b_c$$

where i_t denotes the activation of the input gate.

Then, the old cell C_{t-1} is updated by both f_t and $i_{t^*}\tilde{C}_t$ simultaneously. The update is made by multiplying the old state by f_t forgetting things decided to forget earlier, and then adding new information through $i_{t^*}\tilde{C}_t$.

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

2.5.4 Output Gate

It is the final operation which decides what the output of the current iteration will be. To do this, it enlists the help of the block's input and the memory. First, the

[112] X. LI, L. PENG, X. YAO, S. CUI, Y. HU, C. YOU, T. CHI, *Long short – term memory neural network for air pollutant concentration predictions: Method development and evaluation*, Environ, Pollut, 231, pp.997-1004, 2017.

information of the previous output and those of the current one is transformed by a sigmoid function, which decides what part of the cell state has to be outputted. The signal exited from the sigmoid function is then multiplied by the cell state passed from a $tanh$ function. In this way, it is decided what information the output status should contain.

The first formula shows how the gate output (o_t) is formed, and the second equation explains how the output of h_t is calculated.

$$O_t = \sigma(w_0 \cdot [h_{t-1}, x_t] + b_0)$$

where o_t denotes the activation of the output gate.

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

This process helps to output only those inputs that are decided to update. Finally, this output, o_t and the current memory state, S_t, will be passed to the next LSTM cell as it's inputs, as shown in Figure 2.12.

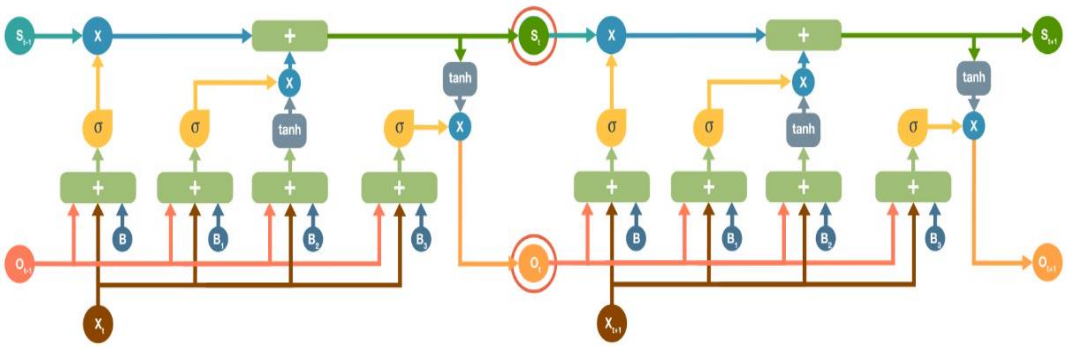

Figure 2.12: interaction model between two LSTM cells

Therefore, a LSTM cell can set up these three gateways for the management of the information flow in and out of the cell. In this way, in a time-series data prediction task, it can recall memories at arbitrary time intervals. The use of the output of each LSTM cell as the input of the next cell allows the status of one cell to affect the way the next cell works. Hence, thanks to the ability of the LSTM neural network to control long-term memory, it turns out to be more durable than the RNN neural network.

Chapter 3
MACHINE LEARNING PIPELINE

In a typical data science scenario, a machine learning pipeline illustrates the entire process from data collection to model optimization. In this chapter, it is detailed every stage of the data pipeline except the LSTM model building and model optimizations stages, which are covered in chapter 4.

3.1 Research Architecture

The purpose of this research is to forecast the price of five different cryptocurrencies through a Long Short-Term memory approach and evaluate the accuracy of the prediction. The cryptocurrency market is heavily influenced by a multitude of variables, which makes digital assets extremely volatile. However, thanks to the volume of data available for the digital assets selected, a detailed study using the proper neural network algorithm model will provide a precise price prediction. Therefore, the empirical project proposed is focused on price and consider it as a variable for forecasting. For this goal, a LSTM model is developed. The latter does not predict the future, but it can suggest a broad trend and the overall direction in which to expect the price movement of the cryptocurrencies utilized.

The methodological approach, which is used to create this thesis' practical part, is divided in different stages. The whole process starts from collecting data from an online market data provider, in this case Yahoo Finance. The dataset includes five cryptocurrency close prices based on USD currency with a daily frequency. The historical data prices go from the 1st of January 2021 to the 1st of September 2022.

Filtering and cleaning the data set is the second phase. This step involves removing and replacing all the missing, empty, and non-matching data from the rows. As the model requires only the close column labelled, it is also needed to drop out unnecessary variables present in the data collected.

The next stage is the data exploration and visualisation process where, performing descriptive analysis, it is shown and explored the behaviour and distribution on data and the relationship between digital assets.

Then, normalisation is applied to improve the performance and training stability of the model. This is reached transforming features to be on a similar scale.

The further step is splitting the dataset into training and testing parts. In order to estimate the future price of cryptocurrencies, the model is trained using the algorithm and the features considered to assist the model. After testing data, we evaluated the accuracy of the algorithm that the LSTM model is using to forecast the price of the different digital assets selected.

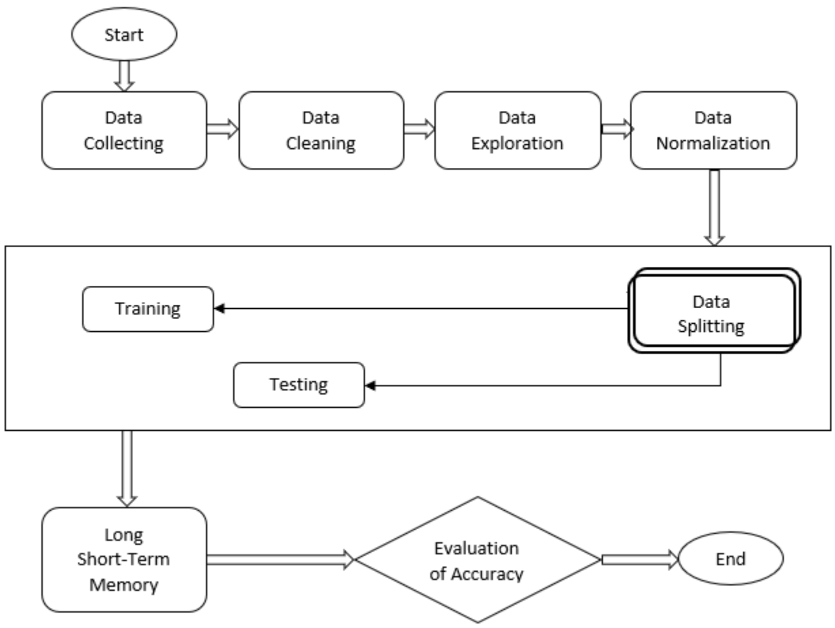

Figure 3.1: System Architecture of the proposed model

3.2 Cryptocurrency Selection

There are more than 20,000 cryptocurrencies in the market, with digital assets being born and dying every day, but regardless of the large number, a lot of problems remain unsolved. The first issue is data related; since many cryptocurrencies are still relatively new, there is a lack in history data, which does not allow a trustable and useful model building or price predicting. A second problem is that, although some digital assets are not highly ranking, they are market drivers based on their market capitalization[113].

One way to rank cryptocurrencies is using the market capitalization parameter, which is based on aggregate data obtained from different exchanges as Binance, Coinbase or Kraken. Then, the classification is displayed on public portals such as CoinMarketCap, CoinDesk and CoinGeko, where it is possible to find important information for each digital asset.

For the empirical part of the study, the choice of cryptocurrencies is made through CoinMarketCap[114], the most-referenced price-tracking website for this purpose. The selection is performed taking into consideration criteria such as market capitalization, particularities in the predominant characteristics and operation. Hence, in the data set are included cryptocurrencies which are not backed by any institutions or pegged on the value of another asset. Therefore, in this case, digital assets such as BNB are excluded, because it is issued by the largest centralised exchange Binance[115], or Thether and USD Coin are not included, since their prices are based on US dollars value. On the 1st of September 2022, the choice fell on Bitcoin, Ethereum, Cardano, Solana and Polkadot.

To give more context to the research, a brief description of the selected cryptocurrencies follows:

3.2.1 Bitcoin

Satoshi Nakamoto starts his white paper, "*Bitcoin: A Peer-to-Peer Electronic Cash System*", with the following statement, through which he explains the idea behind his creation: "*A purely peer-to-peer version e-money peer would allow*

[113] N. MALSA, V. VYAS, J. GAUTAM, *RMSE calculation of LSTM models for predicting prices of different cryptocurrencies*, Lulea University of Technology, Sweden, 2021. Available at: https://-link.springer.com/article/10.1007/s13198-021-01431-1
[114] CoinMarketCap [Online]. Available at: https://www.coinmarketcap.com
[115] *Centralized exchanges (CEXs) are organisations that coordinate cryptocurrency trading on a large scale, using a similar business model to traditional asset exchanges like stock exchanges.* Available at: https://www.coindesk.com/learn/what-is-a-cex-centralized-exchanges-explained/

online payments to be sent directly from one entity to another without going through a financial institution"[116].

For the first time, Bitcoin shows the solution to the problem of double spending[117], as well as that of Byzantine generals[118], using a global network, free of borders, open and decentralized, without any form of central authority.

First, it is necessary to distinguish two different concepts: Bitcoin and bitcoin. Bitcoin, with a capital "B", is the noun used to refer to the entire ecosystem of Bitcoin, with everything related to it:

- the decentralized peer-to-peer network:
- the distributed ledger, i.e. the blockchain;
- the mining process, which represents the model of generation and distribution of new money, as well as the remuneration system for miners;
- the decentralized verification protocols of the validity of transactions and blocks.

While bitcoin, with a lowercase "b", refers to the mere cryptocurrency, universally identified with the symbol ₿ and the BTC code[119].

Satoshi defines the Bitcoin open-source protocol as "a cryptographic and pseudonymous structure capable of both protecting the identity of users and facilitating exchanges of value through a decentralized and disintermediate monetary paradigm, therefore without any need for a fiduciary third party"[120].

---

116 S. NAKAMOTO, *Bitcoin: A Peer-to-Peer Electronic Cash System*, October 31, 2008. Available at: https://bitcoin.org/bitcoin.pdf

117 *The phenomenon of "double spending" occurs when a given amount of coins are spent more than once. Usually as a result of a race attack or a 51% attack.* Available at: https://academy.binance.-com/en/search?page=1&term=double%20spending%20

118 *The problem of the Byzantine generals, a computer problem, known in the field of distributed systems, which takes into consideration the search for an agreement through the use of messages between the nodes of a network.*

119 G. CHIAP, J. RANALLI, R. BIANCHI, *Blockchain, tecnologia e applicazioni per il business*, Hoelpi, Milano, pp. 93-94, 2019

120 G. COMANDINI, *Da Zero alla Luna, quando come e perché la Blockchain sta cambiando il mondo*, Flacovio, Palermo, p.44, 2020.

The main features of Bitcoin are:

- decentralization, since no type of central authority, from banks to governments, can control and manage the value and flows of bitcoin;
- security, given by the use of cryptography, which makes it impossible to access coins without authorization;
- transparency, for which it must be consider that transactions held in bitcoin are stored forever on the blockchain and are visible to all, in such a way as to allow the origin of cryptocurrencies to be established, to certify their existence and to associate them with an owner;
- speed, as it takes a few minutes to send and receive bitcoins, while for many types of payments it is necessary to wait even days;
- convenience, given its purely digital nature;
- being anti-fraud, as it solves the problem of double spending, solved through the Proof of Work consensus algorithm[121].

Within the Bitcoin ecosystem, transactions are held by carrying out transfers of cryptocurrencies, which take place through cryptographic keys, inextricably linked to a specific address created by the node.

There are two types of keys: the public key, useful for sending and receiving payments from other nodes, and the private key, necessary to provide payment authorization, which must be owned exclusively by the owner, the actual recipient of the transaction. This mechanism consists of asymmetric cryptography, which, in the specific case of the Bitcoin protocol, is combined with a Hash function: the SHA-256 algorithm.

As a peer-to-peer network, all nodes of the Bitcoin Blockchain are notified of every transaction held in the system. The bitcoin transfer requires the paying node to have the public key of the beneficiary to which to allocate the sum to be transferred. The transaction held is subsequently verified and validated, so that a node cannot spend its currency twice. The validation of transactions is the task

[121] G. COMANDINI, *Da Zero alla Luna, quando come e perché la Blockchain sta cambiando il mondo,* Flacovio, Palermo, p.44-45, 2020.

entrusted to specific nodes in the network, the miners, who, through their computational power, validate all the transactions held within ten minutes and bring them together in a single block of the chain[122].

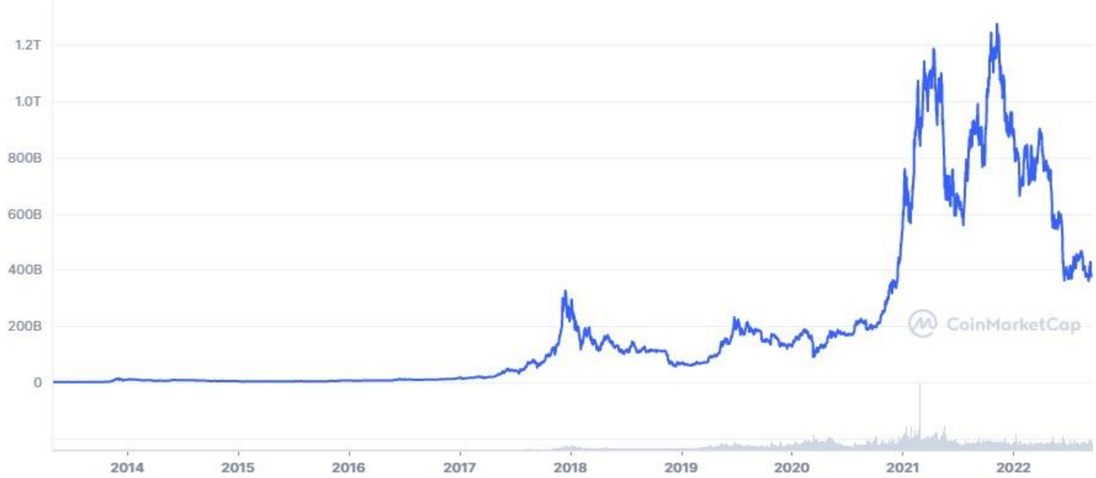

Figure 3.2: Bitcoin market capitalization

3.2.2 Ethereum

The creator of Ethereum is Vitalik Buterin, a Canadian developer who defines and publishes his project in 2013, describing its technical characteristics, first in a white paper, and then providing further specifications in a yellow paper, in 2014, while on 30 July 2015 it released the first version[123].

When we talk about Ethereum we refer to a blockchain with the aim of replacing the third parties of the Internet, dedicated to data storage, which supports the creation and execution of smart contracts.

"Ethereum can be seen as a global computer, in which programs, smart contracts, are executed in a decentralized, continuous and uncensored way"[124].

This platform allows you to move from the concept of distributed ledger to that of distributed computing, a worldwide virtual computer, which aims to

[122] Binance Academy, *What is Bitcoin (BTC)?*, August 29, 2022. Available at: https://academy.binance.-com/en/articles/what-is-bitcoin

[123] Coinbase Learn, *What is Ethereum?*, 2022. Available at: https://www.coinbase.com/it/learn/crypto-basics/what-is-ethereum

[124] G. CHIAP, J. RANALLI, R. BIANCHI, *Blockchain, tecnologia e applicazioni per il business*, Hoelpi, Milano, 2019, p.115.

decentralize the system based on the client / server model and it is composed of all the computers connected to the Ethereum network, from which, however, they are autonomous.

Ethereum turns out to be a transparent, safe, and reliable network, working everywhere. It is also a programmable blockchain, as, in addition to allowing predefined and standardized operations, it allows its users to create their own solutions and develop the most diverse decentralized blockchain applications (DApps), not limited to mere cryptocurrencies[125].

In fact, Ethereum is a generalized blockchain, known as a chain of blocks, that is devoid of a specific purpose, but programmable and adaptable through smart contracts. The Ethereum platform allows you to create dApps, applications running on a decentralized network or peer-to-peer network of computers that do not need the control of a central entity. The development of these applications is granted by Solidity, the programming language of Ethereum, through which it is possible to program smart contracts[126].

Ethereum has its own cryptocurrency, known as ether, or ETH. Similar to bitcoin, ether is a decentralized and equal cryptocurrency, based on cryptography and having a double functionality, as it represents both the computational power needed to produce smart contracts, and the reward for the users who make them[127].

Ether is divisible into *wei*, its smallest part, equal to one quintillionth of an ether, which serves as a unit of measurement in the context of gas. The term gas refers to the unit of measurement used when determining the amount of computational power required in order to perform an operation on the chain of blocks. Each operation supported on the blockchain has a predefined cost; for example, calculating a hash or even adding two numbers requires gas[128].

---

[125] G. Comandini, *Da Zero alla Luna, quando come e perché la Blockchain sta cambiando il mondo*, Flacovio, Palermo, pp. 78-79, 2020

[126] Ethereum Learn, *What is Ethereum? The foundation for our digital future*, Ethereum Foundation, 2022. Available at: https://ethereum.org/en/what-is-ethereum/

[127] G. Comandini, *Da Zero alla Luna, quando come e perché la Blockchain sta cambiando il mondo*, Flacovio, Palermo, pp. 78-79, 2020.

[128] G. Wood, *Ethereum: a secure decentralised generalised transaction ledger*, Yellow Paper, p. 26, 2014. Available at: https://ethereum.github.io/yellowpaper/paper.pdf

The concept of gas is purely related to transactions and each transaction sustained on Ethereum consists of two parameters: the price of gas, understood as the number of wei to be paid for each unit of gas, and the gas limit, i.e. the maximum quantity of gas consumable in one transaction[129].

The previous Ethereum system was particularly expensive, an aspect that led its developers to decide to improve scalability by migrating from a Proof of Work protocol to a Proof of Stake one. Hence, on September 15th at 06:42:42 UTC, at block number 15,537,393, when the Terminal Total Difficulty of the network reached a value of 58750000000000000000000, the Ethereum mainnet execution Layer merged with the Beacon Chain's consensus layer. In this way, the network abandoned the previous Proof-of-Work consensus system and switched to Proof-of-Stake.

Vitalik Buterin said Ethereum will be able to process 100,000 transactions per second after completing five key steps:

- The Merge
- The Surge
- The Verge
- The Purge
- The Splurge

The "Surge" will increase the scalability of the network through sharding, while the "Verge" will reduce the dependence on centralized entities. The "Purge" will eliminate historical data and technical debt, and finally the "Splurge" will introduce a series of small updates[130]

[129] Binance Academy, *What is Ethereum (ETH)?*, August 30, 2022. Available at: https://academy.binance.com/en/articles/what-is-ethereum

[130] B. LINDREA, *Breaking: Historic day for crypto as Ethereum Merge to proof-of-stake occurs*, CoinTelegraph, September 15, 2022. Available at: https://cointelegraph.com/news/breaking-historic-day-for-crypto-as-ethereum-merge-to-proof-of-stake-occurs

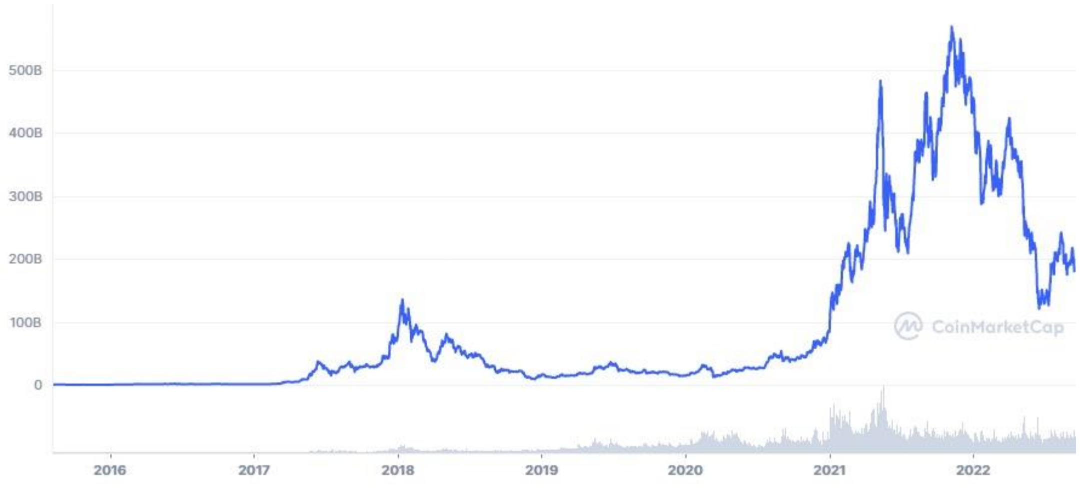

Figure 3.3: Ethereum market capitalization

3.2.3 Cardano

Cardano was founded by Charles Hoskinson, former co-founder of Ethereum, who launched the project in September 2017. Cardano is a blockchain platform for smart contracts, that offers a scalable and secure framework for its developers and users. It is based on a Proof of Stake system and is characterized by its peculiar development process, based on a philosophical approach and academic research[131].

Cardano is considered a third-generation cryptocurrency, with the aim of addressing and solving the problems inherent to the scalability, interoperability and security of the first generation (e.g. Bitcoin) and the second (e.g. Ethereum)[132].

Cardano programming is held with the Haskell language, which is based on mathematical principles and, therefore, it involves fewer human errors and easiness of verification.

The implementation of scalability is required by increasing throughput, through a consensus mechanism based on a specific Proof of Stake algorithm, Ouroboros, and bandwidth, using new types of networks.

[131] Coinbase Learn, *What is Cardano?*, 2022. Available at: https://www.coinbase.com/it/learn/crypto-basics/what-is-cardano

[132] C. TERENZI, *What Is Cardano (ADA)? – Guide About Everything You Need To Know*, Use The Bitcoin, 2019. Available at: https://usethebitcoin.com/what-is-cardano-ada-guide-about-everythin-g-you-need-to-know/

The Cardano protocol is developed on two distinct levels. The first, Cardano Settlement Layer, or CSL, where all the data relating to transactions are concentrated and on which the native tokens of the platform (ADA), are transferred[133].

Instead, Cardano Control Layer, or CCL, manages account and smart contract information, such as digital identities. The division into layers allows to efficiently make updates, separately and in a targeted manner, increasing security, since the compression of one layer does not affect the other.

Cardano has its own roadmap, consisting of five phases. The first is the Byron phase, in which users are allowed to exchange and transfer ADA and where the Cardano *mainnet* was launched[134]. Subsequently, there is the Shelley phase, where it is ensured that the technology is suitable to become an integrally decentralized and autonomous system. In the Goguen phase, the integration of smart contracts is expected. The fourth, Basho, aims at the general improvement of performance and in the last phase, Voltaire, it will be integrated a treasury and governance system that will make the protocol completely decentralized.

Cardano uses a proof-of-stake protocol, which allows developers to precisely control under what conditions a user can become a stakeholder. By becoming a stakeholder, it is assigned a role of control and verification function of transactions. Obviously, the more stakeholders confirm transactions, the more are successful. Finally, the system rewards stakeholders for checking blocks[135].

---

[133] Cardano Foundation, *What is ADA?*, 2022. Available at: https://cardano.org/what-is-ada/

[134] *"Mainnet" is the term used to describe when a blockchain protocol is fully developed and deployed, meaning that cryptocurrency transactions are being broadcasted, verified, and recorded on a distributed ledger technology (blockchain).* Available at: https://academy.binance.com/en/glossary-/mainnet

[135] Binance Academy, *What is Cardano (ADA)?*, June 2021. Available at: https://www.coinbase.com/it-/learn/crypto-basics/what-is-cardano

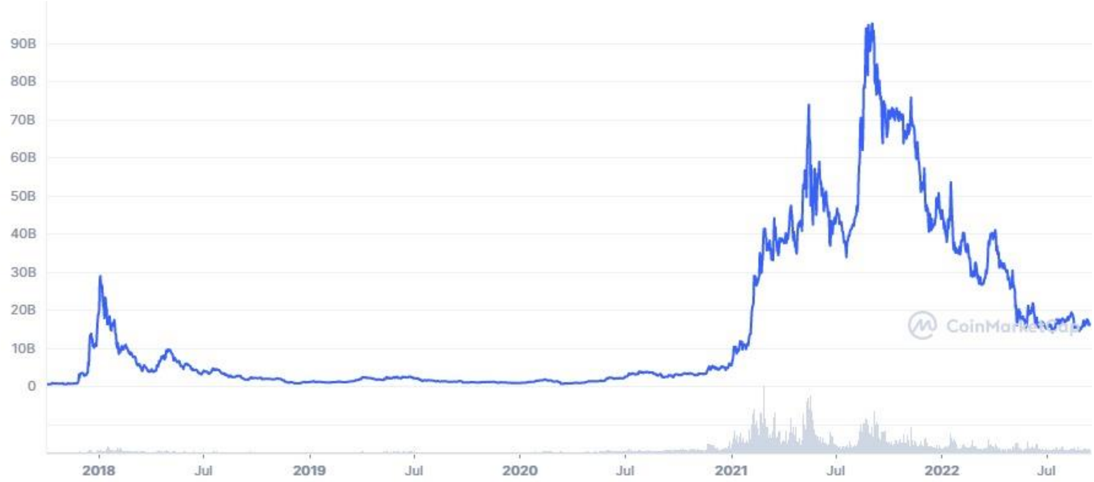

Figure 3.4: Cardano market capitalization

3.2.4 Solana

The Solana single-layer protocol was created with the aim of simplifying the creation of decentralized applications, or dApps. A distinctive feature of Solana is the incredible speed of processing times offered by its blockchain: it can manage up to a maximum of sixty thousand transactions per second and provides for the creation of a block in four hundred milliseconds[136].

Its system of fees is fixed and does not increase based on time and operations exercised, as in Ethereum. The Solana project has been conceived since its inception with low transaction costs, an aspect that must not in any way affect the scalability of the network and its processing speed[137].

Its peculiarity is the combination of two different consensus algorithms: Proof of Stake and Proof of History. In general, the Proof of History is a high-frequency delay verifiable function, i.e. a function that needs a certain number of sequential steps in order to evaluate and produce a unique and reliable result that is subsequently made public. In the case of Solana, it is employed a function that

[136] C. COSTA, *Solana: The 65,000 TPS Blockchain: Warp Speed*, June 26, 2020. Available at: https://coreycrypto.medium.com/solana-the-65-000-tps-blockchain-warp-speed-ab34d3ebb85c

[137] Coinbase Learn, *What is Solana?*, 2022. Available at: https://www.coinbase.com/it/learn/crypto-basics/what-is-solana

uses a sequential hashing system, capable of resisting ready-made hash images, or pre-images[138].

Solana's native token, SOL, is used to pay transaction fees when making transfers or interacting with smart contracts developed on the platform and for staking. It also gives holders the right to vote in future upgrades.

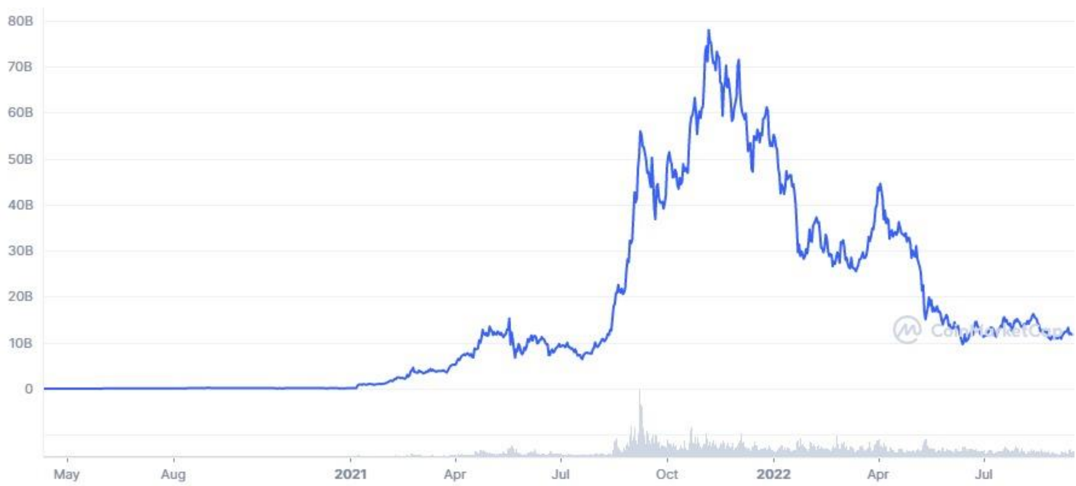

Figure 3.5: Solana market capitalization

3.2.5 Polkadot

Polkadot is a scalable blockchain platform, founded by Gavin Wood, former co-founder of Ethereum, which guarantees interoperability between different networks and security for the connection between different chains[139].

The implementation of scalability makes it possible to accommodate an increasing number of projects on the network, to ensure increasing connectivity, without recording an increase in consumption and costs and without any slowdowns. Unlike other projects that aim at the interconnection between blockchains, Polkadot allows the transmission of raw data, i.e. non-tokenized data, a particularly important aspect for the creation and development of smart contracts through information that comes from external sources.

[138] Binance Academy, *What is Solana (SOL)?*, June 2021. Available at: https://academy.binance.com/-en/articles/what-is-solana-sol

[139] Binance Academy, *What is Polkadot (DOT)?*, June 2021. Available at: https://academy.binance.-com/en/articles/what-is-polkadot-dot

It is structured as a shared multichain: the platform can process any information and any transaction using the support of parallel blockchains. This scheme allows to prevent any possible bottleneck, reason of slowness and cost of old generation blockchains, and to be able to have a potentially infinite scalability.

Unlike past blockchains, which are unable to adapt to purposes other than those for which they were created, Polkadot has a flexibility so-called "by design". In this sense, parallel blockchains, or parachains, can be developed on Polkadot, i.e. sovereign blockchains that are nevertheless integrated into the main project, operating according to their own rules and adaptable to its needs. Polkadot allows, in addition to communication between blockchains, the development of native projects, which are however naturally integrated with the central block chain.

To take advantage of the network nodes, their validation capacity and their security, the platform provides for costs, the payment of which takes place through DOT, the native token of Polkadot.

Furthermore, the DOT is not an instrument dedicated to the exchange of value, which can be used in payments for services or goods; it is designed to be used only within the Polkadot block chain.

The functioning of Polkadot is developed on three fundamental operational elements:

- *Relay chain*: the engine of the platform, on which its security, the functioning of consensus and the cross-chain interoperability of the network depend.
- *Parachain*: parallel blockchains of any type, which can have their own native tokens and optimize their functionality based on individual use cases.
- *Bridges*: to be understood literally as bridges, with which the parachains can connect and communicate with external networks, such as Bitcoin and Ethereum[140].

[140] Polkadot.network [Online]. Available at: *https://polkadot.network/technology/*

In order to manage the large amount of data entered on the Polkadot platform, strict consent rules are required.

In this sense, different types of nodes are identified, each having a specific role. They are distinguished:

- *Nominators*: guarantee the safety of the relay chain through the selection of validators and staking DOT,
- *Validators*: guarantee the safety of the relay chain through the staking of DOT, validate the collators' tests, and participate in the definition of the agreement with the other validators,
- *Collators*: they keep the fragments of the transactions operated by the users and produce evidence for the validators,
- *Fishermen*: a role that can be covered by collators and parachain full nodes, consists of monitoring the network and reporting any incorrect behavior by validators[141].

As can be guessed, Polkadot is not based on the Proof of Work consensus algorithm. It employs a Nominated Proof of Stake algorithm, a variant of the Proof of Stake, in which a process is implemented by which the validators that can take part in the consensus protocol of the blockchain are selected.

To conclude, it can be said that Polkadot represents the guarantor figure within the ecosystem of the blockchains involved in the network, which allows the operation of transactions in a secure, cross-chain and stable manner.

[141] Coinbase Learn, *What is Polkadot?,* 2022. Available at: https://www.coinbase.com/it/learn/crypto-basics/what-is-polkadot

Figure 3.6: Polkadot market capitalization

3.3 Data Understanding

Before starting with the creation of the LSTM model, it necessary to ensure data gathered must be perfectly in line with the requirements of the business or the research. If not, there is a chance that the information collected and analysed will not meet the established standards. Therefore, it is crucial that the data collected are completely understood and clear from the beginning. In order to address the demands of data interpretation, a step-by-step procedure is used. First, it is important to confirm that the data acquired will support goals and plans; occasionally, data collection can reveal the need to revise business strategies. The second step is to make sure that the data obtained is thoroughly examined to use the right formats, samples, and dates. The third step is to guarantee the data quality is verified. These procedures will ensure that the data interpretation process is suitable, and that the analysis produces result that are beneficial to the empirical project.

3.4 Data Gathering and Cleaning

The dataset is entirely collected from the Yahoo Finance database, but in case there would have been missing data, they would be gathered from CoinMarketCap. In this specific case there was no need, as there were no cases of missing data, empty data, and non-matching data. Missing data means data which has not been collected at a daily time interval; empty data refers to data collected at a daily frequency, but it is empty; non-matching data is data that does not perfectly fit every time day period. For all data sets if NaN values are found to be existent, they are replaced with the mean of the respective attribute. In order to have an overview of all fluctuations of digital assets selected, the dataset collection period covers the maximum period of time made available by the data provider, until the day this study was done, which is the 9th of September 2022. The data is acquired with a daily frequency and cryptocurrency prices are all expressed in dollars.

The input parameters gathered are the common features: open price, high price, low price, close price, and volume at each time epoch (i.e. every day). Table 3.1 shows the characteristic of the dataset used:

No	*Features Name*	*Type*
1	Date	Object
2	Open	Float
3	High	Float
4	Low	Float
5	Close	Float
6	Adj Close	Float
7	Volume	Int

Table 3.1: dataset info

While Table 3.2 illustrates an example of the collected raw BTC price data from September 17, 2014 to September 01, 2022.

No	*Date*	*Open*	*High*	*Low*	*Close*	*Adj Close*	*Volume*
1	2014-09-17	465.86	468.17	452.42	457.33	457.33	21,056,800
2	2014-09-18	456.86	456.86	413.10	424.44	424.44	34,483,200
3	2014-09-19	424.10	427.84	384.53	394.80	394.80	37,919,700
4	2014-09-20	394.67	423.30	389.88	408.90	408.90	36,863,600
5	2014-09-21	408.08	412.43	393.18	398.82	398.82	26,580,100

Table 3.2: Bitcoin OHLCV data example (2014-09-17; 2022-09-01 dataset)

The date feature is the current date of the transaction on the bitcoin market. The open feature is the opening price of bitcoin on that date. The high feature is the highest bitcoin value that occurred on that day. The low feature is the lowest bitcoin value that occurred on that day. The close feature is the closing price of bitcoin on that date. The adj close attribute is an adjusted closing share price. The volume feature is the number of bitcoins buying and selling transactions that occurred on that.

The same procedures are conducted for the period considered for the LSTM model building. In this case from January 01, 2021 to September 09, 2022 (the exact day in which this study is done). Since the price of bitcoin and the other cryptocurrencies have extremely fluctuated along their existence, a limited time series is considered in order to avoid problems related to high oscillations.

No	*Date*	*Open*	*High*	*Low*	*Close*	*Adj Close*	*Volume*
1	2021-01-01	28994.01	20600.63	28803.59	29374.15	29374.15	40,730,301,359
2	2021-01-02	29376.46	33155.17	29091.18	32127.27	32127.27	67,865,420,765
3	2021-01-03	32129.41	34608.56	32052.32	32782.02	32782.02	78,665,235,202
4	2021-01-04	32810.95	33440.22	28722.76	31971.91	31971.94	81,163,475,344
5	2014-01-05	31977.04	34437.59	30221.19	33992.43	33992.43	67,547,324,782

Table 3.3: Bitcoin OHLCV data example (2021-01-01; 2022-09-01 dataset)

3.5 Data Exploration

To analyse and understand the story that data tells, it is helpful and crucial to comprehend the data distribution and behaviour using consistent and relevant charts. Figure 3.7 shows the example of a time series distribution of the targeted cryptocurrencies from the first historical price which Yahoo Finance makes available to the 1st of September 2022. OHLC prices are plotted in each graph to provide a deeply overview of fluctuations over the selected period.

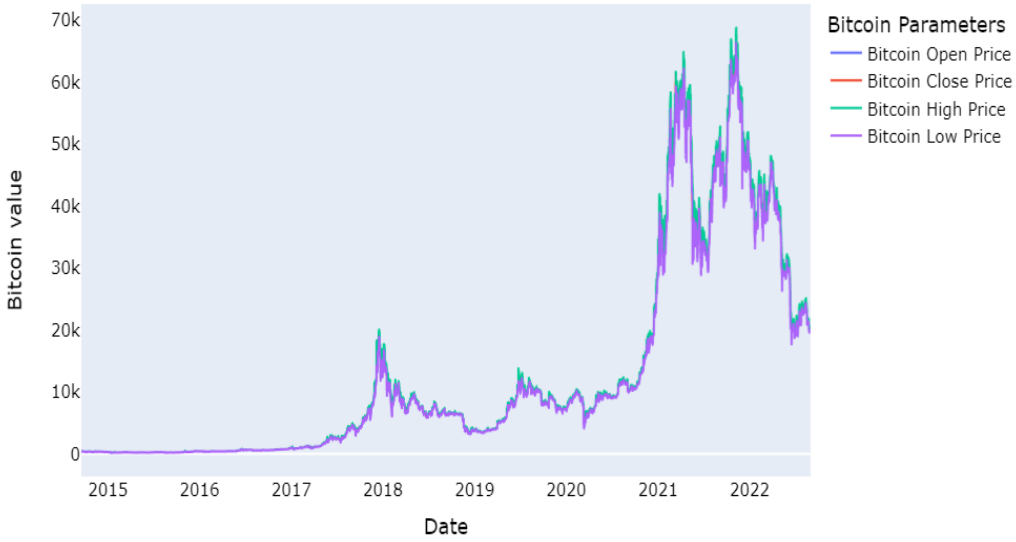

Figure 3.7: Bitcoin chart from 2014-09-17 to 2022-09-01

A similar data analysis is conducted for a shorter period, the one selected for the LSTM model building. Hence, Figure 3.8 shows the Bitcoin closing price development from the 1st of January 2021 to the 9th of September 2022.

Figure 3.8: Bitcoin closing price from 2021-01-01 to 2022-09-01

As it can be visualized from the graphs, cryptocurrencies are highly intercorrelated, especially during volatile bull and bear runs. The chosen time frame is rather interesting because it includes multiple huge runs and crashes but also a more stable period in the last weeks of the plot. Therefore, the examination of the closing prices points out that the data set has a time-by-time downward and rising trend. This means that the time series has a dynamic characteristic.

Another parameter detailed in this research is the daily return on crypto closing prices. In general, the return of a digital asset (R_i) is a ratio or percentage value which reflects the profitability or efficiency of a certain trade or investment. Specifically, the return evaluates the profit or loss of an investment in relation to its purchasing cost. This means that the calculation of the return of a cryptocurrency is simply the ratio between the net profit and net cost (total acquisition cost). In this case specific to calculate the daily returns dataset of each cryptocurrency i at time t, it is used the following formula:

$$R_i = \frac{P_{i,t} - P_{i,t-1}}{P_{i,t-1}} = \frac{P_{i,t}}{P_{i,t-1}} - 1$$

where P_t is the price at time t and P_{t-1} is the price at time t-1.

Date	*BTC-USD Return*	*ETH-USD Return*	*ADA-USD Return*	*SOL-USD Return*	*DOT-USD Return*
2021-01-01	NaN	NaN	NaN	NaN	NaN
2021-01-02	0.093726	0.060473	0.011822	-0.023239	0.108588
2021-01-03	0.020380	0.259475	0.155403	0.201457	0.0895528
2021-01-04	-0.024712	0.066350	0.096427	0.149575	-0.056180
2021-01-05	0.063197	0.057461	0.149278	-0.131938	0.024504

Table 3.4: Cryptocurrency daily return data example (2021-01-01; 2022-09-01 dataset)

Using this returns DataFrame, it figures out in which dates each cryptocurrency had the best and worst single day return. Tabel 3.5 shows that 4 of 5 digital assets share the same day for the worst drop. On that day something significant happened, cryptocurrencies plunged after regulatory moves from China. The financial industry regulators decided to force banks and payment firms to stop offering customers services involving cryptocurrencies[142].

[142] C. VALETKEVITCH, *Bitcoin drops to lowest since Jan; stocks fall before Fed minutes*, Reuters, May 19, 2021. Available at: https://www.reuters.com/business/global-markets-wrapup-4-2021-05-19/

BTC-USD Return	2022-06-13
ETH-USD Return	2021-05-19
ADA-USD Return	2021-05-19
SOL-USD Return	2021-05-19
DOT-USD Return	2021-05-19

Table 3.5: Worst single day drop

BTC-USD Return	2022-06-13
ETH-USD Return	2021-05-19
ADA-USD Return	2021-05-19
SOL-USD Return	2021-05-19
DOT-USD Return	2021-05-19

Table 3.6: Best single day gain

Another helpful tool to have a better data visualisation is the distribution of average daily returns. Figures 3.9 – 3.13 illustrate the variation in data distribution by a histogram and a line in combination to it.

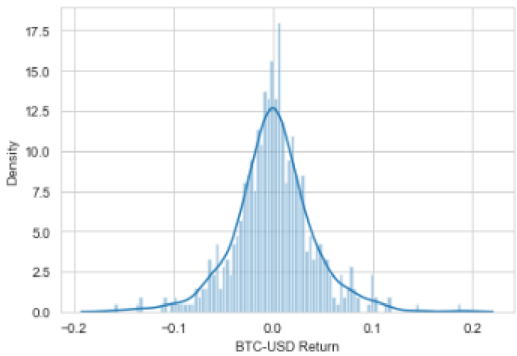

Figure 3.9: Bitcoin distribution plot

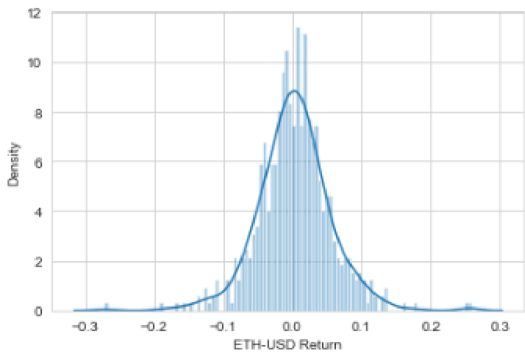

Figure 3.10: Ethereum distribution plot

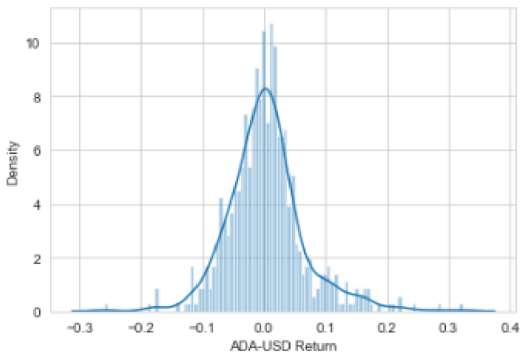

Figure 3.11: Cardano distribution plot

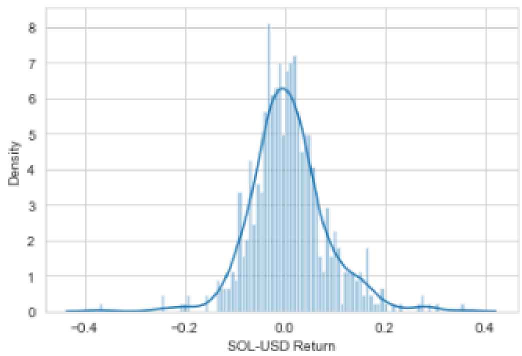

Figure 3.12: Solana distribution plot

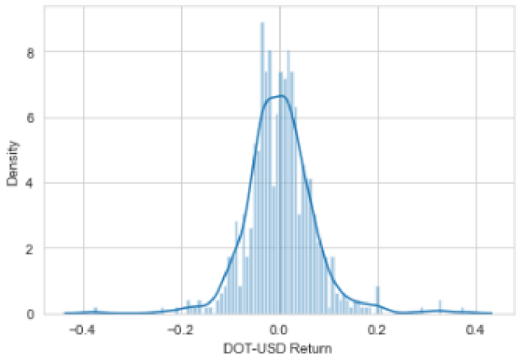

Figure 3.13: Polkadot distribution plot

In order to determine which cryptocurrency is classified as the riskier over the entire time period, the standard deviation of the returns is measured. Standard deviation is an essential parameter to quantify the spread in the data. It can be applied to statistical testing, describing the data, etc. This implies that if the standard deviation is higher, the data is more spread out and if it is lower, the data is more centered. It is obtained by taking the variance's square root. The standard deviation formula is as follows:

$$\sigma_i = \sqrt{Var(R_i)}$$

where the variance, which is the index that allows to identify the difference between the actual return of the cryptocurrency and its expected return, is equal to:

$$\sigma_i^2 = Var(R_i) = \sum_{j=1}^{k} \left(r_{i,j} - \mu_i\right)^2 p_{i,j}$$

and where the expected return is given by:

$$\mu_i = E(R_i) = \sum_{j=1}^{k} r_{i,j} p_{i,j}$$

Table 3.7 demonstrates that the riskiest digital asset, denoted by the highest standard deviation value, is Solana (SOL).

BTC-USD Return	0.039595
ETH-USD Return	0.052870
ADA-USD Return	0.062880
SOL-USD Return	0.076326
DOT-USD Return	0.068863

Table 3.7: Standard deviation value of the period from 2021-01-01 to 2022-09-01

A further important data analytics metric is the Correlation matrix. It is a useful tool for comparing the coefficients of correlation between several features (or qualities) in dataset. It enables to visualize how strongly or weakly different variables are correlated. Each number in this matrix is the correlation coefficient between the variables represented by the corresponding row and column. The correlation matrix also represents an important data stage in the pre-processing step in machine learning pipelines, especially with high-dimension dataset. It is possible to decrease the number of features in a dataset by using the correlation matrix which allow to find variables that have a high degree of correlation. This process, also known as dimensionality reduction, can be utilized to speed up and enhance the performance of the model.

The correlation coefficient expresses the linearity relationship between two variables, in this case two cryptocurrencies, and enables to visualize how the return of a digital asset varies as the return of another one. It has a value between -1 and +1 that denotes both the strength and directionality of a relationship between two cryptocurrencies. The closer the value is to +1 or -1, the stronger the relationship is. The closer a number is to 0, the weaker the relationship is. A positive coefficient indicates the presence of a positive relation between the two cryptocurrencies, meaning that as one value increases, so does the other. On the other hand, a negative coefficient points out a negative relation, denoting that as one digital asset increases, the other decreases. Values near to zero indicates there is an absence of any correlation between the two cryptocurrencies, and hence, those digital assets are independent of each other. The correlation coefficient is express as follows:

$$\rho_{i_{j}j} = \frac{Cov_{i,j}}{\sigma_i \sigma_j}$$

where the covariance is calculated on single pairs of cryptocurrencies (i, j) as the expected value of the product of the spreads from the average of the two digital assets.

$$Cov_{i,j} = E\left\{\left(\left(r_{t,i} - E(R_i)\right)\right)\left(r_{t,i} - E(R_i)\right)\right\}$$

Figure 3.14 illustrates the heatmap of the correlation between the variables, in this case cryptocurrency closing prices. Each cell in the below matrix is also distinguished by shades of colors. Here, shades of red indicate larger values of correlations (near to 1), while shades of blue correspond to smaller values. The matrix points out a strong positive correlation between the pairs: bitcoin-polkadot and ethereum-solana. This means if one of the targeted cryptocurrencies of the pair increases or decreases, the other behaves accordingly.

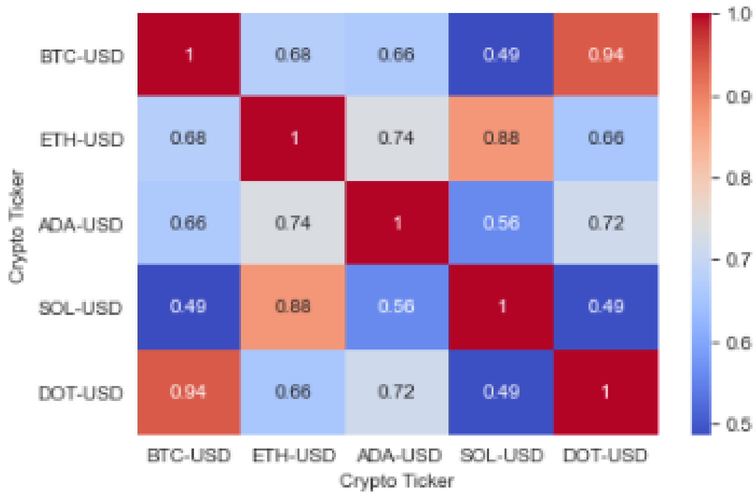

Figure 3.14: Correlation matrix between Crypto closing prices

3.6 Data Normalisation

In machine learning field, the transformation data used as input features plays a crucial role. In fact, it frequently happens that they are completely different from the perspective of distribution or order of magnitude, leading to significant imbalances in the learning phase of the model. Therefore, a pre-processing phase is necessary to ensure that the agent can interpret data appropriately and that the knowledge obtained is more pertinent. Particularly in neural networks, scaling data is crucial to minimize bias across features and, as a result, it significantly reduces the training time of the algorithm.

The traditional standardization techniques foresee the need to be able to estimate the maximum and minimum values of the dataset, or to calculate functions on the entire dataset such as mean, variance or standard deviation. This is feasible in the situation where predictions are only required offline, and the distribution and scales of the data are well defined. In this way, it is possible to compute all the metrics without encountering any issues, to the benefit of the algorithm training.

This is not at all possible when we are in the presence of unknown data or whose minimum and maximum values are not established a priori, as in the case of time series in the financial field. The major cause of this incompatibility lies in the fact that predictions must be made on data never seen before and in real time, which could therefore exceed the minimum and maximum levels of the data known a priori. For this reason, it is essential to adopt normalisation approaches that do not assume a priori knowledge of future data in the training phase.

There are various methodologies to achieve the desired result. For example, it is possible to define a priori knowledge of the application domain and establish maximums and minimums depending on the latter. In this case, an intrinsic knowledge is used, not necessarily due to the data at that moment, but to a more general awareness of the field of application. The methodologies used to scale data consistently are called normalization and standardization techniques. The whole process makes the mean of all the input features equal to zero and also converts ther variance to 1. This ensures that there is no bias while training the model due to different scales of all input features. If this is not done, the neural network might get confused and give a higher weight to those features which have a higher average value than the others.

The most famous and used normalization technique is the Min-Max Normalization. In this case all the input data are mapped in a predefined range between $[new_{min}, new_{max}]$, using the minimum and maximum value of the attribute to normalize. To normalize a value x of feature X to x' in the range $[new_{min}, new_{max}]$ the following formula is applied:

$$x' = new_{min} + \frac{(x - X_{min}) \cdot (new_{max} - new_{min})}{X_{max} - X_{min}}$$

where X_{min} and X_{max} represent the current minimum and maximum values of attribute X.

It has been decided to apply Min-Max Normalisation approach as part of the data preparation for the LSTM model building. This is because we have the knowledge of information available a priori, i.e. the maximum and minimum of the time series. Through the MinMax Scaler, the values are scaled to a value range of 0 to 1 without changing the shape of the original distribution.

3.7 Data Training and Testing

Once the dataset is prepared and preprocessed, the next stage is the split of it into two parts: the training test, where the agent has the opportunity to train trying to derive strategies with best performance on them, and the test set used to assess the model accuracy. The final goal is not to learn how to earn on the training dataset, but to maximize the reward on the test dataset, without the agent ever seeing data from the latter.

The division of the dataset into 2 parts places in front of the choice of the size of each of them. This is a crucial parameter, since it is necessary to find a trade-off between the amount of data the agent has access to during the training phase, and on how many has the opportunity to verify the strategies found. To determine the split size, it has to be considered the trade-off between: an excessively large training set and one too small.

Generally, for algorithms involving neural networks, it is normal to take into consideration a large training set, since networks naturally tend to learn much more when given access to the largest training dataset available. On the other side, it is important to remember that a bigger training set requires more time to learn, and an overtrained model might result in overfitting. In the case of a training set too small, the network may not have enough data available on which to develop a winning strategy. A common feature of neural networks lies in the need to have a large amount of data available in order to converge to an acceptable solution with the possibility of generalization.

In this case, a ratio of 70 and 30 percent is used to create training and test sets for the five cryptocurrency datasets. This decision is driven by the amount of data that is available in the initial dataset of each digital asset. For the entire initial dataset is considered a period which goes from 2021-01-01 to 2022-09-01, hence it includes 608 data points. At the end, the training set consists of 456 data points, and the testing set contains 152, as shown in Table 3.8.

Cryptocurrency	*Training size*	*Test size*	*Data set size*
Bitcoin	456	152	608
Ethereum	456	152	608
Cardano	456	152	608
Solana	456	152	608
Polkadot	456	152	608

Tabel 3.8: Summary of the dataset sizes

Chapter 4
METHODOLOGY

Chapter 4 focuses on illustrating the design of the Long Short-Term Memory model used for Bitcoin, Ethereum, Cardano, Solana and Polkadot cryptocurrencies price prediction. After training and testing the model for all the different digital assets, we extract and evaluate the model performance in terms of accuracy and error rates involved. Finally, we determine for which cryptocurrencies the LSTM approach works best.

4.1 Tools and Platform

In order to construct the empirical project for this thesis, the code part has been developed in Python, a high-level, general-purpose programming language. Among the coding languages, Python is one of the most ideal candidates for the implementation of machine learning and neural network algorithms, and so to build the Long Short-Term Memory model proposed. It is also well suited for graphics and statistical computing, since the integrated open-source environment is provided with a variety of pre-existing libraries, which are employed throughout all the processes, from the data pre-processing to the model building.

Pandas and Numpy libraries are employed in the data preparation procedure. Pandas is largely utilized for data related tasks which help in analysing, managing and manipulating data in a convenient and efficient manner. Numpy is a fundamental package for scientific computing, and it is used for calculations on the dataset, such as matrix/vector operations, and for storing training and test data sets. The performance metrics and the min-max normalisation, through MinMaxScaler, are obtained using Sklearn library. Since it offers a wide range of efficient tools for statistical and machine learning modelling such as regression, classification and clustering, it is considered as one of the most robust Python libraries for machine learnings. It also provides a variety of additional tools for data pre-processing, model development and evaluation selection.

For the LSTM front-end program, it has been chosen Keras, which is a Neural Network library[143]. By providing high-level building blocks, it facilitates faster experimentation with deep neural networks[144]. However, since it cannot handle low-level calculations, a backend library must be used to resolve this problem. In this case, TensorFlow was selected as the specialized tensor manipulation library which is optimized to act as the back-end layer. It is a widely adopted, scalable and production-ready Python library for research in the machine learning and deep learning area, and it is built to be easily run on several CPUs[145] and GPUs[146]. Lastly, Plotly and Matplotlib libraries are used for visualisation purposes and to display the charts.

The project is run using NVIDIA GeForce 920MX. Deep learning computing tasks are automatically dispersed and executed in parallel on NVIDIA GPU[147]. The experiments are performed on a device whose specifications are shown in Table 4.1.

CPU	Intel® Core™ i5-7200U CPU @ 2.50GHz, 2712 Mhz
GPU	NVIDIA GeForce 920MX
Memory	8GB
OS	Microsoft Windows x64

Table 4.1: Machine specifications

[143] Keras [Online]. Available at: https://keras.io/

[144] F. CHOLETT, *Deep Learning with Python*, Manning Publications Co., Shelter Island NY, 2018, p. 61.

[145] *A central processing unit (CPU) is the electronic circuity which carries out the instructions included in a computer programme. The CPU executes the program's instructions for arithmetic, controlling, logic, and input/output tasks.* Available at: https://www.ibm.com/docs/en/ztpf/1.1.0.15?topic=resources-processor-definition

[146] *A graphics processing unit (GPU) is a specialized electronic circuit created to operate and change memory to speed up the production of images in a frame buffer intended for output to a display device.* Available at: https://www.intel.co.uk/content/www/uk/en/products/docs/processors/what-is-a-gpu.html

[147] Nvidia [Online]. Available at: https://www.nvidia.com/en-us/

4.2 Time Series Data

After the data pre-processing stages discussed in chapter 3, data is prepared for modelling. Specifically, the input needed for the LSTM model is reshaped on the base of Time Series forecasting requirement. Typically, a time series consists of a sequence of numbers along time. Contrary to its autoencoder version[148], LSTM for sequence prediction works as a supervised method. As a result, the entire dataset has to be divided into inputs and outputs. Additionally, LSTM is superior to traditional statistical linear models because it can more easily handle different input forecasting problems[149].

As regard this study, the LSTM model prosed uses historical data to forecast 30 days in advance of closing price. The first parameter to be decided is the window size, which is the number of previous daily close data that each prediction has access to. The decision for the window size fell back on 15-time steps, hence the input data set consists of a tensor of matrix with dimension 15x1, with 1 feature and 15 rows for each window. Since Python is zero-indexed, the first window is made up of 0 to the 14 rows, the second from 1 to 15 and so on[150]. The selection in value for the number of timesteps is made taking into consideration the dataset size and through a discretionary approach after trying various window sizes. Hence, the 15-window length is chosen in part because a smaller one tends to exclude patterns that may appear in a longer sequence and since a higher one tends to decrease the LSTM accuracy. The output data accounts not only the window size, but also the 30 days prediction range. Therefore, the output dataset extends from row 15 up until the end and it is composed of chunks of length 30. The LSTM network's output size is also based on the prediction range.

---

[148] *An LSTM Autoencoder is an implementation of an autoencoder which uses an Encoder-Decoder LSTM architecture for sequence data. An encoder-decoder LSTM is set up to read the input sequence, encode it, decode it, and replicate it for a given dataset or sequences. Based on the model's capacity to duplicate the input sequence, the performance of the model is assessed.* Available at: https://analyticsindiamag.com/introduction-to-lstm-autoencoder-using-keras/

[149] K. STRUGA, O. QIRICI, *Bitcoin Price Prediction with Neural Networks*, International Conference on Recent Trends and Applications in Computer Science and Information Technology, University of Tirana, 2018. Available at: https://ceur-ws.org/Vol-2280/paper-06.pdf

[150] MathWorks [Online]. Available at: https://www.mathworks.com/help/econ/rolling-window-estimation-of-state-space-models.html.

4.3 Turn data into tensors

The input for LSTM is provided as a three-dimensional vector of float values. A fundamental feature of tensors is their shape, which in Python is a tuple of integers reflecting its dimension along the 3 axes. For example, in the testing data for Bitcoin, the shape of training inputs is: 440, 15, 1, where 440 stands for the number of samples, 15 is the value of the window size and 1 represents the features. Technically, the data is divided into chunks of 15 and each of these small windows of data are insert into a numpy array. Each window has a 15x1 dimension, so all windows create the tensor.

4.4 LSTM Implementation

The lack of memory retention is a key characteristic of feedforward networks (FNNs). Given that no state is saved between inputs, each input is processed separately. However, since we are dealing with time series where information from previous cryptocurrency prices is needed, it is important to maintain some information to forecast future prices.

A branch of Deep Learning approaches, that provides this characteristic is the Recurrent neural network (RNN) architecture, which features a self-directing loop along with the output. As a result, the step set provided as input is processed in a sequence rather than in a single step. On the other hand, the gradient becomes too small/large when the size of window is quite large, which is often the case. This implies the phenomenon known as vanishing/exploding gradient respectively. This issue arises while the optimizer backpropagates and it will make the algorithm run, while the weights are unlikely to change at all.

RNN variants, particularly LSTM, reduce the problem. The LSTM layer adds some cells which carry information over several timesteps. The status of these cells modulates the LSTM's output. This is generally important in making predictions based on historical context, rather than on just the most recent input. While in RNN loops are absent, LSTM networks are able to remember inputs by using a loop. However, the likelihood that the subsequent output would be dependent on a very old input likely decreases with time, so forgetting is necessary. By using their forget-gates, LSTM learns when to remember and when to forget. Each gate has a unique set of weights.

The input gate and intermediate cell state are combined with the old cell state and the forget state in order to generate the output which will be subsequently utilized to determine the new cell state. Thus, this complex cell with four interacting layers makes LSTM network the ideal model for sequence prediction.

The decision to use a Long Short-term Method model for the price prediction of cryptocurrencies also depended on the fact that LSTMs have a memory capacity, which is useful in analysing timeseries data sets, such as cryptocurrency price over time. LSTMs may deliberately choose which data to use as input and which to disregard for the model[151]. It can also remember historical patterns and use them to improve future forecasts.

4.5 LSTM Structure

The LSTM architecture consists of three components: one LSTM layer, one Dense layer and one Activation Layer.

The LSTM layer is the inner one, and all the gates previously mentioned are already implemented by Keras, with a default activation of hard-sigmoid. The below-mentioned input mode and neurons count are the LSTM parameters.

The Dense Layer is a standard layer which is fully connected[152]. It gives the model a memory capacity. The number of Dense layers represents the measure of hidden units of the layer. A hidden unit is a dimension in the representation space of the layer. Unit is one of the most fundamental and essential parameters of the Dense layer and, given that it represents the dimensionality of the output vector, it must be a positive integer[153]. Having a higher-dimensional representation space and so more hidden units enable the network to learn more complex representations, on the other hand it

[151] D. ZOMER, *Using machine learning to predict the future bitcoin prices*, Towards Data Science, 2020. Available at: https://towardsdatascience.com/using-machine-learning-to-predict-future-bitcoin-prices-6637e7bfa58f

[152] *In a fully connected layer, all the neurons of the input are connected to every neuron of the output layer.* Available at: https://www.baeldung.com/cs/neural-networks-conv-fc-layers

[153] Y. VERMA, *A Complete Understanding of Dense Layers in Neural Networks*, Developers Corner, 2021. Available at: https://analyticsindiamag.com/a-complete-understanding-of-dense-layers-in-neural-networks/

increases the computational cost and the risk of learning undesirable patterns, such as patterns that improve performance on the training data but not on the test data.

As regard the Activation Layer, considering we are solving a complex regression problem, the final layer should provide a linear combination of the activations for the previous layer and the weight vectors. As an alternative, it could be transferred as a parameter to the previous dense layer.

4.6 Hyperparameters

4.6.1 Loss function

To determine the model error, LSTM neural networks are trained using an optimization procedure that require a loss function. Since there are too many unknown factors, it is impossible to compute the perfect weights for a neural network. Instead, the learning issue is framed as a research or optimization problem, and an algorithm is employed to numerically search in the domain of the potential weight sets that the model may use, in order to provide accurate predictions. Commonly, LSTM neural network model's weights are updated using the backpropagation of error process after it has been trained through the gradient descendent optimization technique.

Gradient descent is an optimization technique which iteratively moves in the direction of the steepest descent, as indicated by the gradient's negative, in order to locate the minimum of the function. At each step, this method examines, for each parameter, which way the training set loss would move if you perturbed that parameter just a small amount. The parameter is then updated in a way that may reduce the loss[154]. Since it is a first-order optimization algorithm, it only considers the first derivative when performing the updates of the parameters[155]. An error gradient is referred to as the "gradient" in gradient descendent. Predictions are made using the model with specific weights, and the error associated with those predictions is determined. The gradient descendent algorithm aims to modify the weights so that the following

[154] A ZHANG, Z.C. LIPTON, M. LI, A.J. SMOLA, *Dive into Deep Learning (Release 0.17.4)*, 2022, p.22.
[155] J. BROWNLEE, *Optimization for Machine Learning: Finding Function Optima with Python*, Machine Learning Mastery, 2021.

evaluations reduce the error. The algorithm accomplishes this by iteratively determining step size at each iteration.

The loss function or objective function is what is used to evaluate a set of weights in the context of an optimization method. Since it represents a measure of success for the task at hand, the loss function must accurately concentrate all the aspects of the model to a single value in such a manner where changes in that number are an indication of a better model. Hence, a loss function must be selected to determine the model's error during the optimization model. This can be a difficult challenge to solve given that the function has to include the characteristics of the problem and be driven by concerns which are crucial to the project. Selecting the appropriate objective function for the problem at hand is crucial, since the LSTM neural network model will take any shortcut it can to reduce the loss; therefore, if the objective does not correspond with success for the task at hand, the network may wind up acting in ways that may not be expected.

A variety of functions may be utilized to estimate the accuracy of a set of weights in a neural network. It is significant to note that the activation function used in the neural network's output layer directly affects the loss function chosen. Hence, it is useful to consider the configuration of the output layer as a decision for how to frame the prediction issue, and the loss function as the method to calculate the error for a certain framing of the problem.

Given that the main issue of this study is the cryptocurrency price prediction, we are facing a problem where it has to be forecasted a real-value quantity. This leads to a regression problem. The typical performance measures for regression problems are: Root Mean Square Error (RMSE), Mean Absolute Error (MAE) or Mean Squared Error (MSE). Since the errors are squared before they are averaged, the RMSE gives a relatively high weight to large errors. This means that the RMSE is more sensible to observation which are further from the mean rather than MSE and MAE. The Mean Squared Error is the default loss to utilize for regression issues. Mathematically, it is the preferred loss function within the maximum likelihood inference approach if the distribution of the target is Gaussian. However, on some regression issues, the distribution of the target variable may be primarily Gaussian, but it may also contain outliers, observations which lie outside the overall patter of a distribution[156]. Given the typical cryptocurrencies price spikes and the will to assign a low weight to outliers, the choice fell on the MAE for the most reasonable 30 days close price predictions. Since this parameter is the aggregated mean of the errors, this enables the MAE to be more resilient to outliers and considered a suitable loss function for this scenario[157].

4.6.2 Optimizer

The optimizer plays a fundamental role in the increase of the accuracy of the LSTM model. Optimizers are techniques that modify the weights and learning rates of the neural network in order to minimize losses. Therefore, it determines how the network is updated based on the loss function.

There are several optimizer variants that can be used such as Gradient Descent or Stochastic Gradient Descent (SDG). The Stochastic Gradient Descent is a probabilistic approximation of the Gradient Descent. Because the technique only calculates the gradient for one randomly chosen observation at each step rather than the gradient for the whole dataset, the result is an approximation. Using a single training sample to calculate the gradient of the cost function for each iteration permits the SDG to require less memory. This allows the Stochastic Gradient Descent to be substantially faster than the Gradient Descent and better suited for large-scale datasets, but computationally more expensive due to the parameters are updated for each sample. At the same time, given that the gradient is only calculated for one random point every iteration rather than the full dataset, the frequent updates of the weights can provoke stableness problems through the production of noisy gradients. This causes the loss function to fluctuate more on each interaction, instead of slowly declining, when compared to Gradient Descent. Therefore, Stochastic gradient descent tends to have higher variance and to diverge instead of converging to the global minimum. This implies problems considering cryptocurrency prices[158]. Other excellent optimizers

---

[156] D. YATES, D. MOORE, G. MCCABE, *The Practice of Statistics*, W.H. Freeman and Company, New York, 1999.

[157] T.O. HODSON, *Root-mean-squared error (RMSE) or mean absolute error (MAE): when to use them or not*, U.S. Geological Survey Central Midwest Water Science Center, Urbana, IL, USA, 2022. Available at: https://gmd.copernicus.org/articles/15/5481/2022/gmd-15-5481-2022.pdf

[158] K. STRUGA, O. QIRICI, *Bitcoin Price Prediction with Neural Networks*, International Conference on Recent Trends and Applications in Computer Science and Information Technology, University of Tirana, 2018, p.5. Available at: https://ceur-ws.org/Vol-2280/paper-06.pdf

include Adagrad, AdaDelta, RMSProp and Adam, which are variations of adaptive learning algorithms. They are called first-order optimization algorithms since they use the first derivative of the function to minimize the loss.

Adagrad is a gradient-based optimization algorithm that adapts the learning rate to the parameters, performing larger updates, in this case high learning rates, for parameters associated with infrequent features and low learning rates for parameters associated with frequently occurring features[159]. This makes it a well-suited method for handling sparse data. The primary flaw in Adagrad is the accumulation of squared gradients in the denominator of its formula. Since every additional term is positive, the cumulative amount keeps growing during training. As a result, the learning rate decreases until eventually become infinitesimally small, at which time the algorithm is unable to learn anything new. To fix this problem, the following algorithms are used. AdaDelta is an extension of Adagrad that aims to lessen its monotonically diminishing learning rates[160]. Adadelta limits the window of accumulated prior gradients to some fixed size rather than accumulating all previously squared gradients.

RMSprop addresses this problem by normalising the gradient using a moving average of squared gradients. The step size (momentum) is balanced by this normalisation, which increases the step for small gradients to prevent vanishing issues and decreases the step for large gradients to avoid exploding problems. Instead of considering the learning rate as a hyperparameter, RMSprop employs an adaptive learning rate. RMSprop is very similar to AdaDelta algorithm, except for the fact that Adadelta employs the RMS of parameter updates in the numerator update rule. On the other hand, Adam enhances RMSprop with bias-correction and momentum.

The one used for this LSTM model is the Adaptive Moment Estimation (Adam), since it was found to work slightly better than the rest techniques. It is demonstrated that this approach is computationally efficient, it has small memory requirement, it is invariant to diagonal rescaling of gradients, and it is well suited for problems that are

[159] J. DUCHI, E. HAZAN, Y. SINGER, *Adaptive Subgradient Methods for Online Learning and Stochastic Optimization*, Journal of Machine Learning Research, 2011, pp. 2121-2159. Available at: https://dl.acm.org/doi/10.5555/1953048.2021068

[160] M.D. ZEILER, *ADADELTA: An Adaptive Learning Rate Method*, Google Inc., New York University, USA, 2012. Available at: https://arxiv.org/pdf/1212.5701.pdf

vast in terms of data/parameters[161]. Therefore, since it is shown that its bias-correction helps Adam to slightly outperform other comparable algorithms towards the end of optimization as gradient become sparse, Adam algorithm was selected for this research. Adam is another technique which determines adaptive learning rates for each parameter. Its approach is based on adaptive estimation of first and second-order moments. This is the reason why, instead of rolling quickly just to get the minimum as fast as possible, the idea behind Adam is to slow down the velocity gradually to allow for a more detailed search. Hence, Adam also preserves an exponentially decaying average of past gradients m_t, in addition to maintain an exponentially fall of past squared gradients v_t, as AdaDelta and RMSprop. The values of the first and second moments, m_t and v_t, respectively, represent the gradients' mean and uncentered variance, therefore the name of the technique. The formulas for the decaying averages past and past squared gradients m_t and v_t respectively are as follows:

$$m_t = \beta_1 m_{t-1} + (1 - \beta_1) g_t$$

$$v_t = \beta_2 v_{t-1} + (1 - \beta_2) g_t^2$$

where the values of the first and second moments, m_t and v_t, respectively, represent the gradients' mean and uncentered variance, therefore the name of the technique. For brevity, g_t is used to denote the gradient at time step t, while betas represent hyper-parameters of the algorithm. D. Kinga and J. Ba, Adam's authors, note that m_t and v_t are biased towards zero when they are first initialised as vector of o's, especially in the early time steps and when the decay rates are small, hence when β_1 and β_2 are close to 1. To offset these biases, they compute bias-corrected first and second moment estimates as follows:

$$\widehat{m}_t = \frac{m_t}{1 - \beta_1^t}$$

$$\hat{v}_t = \frac{v_t}{1 - \beta_2^t}$$

[161] D.P. KINGMA, J.L. BA, *Adam: a method for stochastic optimization*, International Conference on Learning Representations, 2015. Available at: https://arxiv.org/pdf/1412.6980.pdf.

The Adam's creators suggest default values of 0.9 for β_1 and 0.999 for β_2. They empirically demonstrate that Adam performs well in practise and outperforms other adaptive learning-method algorithms[162].

4.6.3 Activation function

Activation functions are mathematical functions. Their primary characteristic is that functions are differentiable given that this is a condition for backpropagation in model training. The main challenge is selecting the appropriate activation function, which can be thought of as a type of hyperparameter tuning in which the programmer selects the activation function by comprehending the problem definition and taking into account the model's performance and the convergence of the loss function. In the nodes of a neural network's input layer, no activation function is necessary. It only stores the input data, so no calculation is needed. In a neural network's hidden layers, a non-linear activation function is generally needed. This is because in order for the network to learn complicated patterns, nonlinearity must be included. Without the application of non-linear activation functions, a neural network with several hidden layers would transform into a massive linear regression model which is ineffective for learning intricate patterns from real-world data. The type of activation function chosen inside the hidden layers will have a substantial impact on the performance of a neural network model. In a neural network's output layer, an activation function is also necessary. The kind of problem we wish to solve determines the activation function to use.

The most popular activation functions are: Sigmoid, Tanh and ReLU. The Sigmoid function is employed in logistic regression models; its graph is s-shaped, therefore is a non-linear function (as shown in Figure 4.1). Through the sigmoid, the input is mapped in [0,1], where large negative values are converted to 0 and large positive values are converted to 1. Sigmoid suffers from vanishing gradient, hence almost no signal travels form the neuron to its weight. Since the outputs are not zero-centered, it makes the optimization process harder.

---

[162] D.P. KINGMA, J.L. BA, *Adam: a method for stochastic optimization*, International Conference on Learning Representations, 2015. Available at: https://arxiv.org/pdf/1412.6980.pdf.

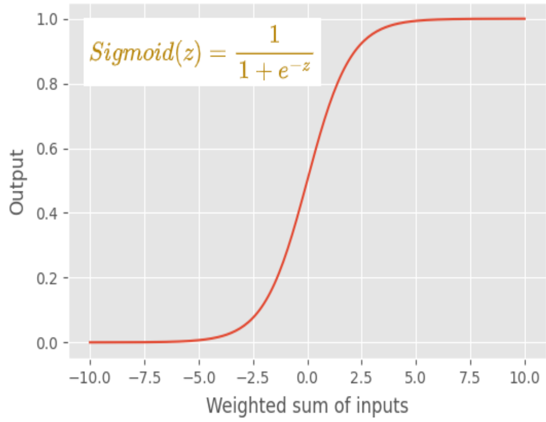

Figure 4.1: Sigmoid activation function

The tanh (tangent hyperbolic) function features an s-shaped graph just like the sigmoid function, and its result is always between -1 and +1 (as shown in Figure 4.2). This function is likewise non-linear and zero-centered, which gives it an advantage over the sigmoid function. This simplifies the optimization procedure. As for the sigmoid function, the tanh has also the vanishing gradient problem.

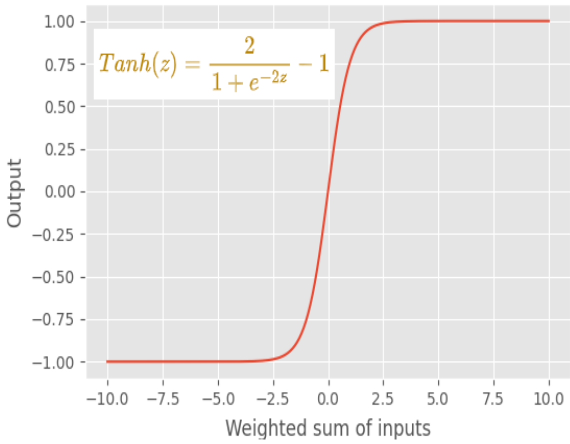

Figure 4.2: Tanh activation function

As regard the ReLU (Rectified Linear Unit) activation function, it is an excellent substitute for the sigmoid and tanh functions. Unlike the latter, the vanishing gradient problem is not present in this function. The computing cost of ReLU is very low. The Rectified Linear Unit is thought to converge 6 times more quickly than sigmoid and tanh functions. The ReLU function outputs the input exactly as it is if the input value is 0 or above. Otherwise, if it is lower than 0, the function returns 0. The ReLU's output can range from 0 to positive infinity (as shown in Figure 4.3). In the end, it was decided to use the Tanh function, since it is preferred in Recurrent Neural Networks[163].

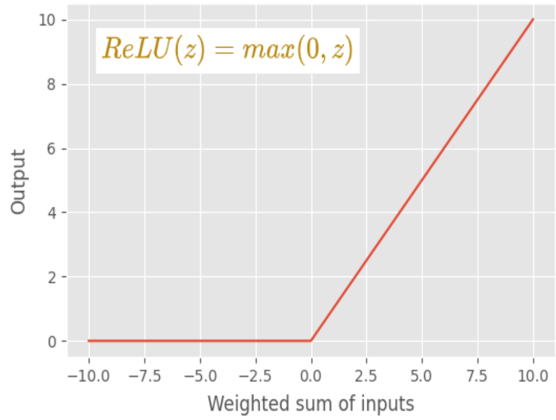

Figure 4.3: ReLU activation function

4.6.4 Numbers of Neurons

This measure specifies the number of nodes or neurons in the hidden layer. The choice of the ideal number of hidden nodes for the LSTM model proposed is based taking in account the complexity of the dataset and the data-generating process, and through a trial-and-error process. The choice fell on 20 neurons, since a higher number of neurons did not provide better outcomes and it generally increases the cost and the lengthen of the training time.

4.6.5 Epochs

Epoch is a parameter that quantifies mow many times the algorithm will process the entire training dataset. Since each epoch represents a complete loop through the training data, the number of epochs has a similar effect to the number of hidden layer neurons in terms of how long training takes to complete. The amount of training epochs to use might be a problem with training neural networks, given that too many epochs can result in an overfitted model[164], while too few may lead to underfit the training dataset. After experimenting with alternative values, it was selected the 200

[163] C. NWANKPA, W. IJOMAH, A. GACHAGAN, S. MARSHALL, *Activation Functions: Comparison of trends in Practice and Research for Deep Learning*, arXivLabs, Cornell University, 2018. Available at: https://arxiv.org/pdf/1811.03378.pdf

[164] *Overfitting is a statistical modelling error which happens when a function is too tightly aligned to a limited set of data points. Because of this, the model is helpful in relation only to its initial data set, and not to any other data sets.* Available at: https://www.investopedia.com/terms/o/overfitting.asp.

epochs option. To overcome and improve this discretionary choice, it was decided to use the Early stopping function, a technique that enables to define an arbitrary large number of training epochs and stop training as soon as the model performance stops improving on the validation dataset. This needs that a validation split must be provided to the fit() function along with an EarlyStopping callback to define the performance metric that will be tracked during the validation split[165]. Keras supports the early stopping of training via a callback called EarlyStopping. The callback permits to determine the performance measure to monitor, the trigger, and once triggered, it will arrest the training process. The parameters of the function are as follows:

- Monitor: the quantity to be monitored. In this study, the validation loss.
- Verbose: is considered as a verbosity mode.
- Mode: there are three types of mode: "*min*" and "*max*" and "*auto*". The "*min*" mode is used while training will stop when the quantity monitored has stopped decreasing, while the "*max*" mode when it has stopped increasing. In the "auto" mode, the direction is automatically determined based on the name of the quantity being monitored. In this research, it was chosen the "min" mode since the aim is to reduce the *val_loss*.
- Patience: it is used to check the number of epochs with no improvement after which training will be stopped. In this case, it was set to 5. The choice is based on a trial-and-error process.

4.6.6 Batch size

The batch size parameter is used to specify the number of samples to be worked on. It refers to the number of data points that the model uses to compute the error before backpropagating it and making modifications to the weights. Using the same method for epochs, it was decided to feed the network with batches of 120 data.

With these parameters, the LSTM model is set and is ready to make predictions.

---

[165] Keras API reference / Callbacks API / EarlyStopping [Online]. Available at: https://keras.io/api-/callbacks/early_stopping/.

4.7 Performance Evaluation Measures

Since the study focuses on five different cryptocurrencies, it is essential to evaluate the LSTM model's application using the appropriate metrics to determine on which digital asset the model has the best fit. Metrics are used to monitor and measure the performance of a model both during training and testing. In the field of Neural Networks, the selection of the evaluation metrics to use is important, as it affects how a model's performance is assessed and compared, as well as how much importance you decide to assign to different features of the model on the results[166].

Given that cryptocurrency price prediction is a regression problem, we require measures based on estimating some type of distance between predicted and real value. To evaluate the LSTM methodology several performance measures are used: Root Mean Squared Error (RMSE), Mean Squared Error (MSE), Mean Absolute Error (MAE) and R Square Score (R^2). The detailed explanation for each performance metric is provided below.

4.7.1 Mean Squared Error

Mean squared error is the most commonly used metric for regression problems. Taking it from a statistical perspective, the MSE measures the ways of the squares of observed errors[167]. This is essentially the average of the squared difference between the actual observed value and the predicted or estimated value by the regression model. This metric is a function which corresponds to the expected value of the squared error loss and that assesses risk. The Mean Squared Error is calculated using the following formula:

$$MSE = \frac{1}{N}\sum_{j=1}^{N}\left(y_j - \hat{y}_j\right)^2$$

[166] A. BENINCASA, *An LSTM-based model to Trading Energy Stocks*, Politecnico di Torino, 2020, p.51. Available at: https://webthesis.biblio.polito.it/16736/

[167] K.B. SABOOR, Q.U.A. SABOOR, L. HAN, A.S. ZAHID, *Predicting the Stock Market using Machine Learning: Long short.term Memory*, Electronic Research Journal of Engineering, Computer and Applied Sciences, 2020. Available at: https://papers.ssrn.com/sol3/papers.cfm?abstract_id=381012-8

where, N is the number of samples, y_j is the observed value, $\hat{y}_j$ is the predicted value. Hence, the sum of squares of the prediction errors, that represents the difference between the real and predicted values, is divided by the number of predicted data.

The MSE is differentiable, allowing for improved optimization. Because of the squaring factor, it is essentially more prone to outliers than other measures. Since it squares even small errors, this penalizes the model evaluation leading to an overestimation of how bad it is. Therefore, the squaring factor must be taken into consideration while interpreting errors.

4.7.2 Mean Absolute Error

Mean Absolute Error is also a widely used function for regression. It is calculated as the average measure of the errors, which is the absolute difference between the real and predicted values. The primary distinction between MSE and MAE is that, for the Mean Squared Error, the squared of the difference tells us that the MSE value increases as the mistake rises. As a result, we communicate to our network that we do not accept values that are significantly dissimilar from the true ones. The Mean Absolute Error is computed using the following equation:

$$MAE = \frac{1}{N}\sum_{j=1}^{N}\left|y_j - \hat{y}_j\right|$$

where, N is the number of datums, y_j is the ground-truth value, $\hat{y}_j$ is the predicted value from the regression model. Therefore, MAE measures the difference between two continuous variables.

In contrast to MSE, the Mean Absolute Error is not differentiable, and it is more resistant to outliers since it does not inflate errors. Hence, MAE can be used when the researcher does not want outliers to have a significant impact. It can also be worthwhile if it is known that the distribution is multimodal and it is preferred to make prediction at one of the models, rather than at their mean of them. On the other hand, because of MAE employs the absolute value of the residual, it does not give a precise idea of the direction of the error, such as if the predictions of the data are under or overrepresentative, but it provides a gauge for how far the forecasts were from the actual output. This last concept is also true for the MSE parameter.

4.7.3 Root Mean Squared Error

The Root Mean Squared Error (RMSE) is another performance measure extensively used in prediction literature. The RMSE is very accurate to predict in a single measurement when large errors from several data points are combined[168]. Because it is scale dependent, RMSE is not applicable to multiple datasets, but it is an accuracy method to compare the prediction errors of many models in a particular dataset[169].

Mathematically the RMSE is given by:

$$RMSE = \sqrt{MSE} = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2}$$

where N is the number of observations, y_i is the true value and $\hat{y}_i$ is the forecasted value. Using RMSE, the MSE error is squared-rooted to restore it to its original unit while keeping the feature of penalizing higher errors. But, since RMSE is the square root of the mean squared errors, the RMSE suffers in case of outliers[170]. Therefore, the impact of each observed error on the RMSE will be proportional to the magnitude of their squared forms. The Root Mean Squared Error retains the differentiable property of MSE, and it displays a graph similar to the Mean Squared Error one, with the difference that here data is squeezed by a calculation of the root. Considering that the scale is the same as the random variable, error interpretation can be done smoothly.

---

[168] I. NASIRTAFRESHI, *Forecasting cryptocurrency prices using Recurrent Neural Network and Long Short-term Memory*, Data & Knowledge Engineering, 2022. Available at: https://doi.org/10.1016/j.-datak-.2022.102009

[169] S. AGARWAL, *An Approach of SLA Violation Prediction and QoS Optimization Using Regression Machine Learning Techniques*, University of Windsor, Canada, 2020.

[170] K.B. SABOOR, Q.U.A. SABOOR, L. HAN, A.S. ZAHID, *Predicting the Stock Market using Machine Learning: Long short-term Memory*, Electronic Research Journal of Engineering, Computer and Applied Sciences, 2020. Available at: https://papers.ssrn.com/sol3/papers.cfm?abstract_id=381012-8

4.7.4 R-Squared Score

The R^2 score, also known as the coefficient of determination, is one of the performance metrics for regression-based models; hence, it measures how well the model fits the data. The R^2 score works as a post metric, meaning it is a measure that is computed using other metrics, in this case the variance. In fact, the coefficient of determination can be defined as the proportion of the deviance of the dependent variable that is predictable from the independent variable[171]. The R-Squared measure is calculated using the sum of squared errors and it is given by:

$$R^2 = 1 - \frac{SS_{res}}{SS_{tot}}$$

where SS_{res} is the sum of squares of residuals of the data set and SS_{tot} is the total sum of squares proportional to the variance of the data.

The Total Sum of Squares is equal to:

$$SS_{tot} = \sum_{i=1}^{N} (y_i - \bar{y})^2$$

where y_i is the i^{th}item in the set, $\bar{y}$ is the mean of all items in the set and $(y_i - \bar{y})$ is the deviation of each item from the mean, while the Sum of Squares of residuals is computed as follow:

$$SS_{res} = \sum_{i=1}^{N} (y_i - \hat{y})^2$$

where $\hat{y}$ is the value estimated by regression line.

The following diagram shows the elements explained above, necessary to understand the concepts of R-Squared.

[171] I. NASIRTAFRESHI, *Forecasting cryptocurrency prices using Recurrent Neural Network and Long Short-term Memory*, Data & Knowledge Engineering, 2022. Available at: https://www.sciencedirect.com/science/article/abs/pii/S0169023X22000234?via%3Dihub

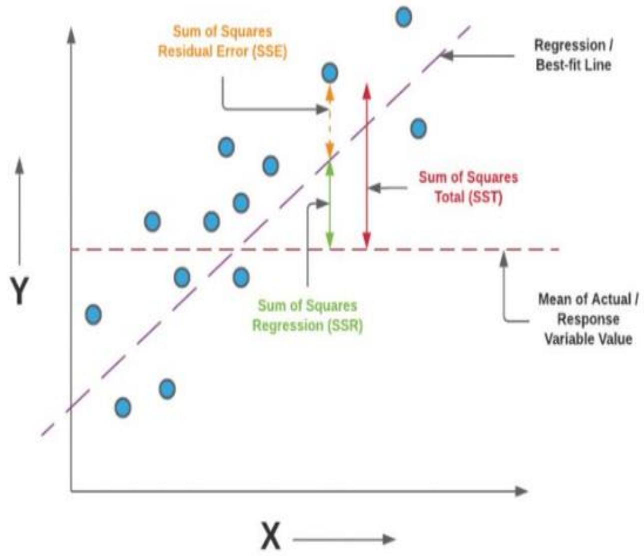

Figure 4.4: Sum of Squares (Total Sum, Regression Sum, Residual Sum)

In the best ideal scenario, where the modelled values precisely match the observed ones, R-Squared will be equal to 1 and the Residual Sum of Squared will be equal to 0. This would mean that the regression was able to capture 100% of the deviance in the target variable. Conversely, if the R-Squared will be equal to 0 and the Residual Sum of Squared will have a high value, it would intend the regression was unable to capture any variance in the target variable.

4.8 LSTM Results

This section illustrates the results of the project obtained from the Long Short-Term Memory algorithm using five popular cryptocurrencies: Bitcoin, Ethereum, Cardano, Solana and Polkadot. First, it shows the comparison of the predicted and actual prices and presents the evaluation of the different performance measures used in this study, then, it defines in which digital asset the LSTM model has the best fit for the price prediction based on the dataset collected.

Figures 4.5 - 4.9 display a graphical representation, comparing the original and the predicted daily closing prices of the training and testing data of the LSTM model for all the cryptocurrencies. Here, we can see that the red line, corresponding to the train predicted cryptocurrency prices, and the dark blue line, which stands for the actual close prices, are moving in tandem, as they are trained data points. As concerns the testing part, the graphs show that the predicted test data, figured in light blue, and the actual test data are quite synced. From a graphical point of view, the LSTM model was able to properly predict upward and downward trends. The only parts where it is possible to notice a lag in prediction price is when ample volatility has occurred, with price accelerating both to the downside or to the upside. An example of this trait is visible in Bitcoin and Ethereum plots when it was registered a strong drawdown in price in July 2022.

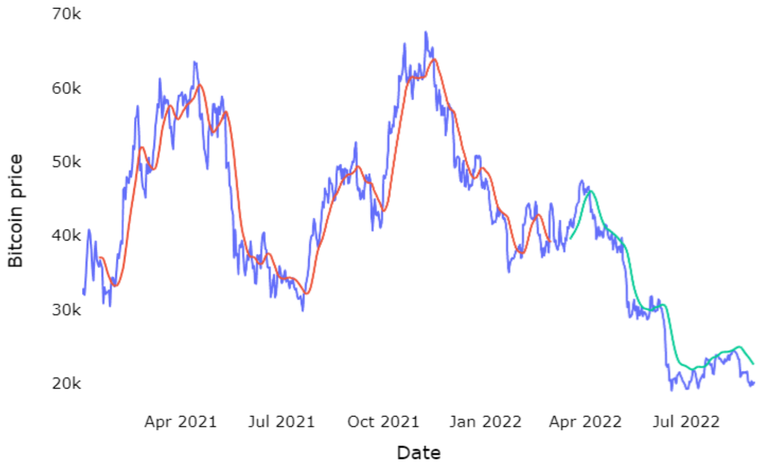

Figure 4.5: Comparison between Bitcoin original and predicted close price

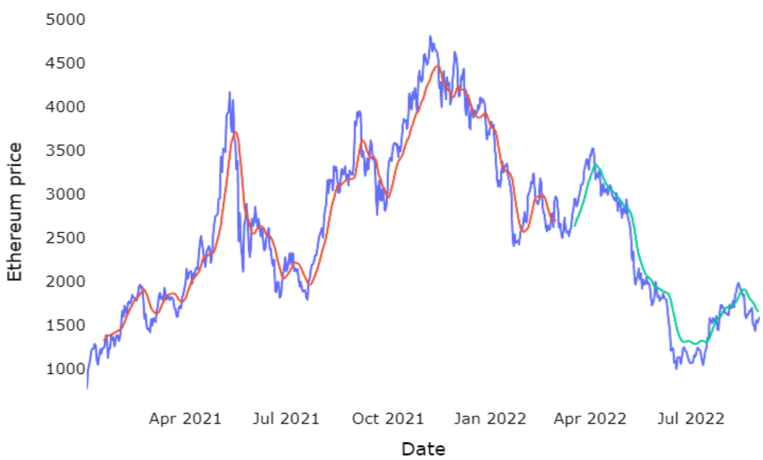

Figure 4.6: Comparison between Ethereum original and predicted close price

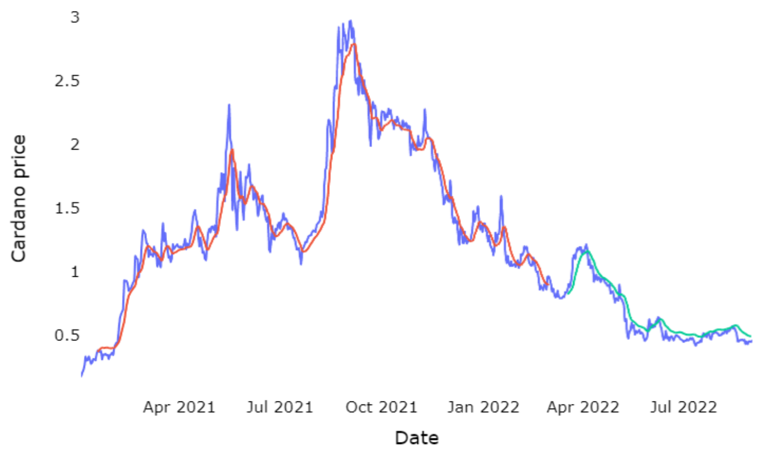

Figure 4.7: Comparison between Cardano original and predicted close price

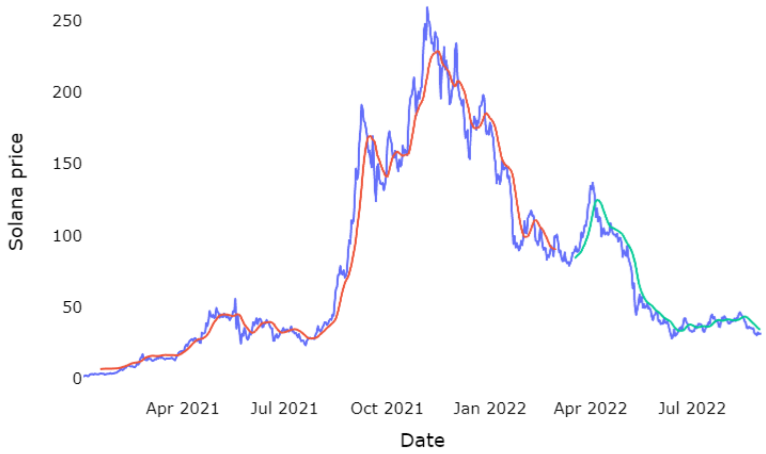

Figure 4.8: Comparison between Solana original and predicted close price

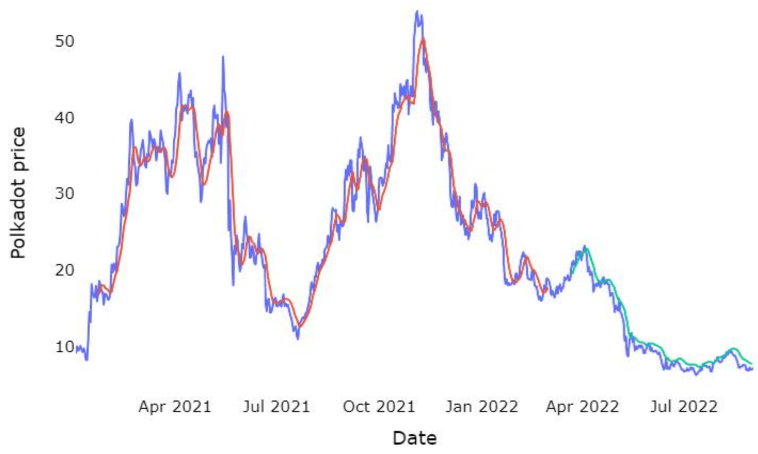

Figure 4.9: Comparison between Polkadot original and predicted close price

The train history of the LSTM approach can be used to diagnose the behaviour of the model. To do that, we consider model skill on the train and validation sets in terms of loss that is minimized. As previously explained, we will use the Mean Absolute Error metric, which was decreed meaningful for our goal.

Therefore, the diagnostic line plot based on Mean Absolute Error loss function history of the LSTM model in training and testing are shown in Figures 4.10 – 4.14. The red line represents the training loss function, while the blue one the validation loss. The x-axis takes into account the number of epochs and on the y-axes are signed the MAE loss values. Graphically, the results illustrate an overall quite good fit for the LSTM model, since it performs fairly well on both the train and validation loss. Generally, the grater the number of epochs, the sharper and more accurate the prediction becomes. This approach is visible in the Bitcoin, Cardano and Solana plots, where the EarlyStop function stops training a bit later than the Polkadot validation dataset. Except for Polkadot, in other graphs the train and validation loss both decline and stabilize at the same time. The point where predictions does not vastly improve is quite similar for all cryptocurrencies and it corresponds to the value around the 6th epoch. Through the use of an EarlyStopping technique, there were no cases of overfitting. This can be diagnosed given that, until the last epochs, the train and validation losses continued to slope down gradually, without hitting an inflection point for then beginning to degrade sloping up again.

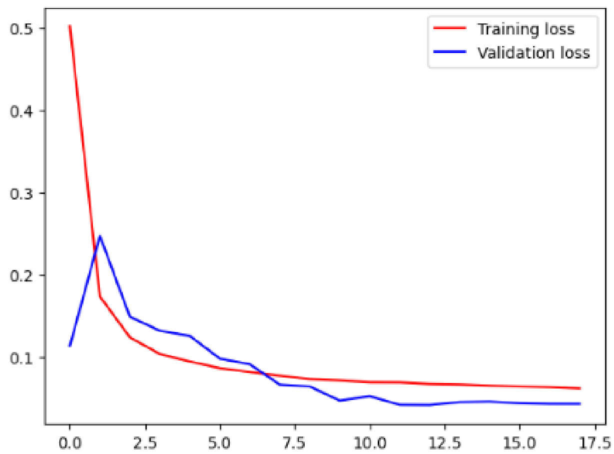

Figure 4.10: Loss function history of Bitcoin in train & test

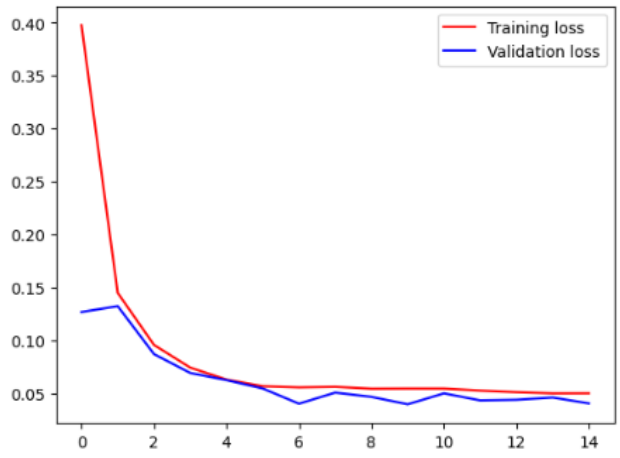

Figure 4.11: Loss function history of Ethereum in train & test

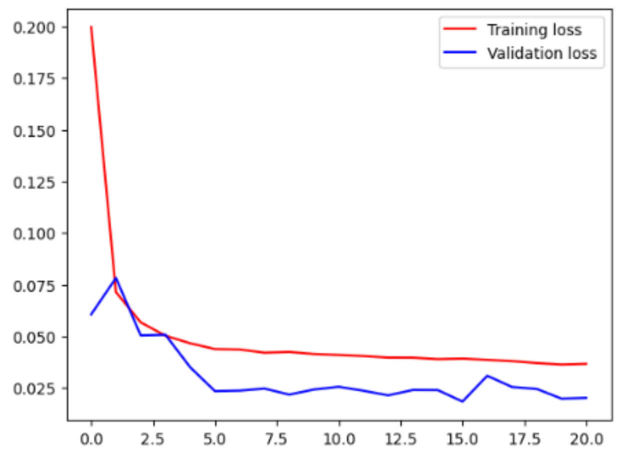

Figure 4.12: Loss function history of Cardano in train & test

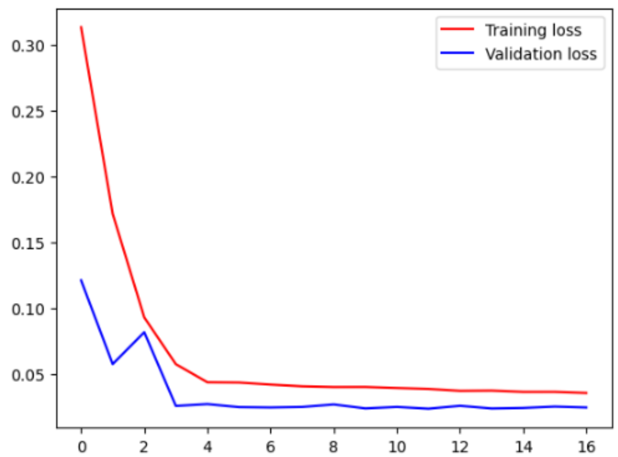

Figure 4.13: Loss function history of Solana in train & test

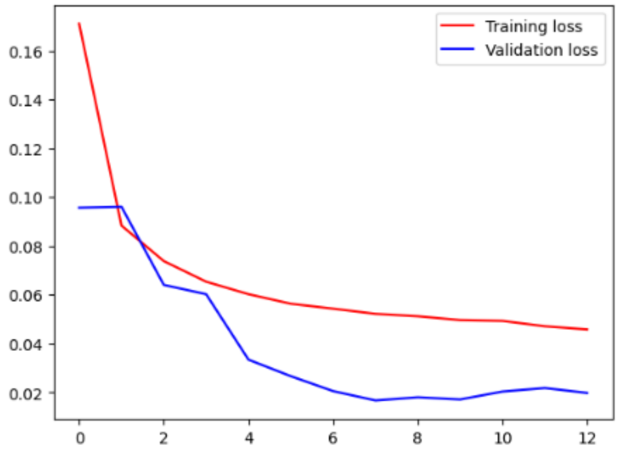

Figure 4.14: Loss function history of Polkadot in train & test

Now that we have an LSTM model which has been trained on past data, we are able to forecast cryptocurrency prices for the next days. According to the dataset used to build the neural network model, the final historical data that we have is the daily close price on September 01, 2022. Since the goal is to predict the five cryptocurrency prices for the following thirty days, we are now looking beyond that specific last close value. Therefore, we are using the lookback period to forecast the future prices on the next days. In this specific case, the lookback period is set to fifteen days, meaning that we are forecasting the digital assets' prices for the next thirty days using data on closing prices from the fifteen days prior. As LSTM requires the input to be fed into its model, it is to note that the test data has been reshaped into a three-dimensional array in the form of samples, timesteps, and features. Then, we are utilizing the last fifteen elements in the three-dimensional tensor. As a result, we are repeating this procedure thirty times, with each iteration producing the forecasted close price for the subsequent thirty days in a row. Finally, as illustrated in Figures 4.15 – 4.19, we are creating a graph plotting the whole real closing price period from the 1st of January 2021 to the 1st of September 2022 adding the predicted next thirty days. What emerges from the plots, according to the LSTM model's forecasts, is that a general rise in price for all five cryptocurrencies is expected for the next 30 days after the 1st of September 2022.

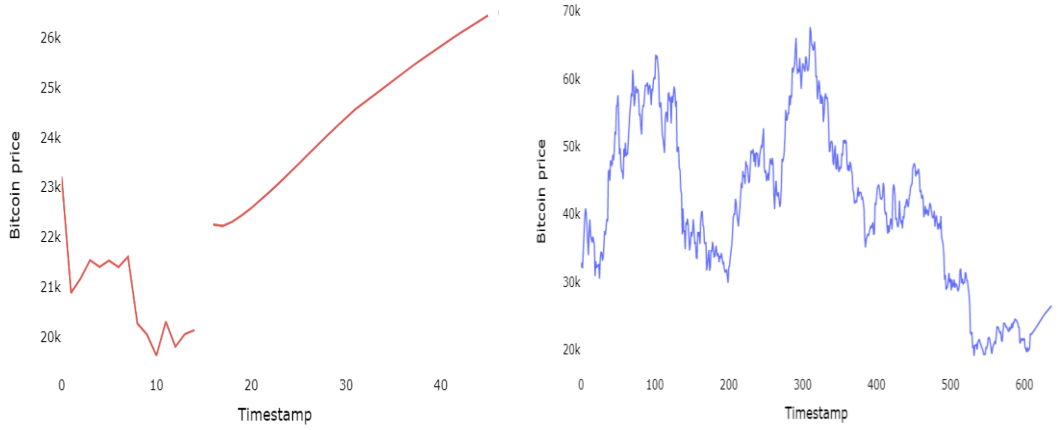

Figure 4.15: Prediction of the next 30 days Bitcoin close price

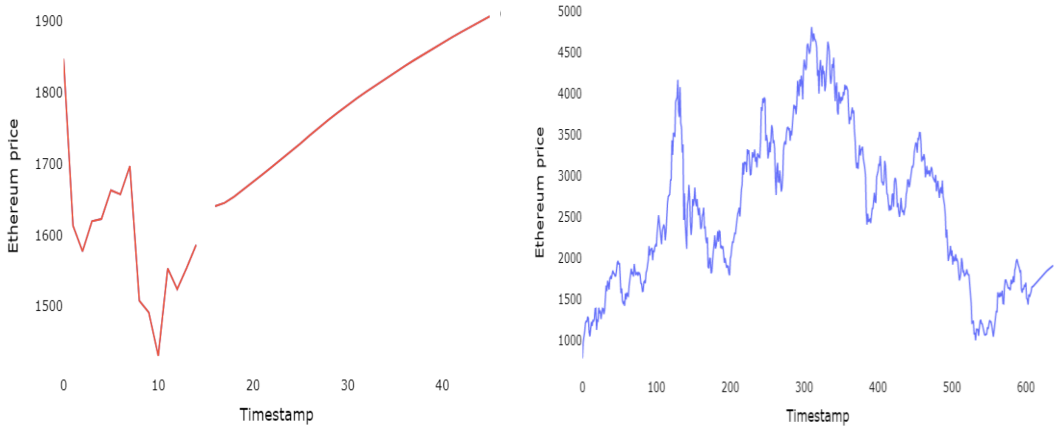

Figure 4.16: Prediction of the next 30 days Ethereum close price

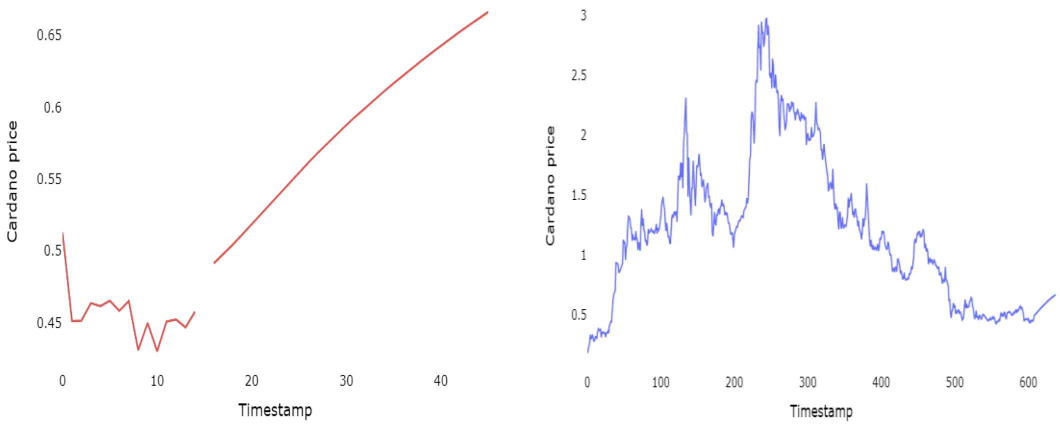

Figure 4.17: Prediction of the next 30 days Cardano close price

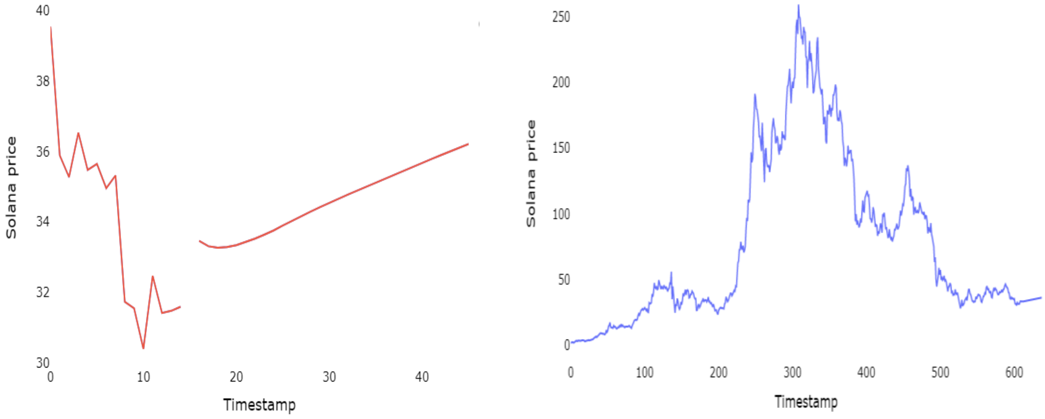

Figure 4.18: Prediction of the next 30 days Solana close price

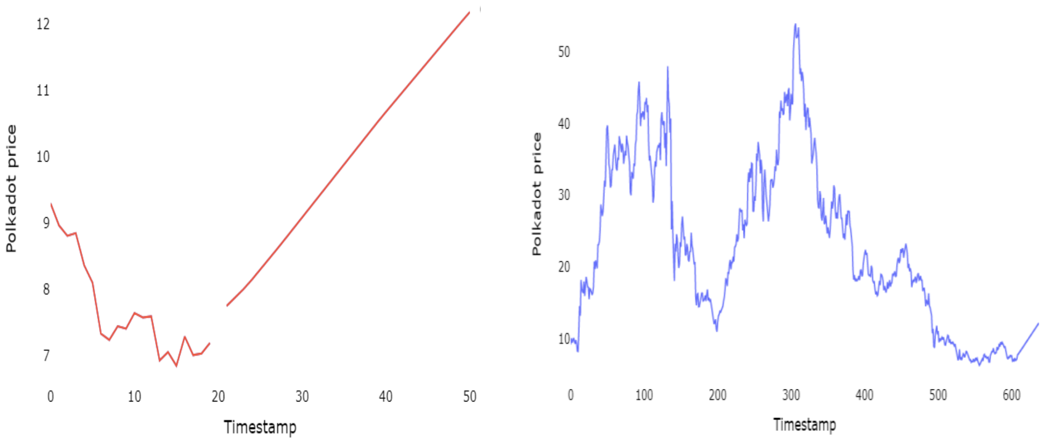

Figure 4.19: Prediction of the next 30 days Polkadot close price

In order to study the performance of the LSTM technique, we experiment with different accuracy measures. Tables 4.2 and 4.3 consist of the scores of MSE, MAE, RMSE and R^2 for the training and testing data. The aspect which emerges from the observation of the two tables is that the MSE, MAE and RMSE losses achieved for test data are lesser compared to those for train data. This means that the LSTM approach is quite underfitting; one reason of this condition could derive from an exceed in preventing the model proposed from overfitting the data. To effectively evaluate the goodness in fit of the LSTM model and to enable a comparison between deep learnings approaches to more traditional techniques in financial forecasting, an ARIMA model is built, as it has been widely employed in price prediction problems[172].

	Bitcoin	*Ethereum*	*Cardano*	*Solana*	*Polkadot*
# of obs. (train/test)	425/183	425/183	425/183	425/183	425/183
MSE	3780.0193	263.4391	0.1406	13.7817	2.9409
MAE	54.2384	14.2083	0.3168	2.9917	1.4697
RMSE	61.4821	16.2312	0.3750	3.7123	1.7149
R^2	0.8353	0.9174	0.9349	0.9640	0.9082

Table 4.2: Accuracy measure values for train data

	Bitcoin	*Ethereum*	*Cardano*	*Solana*	*Polkadot*
# of obs. (train/test)	425/183	425/183	425/183	425/183	425/183
MSE	2720.2331	204.6703	0.0708	8.6913	1.2093
MAE	45.5891	12.7662	0.2370	2.4946	0.9707
RMSE	52.1559	14.3061	0.2661	2.9481	1.0997
R2	0.9062	0.9243	0.9068	0.9180	0.9425

Table 4.3: Accuracy measure values for test data

4.9 ARIMA Model Implementation

It is necessary to take into account some factors in order to successfully conduct a comparison between the ARIMA and LSTM approaches:

- Using the same dataset: the dataset from the 1st of January 2021 to the 1st of September 2022 are used for all five cryptocurrencies price predictions. The same are divided into training and test portions (70% and 30%, respectively) both in ARIMA and LSMT models.

[172] P.F. PAI, C.S. LIN, *A hybrid ARIMA and support vector machines model in stock price forecasting*, Omega, vol. 33, no. 6, pp. 497-505, 2005. Available at: https://www.science-direct-.com/science-/article/abs/pii/S0305048304001082

- Making the same kind of forecast: we are forecasting the cryptocurrency's prices for the next thirty days utilizing data on daily closing prices from the prior fifteen days.
- Compute the same error metrics: the MSE, MAE, RSME and R^2 evaluation parameters are used.

4.9.1 ARIMA Composition

The ARIMA approach is one of the most often used tools for making prediction about time series[173]. This model is widely used in the fields of economics and finance since it is recognised for being effective, durable, and having a strong potential for forecasting the price of the market at a short period[174]. The acronym ARIMA stands for Autoregressive Integrated Moving Average and refers to a more generalised version of a more straightforward auto-regression model called ARMA; hence, it is made up of the AR (Auto-Regressive) and MA (Moving Average) models with the addition of an integration phase (I). Both these models have been used to temporal series data in order to better understand the data or predict the series' future plots (prediction).

The ARIMA model is used in some cases where the data show evidence of non-stationarity. In these cases, it is possible to apply an initial differentiation phase, which corresponds to the model's integrated part (I) more than once to remove the non-stationarity of the function (the trend and seasonality). Therefore, in order to proceed to the next phases, the ARIMA model first transforms non-stationary data into stationary data.

The assumptions behind auto-regressive models are that historical values have an impact on current values. AR models are usually used for the analysis of financial, economic, and other time-varying processes. Therefore, it is possible to create a linear

[173] Y. HUA, *Bitcoin price prediction using ARIMA and LSTM*, Jacobs School of Engineering, University of California San Diego, 2020. Available at: https://www.e3sconferences.org/articles/e3sconf/abs/-2020/78/e3sconf_iseese2020_01050/e3sconf_iseese2020_01050.html

[174] V.S. EDIGER, S. AKAR, *ARIMA forecasting of primary energy demand by fuel in Turkey*, Energy policy, vol. 35, pp. 1701-1708, 2007. Available at: https://www.sciencedirect.com/science/article/-abs/pii/S030-1421506002291

regression model that attempts to predict the value of a dependent variable (current value), given the values that it had in the prior days (past value).

$$Y_t = \alpha + \beta_1 Y_{t-1} + \beta_2 Y_{t-2} + \cdots \beta_p Y_{t-p} + \varepsilon_1$$

The order of the AR model corresponds to the number of days included in the formula. The value of "p" is known as the AR model's order and it refers to the number of Y's lags that may be used as predictors.

MA models assume that the value of the dependent variable on the current day depends on the error terms from the previous days. It is mathematically represented in the following formula.

$$Y_t = \alpha + \varepsilon_t + \theta_1 \varepsilon_{t-1} + \theta_2 \varepsilon_{t-2} + \cdots + \theta_q \varepsilon_{t-q}$$

The "q" value is known as the MA model's order value and it refers to the number of delayed prediction errors (lags) that should be implemented in the ARIMA model.

The ARMA model is just a combination of the AR and MA models[175]:

$$Y_t = \alpha + \beta_1 Y_{t-1} + \beta_2 Y_{t-2} + \cdots \beta_p Y_{t-p} \varepsilon_t \ + \theta_1 \varepsilon_{t-1} + \theta_2 \varepsilon_{t-2} + \cdots + \theta_q \varepsilon_{t-q}$$

The ARIMA model adds a phase of differentiation to the ARMA model since the time series must be differentiated at least once in order to become stationary (if it is not already stationary). Therefore, the value of "d" is the minimum number of differences required to make the series stationary. If the time series is already stationary, then "d" is equal to zero.

4.9.2 Stationarity

After completing the initial pre-processing steps, it's important to check the stationarity of the time series. The Data Frame acquisition and preservation procedures as well as the identification and management of null values used for the implementation of the ARIMA model are the same of those utilized for the LSTM technique. When there is a constant mean and a relatively constant variance, hence

[175] J. CONTRERAS, R. ESPINOLA, F.J. NOGALES, A.J. CONEJO, *ARIMA models to predict next-day electricity prices*, IEEE Transactions on Power Systems, vol. 18, no. 3, pp. 1014-1020, 2003. Available at: https://ieeexplore.ieee.org/document/1216141

these parameters do not change with time, a temporal series is said to be stationary. Stationary time series are easier to be predicted.

There are different methods to determine if a time series is stationary or not; in this case the approach used is the Augmented Dickey-Fuller (ADF) Test. In this test, the time series is regarded as stationary if the resulting value called p-value is less than a threshold of 0.05 and the critical values in the confidence intervals of 1%, 5%, and 10% are the closest to the ADF statistics. For all the five cases, the ADF test has provided p-values that are much larger than the threshold of 0.05 and the value of the ADF statistics was outside the critical ranges of 1%, 5%, and 10%. Therefore, in light of the findings, it can be concluded that all temporal series are not stationary.

As the previous results indicate that the initial time series are non-stationary, it is required to execute transformations to make them stationary. The most typical way is differencing. Differencing is a technique to transform a time series dataset. It is applied to eliminate the series dependence on time, so-called temporal dependence. Structures like patterns and seasonality are included in this. This methodology produces differences between current and previous values, leading to differences between historical series observations.

Different orders of differencing may be employed. For instance, the first order difference uses the transformation $Y_i = Z_i - Z_i - 1$ and addresses linear tendencies. The second order difference confronts quadratic tendencies by leading to an earlier difference starting from the already differentiating series and it is expressed as $Y_i = (Z_i - Z_i - 1) - (Z_i - 1 - Z_i - 2)$. The proper order of differencing is the minimum differencing necessary to obtain the near-stationary series and prevent over-difference.

The data set must be differencing, however this process frequently involves the autocorrelation of the data set, since differencing produces an uncorrelated data set. Autocorrelation describes how the data set values are associated with past values and the ACF plot demonstrates how points, such as the lag variable, are related. To build the ARIMA model, we need to choose how many times we depart from our data set. After a single difference, the data set has significantly become stationary, and the ACF plot demonstrates that the time lag has decreased to zero levels. After two variations, the data set became significantly stationary, but the time lag quickly fell below zero levels, raising the possibility that the series may have been over distinguished. The trend of the data set was eliminated after one differencing, leading it to be close to stationary. The result of lag-1 differencing and the corresponding ACF explain its fast-decaying aspect of the result, which is a way to demonstrate the stationarity of the differencing data. Both ACF and PACF decayed exponentially to zero simultaneously, demonstrating that an ARIMA model may be built with revised data.

4.9.3 Auto ARIMA

It is necessary to first divide the entire data set into training and testing sets before building the ARIMA model and setting its parameters. It has been determined to split the Data Frame in the same way as for the LSTM model: 70% for the training set and 30% for the test set.

The Auto ARIMA function of the Pmdarima library[176] is used to determine the optimal order of the model's parameters without even looking at the ACF and PACF graphs. First, the training set upon which it is based is put into operation. Then, the algorithm performs the Augmented Dickey-Fuller test successfully determining the differencing order (d). In order to find the best model, Auto ARIMA optimises for a certain criterion and returns ARIMA that reduces the value to the absolute minimum. In this case, the value used is the Akaike Information Criterion (AIC). The "AIC" informational criterion evaluates the quality of a group of statistical models between them and it ranks them from best to worst. The model that will be chosen is the one that is neither underfitting nor overfitting. At this point, the model is built by choosing the ideal values for "p," "d," and "q". The parameters set are practically the default ones.

Another library used in the construction of the ARIMA model is statsmodels[177]. It provides functions and classes for the evaluation of various statistical models; hence, to conduct statistical tests and statistical data exploration (Augmented Dickey-Fuller test, visualisation of the ACF and PACF graphs, etc.).

---

[176] Pmdarima [Online]. Available at: https://alkaline-ml.com/pmdarima/modules/generated/pmdarima.-arima.auto_arima.html

[177] Statsmodels [Online]. Available at: https://pypi.org/project/statsmodels/

4.9.4 ARIMA Implementation

The next steps after the development of the ARIMA model are the prediction of future values using the train data set and then, the determination of its accuracy through the comparison between the predictive values and the test data.

In this case, the "create model auto arima()" function is used to create the ARIMA model. It only takes the training set as input, which is then used by the function "auto arima()" to identify the ideal model parameters (p, d, and q). As stated, the majority of the parameters imposed for the construction of the model are the default values. The crucial factor to take into account is the "seasonal" parameter, which is set to True by default. Therefore, since a seasonal ARIMA is not being used, this parameter must be set to False.

Another parameter that has been changed is the "test" one. In place of the default "kpss," the use of the ADF to calculate the difference parameter (d) has been implemented. Other parameters have also been changed (such as "error_action"), but these have no influence on the final result. During this phase, the model is already trained on the set of training data passed to the function.

After determining the order, the model is tested on the remaining 30% test data where the predicted value is obtained.

4.10 Performance Comparison

The test results are based on five different cryptocurrencies: Bitcoin, Ethereum, Cardano, Solana and Polkadot. Even if the time period considered is the same, LSTM and ARIMA models operate on five different types of time series. The movements in price of the five digital assets are mostly similar; the main difference is the value that each individual cryptocurrency assumes in terms of daily closing price, in fact all of them have very different values.

From the visualisation of Tables 4.4 – 4.8, it is possible to figure out that the difference in prediction error between the two models is not very large. In fact, the resulting metrics are very similar in most of the comparisons. For both ARIMA and LSTM, the achieved prediction results can be considered quite good; the calculated errors are not very high, and this means that both models have good precision in terms of accuracy. However, there are still significant room for improvement and optimization of the two models. In particular, this can be seen in the percentage result of R^2, which consistently yields high values, averaging about 90% of value. Since R-squared values range from 0% to 100%, an R-Squared value of 90% would indicate that 90% of the variance of the dependent variable being studied is explained by the variance of the independent variable. Taking into account the various assessment factors, the LSTM model performs better than the ARIMA model, even though the results are not drastically different.

The LSTM proposed model in this study proved its superiority over the ARIMA model by obtaining the lowest MSE, MAE, RMSE and the highest R-squared in the cryptocurrency price forecast test for all the five digital assets. The results show that the LSTM neural network technique fit reasonably well with the data but do not outperform the ARIMA approach by much.

Moreover, LSTM modelling does not need the input dataset to be stationary, as required by the ARIMA-type models. Such makes LSTM a potential candidate in time-series forecasts with the possibility of encountering historical shocks across units and layers through the memory cell at each sequence, which is the time steps in time-series context.

Bitcoin				
	MSE	MAE	RMSE	R^2
LSTM	2720.2331	45.5891	52.1559	0.9062
ARIMA	3018.6366	46.2167	54.9421	0.8955

Table 4.4: Model results for Bitcoin test data

Ethereum				
	MSE	MAE	RMSE	R^2
LSTM	204.6703	12.7662	14.3061	0.9243
ARIMA	256.2311	13.9497	16.0072	0.8943

Table 4.5: Model results for Ethereum test data

Cardano				
	MSE	MAE	RMSE	R^2
LSTM	0.0708	0.2370	0.2661	0.9068
ARIMA	0.1016	0.2674	0.3188	0.8845

Table 4.6.: Model results for Cardano test data

Solana				
	MSE	MAE	RMSE	R^2
LSTM	8.6913	2.4946	2.9481	0.9180
ARIMA	10.8604	2.8142	3.2955	0.8932

Table 4.7: Model results for Solana test data

Polkadot				
	MSE	MAE	RMSE	R^2
LSTM	1.2093	0.9707	1.0997	0.9425
ARIMA	1.8774	1.1719	1.3702	0.8934

Table 4.8: Model results for Polkadot test data

4.11 General Evaluation

In a technical analysis, parameters like the Moving Average (MA), Exponential Moving Average (EMA), Moving Average Convergence/Divergence (MACD), and Relative Strength Index (RSI) are frequently used to help analysts better understand market movements. None of these methodologies has demonstrated to be a consistently accurate prediction tool. However, a variety of cutting-edge techniques exist for price prediction in the financial market, including: Randomwalk (RW), Linear and Multilinear Regression (LR, MLR), Auto-Regressive Integrated Moving Average (ARIMA), Generalized Auto-Regressive Conditional Heteroscedasticity (GARCH), Artificial Neural Networks (ANNs), K-Nearest Neighbors (KNN), Long Short-Term Memory (LSTM), and others. Each of these techniques can have pros and cons.

In a study led by Kenneth Page[178], it is demonstrated that ARIMA and neural networks are the two most effective machine learning models for predicting Amazon's market price. Other algorithms that have been compared include GARCH, Prophet, KNN, and recurrent neural networks. One of the most popular and widely used technologies for financial forecasting is ARIMA. In fact, the experimental values obtained with ARIMA model in [37] have demonstrated the capacity to produce better error results with a short prediction period for the forecasting of Bitcoin price[179]. In contrast, due to their inherent ability to adapt on data, artificial neural network (ANN) models may be very effective for modelling predictive data, according to Mangla et al.[180]. Recurrent neural networks (RNNs) are an example of artificial neural networks that employ feedback connections. Numerous RNN model types are employed in the forecasting of financial time series. Long Short-Term Memory (LSTM) models are one example. As demonstrated in the study [39], they are able to forecast financial time series by avoiding long-term dependency issues[181].

The reason why it was decided to compare ARIMA and LSTM approaches in this research is because these two techniques are widely used in the economic and financial fields. Both the models are successful in achieving a good level of prediction accuracy, and sometimes it may not be so obvious to understand which of the two is the best. In some cases, it is ARIMA to outperform LSTM model[182], whereas in other cases, the LSTM technique performs better than the ARIMA approach[183]. In [42], the authors compare ARIMA, FBProphet, and XGBoost approaches in their ability to predict the

[178] KENNETH. PAGE, *Stock Price Forecasting Using Time Series Analysis, Machine Learning and single layer neural network Models*, 2019. Available at: https://rpubs.com/kapage/523169

[179] T.K. TOAI, R. SENKERIK, I. ZELINKA, A. ULRICH, V.T.X. HANH, V.M. HUAN, *ARIMA for Short-Term and LSTM for Long-Term in Daily Bitcoin Price Prediction*, Artificial Intelligence and Soft Computing, Springer International Publishing, 2023.

[180] N. MANGLA, A. BHAT, G. AVARBRATHA, N. BHAT, *Bitcoin Price Prediction Using Machine Learning*, International Journal of Information and Computer Science, Volume 6, Issue 5, 2019. Available at: https://www.ijics.com/gallery/52-may-1142.pdf.

[181] S.J. BROWN, W.N. GOETZMANN, A. KUMAR, *The Dow Theory: William Peter Hamilton's Track Record Reconsidered*, The Journal of Finance, Vol. 53, No. 4, Papers and Proceedings of the Fifty-Eighth Annual Meeting of the American Finance Association, Chicago, Illinois, pp. 1311-1333, 1998. Available at: https://www.jstor.org/stable/117403

[182] J.H. HAN, *Comparing Models for Time Series Analysis*, University of Pennsylvania, 2018. Available at: https://core.ac.uk/download/pdf/219378739.pdf

[183] S.S. NAMINI, A.S. NAMIN, *Forecasting Economics and Financial Time Series: ARIMA vs. LSTM*, Cornell University, 2018. Available at: https://arxiv.org/abs/1803.06386

cryptocurrency markets[184]. The models were assessed using Bitcoin historical data with RMSE, MAE, and R^2 as the performance indicators. ARIMA had the greatest performance with an RMSE of 322.4. While in the research conducted by Çıbıkdiken, et al. [185] the LSTM deep learning algorithm for Bitcoin price prediction is compared with the ARIMA time series model. Also in this case, the results are good; the conclusions show that approximately MAPE 1.40% with LSTM. The results obtained in this study are quite good. They are based on the daily closing price forecasting for the next thirty days using data on closing prices from the fifteen prior days. For the proposed LSTM model, the average R-Squared error of the five considered cryptocurrencies is around 90%.

However, it is not simple to compare the results obtained through the model proposed with those from other empirical studies; this is mainly due to the use of different types of data. The results of the model forecasting depend a lot on the type of the market being examinate. The cryptocurrency market can be defined as a very volatile market, i.e. the daily price changes are very high; sometimes it can even reach 30% of its value. While if a much less volatile market is analysed, such as the S&P 500 and the NASDAQ where the daily price variation is not very high (on average around 1% or even less), it is easy to understand that the two things are difficult to compare. Even using the same model with different time series might lead to quite different outcomes. Many investment and trading firms are working to develop algorithms and models to be able to calculate predictions in relation to financial markets. Most of the time, these models are not exposed publicly; instead, these companies use them to reap profits based on their projections. Therefore, it is often also challenging to determine the results that have been achieved so far in this field and whether there are any algorithms that can perfectly work.

---

[184] M. IQBAL, M.S. IQBAL, F.H. JASKANI, K. IQBAL, A. HASSAN, *Time-Series Prediction of Cryptocurrency Market using Machine Learning Techniques*, EAI Endorsed Transactions on Creative Technologies, 2021. Available at: https://eudl.eu/pdf/10.4108/eai.7-7-2021.170286

[185] E.S. KARAKOVUN, A.O. CIBIKDIKEN, *Comparison of ARIMA Time Series Model and LSTM Deep Learning Algorithm for Bitcoin Price Forecasting*, The 13th Multidisciplinary Academic Conference in Prague 2018, ed. Czech Republic, pp. 171-180, 2018. Available at: https://www.researchgate.net/publication-/340417228_Comparison_of_ARIMA_Time_Series_Model_and_LSTM_Deep_Learning_Algorithm_for_Bitcoin_Price_Forecasting#:~:text=The%20computed%20value%20of%20RMSE,%2C%204.665%25%20and%200.9505%20respectively

CONCLUSION

The general purpose of this thesis is to determine how effective Long Short-Term Memory models are in cryptocurrency prediction tasks. Although there is a vast amount of prior literature on financial time-series prediction, cryptocurrencies have not been examined that thoroughly. Therefore, this research contributes to the existing literature by evaluating the forecasting power of LSTM method in the cryptocurrency sector. Furthermore, this study tries to increase the credibility and reliability of cryptocurrency price forecasts, for assisting traders and investors in their decision-making process. More in detail, the goal is to evaluate the accuracy of the proposed LSTM model in terms of price prediction for a basket of five cryptocurrencies, for then determining in which of these the method fits best.

To properly assess the goodness in fit of the LSTM model proposed and to permit a comparison of deep learning methods to more conventional financial forecasting methodologies, an ARIMA model is built with the same parameters to obtain a most fair confrontation. Five cryptocurrencies are taken into consideration (Bitcoin, Ethereum, Cardano, Solana, Polkadot) for a time interval which goes from the 1st of January 2021 to the 1st of September 2022. The data collected is pre-processed to normalize and eliminate outliers. After then, the data is split into 70% and 30% for training and testing sets respectively. The LSTM and ARIMA models have been used to analyse and calculate the digital assets price forecasts for the next thirty days using as time step the prior fifteen daily closing prices. From the obtained results, it is possible to conclude that the LSTM model's performance is superior to that of the ARIMA, even though their error margin is not particularly wide. By accumulating experience in this field, better LSTM models could be built, thus also succeeding in significantly outperforming traditional price prediction approaches. The initial purpose of this project has been achieved, but it leaves ample room for the addition of new features. It has been demonstrated that the LSTM model is an overall quite good approach for predicting the prices of digital currencies after it obtained the lowest MSE, MAE, RMSE, and highest R-squared for all five digital assets in comparison with the ARIMA model. Taking into consideration the R2 evaluation parameter, the cryptocurrency on which the LSTM model fitted best is Polkadot, with an R2 value of 94%, the highest among all five cryptocurrencies.

In this thesis the LSTM prediction model is developed using only one variable, which is the daily closing price. As a result, the research is restricted in certain aspects. Enhancing these elements will provide the occasion for researchers to conduct future studies that will increase the accuracy of the results. First, the prediction model's foundation, which simply consider one element, the closing price. But prices are affected by a multitude of factors, including market sentiments, blockchain factors and governmental regulations. Secondly future research can be based on a study of price prediction that considers opening, intraday high, intraday low, and closing prices. A better prediction accuracy could be achieved by using more complex models or by optimizing the hyperparameters. This research is focused on two models ARIMA and LSTM, but cryptocurrency price prediction can be explored using other machine learning models. There are numerous technologies that may be combined with what has already been developed to produce even more precise predictions. One example is "Sentiment Analysis," which is a machine learning technique that determines whether a "sentiment" is positive, negative, or neutral through test analysis. This could be helpful in understanding what investors' intentions are and, consequently, in figuring out what the trend of cryptocurrency market could be. For better results, deep learning models require a large quantity of data. Consequently, a further investigation can potentially be conducted using datasets larger than the 608 observations employed in this study.

APPENDICES

Portions of the code developed for the case study are presented below. The scripts refer to a single digital asset since the procedure remains, with the necessary adjustments, the same even with different cryptocurrencies.

A Packages

1. IMPORT LIBRARIES

```
# First we will import the necessary Library

import os
import pandas as pd
import numpy as np
import math
import datetime as dt

# For evaluation we will use these library

from sklearn.metrics import mean_squared_error, mean_absolute_error, r2_score
from sklearn.preprocessing import MinMaxScaler

# For model building we will use these library

import tensorflow as tf
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense, Dropout
from tensorflow.keras.layers import LSTM

# For Plotting we will use these library

import matplotlib.pyplot as plt
from itertools import cycle
import plotly.graph_objects as go
import plotly.express as px
from plotly.subplots import make_subplots
```

B Functions

2. LOADING DATASET

```
# Load our dataset
maindf = pd.read_csv ('BTC-USD.csv')

print('Total number of days present in the dataset:',maindf.shape[0])
print('Total number of fields present in the dataset:',maindf.shape[1])

maindf.shape

maindf.head()

maindf.tail()

maindf.info()

maindf.describe()
```

Checking for Null values

```
print('Null Values:',maindf.isnull().values.sum())

# Final shape of the dataset after dealing with null values
maindf.shape
```

3. EDA (Explanatory Data Analysis)

```
# Printing the start date and End date of the dataset
sd = maindf.iloc[0][0]
ed = maindf.iloc[-1][0]
print('Starting Date',sd)
print('Ending Date',ed)
```

Overall Analysis from 2014 to 2022

```
maindf['Date'] = pd.to_datetime(maindf['Date'], format='%Y-%m-%d')
y_overall = maindf.loc[(maindf['Date'] >= '2014-09-17') & (maindf['Date'] <='2022-09-01')]
y_overall.drop(y_overall[['Adj Close','Volume']],axis=1)

monthwise=y_overall.groupby(y_overall['Date'].dt.strftime('%B'))[['Open','Close']].mean()
new_order = ['January', 'February', 'March', 'April', 'May', 'June', 'July', 'August',
             'September', 'October', 'November', 'December']
monthwise = monthwise.reindex(new_order, axis=0)
monthwise

names = cycle(['Bitcoin Open Price','Bitcoin Close Price','Bitcoin High Price','Bitcoin Low Price'])
fig = px.line(y_overall, x=y_overall.Date,y=[y_overall['Open'],y_overall['Close'],y_overall['High'],
             y_overall['Low']],labels={'Date': 'Date','value':'Bitcoin value'})
fig.update_layout(title_text='Bitcoin chart from 2014-09-17 to 2022-09-01',
             font_size=15, font_color='black',legend_title_text='Bitcoin Parameters')
```

```
fig.for_each_trace(lambda t:  t.update(name = next(names)))
fig.update_xaxes(showgrid=False)
fig.update_yaxes(showgrid=False)
fig.show()
```

4. Building LSTM model

```
# Lets First Take all the Close Price
closedf = maindf[['Date','Close']]
print("Shape of close dataframe:", closedf.shape)

fig = px.line(closedf, x=closedf.Date, y=closedf.Close,labels={'date':'Date','close':'Close Bitcoin'})
fig.update_traces(marker_line_width=2, opacity=0.8, marker_line_color='orange')
fig.update_layout(title_text='Whole period of timeframe of Bitcoin close price 2014-2022',
plot_bgcolor='white', font_size=15, font_color='black')
fig.update_xaxes(showgrid=False)
fig.update_yaxes(showgrid=False)
fig.show()
```

Now we will take data from 2021-01-01 to 2022-09-01

```
closedf = closedf[closedf['Date'] > '2021-01-01']
close_stock = closedf.copy()
print("Total data for prediction: ",closedf.shape[0])
closed

fig = px.line(closedf, x=closedf.Date,
y=closedf.Close,labels={'date':'Date','close':'Close Bitcoin'})
fig.update_traces(marker_line_width=2, opacity=0.8, marker_line_color='orange')
fig.update_layout(title_text='Considered period to predict Bitcoin close price',
plot_bgcolor='white', font_size=15, font_color='black')
fig.update_xaxes(showgrid=False)
fig.update_yaxes(showgrid=False)
fig.show()
```

Normalizing Data

```
# Deleting date column and normalizing using MinMax Scaler
del closedf['Date']
scaler=MinMaxScaler(feature_range=(0,1))
closedf=scaler.fit_transform(np.array(closedf).reshape(-1,1))
print(closedf.shape)
```

Slicing Data into Training and Testing set

```
# We keep the training set as 70% and 30% testing set
training_size=int(len(closedf)*0.70)
test_size=len(closedf)-training_size
train_data,test_data=closedf[0:training_size,:],closedf[training_size:len(closedf),1]
print("train_data: ", train_data.shape)
print("test_data: ", test_data.shape)
```

Slicing Now we Transform the Close price based on Time-series-analysis forecasting requirement. Here we will take 15 steps.

```
# Convert an array of values into a dataset matrix

def create_dataset(dataset, time_step=1):
    dataX, dataY = [], []
    for i in range(len(dataset)-time_step-1):
        a = dataset[i:(i+time_step), 0]   ###i=0, 0,1,2,3-----99 100
        dataX.append(a)
        dataY.append(dataset[i + time_step, 0])
return np.array(dataX), np.array(dataY)

time_step = 15
X_train, y_train = create_dataset(train_data, time_step)
X_test, y_test = create_dataset(test_data, time_step)

print("X_train: ", X_train.shape)
print("y_train: ", y_train.shape)
print("X_test: ", X_test.shape)
print("y_test", y_test.shape)

# Reshape input to be [samples, time steps, features] which is required for LSTM
X_train =X_train.reshape(X_train.shape[0],X_train.shape[1] , 1)
X_test = X_test.reshape(X_test.shape[0],X_test.shape[1] , 1)

print("X_train: ", X_train.shape)
print("X_test: ", X_test.shape)
```

Actual Model Building

```
model=Sequential()
model.add(LSTM(20,input_shape=(None,1),activation="tanh"))
model.add(Dense(1))
model.compile(loss="mean_absolute_error",optimizer="adam")

from keras.callbacks import EarlyStopping
earlyStop=EarlyStopping(monitor="val_loss",verbose=2,mode='min',patience=5)

history = model.fit(X_train,y_train,validation_data=(X_test,y_test),epochs=200,batch_siz
                    =32,verbose=1,callbacks=[earlyStop])
```

Plotting Loss vs. Validation loss

```
loss = history.history['loss']
val_loss = history.history['val_loss']

epochs = range(len(loss))

plt.plot(epochs, loss, 'r', label='Training loss')
plt.plot(epochs, val_loss, 'b', label='Validation loss')
plt.title('Training and validation loss')
plt.legend(loc=0)
plt.figure()
plt.show()
```

```
### Lets Do the prediction and check performance metrics
train_predict=model.predict(X_train)
test_predict=model.predict(X_test)
train_predict.shape, test_predict.shape
```

Model Evaluation

```
# Transform back to original form
train_predict = scaler.inverse_transform(train_predict)
test_predict = scaler.inverse_transform(test_predict)
original_ytrain = scaler.inverse_transform(y_train.reshape(-1,1))
original_ytest = scaler.inverse_transform(y_test.reshape(-1,1))

# Evaluation metrices RMSE and MAE
print("Train data RMSE: ",
math.sqrt(mean_squared_error(original_ytrain,train_predict)))
print("Train data MSE: ", mean_squared_error(original_ytrain,train_predict))
print("Train data MAE: ", mean_absolute_error(original_ytrain,train_predict))
print("-------------------------------------------------------------------------------------")
print("Test data RMSE: ",
math.sqrt(mean_squared_error(original_ytest,test_predict)))
print("Test data MSE: ", mean_squared_error(original_ytest,test_predict))
print("Test data MAE: ", mean_absolute_error(original_ytest,test_predict))

# R squared score for regression
print("Train data R2 score:", r2_score(original_ytrain, train_predict))
print("Test data R2 score:", r2_score(original_ytest, test_predict))
```

Comparison of original Bitcoin close price and predicted close price

```
# shift train predictions for plotting

look_back=time_step
trainPredictPlot = np.empty_like(closedf)
trainPredictPlot[:, :] = np.nan
trainPredictPlot[look_back:len(train_predict)+look_back, :] = train_predict
print("Train predicted data: ", trainPredictPlot.shape)

# shift test predictions for plotting
testPredictPlot = np.empty_like(closedf)
testPredictPlot[:, :] = np.nan
testPredictPlot[len(train_predict)+(look_back*2)+1:len(closedf)-1, :] = test_predict
print("Test predicted data: ", testPredictPlot.shape)

names = cycle(['Original close price','Train predicted close price','Test predicted close price'])

plotdf = pd.DataFrame({'date': close_stock['Date'],
                       'original_close': close_stock['Close'],
                       'train_predicted_close': trainPredictPlot.reshape(1,-1)[0].tolist(),
                       'test_predicted_close': testPredictPlot.reshape(1,-1)[0].tolist()})

fig = px.line(plotdf,x=plotdf['date'],y=[plotdf['original_close'],plotdf['train_predicted_close'],
          plotdf['test_predicted_close']],labels={'value':'Bitcoin price','date': 'Date'})
```

```
fig.update_layout(title_text='Comparision between original close price vs predicted
close price', plot_bgcolor='white', font_size=15, font_color='black',    legend_title_text='Close
Price')

fig.for_each_trace(lambda t:  t.update(name = next(names)))

fig.update_xaxes(showgrid=False)
fig.update_yaxes(showgrid=False)
fig.show()
```

Predicting next 30 days

```
x_input=test_data[len(test_data)-time_step:].reshape(1,-1)
temp_input=list(x_input)
temp_input=temp_input[0].tolist()

from numpy import array

lst_output=[]
n_steps=time_step
i=0
pred_days = 30
while(i<pred_days):
 if(len(temp_input)>time_step):

  x_input=np.array(temp_input[1:])
  #print("{} day input {}".format(i,x_input))
  x_input = x_input.reshape(1,-1)
  x_input = x_input.reshape((1, n_steps, 1))

  yhat = model.predict(x_input, verbose=0)
  #print("{} day output {}".format(i,yhat))
  temp_input.extend(yhat[0].tolist())
  temp_input=temp_input[1:]
  #print(temp_input)

  lst_output.extend(yhat.tolist())
  i=i+1

else:

  x_input = x_input.reshape((1, n_steps,1))
  yhat = model.predict(x_input, verbose=0)
  temp_input.extend(yhat[0].tolist())

  lst_output.extend(yhat.tolist())
  i=i+1

print("Output of predicted next days: ", len(lst_output))
```

Plotting last 15 days od dataset and next predicted 30 days

```
last_days=np.arange(1,time_step+1)
day_pred=np.arange(time_step+1,time_step+pred_days+1)
print(last_days)
print(day_pred)

temp_mat = np.empty((len(last_days)+pred_days+1,1))
temp_mat[:] = np.nan
temp_mat = temp_mat.reshape(1,-1).tolist()[0]

last_original_days_value = temp_mat
next_predicted_days_value = temp_mat

last_original_days_value[0:time_step+1] =
scaler.inverse_transform(closedf[len(closedf)-time_step:]).reshape(1,-1).tolist()[0]
next_predicted_days_value[time_step+1:] =
scaler.inverse_transform(np.array(lst_output).reshape(-1,1)).reshape(1,1).tolist()[0]

new_pred_plot = pd.DataFrame({'last_original_days_value':last_original_days_value,
                              'next_predicted_days_value':next_predicted_days_value})

names = cycle(['Last 15 days close price','Predicted next 30 days close price'])

fig = px.line(new_pred_plot,x=new_pred_plot.index,
      y=[new_pred_plot['last_original_days_value'],
      new_pred_plot['next_predicted_days_value']],
      labels={'value': 'Bitcoin price','index': 'Timestamp'})

fig.update_layout(title_text='Compare last 15 days vs next 30 days',plot_bgcolor='white',
                  font_size=15, font_color='black',legend_title_text='Close Price')

fig.for_each_trace(lambda t:  t.update(name = next(names)))
fig.update_xaxes(showgrid=False)
fig.update_yaxes(showgrid=False)
fig.show()
```

Plotting entire Closing Bitcoin Price with next 30 days period of prediction

```
lstmdf=closedf.tolist()
lstmdf.extend((np.array(lst_output).reshape(-1,1)).tolist())
lstmdf=scaler.inverse_transform(lstmdf).reshape(1,-1).tolist()[0]

names = cycle(['Close price'])

fig = px.line(lstmdf,labels={'value': 'Bitcoin price','index': 'Timestamp'})
fig.update_layout(title_text='Plotting whole closing Bitcoin price with
                  prediction',plot_bgcolor='white', font_size=15,
                  font_color='black',legend_title_text='Bitcoin')

fig.for_each_trace(lambda t:  t.update(name = next(names)))
fig.update_xaxes(showgrid=False)
fig.update_yaxes(showgrid=False)
fig.show()
```

BIBLIOGRAPHY

ADAMS Richard, KEWELL Beth and PARRY Glenn, *Blockchain for Good? Digital Ledger Technology and Sustainable Development Goals*, in Handbook of Sustainability and Social Science Research. World Sustainability Series. Springer, Cham., (2017). Available at: https://link.springer.com/chapter/10.1007/978-3-319-67122-2_7.

ADCOCK Robert and GRADOJEVIC Nikola, *Non-fundamental, non-parametric bitcoin forecasting*, Physica A: Statistical Mechanics and its Applications, vol. 531, p. 121, (2019). Available at: https://doi.org/10.1016/j.physa.2019.121727.

AGARWAL Saurav, *An Approach of SLA Violation Prediction and QoS Optimization Using Regression Machine Learning Techniques*, University of Windsor, Canada, 2020. Available at: https://core.ac.uk/download/pdf/344950163.pdf.

ALKHODHAIRI Reem K., ALJALHAMI Shahad R., RUSAYNI Norah K., ALSHOBAILI Jowharah F., AL-SHARGABI Amal and ABDULATIF Al-abdulatif, *Bitcoin candlestick prediction with deep neural networks based on real time data*, Computers, Materials and Continua, Vol. 68, pp. 3215-3233, (2021). Available at: https://www.techscience.com/cmc/v68n3/42488.

ARIYO Adebiyi A., ADEWUMI Adewumi O. and AYO Charles K., *Stock price prediction using the arima model*, 2014 UKSim-AMSS 16th International Conference on Computer Modelling and Simulation, Cambridge, UK, pp. 106-112, (2014). Available at: https://ieeexplore.ieee.org/document/7046047.

BASHIR Imran, *Mastering Blockchain: A deep dive into distributed ledgers, consensus protocols, smart contracts, DApps, cryptocurrencies, Ethereum, and more*, Packt, Birmingham – Mumbai, (2020).

BENINCASA Alberto, *An LSTM-based model to Trading Energy Stocks*, Corso di laurea magistrale in Ingegneria Informatica (Computer Engineering), (2020). Available at: https://webthesis.biblio.polito.it/16736/.

BROWN Stephen J., GOETZMANN William N. and KUMAR Alok, *The Dow Theory: William Peter Hamilton's Track Record Reconsidered*, in The Journal of Finance, Vol. 53, No. 4, Papers and Proceedings of the Fifty-Eighth Annual Meeting of the American Finance Association, Chicago, Illinois, pp. 1311-1333, (1998). Available at: https://www.jstor.org/stable/117403.

BROWNLEE Jason, *Optimization for Machine Learning: Finding Function Optima with Python*, Machine Learning Mastery, (2021). Available at: https://machinelearningmastery.com/optimization-for-machine-learning/.

CAO Jian, LI Zhi and LI Jian, *Financial time series forecasting model based on CEEMDAN and LSTM*, in Physica A: Statistical Mechanics and its Applications, Vol. 519, pp. 127-139, (2019). Available at: https://doi.org/10.1016/j.physa.2018.11.061.

CHAUM David, *Blind Signatures for Untraceable Payments*, Advances in Cryptology. Springer, Boston, MA, (1983), pp. 199-203. Available at: https://doi.org/10.1007/978-1-4757-0602-4_18.

CHAUM David, *Untraceable Electronic Mail, Return Addresses, and Digital* Pseudonyms, Advances in Information Security, vol 7. Springer, Boston, MA., (2003), pp. 211-219. Available at: https://doi.org/10.1007/978-1-4615-0239-5_14.

CHEVALLIER Julien, GUEGAN Dominique and GOUTTE Stéphane, *Is it possible to forecast the price of bitcoin?*, Forecasting, Vol. 3, pp. 377-420, (2021). Available at: https://www.mdpi.com/2571-9394/3/2/24.

CHIAP Gianluca, RANALLI Jacopo and BIANCHI Raffaele, *Blockchain, tecnologia e applicazioni per il business*, Hoelpi, Milano, (2020).

CHOLETT Francois, *Deep Learning with Python*, Manning Publications Co., Shelter Island NY, (2018).

CHOWDHURY Reaz, RAHMAN M. Arifur, RAHMAN M. SOHEL and MAHDY M.R.C., *Predicting and Forecasting the Price of Constituents and Index of Cryptocurrency Using Machine Learning*, Cornell University, (2019). Available at: https://arxiv.org/abs/1905.08444.

COMANDINI Gianluca, *Da Zero alla Luna, quando come e perché la Blockchain sta cambiando il mondo*, Flaccovio, Palermo, (2020).

CONTRERAS Javier, ESPINOLA Rosario, NOGALES Francisco J. And CONEJO Antonio J., *ARIMA models to predict next-day electricity prices*, in IEEE Transactions on Power Systems, Vol. 18, no. 3, pp. 1014-1020, (2003). Available at: https://ieeexplore.ieee.org/document/1216141.

CORBET, LUCEY and YAROVAYA, *Date stamping the Bitcoin and Ethereum Bubbles*, Finance Research Letters, Elsevier, vol. 26, pp. 81 – 88, (2018). Available at: https://papers.ssrn.com/sol3/papers.cfm?abstract_id=3079712.

DERBENTSEV Vasily, MATVIYCHUK Andriy and SOLOVIEV Vladimir N., *Forecasting of Cryptocurrency Prices Using Machine Learning*, In Advanced Studies of Financial Technologies and Cryptocurrency Markets, Springer, Berlin/Heidelberg, Germany, (2020), pp. 211-231. Available at: http://elibrary.kdpu.edu.ua/jspui/bitstream/123456789/4137/1/DMS.pdf.

DUCHI John, HAZAN Elad and SINGER Yoram, *Adaptive Subgradient Methods for Online Learning and Stochastic Optimization, in* Journal of Machine Learning Research, Vol. 12, pp. 2121-2159, (2011). Available at: https://dl.acm.org/doi/10.5555/1953048.2021068.

EDIGER Volkan S. and AKAR Sertac, *ARIMA forecasting of primary energy demand by fuel in Turkey*, Energy policy, Vol. 35, Issue 3, pp. 1701-1708, (2007). Available at: https://doi.org/10.1016/j.enpol.2006.05.009.

FISHER Thomas and KRAUSS Christopher, *Deep learning with long short-term memory networks for financial market predictions*, in European Journal of Operational Research, Vol. 270, no. 2, pp. 654-669, (2018). Available at: https://doi.org/10.1016/j.ejor.2017.11.054.

GARAVAGLIA Roberto, *Tutto su Blockchain, Capire la tecnologia e le nuove opportunità*, Hoepli, Milano, (2019).

GARCIA David, TESSONE Claudio J., MAVRODIEV Pavlin and PERONY Nicolas, *The digital traces of bubbles: feedback cycles between socioeconomic signals in the Bitcoin economy*, Journal of the Royal Society Interface, (2014). Available at: https://royalsocietypublishing.org/doi/10.1098/rsif.2014.0623.

GAUTAM Kartik, SHARMA Nitin and KUMAR Praveen, *Empirical Analysis of Current Cryptocurrencies in Different Aspects*, 2020 8th International Conference on Reliability, Infocom Technologies and Optimization (Trends and Future Directions) (ICRITO), Noida, India, (2020), pp. 344-348. Available at: https://ieeexplore.ieee.org/document/9197891.

GERON Aurelien, *Hands-on machine learning with Scikit-Learn, Keras, and TensorFlow: Concepts, tools, and techniques to build intelligent systems,* O'Reilly Media, (2019).

GOH Hui Hwang, HE Ronghui, ZHANG Dongdong, LIU Hui et al., *Short-term wind power prediction based on pre-processing and improved secondary decomposition*, in Journal of Renewable and Sustainable Energy, Vol. 13, Issue 5, (2021). Available at: https://doi.org/10.1063/5.0059809.

GROSSI Enzo and BUSCEMA Massimo, *Introduction to artificial neural networks*, in European Journal of Gastroenerology & Hepatology, Vol. 19(12), pp. 1046-1054, (2007). Available at: https://journals.lww.com/eurojgh/Abstract/2007/12000/Introduction_to_artificial_neural_networks.5.aspx.

HAN Jae Hyuk, *Comparing Models for Time Series Analysis*, University of Pennsylvania, 2018. Available at: https://core.ac.uk/download/pdf/219378739.pdf.

HOCHEREITER Sepp and SCHMIDHUBER Jürgen, *Long Short-term Memory*, in Neural Computation, Vol. 9, no. 8, pp. 1735-1780, (1997). Available at: https://doi.org/10.1162/neco.1997.9.8.1735.

HOCHEREITER Sepp, *The vanishing gradient problem during learning recurrent neural nets and problem solutions*, in International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems, Vol. 6, no. 02, pp. 107–116, (1998). Available at: https://doi.org/10.1142/S0218488598000094.

HODSON Timothy O., *Root-mean-squared error (RMSE) or mean absolute error (MAE): when to use them or not*, U.S. Geological Survey Central Midwest Water Science Center, Urbana, IL, USA, (2022). Available at: https://gmd.copernicus.org/articles/15/5481/2022/gmd-15-5481-2022.pdf.

HUA Yiqing, *Bitcoin price prediction using ARIMA and LSTM*, Jacobs School of Engineering, University of California San Diego, 2020. Available at: https://doi.org/10.1051/e3sconf/202021801050.

IEEE Electron Devices Society, Institute of Electrical and Electronics Engineers, and Vaigai College of Engineering, *Proceeding of the 2018 International Conference on Intelligent Computing and Control Systems (ICICCS)*, IEEE, Piscataway, New Jersey, June 14- 15, (2018). Available at: https://ieeexplore.ieee.org/xpl/conhome/8651405/proceeding.

IQBAL Mahir, IQBAL Muhammad Shuaib, JASKANI Fawwad Hassan, IQBAL Khurum and HASSAN Ali, *Time-Series Prediction of Cryptocurrency Market using Machine Learning Techniques*, EAI Endorsed Transactions on Creative Technologies, Vol. 8, no 28, (2021). Available at: https://eudl.eu/pdf/10.4108/eai.7-7-2021.170286

JANG Huisu and LEE Jaewook, *An empirical study on modelling and prediction of bitcoin prices with Bayesian neural networks based on blockchain information*, in IEEE Access, vol. 6, pp. 5427-5437, (2018). Available at: https://ieeexplore.ieee.org/document/8125674.

JUDMAYER Aljosha, STIFTER Nicholas, KROKBHOLZ Katharina and WEIPPL Edgar, *Blocks and Chains: Introduction to Bitcoin, Cryptocurrencies, and Their Consensus Mechanisms*, Synthesis Lectures on Information Security, Privacy, and Trust, (2017). Available at: https://doi.org/10.1007/978-3-031-02352-1.

KARAKOVUN Ebru and CIBIKDIKEN Ali, *Comparison of ARIMA Time Series Model and LSTM Deep Learning Algorithm for Bitcoin Price Forecasting*, The 13th Multidisciplinary Academic Conference in Prague 2018, ed. Czech Republic, pp. 171-180, (2018). Available at: https://www.researchgate.net/publication-/340417228_Comparison_of_ARIMA_Time_Series_Model_and_LSTM_Deep_Learning_Algorithm_for_Bitcoin_Price_Forecasting#:~:text=The%20computed%20value%20of%20RMSE,%2C%204.665%25%20and%200.9505%20respectively

KIM Ha Young and WON Chang Hyun, *Forecasting the volatility of stock price index: A hybrid model integrating LSTM with multiple GARCH-type models*, in Expert Systems with Applications, Vol. 103, pp. 25-37, (2018). Available at: https://doi.org/10.1016/j.eswa.2018.03.002.

KINGMA Diederik P. and BA Jimmy, *Adam: a method for stochastic optimization*, in International Conference on Learning Representations, (2015). Available at: https://arxiv.org/pdf/1412.6980.pdf.

KITA Eisuke, HARADA Masaaki and MIZUNO Takao, *Application of Bayesian network to stock price prediction*, Artificial Intelligence Research, vol. 1, no. 2, pp. 171-184, (2012). Available at: https://doi.org/10.5430/air.v1n2p171.

KRISTOUFEK Ladislav, *What are the main drivers of the bitcoin price? Evidence from wavelet coherence analysis*, PloS one, (2015). Available at: https://arxiv.org/pdf/1406.0268.pdf.

KUMAR Deepak and RATH S.K., *Predicting the trends of price for Ethereum using deep learning techniques*, In Artificial Intelligence and Evolutionary Computations in Engineering Systems, Springer, (2020), pp. 103–114. Available at: https://link.springer.com/chapter/10.1007/978-981-15-0199-9_9

LAHMIRI and BEKIROS, *Cryptocurrency forecasting with deep learning chaotic neural networks,* in Chaos, Solions and Fractals, Vol. 118, pp. 35-40, (2019). Available at: https://doi.org/10.1016/j.chaos.2018.11.014.

LE Xuan-Hien, HO Hung Viet, LEE Giha and JUNG Sungho, *Application of Long Short-Term Memory (LSTM) Neural Network for Flood Forecasting*, Water 11, no. 7: 1387, (2019). Available at: https://doi.org/10.3390/w11071387.

LI Xiang, PENG Ling, YAO Xiaojing, CUI Shaolong, HU Yuan, YOU Chengzeng and CHI Tianhe, *Long short – term memory neural network for air pollutant concentration predictions: Method development and evaluation*, in Environmental Pollution, Vol. 231, Part 1, pp. 997-1004, (2017). Available at: https://doi.org/10.1016/j.envpol.2017.08.114.

LIU Weibo, WANG Zidong, LIU Xiaohui, ZENG Nianyin, LIU Yurong and ALSAADI Fuad E., *A survey of deep neural network architectures and their applications*, in Neurocomputing, Vol. 234, pp. 11-26, (2017). Available at: https://doi.org/10.1016/j.neucom.2016.12.038.

LIVIERIS Ioannis E., PINTELAS Emmanuel and PINTELAS Panagiotis, *A CNN–LSTM model for gold price time-series forecasting*, Neural Comput. Appl. 32, pp. 17351–17360, (2020). Available at: https://doi.org/10.1007/s00521-020-04867-x.

LIVIERIS Ioannis E., STAVROYIANNIS Stavros, PINTELAS Emmanuel and PINTELAS Panagiotis, *A novel validation framework to enhance deep learning models in time-series forecasting*, Neural Comput & Applic 32, 17149–17167 (2020). Available at: https://doi.org/10.1007/s00521-020-05169-y.

LIVIERIS Ioannis E., STAVROYIANNIS Stavros, PINTELAS Emmanuel and PINTELAS Panagiotis, *Ensemble Deep Learning Models for Forecasting Cryptocurrency Time-Series*, Computer Science, Algorithms, Vol. 13, p. 121, (2020). Available at: https://www.mdpi.com/1999-4893/13/5/121.

LU Wenjie, LI Jiazheng, WANG Jingyang and QIN Lele, *A CNN-BiLSTM-AM method for stock price prediction*, Neural Computing and Applications, 33, pp. 4741-4753, (2021). Available at: https://link.springer.com/article/10.1007/s00521-020-05532-z.

MALSA Nitima, VYAS Vaibhav and GAUTAM Jyoti, *RMSE calculation of LSTM models for predicting prices of different cryptocurrencies*, in International Journal of System Assurance Engineering and Management, (2021). Available at: https://doi.org/10.1007/s13198-021-01431-1.

MANGLA Neha, BHAT Akshay, AVABRATHA Ganesh and BHAT Narayana, *Bitcoin Price Prediction Using Machine Learning*, in International Journal of Information and Computer Science, Volume 6, Issue 5, (2019). Available at: https://www.ijics.com/gallery/52-may-1142.pdf.

MCCULLOCH Warren S. and PITTS Walter, *A logical calculus of the ideas immanent in nervous activity*, in The Bulletin of Mathematical Biophysics, Vol. 5, Issue 4, pp 115-133, (1943). Available at: https://link.springer.com/article/10.1007/bf02478259.

MCNALLY Sean, ROCHE Jason and CATON Simon, *Predicting the Price of Bitcoin Using Machine Learning*, 2018 26th Euromicro International Conference on Parallel, Distributed and Network-based Processing (PDP), Cambridge, UK, pp. 339-343, (2018). Available at: https://ieeexplore.ieee.org/document/8374483.

MITCHELL Tom M., *Machine Learning*, McGraw-Hill, 1997.

NAKANO Masafumi, TAKAHASHI Akihiko and TAKAHASHI Soichiro, *Bitcoin technical trading with artificial neural network*, Physica A: Statistical Mechanics and its Applications, vol. 510, pp. 587 – 609, (2018). Available at: https://doi.org/10.1016/j.physa.2018.07.017.

NARAYANAN Arvind, BONNEAU Joseph, FELTEN Edward, MILLER Andrew and GOLDFEDER Steven, *Bitcoin and cryptocurrency technologies: a comprehensive introduction*, Princeton University Press, (2016).

NASIRTAFRESHI I., *Forecasting cryptocurrency prices using Recurrent Neural Network and Long Short-term Memory*, in Data & Knowledge Engineering, Vol. 139, (2022). Available at: https://doi.org/10.1016/j.datak.2022.102009.

NIKOU Mahla, MANSOURFAR Gholamreza and BAGHERZADEH Jamshid, *Stock price prediction using deep learning algorithm and its comparison with machine learning algorithms*, in Intelligent Systems in Accounting, Finance and Management, Vol. 26, Issue 4, pp.164-167. (2019). Available at: https://onlinelibrary.wiley.com/doi/10.1002/isaf.1459.

NWANKPA Chigozie, IJOMAH Winifred, GCHAGAN Anthony and MARSHALL Stephen, *Activation Functions: Comparison of trends in Practice and Research for Deep Learning*, in arXivLabs, Cornell University, 2018. Available at: https://arxiv.org/pdf/1811.03378.pdf.

PAI Ping-Feng and LIN Chih-Sheng, *A hybrid ARIMA and support vector machines model in stock price forecasting*, in Omega, Vol. 33, no. 6, pp. 497-505, (2005). Available at: https://doi.org/10.1016/j.omega.2004.07.024.

PHALADISAILOED Thearasak and NUMNONDA Thanisa, *Machine Learning Models Comparison for Bitcoin Price Prediction*, 2018 10th International Conference on Information Technology and Electrical Engineering (ICITEE), Bali, Indonesia, pp. 506-511, (2018). Available at: https://ieeexplore.ieee.org/document/8534911.

PINTELAS Emmanuel, LIVIERIS Ioannis E., STAVROYIANNIS Stavros, KOTSILIERIS Theodore and PINTELAS Panagiotis, *Investigating the Problem of Cryptocurrency Price Prediction: A Deep Learning Approach*, In IFIP International Conference on Artificial Intelligence Applications and Innovations, Springer, Berlin/Heidelberg, Germany, pp. 99–110, (2020). Available at: https://link.springer.com/chapter/10.1007/978-3-030-49186-4_9.

RAJPUT Ubaidullah, ABBAS Fizza and OH H., *A solution towards eliminating transaction malleability in bitcoin*, Journal of Information Processing Systems, vol. 14, no. 4, (2018), pp. 837-850. Available at: https://s3.ap-northeast-2.amazonaws.com/journal-home/journal/jips/fullText/20/jips_579.pdf.

ROSENBLATT F., *The perceptron: A probabilistic model for information storage and organization in the brain*, in Psychological Review, Vol. 65(6), pp. 386-408, (1958). Available at: https://doi.org/10.1037/h0042519.

SABOOR Khalid, SABOOR Qurat Ul Ain, HAN Liyan and ZAHID Abdul Saboor, *Predicting the Stock Market using Machine Learning: Long short-term Memory*, in Electronic Research Journal of Engineering, Computer and Applied Sciences, Vol. 2, pp. 202-219, (2020). Available at: https://papers.ssrn.com/sol3/papers.cfm?abstract_id=3810128.

SAMUEL Arthur L., *Some studies in machine learning using the game of checkers,* in IBM Journal of Research and Development, vol. 3, no. 3, pp. 210-229, (1959). Available at: https://ieeexplore.ieee.org/document/5392560.

SCHEIER Christian and TSCHACHER Wolfgang, *Appropriate algorithms for non-linear time series analysis in psychology*, in NonLinear Dynamics in Human Behavior, World Scientific, pp. 27-43, (1996). Available at: https://doi.org/10.1142/9789812830005_0002.

SCHIMIDT Robin M., *Recurrent Neural Networks (RNNs): A gentle Introduction and Overview*, Cornell University, (2019). Available at: https://arxiv.org/pdf/1912.05911.pdf.

SIAMI-NAMINI Sima and NAMIN Akbar Siami, *Forecasting Economics and Financial Time Series: ARIMA vs. LSTM*, Cornell University, (2018). Available at: https://arxiv.org/abs/1803.06386.

STRUGA Kejsi and QIRICI Olti, *Bitcoin Price Prediction with Neural Networks, in* International Conference on Recent Trends and Applications in Computer Science and Information Technology, University of Tirana, (2018). Available at: https://ceur-ws.org/Vol-2280/paper-06.pdf.

SUGOMORI Yusuke, *Java Deep Learning Essentials,* Packt Publishing Ltd, pp. 229-234, (2016).

TOAI Tran Kim, SENKERIK Roman, ZELINKA Ivan, ULRICH Adam, HANH Vo Thi Xuan and HUAN Minh, *ARIMA for Short-Term and LSTM for Long-Term in Daily Bitcoin Price Prediction*, in Artificial Intelligence and Soft Computing, Lecture Notes in Computer Science(), Vol 13588. Springer, Cham., (2023). Available at: https://doi.org/10.1007/978-3-031-23492-7_12.

TSCHORSCH Florian, SCHEUERMANN Bjorn, *Bitcoin and beyond: A technical survey on decentralized digital currencies,* in IEEE Communications Surveys & Tutorials, vol. 18, no. 3, pp. 2084–2123, (2016). Available at: https://ieeexplore.ieee.org/document/7423672.

WHITE Halbert, *Economic prediction using Neural Networks: the case of the IBM daily stock return,* IEEE 1988 International Conference on Neural Networks, San Diego, CA, USA, Vol. 2, pp. 451-458, (1998). Available at: https://ieeexplore.ieee.org/document/23959.

WOOD Gavin, *Ethereum: a secure decentralised generalised transaction ledger,* in Ethereum Project Yellow Paper, 151, pp. 1-32, (2014). Available at: https://gavwood.com/paper.pdf.

YANG Liuqing, LIU Xiao-Yang, LI Xinyi and LI Yinchaun, Price Prediction of Cryptocurrency: An Empirical Study, In: Qiu, M. (eds) Smart Blockchain. SmartBlock 2019. Lecture Notes in Computer Science(), Vol. 11911. Springer, Cham., (2019). Available at: https://doi.org/10.1007/978-3-030-34083-4_13.

YATES Daren, MOORE David S., et al., *The Practice of Statistics*, W.H. Freeman and Company, New York, (1999).

ZEILER Matthew D., *ADADELTA: An Adaptive Learning Rate Method*, Google Inc., New York University, USA, (2012). Available at: https://arxiv.org/pdf/1212.5701.pdf.

ZHANG Aston, LIPTON Zachary C., LI Mu and SMOLA Alexander, *Dive into Deep Learning (Release 0.17.4)*, (2022). Available at: https://arxiv.org/abs/2106.11342.

SITOGRAPHY

ACM Digital Library, https://dl.acm.org/

Analytics Vidhya, https://www.analyticsvidhya.com/

ArXiv, https://arxiv.org/

Baeldung, https://www.baeldung.com/

Binance Academy, https://academy.binance.com/en

Bitpanda Academy, https://www.bitpanda.com/academy/en/

Blockgeeks, https://blockgeeks.com/

Canadian Association for Neuroscience, https://can-acn.org/

Cardano Foundation, https://cardano.org/

Cisco, https://www.cisco.com/site/us/en/index.html

Coinbase Learn, https://www.coinbase.com/it/learn

CoinDesk, https://www.coindesk.com/

CoinMarketCap, https://coinmarketcap.com/

Cointelegraph, https://cointelegraph.com/

Dataversity, https://www.dataversity.net/

Ethereum Foundation, https://ethereum.org/

Forbes, https://www.forbes.com/

Geoscientific Model Development, https://gmd.copernicus.org/

GitHub, https://github.com/

Grayscale, https://grayscale.com/

IBM, https://www.ibm.com/

IEEE Xplore, https://ieeexplore.ieee.org/

Intel, https://www.intel.com/

Investopedia, https://www.investopedia.com/

Javatpoint, https://www.javatpoint.com/

Keras, https://keras.io/

MathWorks, https://www.mathworks.com/

Medium, https://medium.com/

MIT Technology Review, https://www.technologyreview.com/

NVIDIA Developer, https://developer.nvidia.com/

Polkadot network, https://polkadot.network/

Python Package Index, https://pypi.org/

ResearchGate, https://www.researchgate.net/

Reuters, https://www.reuters.com/

RPubs, https://rpubs.com/

ScienceDirect, https://www.sciencedirect.com/

SSRN, https://www.ssrn.com/

TechTarget, https://www.techtarget.com/

TensorFlow, https://www.tensorflow.org/

The World Economic Forum, https://www.weforum.org/

Towards Data Science, https://towardsdatascience.com/

Yahoo Finance, https://finance.yahoo.com/