****

**Model for Predicting the Price of Bitcoin Crypto Currency Using Long Short-Term Memory Recurrent Neural Network**

**KIPKOECH NGETICH JOHNSTONE**

**SC200/0179/2019**

**A report submitted to the Department of Computer Science, School of Computing and Information Technology, Murang'a University of Technology in partial completion of the requirements for the award of a Bachelor's degree in Computer Science.**

# DECLARATION

This idea is entirely original to me and has never before been presented to the school of computer science and information technology at Murang'a University of Technology for the award of a bachelor's degree in computer science. Without my approval, no part of this report may be copied.

**NAME:** KIPKOECH NGETICH JOHNSTONE

**Reg No**: SC200/0179/2019

.…………………………… ……………………

SIGN DATE

SUPERVISOR: **MR. TIRUS MUYA**

………………………. ……………………….

SIGN DATE

Department of Computer Science

School of Computing and Information Technology

Murang’a University of Technology

# DEDICATION

**To the entire software development community**,

I would like to take a moment to express my sincere appreciation and gratitude for your unwavering support, knowledge sharing, and encouragement throughout the development of my computing project.

Your contributions to the field of software development have been instrumental in shaping the industry and inspiring countless developers like myself to push the boundaries of what is possible. Your willingness to share your knowledge and experiences has been a source of inspiration and motivation for me, and I am sure for many others as well.

As I reflect on my journey in software development, I am humbled by the immense support I have received from this vibrant community. From online forums to conferences, meetups, and open-source communities, I have found a wealth of resources and support that have helped me to grow both personally and professionally.

With your support, I was able to complete my computing project successfully, and I could not have done it without your contributions to the software development community. I am proud to be a part of this dynamic and passionate community and look forward to contributing my own experiences to help others in the future.

Thank you once again for your dedication and contributions to software development, and I look forward to continuing to learn and grow alongside you.

# ACKNOWLEDGEMENT

**Dear Mr. Tirus Muya,**

I would like to express my deepest appreciation for your guidance, support, and encouragement throughout the duration of my computing project. Your insightful comments, helpful feedback, and unwavering dedication have been instrumental in my success.

I would also like to extend my gratitude to my friends who provided me with much-needed assistance and motivation during the project. Their willingness to lend a hand and share their knowledge helped me overcome various challenges.

Furthermore, I would like to thank my parents and family for their unwavering support and encouragement. Their belief in my abilities, and their constant encouragement, have been a source of strength and inspiration throughout the project.

Lastly, I would like to acknowledge the contributions of all the individuals who played a role in making my project a success. Your support and assistance have been invaluable, and I could not have achieved this without you.

Thank you all once again for your support and encouragement.

# ABSTRACT

Crypto currency is becoming more and more crucial to the transformation of the financial system as a result of its rising commercial acceptance. Even though a lot of individuals are investing in crypto currencies, little is known about their dynamic properties and predictability, which puts money at risk. This study presents a bitcoin price prediction model using Recurrent Neural Network and Long Short-Term Memory algorithm. Adam is implemented as the learning algorithm.

The model is trained on a dataset obtained from yahoo finance database containing six years of bitcoin historical price data. Due to the volatility of the Bitcoin cryptocurrency prices, the researcher has used an effective pre-processing of the data to improve the accuracy of the forecast. The model achieved an accuracy of 96.2%.

In this study, the researcher has compared this model with other existing models, including Multi-layer Perceptron (MLP), Recurrent Neural Network with Long-Short Memory (RNN-LSTM) and Autoregressive Integrated Moving Average (ARIMA). The comparison analysis shows that the proposed model outperforms these models in terms of accuracy and robustness. The results of this study show that Recurrent Neural Network and Long Short-Term Memory model with Adam as the learning algorithm is a promising approach for predicting future prices of Bitcoin cryptocurrency. This model can be used by investors to make informed decisions in cryptocurrency market.

**TABLE OF CONTENTS**

[**DECLARATION ii**](#_Toc132145186)

[**DEDICATION iii**](#_Toc132145187)

[**ACKNOWLEDGEMENT iv**](#_Toc132145188)

[**ABSTRACT v**](#_Toc132145189)

[**LIST OF FIGURES x**](#_Toc132145190)

[**LIST OF EQUATIONS xi**](#_Toc132145191)

[**DEFINITION OF TERMS xii**](#_Toc132145192)

[**ACRONYMS AND ABBREVIATIONS xiii**](#_Toc132145193)

[**CHAPTER 1 1**](#_Toc132145194)

[Introduction 1](#_Toc132145195)

[1.1: Background Information 1](#_Toc132145196)

[1.2: Problem Statement 2](#_Toc132145197)

[1.3: Objectives 3](#_Toc132145198)

[1.3.1: General Objectives 3](#_Toc132145199)

[1.3.2: Specific Objectives 3](#_Toc132145200)

[1.4: Research questions 3](#_Toc132145201)

[1.5: Significance of the study 4](#_Toc132145202)

[1.6: Scope of the study 4](#_Toc132145203)

[1.7: Limitations 4](#_Toc132145204)

[**CHAPTER 2 5**](#_Toc132145205)

[LITERATURE REVIEW 5](#_Toc132145206)

[2.1 Introduction 5](#_Toc132145207)

[2.2: Survey on existing models 5](#_Toc132145208)

[2.3: Proposed Model: Recurrent Neural Network with Long Short-Term Memory 7](#_Toc132145209)

[2.4: Existing software design and development tools 9](#_Toc132145210)

[2.4.1: Python Programming Language 9](#_Toc132145211)

[2.4.2 Deployment tools 10](#_Toc132145212)

[2.4.3: Justification ..11](#_Toc132145213)

[2.4.3: Conclusion 11](#_Toc132145214)

[**CHAPTER 3 13**](#_Toc132145215)

[RESEARCH METHODOLOGY 13](#_Toc132145216)

[3.1: Introduction 13](#_Toc132145217)

[3.2: Data Collection Techniques 13](#_Toc132145218)

[3.2.1: Interviews 13](#_Toc132145219)

[3.2.2: Questionnaires 14](#_Toc132145220)

[3.2.3: Documents and records 16](#_Toc132145221)

[3.2.5: Justification 16](#_Toc132145222)

[3.3 Software Development Methodologies 17](#_Toc132145223)

[3.3.1: Waterfall Model 17](#_Toc132145224)

[3.3.2: Agile Model 18](#_Toc132145225)

[3.3.3: Prototyping 20](#_Toc132145226)

[3.3.4 Justification 21](#_Toc132145227)

[3.4 System Requirements 21](#_Toc132145228)

[3.4.1: Software Requirements 21](#_Toc132145229)

[3.4.2: Hardware Requirements 22](#_Toc132145230)

[3.4.3: Functional Requirements 22](#_Toc132145231)

[3.4.4: Non-Functional Requirements 22](#_Toc132145232)

[3.5 Conclusion 22](#_Toc132145233)

[**CHAPTER 4 23**](#_Toc132145234)

[System design, Implementation and Testing 23](#_Toc132145235)

[4.1: Introduction 23](#_Toc132145236)

[4.2: System design. 23](#_Toc132145237)

[4.2.1: Logical Design 23](#_Toc132145238)

[Bitcoin price prediction use case model 23](#_Toc132145239)

[4.2.2: Data Collection 24](#_Toc132145241)

[4.2.3: Data Design. 24](#_Toc132145242)

[4.2.4: Process Design. 25](#_Toc132145243)

[4.3: Implementation approaches 25](#_Toc132145244)

[4.3.1: Importing necessary packages and reading the dataset 25](#_Toc132145245)

[4.3.2: Split Train/Test data 26](#_Toc132145246)

[4.3:4 Data-Scaling using MinMaxScaler 26](#_Toc132145247)

[4.3.5: Model development 27](#_Toc132145248)

[4.3.6: Training the model 27](#_Toc132145249)

[4.3.7: Predicting 27](#_Toc132145250)

[4.3.8: Inverting predictions 27](#_Toc132145251)

[4.4.: Prediction for next n days 28](#_Toc132145252)

[4.4.1: The final model saved as JSON file 30](#_Toc132145253)

[4.4.2: User interface design 31](#_Toc132145254)

[4.5: Deploying the model. 32](#_Toc132145255)

[4.5.1: Deploying the model using flask and sample prediction 32](#_Toc132145256)

[4.5.2: Sample prediction 34](#_Toc132145257)

[4.5.3: Evaluation of the Model. 34](#_Toc132145258)

[4.6: Testing Approach 36](#_Toc132145259)

[4.6.1: Functional Testing 36](#_Toc132145260)

[4.6.2: Integration tests 37](#_Toc132145261)

[**CHAPTER 5 38**](#_Toc132145262)

[5.1. Introduction 38](#_Toc132145263)

[5.2: Evaluation and comparison results 38](#_Toc132145264)

[5.2.1: Proposed model evaluation results 38](#_Toc132145265)

[5.2.2: Comparison between the proposed method and existing methods 38](#_Toc132145266)

[5.2.3 Limitations of the proposed model and potential areas for improvement 39](#_Toc132145267)

[5.3: User Documentation 39](#_Toc132145268)

[**CHAPTER 6 41**](#_Toc132145269)

[Conclusions and Future Works 41](#_Toc132145270)

[6.1: Conclusion 41](#_Toc132145271)

[6.2: Future Works 41](#_Toc132145272)

[APPENDICES 43](#_Toc132145273)

[APP 1: Budget 43](#_Toc132145274)

[APP2: Schedule 43](#_Toc132145275)

[References 44](#_Toc132145276)

**TABLE OF FIGURES**

[Table 1: Sample data for the past 5 days 35](#_Toc131020478)

[Table 2: Proposed model evaluation results 38](#_Toc131020479)

[Table 3: Comparison between the proposed method and existing methods 38](#_Toc131020480)

[Table 4: Budget 43](#_Toc131020481)

[Table 5: Schedule 43](#_Toc131020482)

# LIST OF FIGURES

[Figure 1:Bitcoin's price fluctuation since it was invented 2](#_Toc132147242)

[Figure 2: Recurrent Neural Network Architecture 7](#_Toc132147243)

[Figure 3: Long Short-Term Memory architecture 8](#_Toc132147244)

[Figure 4:Process to develop and train the model 9](#_Toc132147245)

[Figure 5:Sample crypto currency price data from yahoo database 16](#_Toc132147246)

[Figure 6: Waterfall model. 17](#_Toc132147247)

[Figure 7:Agile model 19](#_Toc132147248)

[Figure 8:Prototyping model 20](#_Toc132147249)

[Figure 9: Bitcoin price prediction use case model 23](#_Toc132147250)

[Figure 10: Data design using Excel 24](#_Toc132147251)

[Figure 11: Process design diagram 25](#_Toc132147252)

[Figure 12: Reading head data of Bitcoin historical price dataset 26](#_Toc132147253)

[Figure 13: Plot of baseline predictions 28](#_Toc132147254)

[Figure 14: Prediction of Bitcoin Prices for next 15 days 29](#_Toc132147255)

[Figure 15: Final model 30](#_Toc132147256)

[Figure 16: User interface 32](#_Toc132147257)

[Figure 17: Sample prediction of next 5 days 34](#_Toc132147258)

# LIST OF EQUATIONS

[Equation 2‑1 Mean Absolute Error Formulae 8](#_Toc121142256)

# DEFINITION OF TERMS

**Crypto currency:** Any form of money that is issued digitally or virtually, uses encryption to safeguard transactions, lacks a central authority to issue or regulate it, and instead uses a decentralized system to keep track of transactions and create new units.

**Bitcoin:** A digital asset that can be used in peer-to-peer transactions for traditional currency.

**Blockchain:** Distributed database or ledger that is shared by computer network nodes that keeps data electronically in digital format, ensuring a secure and decentralized record of transactions.

**Recurrent neural networks (RNNs)**: A subclass of artificial neural networks in which connections between nodes can form a cycle, allowing output from some nodes to influence input to other nodes in the same network in the future. It can display temporal dynamic behavior as a result of this.

**Artificial Neural Networks**: A group of algorithms that aim to identify underlying connections in a collection of data by simulating how the human brain functions.

**Deep learning:** A subset of machine learning that, in essence, consists of a neural network with three or more layers that tries to mimic the actions of the human brain while being able to "learn" from a lot of data. Additional hidden layers can help to tune and refine for accuracy even if a neural network with only one layer can still make approximation predictions.

**Data preprocessing**: Data manipulation before use in order to improve performance.

# ACRONYMS AND ABBREVIATIONS

**RNN**-Recurrent Neural Networks

**LSTM**-Long Short-Term Memory

**MLP**- Multi-layer Perceptron

**ARIMA**- Autoregressive Integrated Moving Average

**MAE**-Mean Absolute Error

**RMSE**-Root Mean Square Error

**CSV**-Comma Separated Values

**USD**-United States Dollar

# CHAPTER 1

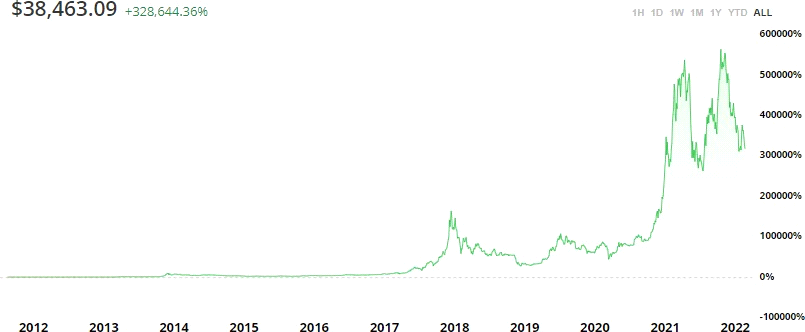
## Introduction

## 1.1: Background Information

Bitcoin is a digital crypto currency introduced in 2008 by anonymous researcher Nakamoto [1]. It was released as an open-source software in 2009 and enabled by the blockchain technology that allows for peer-to-peer transactions secured by cryptography. Investors can buy and sell Bitcoin on an online decentralized network that is not regulated or reliant on a central bank, government or a single administrator [2].

Since Bitcoin is now accepted in more than 40 countries around the world (including the United States and the United Kingdom), new alternative coins like Etherium and Dogecoin have begun to appear. Other crypto currencies, goods, and services are traded using bitcoin [3, 4]. Due to block chain technology, where each electronic coin is encrypted with a unique digital Signature, which makes it easier to track and can be trusted, no hacker has been able to breach it since the debut of this crypto currency. Before passing it on to the next owner, each owner signs a digital hash from the prior transaction and adds the next owner's public key [5, 6].

Because Bitcoin's price is susceptible to sudden and erratic changes, it is difficult to anticipate its future value. Bitcoin cost $1,000 USD in January 2017, increased to $16,000 USD by the end of the year, and then fell sharply to $3,000 USD in 2019. The cost of one bitcoin increased once more to 32818 USD in July 2021 [7, 8]. (Figure 1) displays the historical price fluctuations for bitcoin [15].



**Figure 1:Bitcoin's price fluctuation since it was invented**

According to the study, the Bitcoin market and all crypto currencies are extremely volatile. Therefore, it has become vital to precisely anticipate the value of Bitcoin, which is the most popular cryptocurrency, in order to significantly help today's investing decisions. Deep learning methods and neural network models can be taken into consideration, especially when thinking about price prediction of data that is temporal in nature, as this topic falls within the time series prediction category.

The (LSTM) Recurrent Neural Network technique will be used in this study to forecast Bitcoin prices. The most effective and well-known artificial neural network techniques for finding patterns in data sequences, such as numerical time series data, financial markets, and data from governmental organizations, are LSTM and RNN architectures. RNNs and LSTMs differ from other neural networks in that they take time and sequences into account and contain a temporal component.

## 1.2: Problem Statement

This study's proposed major problem is a model for predicting the price of the Bitcoin crypto currency. The model can assess Bitcoin prices, compare them to historical prices, and accurately estimate future values with a low error rate. The model won't be able to predict future prices precisely, but it may be able to predict the broad trend and the overall direction in which prices are likely to move. As a result, Bitcoin investors can foresee investing risks and opportunities [8].

A Bitcoin trader with $1,000 would purchase 1.52 bitcoin in July 2016 at a price of $656.17 per coin based on a buy and hold decision. That investment would increase in value by 5,805% by 2022 to reach $58,900 [11]. The growth curve is much steeper when zoomed out more (Figure 1). In the previous 11 years, Bitcoin had enormous % growth. A coin cost $13.91 in July 2011, two years after its creation, meaning that $1,000 would buy around 72 Bitcoins which is worth $2,785,737.50, in the first quarter of 2022. This number shows a staggering increase of 278,476.56% [12].

Bitcoin's price has fluctuated dramatically throughout time, reaching a low of $5,165 and a high of $28,990 just in 2020. The impact of Bitcoin's volatility on its investors has always been detrimental, and more traders have lost their hard-earned money [13]. Bitcoin is a highly volatile digital asset that experiences price swings that are unbelievable. This implies that a significant amount of money might be put in this virtual currency and then completely lost in a matter of hours [14]. It is not improbable that a prediction tool will be required. The suggested model will therefore be useful to Bitcoin traders as it seeks to estimate prices over a short time range, such as days, weeks, and months, and aids traders in anticipating investment risks and possibilities.

## 1.3: Objectives

### 1.3.1: General Objectives

To develop a model for Bitcoin crypto currency price prediction using Long Short-Term Memory Recurrent Neural Network.

### 1.3.2: Specific Objectives

1. To analyze existing Bitcoin cryptocurrency price prediction models.
2. To design the proposed Bitcoin market value prediction model.
3. To develop the proposed Bitcoin market value prediction model.
4. To test and validate the newly developed Bitcoin market value prediction model.

## 1.4: Research questions

1. Is crypto currency price prediction a random walk process?
2. What design prototype can be best for Bitcoin price prediction model?
3. Can deep learning efficiently predict the prices of Bitcoin Crypto currencies?
4. What is a proper validation method of Bitcoin cryptocurrency price prediction model?

## 1.5: Significance of the study

Investors may benefit from the proposed model's accurate prediction of the price of bitcoin. Since the price of bitcoin is unstable and it is extensively utilized, it is crucial to forecast its price before making any kind of investment.

## 1.6: Scope of the study

The project seeks to give investors a reliable model for predicting the price of bitcoin. The model examines Bitcoin prices, evaluates them against earlier prices, and forecasts future prices. Using the Python programming language, the researcher will evaluate how well the Recurrent Neural Networks (RNN) model, or long short-term memory cells (LSTM) method, predicts Bitcoin prices. In order to provide a more insightful analysis of the future of Bitcoin crypto currencies, the performance results of this study will be compared to the performance results of other models.

## 1.7: Limitations

1. The cryptocurrency market is extremely unpredictable and cannot ever be completely predicted.

2. The market is based on consumer mood; it is impossible to predict when someone with at least 100 Bitcoins will sell them all at once, sending the price of cryptocurrencies plummeting.

3. No amount of cutting-edge technology can forecast human emotion. A millionaire tweets a rebuttal and the price soars while a negative news feed can make people to believe that Bitcoin is dead. As a result, investors are easily persuaded.

4. Compared to the stock market, currencies, and commodities, crypto currencies have not been around long enough to provide adequate knowledge regarding the resistance and crucial support. Because of this, prediction and practice are challenging.

# CHAPTER 2

## LITERATURE REVIEW

## 2.1 Introduction

It is not unexpected that neural networks have attracted more attention in the area of finance, particularly for time series forecasting, given the extremely non-linear and erratic nature of financial market operations [16]. Since Bitcoin is a relatively new technology, there aren't many models for predicting its price. The researcher will review earlier study on Bitcoin price prediction in this section.

## 2.2: Survey on existing models

Patel provides a hybrid strategy to predicting cryptocurrency prices that is based on LSTM and GRU and only applies to Litecoin and Monero. According to the findings, this method successfully forecasts prices, and it can be applied to many different crypto currency price projections [16]. Authors in McNally created two models for predicting the price of bitcoin using RNN and LSTM, and they contrasted them with an ARIMA model [17]. In the study, the RNN and LSTM models fared better than the ARIMA model.

A deep forward neural network (DFFNN) was created and used in the study by Lahmiri & Bekiros for the modeling and forecasting of Bitcoin high-frequency price data [18]. The Levenberg-Marquardt algorithm and the robust method's impact on DFFNN accuracy were evaluated by the study's authors. The simulation results show that DFFNN trained with the Levenberg-Marquardt algorithm performs better than DFFNN trained with the Powell-Beale restarts approach and DFFNN trained with the robust algorithm. The robust method is faster as a result, suggesting that it might be useful for online training and trading.

The machine learning approach suggested by Sebastio & Godinho [19] predicts the profitability growth of investment strategies and the predictability of the three most important cryptocurrencies (Bitcoin, Ethereum, and Litecoin), with accuracy rates of 80.17% , 89.21% and 91.35% respectively. Ferdiansyah researches bitcoin and stock market forecasts, methodologies, methods, and tools from a range of sources, such as books, journals, and other freely accessible sites [20]. This study shows how to use LSTM to build a prediction model for the bitcoin stock market, although the RMSE of the prediction output is insufficient.

Researchers in Albariqi & Winarko assessed the effectiveness of MLP and RNN models for predicting swings in the price of bitcoin. In both MLP and RNN, long-term price prediction performs better than short-term price prediction. A Multilayer Perceptron with a time window of 3 and 200 epochs, an accuracy of 81.3 percent, a precision of 81 percent, and a recall of 94.7 percent was the model that performed the best in this experiment [21].

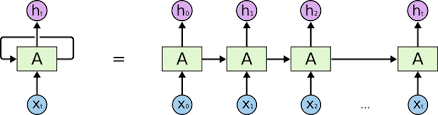
Alkaya presented an optimization strategy for the amount of processing features of hidden layers to predict a stock price measurement using Evolutionary Artificial Neural Networks-EANN to promote the development of Intelligent Systems [22]. Researchers in this study showed that EANN works better with a fixed neural network design and optimal parameters when the number of neurons used in hidden layers is increased.

A deep learning model fed with statistical features calculated from historical benchmark price data is provided by Raşo, H., and Demirci. Their method beats others and achieves high accuracy [23]. Rasheed recommended LSTM and 1D-CNN as two deep learning techniques to improve stock prediction accuracy utilizing three separate datasets [24]. In this work, the researcher looked into the impact of feature extraction using PCA on the accuracy rates of both 1D-CNN and LSTM. This study showed that, despite the 1D-CNN model performing well in terms of computational complexity during training, LSTM with PCA gave substantially better results on the dataset.

In addition, Kemalbay G reported the creation of a machine learning tool based on genetic programming for the forecasting of the national market index for the Istanbul Stock Exchange and contrasted the outcomes with conventional seasonal ARIMA (SARIMA) and ARCH models [25]. These results demonstrate that the GP strategy outperforms conventional approaches in predicting the financial time series data of the XU100 index.

# 2.3: Proposed Model: Recurrent Neural Network with Long Short-Term Memory

Recurrent Neural Networks with Long Short-Term Memory cells are a popular choice for time series prediction tasks such as predicting Bitcoin prices. The main advantage of using LSTMs is to capture long-term dependencies in the input data. The basic idea behind an RNN is to feed the input sequence into a set of interconnected cells that have a feedback loop [26].

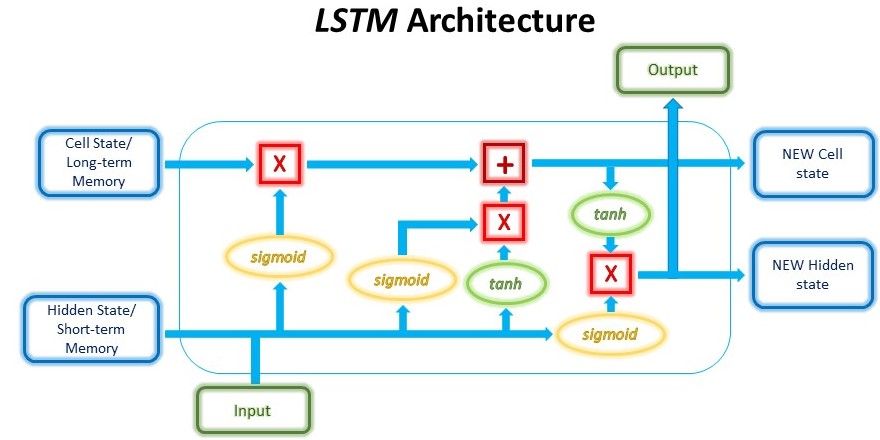


**Figure 2: Recurrent Neural Network Architecture**

The output of each cell is function of the current input and the previous state of the cell. The LSTM is a variant of the RNN that includes a memory cell and a set of gates that control the flow of information into and out of the cell. The memory cell is responsible for storing long-term information, while the gates control the flow of information in and out of the cell [26,27]. The three types of gates in LSTM are;

1. Forget gate: Determines which information to discard from the memory cell.
2. Input gate: Determines which new information to add to the memory cell.
3. Output gate: Determines which information to output from the memory cell.

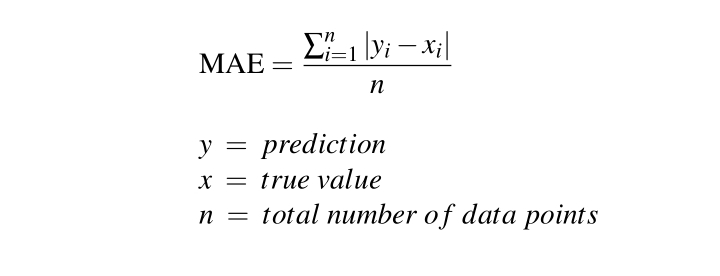
The input and forget gates use a sigmoid activation function to determine which information to keep or discard. The output gate uses a tanh activation function to determine the values to output from the memory cell [27].



**Figure 3: Long Short-Term Memory architecture**

In the context of predicting Bitcoin prices, the LSTM network takes in a sequence of past Bitcoin prices and uses its internal memory cell and gates to predict the next price in the sequence. The network is trained on historical data and learns to identify patterns and trends that can help predict future prices. The sigmoid and tanh functions used in the LSTM architecture help to control the flow of information through the network and allow it to capture complex relationships between past prices and future predictions [28].

The primary methodology used to formulate the outcome in LSTM is Mean Absolute Error.



**Equation 1 :Mean Absolute Error Formulae**

The plan:

1. Crypto currency data overview
2. Time Series
3. Data preprocessing
4. Build and train LSTM model
5. Use the model to predict future Bitcoin price

**LSTM**

Train Dataset

Final Prediction

Preprocessing

Dataset

Model

Test Dataset

Performance Result

**Figure 4:Process to develop and train the model**

## 2.4: Existing software design and development tools

# 2.4.1: Python Programming Language

Python is a dynamically semantic, object-oriented, high level programming language. It is particularly desirable for rapid application development as well as for usage as a scripting or glue language to tie existing components together due to its high-level built-in data structures, dynamic typing, and dynamic binding.

Python libraries include;

**(i) Pandas**

It is an open-source Python package used for machine learning and data science activities. Support for multi-dimensional array is offered.

The following tasks involving the working data are made easy by it: data exploration, data cleaning, and data visualization.

**(ii) Numpy**

NumPy (Numerical Python) is a popular Python library used for scientific computing and data analysis. It provides support for multi-dimensional arrays and matrices, as well as a large collection of mathematical functions to manipulate them. NumPy is an essential library for data scientists and machine learning practitioners due to its fast and efficient array operations.

**(iii) Scikit-Learn**

The program uses Python and offers both supervised and unsupervised learning algorithms. It has effective Recurrent Neural Networks tools, such as Long Short-Term Memory, which the suggested model would make advantage of.

**(iv) Keras**

Keras is a high-level open-source neural network library written in Python. It is designed to make building deep learning models as fast and easy as possible, without sacrificing flexibility or performance. Keras allows to define and train neural networks using a variety of different types of layers and activation functions, as well as providing a range of optimization algorithms and loss functions to fine-tune your models. It also includes a number of pre-trained models and datasets that can be easily used for common machine learning tasks.

**(v) Tensorflow**

TensorFlow is an open-source software library developed by Google for building and training machine learning models. It provides a comprehensive set of tools for building and deploying machine learning models, including a high-level application programming interface for building models using pre-built layers and models, as well as low-level APIs for creating custom models and algorithms. TensorFlow is designed to work with a variety of platforms and devices, making it possible to scale models to large datasets and complex architectures.

# 2.4.2 Deployment tools

**(i) Flask Framework**

Flask is a popular micro web framework for building web applications using the Python programming language. It is classified as a micro framework because it does not require particular tools or libraries, and it has no database abstraction layer, form validation, or any other components where pre-existing third-party libraries provide common functions. Flask provides a simple way to define routes for handling incoming requests and to create views that generate responses to those requests. It also has built-in support for handling cookies, sessions, and static files.

**(ii)Hypertext Markup Language**

HyperText Markup Language (HTML) is a markup language used for creating web pages and web applications. HTML consists of a series of elements, which are the building blocks of web pages. Each element is represented by tags that are enclosed within angle brackets. HTML tags describe the structure and content of a web page, such as headings, paragraphs, links, images, and forms. They also provide additional information for web browsers and other software that may process the web page.

**(iii)Bootstrap**

Bootstrap is a free and open-source front-end web development framework that is designed to help developers create responsive, mobile-first websites and web applications quickly and easily. It was originally developed by Twitter and is now maintained by a large community of developers. Bootstrap provides a collection of CSS and JavaScript components, such as forms, buttons, navigation menus, and modals, that can be easily incorporated into a website or web application. These components are designed to be responsive, meaning they adjust to different screen sizes and devices, so that your website or application looks good on any device, whether it's a desktop computer, tablet, or smartphone.

### 2.4.3: Justification

Without making any original additions, a literature study summarizes the claims and concepts of various prediction systems as well as currently used bitcoin price prediction systems. The suggested system will outperform existing systems with thorough knowledge of the performances of deep learning models exposed in such systems.

The prediction model will be created using Python programming because of its choice of machine learning-specific libraries and frameworks, which streamline the development process and cut down on development time. Python's readable syntax and speedy testing of complex algorithms make it a popular programming language.

# **2.4.3: Conclusion**

Numerous Bitcoin cryptocurrency prediction models have been offered, and each model has unique methods for producing performance results, according to the presented literature assessment. The application of multiple data pre-processing techniques using various hyperparameters, layers, and real-time bitcoin cryptocurrency dataset, however, was not covered by any of the completed research. This motivates the development of the LSTM model to produce more precise Bitcoin price predictions. The LSTM model is employed in assessing predictor potency, predicting an outcome, and forecasting trends.

Deep learning methodologies show to be a crucial component for determining the future of all varieties of cryptocurrencies. In a competitive environment, adopting appropriate stock market tactics is made possible by price prediction in relation to a variety of elements, including prices from prior years.

# CHAPTER 3

## RESEARCH METHODOLOGY

## 3.1: Introduction

A research problem is solved using a specific set of steps and strategies called a methodology. One can assess a study's overall validity and reliability using its methodology. Data collection methods is covered in a research methodology [29]. This study is directed at online investors in the Bitcoin cryptocurrency. A survey was done from at least 100 persons in the vicinity of Murang'a Town in Kenya, which has a population of roughly 1,640,000 persons[30].

## 3.2: Data Collection Techniques

### 3.2.1: Interviews

Open-ended questions are used in interviews, a qualitative research method, to engage participants and obtain information about a topic [31].

**Types of interviews include;**

1. **Personal interviews;** have a greater response rate since respondents are questioned directly and personally.
2. **Telephonic interviews;** which are popular and simple to employ in conjunction with online surveys to do research.
3. **Email/web page interviews;** effective because more people are utilizing the virtual world and online research is expanding**.**

**Sample questions used in the interview**

1. What are Bitcoin and cryptocurrencies?
2. How do you define Bitcoin?

3. Who created Bitcoin?

4. Who is in charge of Bitcoin?

5. Describe a bitcoin wallet.

6. Is Bitcoin the currency of the future?

7. Who decides the price of Bitcoin?

8. What causes fluctuations in Bitcoin prices?

9. How hazardous might it be to make an investment without doing your research?

10. Can stores accept Bitcoin***?***

**Advantages of using Interviews**

1. Ability to gain valuable insights based on the depth of the information gathered and the wisdom.
2. Require only simple equipment and build on conversation skills which researchers already have.
3. Interviews are more flexible.
4. Direct contact at the point of interview means data can be checked for accuracy and relevance.

**Disadvantages of using interviews**

1. Data analysis and preparation can be difficult and time consuming.
2. Consistency and objectivity are hard to achieve.
3. Identity of researcher may affect the statements of the interviewee.
4. Some people may not show up for the interview.

### 3.2.2: Questionnaires

Questionnaire is a set of standardized questions which follow a fixed scheme in order to collect individual data about one or more specific topics [32]. Closed-ended questions were majorly used in this study.

**Sample questionnaires**

1. Have you heard or read about crypto currencies such as Bitcoin or Ethereum?

* A lot
* Some
* Not Much
* Hearing about it now

1. Do you own Crypto currency?

* Yes
* No

1. How much money have you earned or lost trading in cryptocurrency in the last 6 months?

Earned

* More than $1000
* More than $500
* Less than $500

Lost

* More than $1000
* More than $500
* Less than $500

1. In 5 years, do you think Bitcoin will be worth more or less than today?

* Somewhat more
* About the same
* Somewhat less

**Advantages of using questionnaires**

1. Result into wide range of views from users of a system.
2. Questionnaires are the most affordable ways to gather quantitative data.
3. It’s easy and quick to collect results.

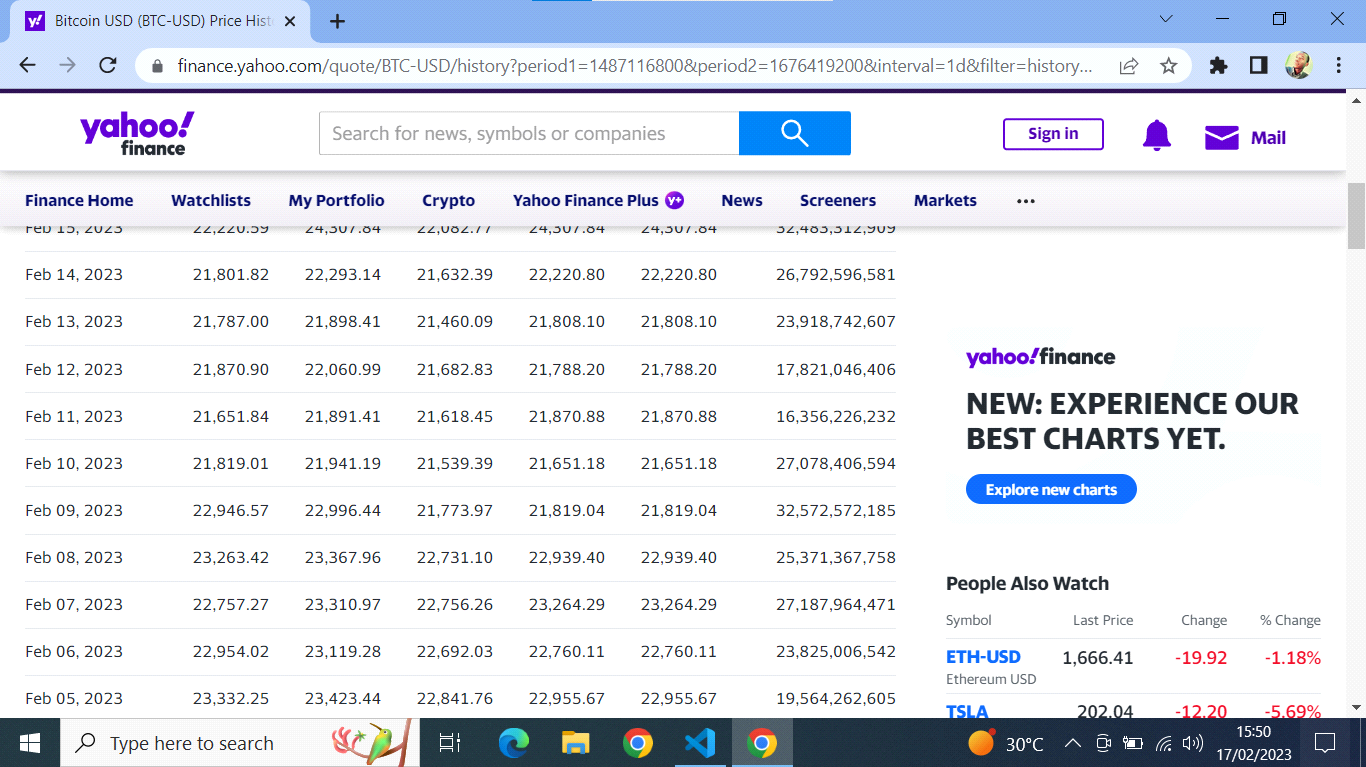
Quantified data can be used to compare and contrast other research and maybe used to measure variance in data.

**Disadvantages of using questionnaires**

1. Chances that some questions will be ignored.
2. Differences in understanding and interpretation.
3. Cannot fully capture emotional responses and feelings.

### 3.2.3: Documents and records

It entails looking over previously collected information from databases, reports, and records that are relevant to the research topic [33]. The researcher will consult various publications on bitcoin as well as Coinmarketcap's daily bitcoin prices. For each month of the year, the data included the daily prices for Bitcoin and other cryptocurrencies. The ability to estimate Bitcoin values makes use of a data set that includes daily prices.



**Figure 5:Sample crypto currency price data from yahoo database**

**Advantages of using Documents and Records**

1. Easy to obtain historical data.
2. Inexpensive way to analyze information.
3. Offers an opportunity for longitudinal analysis.

**Disadvantages of using Documents and Records**

1. Some documents and records are generated for other purposes
2. Exaggeration of documents and records often happen to make good stories.

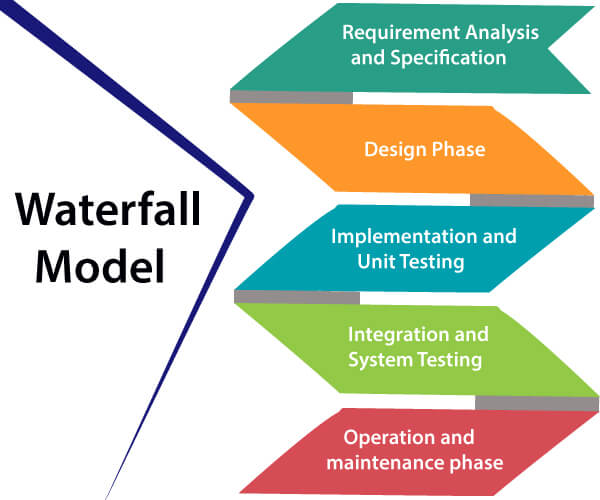
### 3.2.5: Justification

The three methods of data collection (questionnaires, interviews, and documents and records) will be utilized by the researcher to analyze data and information for the suggested model. Data analysis is a critical component of research. Interviewing specific people in an area where cryptocurrency investors congregate allows one to learn things like how much money they make or lose each week, quarter, and month, the factors influencing the rise and fall of the price of Bitcoin, and the potential benefits of using a prediction model to allocate resources like money when it comes to implementation.

## 3.3 Software Development Methodologies

### 3.3.1: Waterfall Model

The Waterfall model is a sequential software development process model that follows a linear and sequential approach. In this model, the development process is divided into several stages or phases, with each stage being completed before moving on to the next [34].



**Figure 6: Waterfall model.**

**1. Requirements analysis and specification phase**: Understanding the precise requirements of the user and accurately capturing them are the goals of this phase. Together, the user and the software developer document all the features, performance needs, and interface specifications of the software. A document called a Software Requirement Specification (SRS) is made and it offers a thorough explanation of what the system will do.

**2. Design Phase**: This stage seeks to convert the SRS requirements into a format that is appropriate for further programming in a computer language. It provides high level and comprehensive design in addition to defining the overall program architecture. A Software Design Document serves as a record of all of this work (SDD).

**3. Implementation and unit testing**: Design is put into practice during this stage. The implementation or coding phase starts once the SDD is finished. The code is carefully reviewed and changed during testing. Small modules are initially tested separately. The interaction between the modules and the flow of the intermediate output are then tested as integrated code.

**4. 4. Integration and System Testing**: As the final product's quality is established during this step, it is extremely important. Users will be happier, maintenance expenses will be lower, and the results will be accurate with a better output. Unit testing establishes each module's effectiveness. The modules' interactions with the system and one another, however, are tested at this phase.

**5. Operation and maintenance phase:** Maintenance is the task performed once the software has been delivered to the user, installed, and operational.

**Circumstances to use Waterfall Model**

1. When the requirements are constant and not changed regularly.
2. A project is short.
3. The situation is calm.
4. Where the tools and technology used is consistent and is not changing.
5. When resources are well prepared and are available to use.

**Advantages of Waterfall model**

1. Simple to implement and the number of resources that are required is minimal.
2. The requirements are simple and explicitly declared.
3. The start and end points for each phase is fixed, which makes it easy to cover progress.
4. The release date for the complete product, as well as its final cost, can be determined before development.

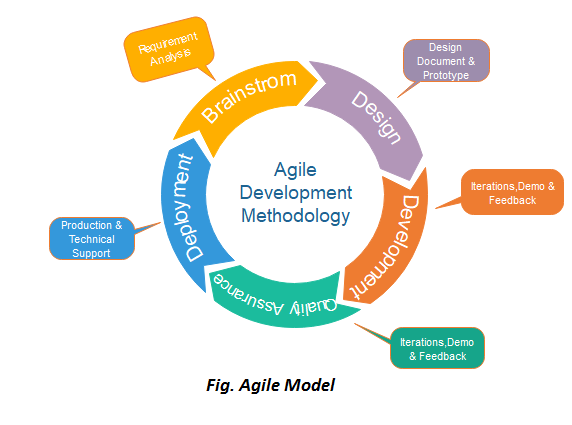
**Disadvantages of Waterfall model**

1. The risk factor is higher, model not suitable for more significant and complex projects.
2. Cannot accept the changes in requirements during development.
3. Risk reduction strategy is difficult to prepare since the testing done at a later stage and it does not allow identifying the challenges.

### 3.3.2: Agile Model

Agile is a term that means quick or adaptable. A software development methodology based on iterative development is referred to as a "agile process model." Agile project management techniques divide work into smaller iterations or pieces without directly including long-term planning. The project's requirements and scope are established at the start of the development phase. Plans for the quantity, length, and scope of each iteration are spelled out in detail in advance.

In the Agile process model, each iteration is viewed as a brief time "frame" that typically lasts one to four weeks. The project risk is reduced and the overall project delivery time requirements are lowered thanks to the project's breakdown into smaller components. Before a working product is shown to the client during each iteration, a team must complete the entire software development life cycle, including planning, requirements analysis, design, coding, and testing [35].



**Figure 7:Agile model**

**Phases of Agile Model:**

**1. Requirements gathering**: Requirements are defined in this stage. Business potential are described, and the project's time and effort requirements are planned. The information can be used to assess the technical and financial viability.

**2. Design the requirements**: Stakeholders are involved in the project identification process to specify requirements. To demonstrate the functionality of new features and demonstrate how they will integrate with an existing model, utilize a user flow diagram or a high-level UML diagram.

**3.** **Construction/ iteration**: The job starts once the requirements are established. On the project, which intends to release a functional product, developers get to work. The product has basic, minimum functionality and will go through numerous stages of refinement.

**4. Testing:** The Quality Assurance team reviews the product's functionality and searches for bugs at this phase.

**5. Deployment:** The development team in this phase releases a product for the user's working environment.

**6. Feedback:** The final step is feedback once the product has been released. Through this, the team collects input on the product and processes it.

**Advantages of Agile Method:**

1. Regular Deliveries
2. Direct communication with customers.
3. Effective design that satisfies the business need.
4. Modifications are welcome at any moment.
5. It cuts down on overall development time..

**Disadvantages of Agile Model:**

1. Lack of formal documents causes uncertainty and makes it possible for team members to interpret important decisions made at any moment during various phases in the wrong way.
2. Because there isn't adequate documentation, maintaining the finished project can be challenging once the project is finished and the developers are assigned to another project.

### 3.3.3: Prototyping

A prototype is produced, tested, and revised until it is an acceptable prototype according to the prototyping paradigm of software development. Additionally, it builds a foundation on which to generate the finished software. It functions well in circumstances where the project requirements are not fully understood. Between the developer and the client, it is an iterative, trial-and-error process [36].

Requirements

Refining Prototype

Implement and Maintain

**Build prototypepeppe**

User Evaluation

Quick Design

**Figure 8:Prototyping model**

**Model phases**

Prototyping Model has following six phases:

**1. Requirements gathering and analysis;** an examination of requirements is the first step in a prototype model. The system requirements are precisely determined during this phase. The system's users are questioned throughout the process to learn what they expect from it.

**2. Quick design;** It is an early design where a straightforward system design is produced. It is not, however, a finished design. It gives the user a quick overview of the system. The prototype development is aided by the rapid design.

**3. Build a Prototype;** based on the data acquired from rapid design, a real prototype is designed in this stage. It is a scaled-down version of the necessary system.

**4. Initial user evaluation;** at this step, the client is given the proposed system for a preliminary assessment. It is useful to identify the working model's advantages and disadvantages. The user's feedback is gathered, and it is given to the developer**.**

**5. Refining prototype;** if the user is unhappy with the existing prototype, it must be improved in accordance with their comments and recommendations. Until all of the user's criteria are satisfied, this phase won't be considered complete. A final system is created based on the final prototype that has been accepted once the user is satisfied with the developed prototype.

**6. Implement Product and Maintain;** after being created based on the ultimate prototype, the final system is rigorously tested before going into production. Regular maintenance is performed on the system to reduce downtime and guard against catastrophic breakdowns.

### 3.3.4 Justification

The waterfall and prototyping models will be mostly used in this study by the researcher since the waterfall model is appropriate for projects with distinct outputs and a review process for each phase. The waterfall model processes and completes phases one at a time without overlapping. The waterfall methodology is simple to use and requires little time and effort.In this study, prototyping is a key model because it aids in organizing a concept and identifying its components that are challenging or impossible to implement.

## 3.4 System Requirements

## 3.4.1: Software Requirements

1. Operating system: Windows 10 and higher version, Linux or MacOS.
2. Programming Language: Python
3. Other softwares: Python installed and Chrome (Jubiter notebook and Kaggle extensions)

## 3.4.2: Hardware Requirements

1. Laptop computer
2. Processor: Intel Core I 5 and above
3. RAM: minimum of 4gb

## 3.4.3: Functional Requirements

Any system must have these features in order to meet business requirements and user acceptance. May consist of;

1. The model can produce and approximatively determine Bitcoin prices.
2. The program can reliably and accurately gather data from Coinmarketcap.
3. Latest Bitcoin prices are used to update the dataset.

## 3.4.4: Non-Functional Requirements

It includes a description of the system's features, traits, and attributes as well as any restrictions that might set the suggested model's boundaries. They are predicated on effectiveness and performance. Non-functional requirements, according to the suggested model, include;

1. The model offers more accurate prediction.
2. The user interface for the model is straightforward.
3. The model performs quickly and effectively.

## 3.5 Conclusion

The forecasting of crypto currency prices is crucial for online trading and investment. Utilizing prediction systems enables traders in the crypto currency market to plan for increased revenue. Long Short-Term Memory has been tested for its ability to forecast time series data, identify crucial variables that affect Bitcoin crypto currency prices, and offer a solution to upcoming trade uncertainty.

# CHAPTER 4

# System design, Implementation and Testing

# 4.1: Introduction

System design is the process of defining the architecture, product design, modules, interfaces, and data for a in order to meet predetermined requirements.

# 4.2: System design.

# The suggested approach aims to create a model that, using historical data, forecasts bitcoin prices for the following n days. The following approach of defining the architecture will be used to construct the suggested system. Iterative in nature, this approach trains the model to obtain the curve that will be most useful for predicting future prices.

# 4.2.1: Logical Design

A system's inputs, outputs, and data flows are all represented abstractly in the system's logical architecture. This is frequently done through modeling, which makes use of an overly abstract (and occasionally graphical) description of the real system.

# Bitcoin price prediction use case model

# 

User

System

**Figure 9: Bitcoin price prediction use case model**

# 4.2.2: Data Collection

The dataset containing bitcoin historical prices was obtained from yahoo finance database. The dataset is in the form of comma separated values(csv) file which can be opened using excel sheet.

# 4.2.3: Data Design.

The dataset consists of 2193 individual data. There are 7 columns in the dataset as follows;

**Date-**Specifies trading date

**Price-**Current price

**Open-**Opening price

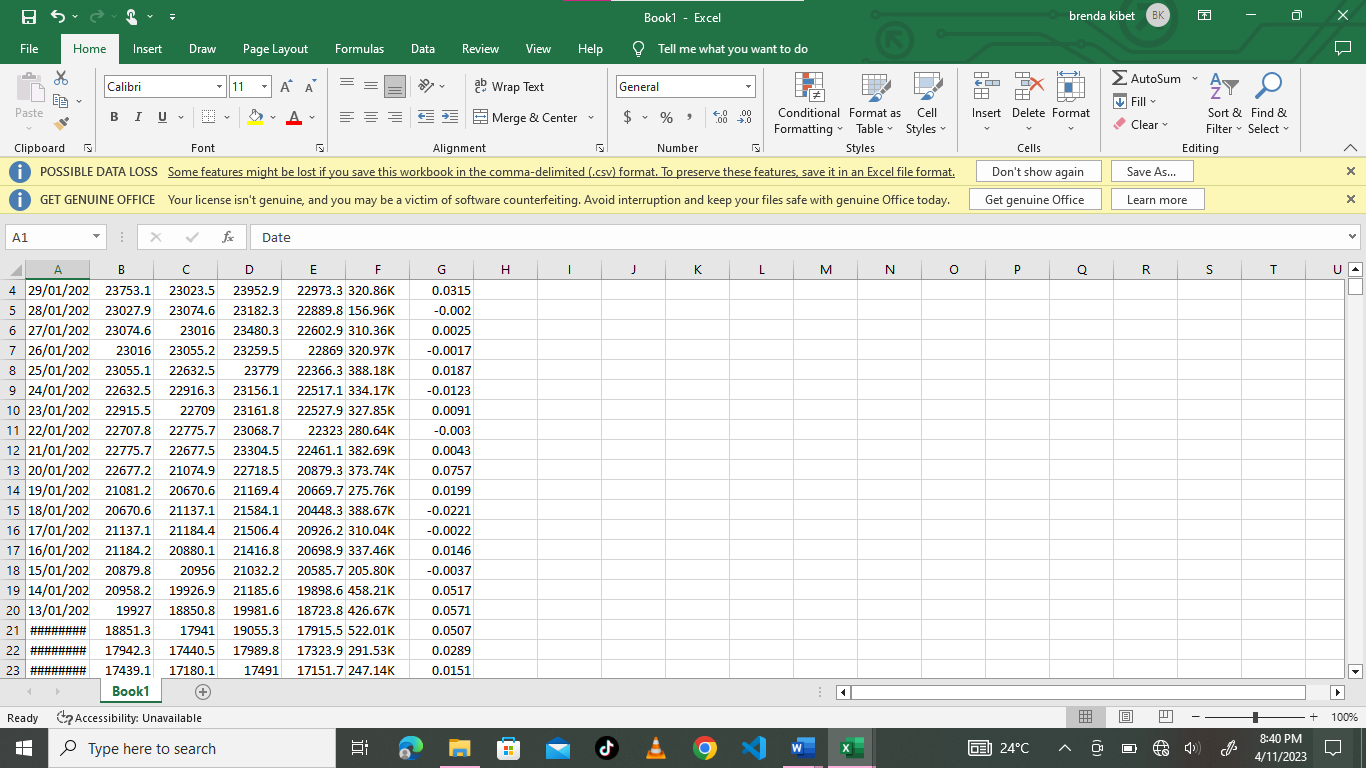
**High-**Maximum price during the day

**Low-** Minimum price during the day

**Volume –** The number of shares that changed hand during a given day.

**Change %-**Percentage change in Bitcoin price over 24 hours.

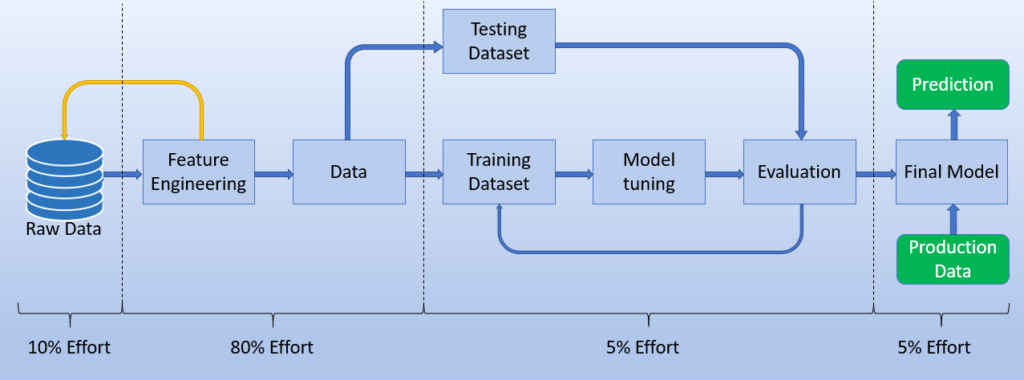
Data is prepared by getting rid of null values and removing some columns. First a list of column names that need to be retained is created.



**Figure 10: Data design using Excel**

# 4.2.4: Process Design.

 Process design is concerned with how data flows into, through, and out of the system as well as how and where information is vetted, secured, and/or changed. The following approach of defining the architecture will be used to construct the suggested system.



**Figure 11: Process design diagram**

# 4.3: Implementation approaches

# Implementation plan is required in order to successfully complete the stated task of creating a model for forecasting sales. A detailed implementation strategy is created to outline the essential tasks required to put solutions into action. The following methods have been used to put the stages outlined in the suggested system design into practice.

# 4.3.1: Importing necessary packages and reading the dataset

import numpy as np

import pandas as pd

import matplotlib.pyplot as plt

**Converting dataset from csv to pickle**

df = pd.read\_csv('btc.csv')

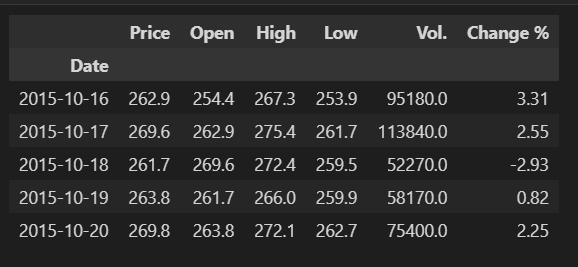
df.to\_pickle('dfe.pkl')

Generally, pickle file data is in binary form and compressed to a smaller size which loads and train faster than comma separated values(csv) file.

**Reading the pickle file.**

df = pd.read\_pickle('./dfe.pkl')

df.head()



**Figure 12: Reading head data of Bitcoin historical price dataset**

# 4.3.2: Split Train/Test data

Once the useful features have been identified, dataset is split into a Train and Test dataset. In the proposed system, model is trained into the Train dataset and tested in the Test dataset. The split is done taking 70% and 30% of the data for train and test respectively.

train\_size = int(len(ds) \* 0.7)

test\_size = len(ds)-train\_size

train = ds[:train\_size]

test = ds[train\_size:]

# 4.3:4 Data-Scaling using MinMaxScaler

from sklearn import preprocessing

from sklearn.preprocessing import MinMaxScaler

min\_max\_scaler = preprocessing.MinMaxScaler(feature\_range=(0, 1))

ds = min\_max\_scaler.fit\_transform(df['Open'].values.reshape(-1, 1))

X value depends on the historical Price data, also I am feeding the model a look\_back value of 15 days, i.e.; it predicts on the basis of previous 15 days data.

look\_back = 15

dataX, dataY = [], []

for i in range(look\_back, train.shape[0]):

    dataX.append(train[i-look\_back:i])

    dataY.append(train[i,0])

x\_train = np.array(dataX)

y\_train = np.array(dataY)

dataX, dataY = [], []

for i in range(look\_back, test.shape[0]):

    dataX.append(test[i-look\_back:i])

    dataY.append(test[i,0])

x\_test = np.array(dataX)

y\_test = np.array(dataY)

# 4.3.5: Model development

from keras import Sequential

from keras.layers import Dense, Dropout, LSTM

from tensorflow.keras.optimizers import Adam

model = Sequential()

model.add(LSTM(150, activation='tanh', input\_shape=(x\_train.shape[1], 1)))

model.add(Dense(1))

adam = Adam(

    learning\_rate=0.00001,

    beta\_1=0.001,

    beta\_2=0.01,

    epsilon=1e-07,

    amsgrad=False,

)

model.compile(loss='mean\_squared\_error', optimizer='adam')

np.random.seed(7)

np.random.seed(7)

# 4.3.6: Training the model

history = model.fit(x\_train, y\_train, validation\_data=(x\_test, y\_test), epochs=50, batch\_size=16, verbose=1)

loss = history.history['loss']

val\_loss = history.history['val\_loss']

# 4.3.7: Predicting

trainPredict = model.predict(x\_train)

testPredict = model.predict(x\_test)

# 4.3.8: Inverting predictions

trainPredict = min\_max\_scaler.inverse\_transform(trainPredict)

trainY = min\_max\_scaler.inverse\_transform([y\_train])

trainY = trainY.reshape(y\_train.shape)

testPredict = min\_max\_scaler.inverse\_transform(testPredict)

testY = min\_max\_scaler.inverse\_transform([y\_test])

testY = testY.reshape(y\_test.shape)

**4.3.9: Plotting baseline and predictions**

plt.figure(figsize=(15,7))

plt.plot(min\_max\_scaler.inverse\_transform(ds))

plt.plot(trainPredictPlot)

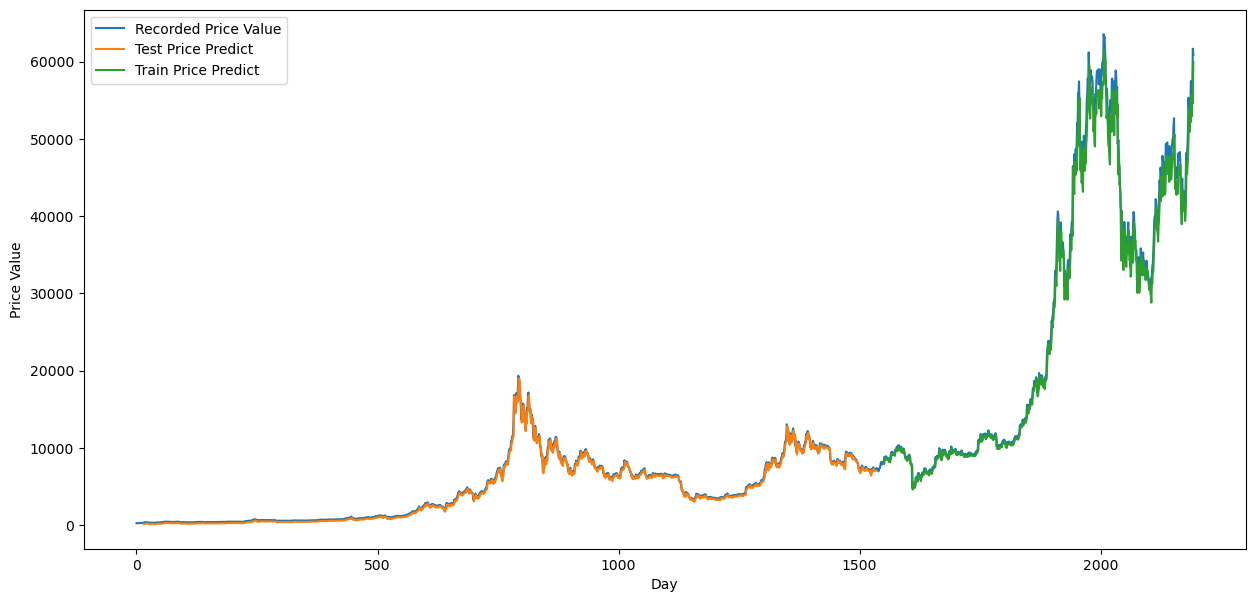
plt.plot(testPredictPlot)

plt.legend(["Recorded Price Value", "Test Price Predict", "Train Price Predict" ], loc=2)

plt.ylabel('Price Value')

plt.xlabel('Day')

plt.show()



**Figure 13: Plot of baseline predictions**

# 4.4.: Prediction for next n days

x\_input = test[len(test)-look\_back:].reshape(1,-1)

temp\_input=list(x\_input)

temp\_input=temp\_input[0].tolist()

from numpy import array

lst\_output=[]

i=0

n=15   # next number of days for which we are predicting

while(i<n):

    x\_input.shape

    if(len(temp\_input)>look\_back):

        x\_input=np.array(temp\_input[1:])

        x\_input=x\_input.reshape(1,-1)

        x\_input = x\_input.reshape((1, look\_back, 1))

        yhat = model.predict(x\_input, verbose=0)

        temp\_input.extend(yhat[0].tolist())

        temp\_input=temp\_input[1:]

        lst\_output.extend(yhat.tolist())

        i=i+1

    else:

        x\_input = x\_input.reshape((1, look\_back,1))

        yhat = model.predict(x\_input, verbose=0)

        temp\_input.extend(yhat[0].tolist())

        lst\_output.extend(yhat.tolist())

        i=i+1

print(lst\_output)

day\_new=np.arange(1, look\_back+1)

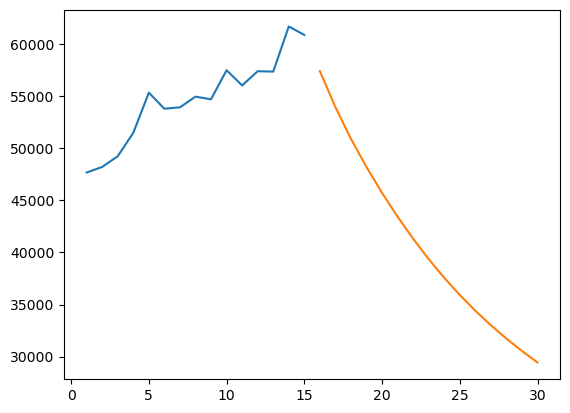
day\_pred=np.arange(look\_back+1, look\_back+n+1)

plt.plot(day\_new, min\_max\_scaler.inverse\_transform(ds[len(ds)-look\_back:]))

plt.plot(day\_pred, min\_max\_scaler.inverse\_transform(lst\_output))

**Prediction of the prices, decreasing:**

Blue curve shows the data for the last 15 days of the given data and the orange curve show the predicted data for the next 15 days



**Figure 14: Prediction of Bitcoin Prices for next 15 days**

Adding the predictions to the larger trend for a better view:

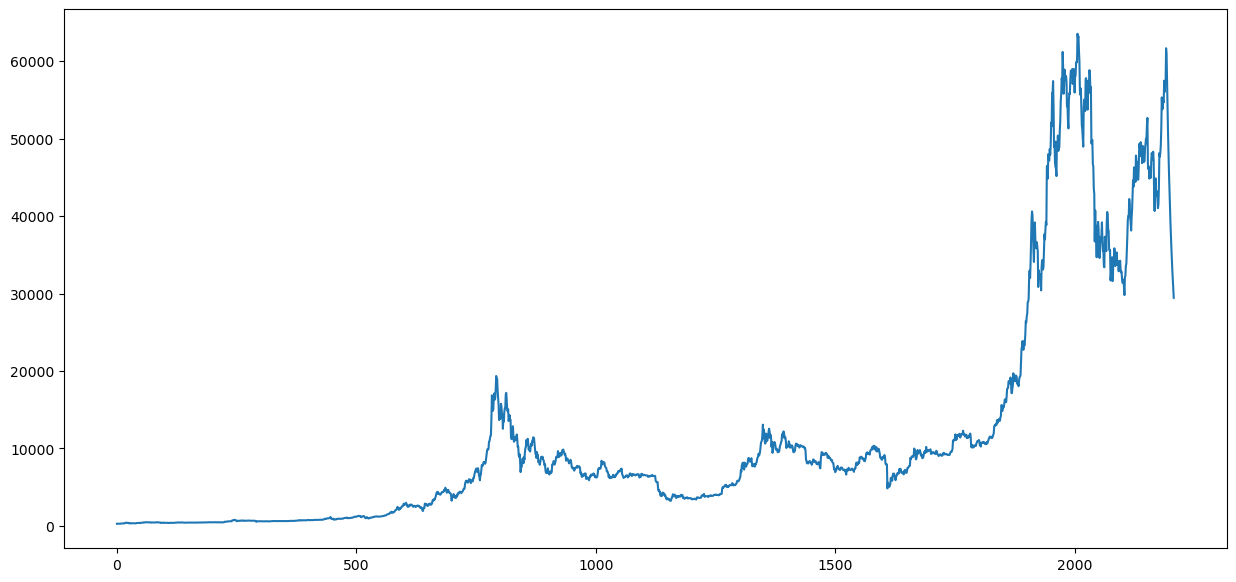
plt.figure(figsize=(15,7))

df0=ds.tolist()

df0.extend(lst\_output)

plt.plot(df0[2000:])

Predicted data of the previous step is appended to the original data( shown from day 2000 onwards)



**Figure 15: Final model**

# 4.4.1: The final model saved as JSON file

The researcher trained the LSTM model on the extended six year Bitcoin Price dataset. The model was then converted to JSON format and written to model.json in the local directory. The network weights are written to model.h5 in the directory.

model\_json = model.to\_json()

with open("model.json", "w") as json\_file:

    json\_file.write(model\_json)

model.save\_weights("model.h5")

print("Saved model to disk")

The model and weight data is loaded from the saved files and a new model is created in app.py.

# 4.4.2: User interface design

The user interface design is purposed to help user on the key functionalities of a system.

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta http-equiv="X-UA-Compatible" content="IE=edge">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <title>Bitcoin Price Prediction</title>

    <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/css/bootstrap.min.css"

        integrity="sha384-Gn5384xqQ1aoWXA+058RXPxPg6fy4IWvTNh0E263XmFcJlSAwiGgFAW/dAiS6JXm" crossorigin="anonymous">

</head>

<body>

    <div class="p-1 mx-auto mt-1" style="width: 38%; background-color: rgb(223, 223, 223);">

        <div class="card border-dark">

            <img class="card-img-top"

                src="https://images.unsplash.com/photo-1590283603385-17ffb3a7f29f?ixlib=rb-1.2.1&ixid=MnwxMjA3fDB8MHxwaG90by1wYWdlfHx8fGVufDB8fHx8&auto=format&fit=crop&w=870&q=80"

                alt="Card image cap">

            <form action="{{ url\_for('predict') }}" method="POST">

                <div class="card-header bg-transparent border-dark font-italic">

                    LSTM model

                </div>

                <div class="card-body">

                    <h1 class="title is-1 mb-3">Bitcoin Price Prediction</h1>

                    <div class="input-group mb-3">

                        <div class="input-group-prepend">

                            <label class="input-group-text" for="date">Date - First day of prediction:</label>

                        </div>

                        <input id="date" type="date" name="date" min="2015-10-31" max="2021-10-16" required

                            class="form-control">

                    </div>

                    <div class="input-group mb-8">

                        <div class="input-group-prepend">

                            <label class="input-group-text" for="days">Number of future days to predict for:</label>

                        </div>

                        <select class="custom-select" name="n" id="days">

                            <option value="1">1</option>

                            <option value="2">2</option>

                            <option value="3">3</option>

                            <option value="4">4</option>

                            <option value="5">5</option>

                            <option value="6">6</option>

                            <option value="7">7</option>

                            <option value="8">8</option>

                            <option value="9">9</option>

                            <option value="10">10</option>

                            <option value="11">11</option>

                            <option value="12">12</option>

                            <option value="13">13</option>

                            <option value="14">14</option>

                            <option value="15">15</option>

                        </select>

                    </div>

                </div>

                <div class="card-footer text-center bg-transparent border-dark">

                    <button type="submit" class="btn btn-info btn-lg" style="width: 35%;">Predict</button>

                    <button type="reset" class="btn btn-danger btn-lg" style="width: 35%;">Reset</button>

                </div>

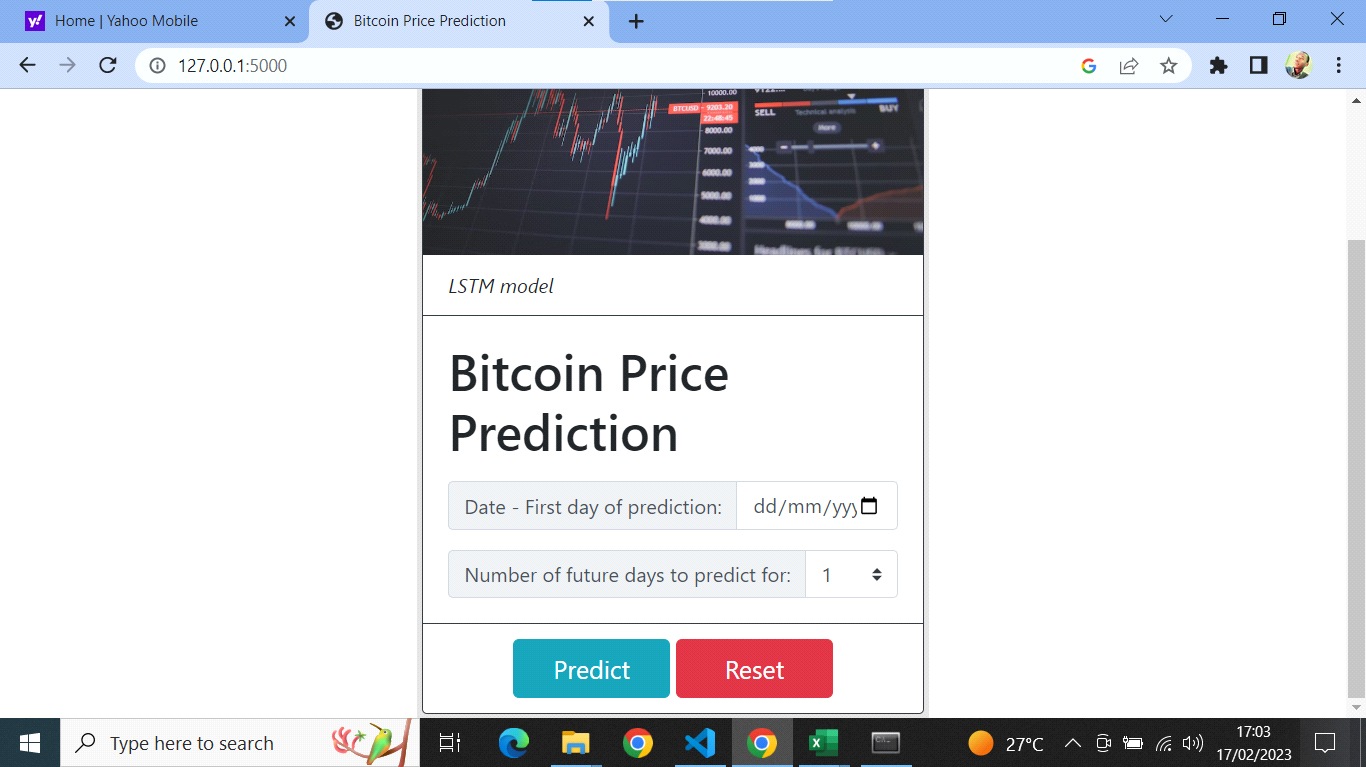
            </form>

        </div>

    </div>

</body>

</html>



**Figure 16: User interface**

# 4.5: Deploying the model.

To implement the user interface, the model uses flask framework. Flask is a micro web framework written in python.

# 4.5.1: Deploying the model using flask and sample prediction

# importing essential libraries

from flask import Flask, render\_template, request, url\_for

from keras.models import model\_from\_json

import numpy as np

import pandas as pd

from sklearn.preprocessing import MinMaxScaler

json\_file = open('model.json', 'r')

loaded\_model\_json = json\_file.read()

json\_file.close()

model = model\_from\_json(loaded\_model\_json)

model.load\_weights("model.h5")

print("Loaded model from disk")

model.compile(loss='mean\_squared\_error', optimizer='adam')

df = pd.read\_pickle('./dfe.pkl')

app= Flask(\_\_name\_\_)

@app.route('/')

def home():

    return render\_template('home.html')

@app.route('/predict', methods=['POST','GET'])

def predict():

    if request.method== 'POST':

        date = request.form['date']

        n = int(request.form['n'])

        loc = df.index.get\_loc(date)

        prev\_data = df.iloc[loc-15:loc].Price.astype(float)

        min\_max\_scaler = MinMaxScaler(feature\_range=(0, 1))

        ds = min\_max\_scaler.fit\_transform(prev\_data.values.reshape(-1, 1))

        ds = ds.reshape(1,15,1)

        look\_back=15

        x\_input = ds[len(ds)-look\_back:].reshape(1,-1)

        temp\_input=list(x\_input)

        temp\_input=temp\_input[0].tolist()

lst\_output=[]

        i=0

        #n=15   # next number of days prediction

        while(i<n):

            if(len(temp\_input)>look\_back):

                x\_input=np.array(temp\_input[1:])

                x\_input=x\_input.reshape(1,-1)

                x\_input = x\_input.reshape((1, look\_back, 1))

                yhat = model.predict(x\_input, verbose=0)

                temp\_input.extend(yhat[0].tolist())

                temp\_input=temp\_input[1:]

                lst\_output.extend(yhat.tolist())

                i=i+1

            else:

                x\_input = x\_input.reshape((1, look\_back,1))

                yhat = model.predict(x\_input, verbose=0)

                temp\_input.extend(yhat[0].tolist())

                lst\_output.extend(yhat.tolist())

                i=i+1

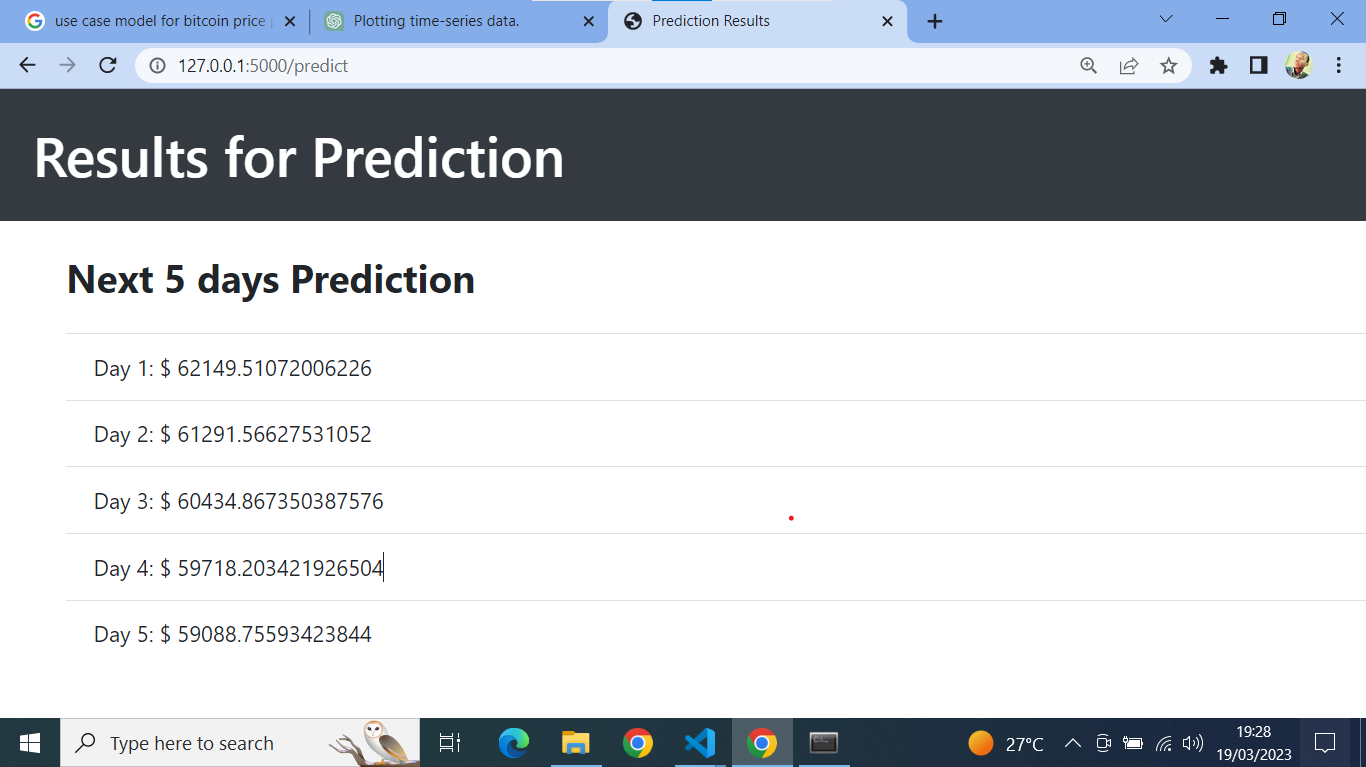
        res = min\_max\_scaler.inverse\_transform(lst\_output)

        return render\_template('result.html', res=res, len=len(res), n=n, date=date)

if \_\_name\_\_ == "\_\_main\_\_":

    app.run()

# 4.5.2: Sample prediction

****

**Figure 17: Sample prediction of next 5 days**

# 4.5.3: Evaluation of the Model.

Model evaluation aims to estimate the generalization accuracy of a model on future data. The proposed model uses the following evaluation metrics to measure its effectiveness;

1. **Root Mean Squared Error (RMSE)**

Evaluation metric for measuring the performance of predictive models, including those used for time series data prediction. RMSE measures the difference between the predicted and actual values of the time series, with a lower RMSE indicating better predictive performance.

1. **Mean Absolute Error**

Measure of the average size of the mistakes in a collection of the mistakes in a collection of predictions, without taking their direction into account. It is measured as the average absolute difference between the predicted values and the actual values and is used to assess the effectiveness of a prediction model.

1. **Accuracy of the model**

Accuracy is one metric for evaluating prediction models and is calculated using the formular;

**Formula Accuracy = 100% - (MAE / Avg. Actual Price) x 100%.**

**Sample data for the past 5 days**

|  |  |  |
| --- | --- | --- |
| Day | Predicted Price in USD | Actual Price in USD |
| 1 | 62,149 | 60,887 |
| 2 | 61,291 | 61,548 |
| 3 | 60,434 | 62,043 |
| 4 | 59,718 | 64,284 |
| 5 | 59,088 | 66,002 |

**Table 1: Sample data for the past 5 days**

**Actual Prices**: [60887, 61548, 62043, 64284, 66002]

**Predicted Prices**: [62149, 61291, 60434, 59718, 59088]

To calculate the % accuracy of the model using the formula, the following steps are followed:

**(i) Mean Absolute Error**

Mean Absolute Error is calculated by taking the average of the absolute differences between the predicted and actual prices for each day in the chosen period:

MAE = (1/5) \* (|60887-62149| + |61548-61291| + |62043-60434| + |64284-59718| + |66002-59088| = 2416.8

**(ii) Avg. Actual Price**

Calculate the Average Actual Price by taking the average of the actual prices for each day in the chosen period:

= (1/5) \* (60887+61548+62043+64284+66002) = 62952.8

Substituting MAE and Avg. Actual Price values into the formula to calculate the % accuracy:

**Accuracy** = 100% - (MAE / Avg. Actual Price) x 100%

= 100% - (2416.8/ 62952.8) x 100% = 96.16%

Therefore, the model's accuracy for the past 5 days is 96.2%.

# 4.6: Testing Approach

Software testing has the ability to identify any error and flaw found during development. Testing in various ways enables us to find flaws that are only apparent during runtime. Machine learning testing is done to make sure that the learnt reasoning will be true no matter how frequently we contact the program.

# 4.6.1: Functional Testing

It is a kind of software testing that verifies the software system in comparison to the functional specifications or requirements. Each function of the software program is tested using functional tests, which involve supplying the right input and comparing the output to the functional requirements. Functional testing mainly involves;

**(i)Black-box testing**- refers to testing without having access to the model's internal information, such as the algorithm used to build it and the features it contains. Black-box testing's primary goal is to consistently maintain the models' quality.

**(ii)Unit tests**-The program is divided into blocks, and each block's component (unit) is tested individually. It entails testing specific pieces of the source code, such as functions, methods, and classes, to make sure they comply with the specifications and provide the desired outcomes. Each each bit of code has been examined, with the findings being performed.

**(iii)Regression tests**- they check already-tested software to make sure it doesn't break unexpectedly and guarantee the new changes will have a high-quality user experience.

# 4.6.2: Integration tests

These tests are meant to check whether assembled components that were created independently perform as anticipated. In terms of a data pipeline, these can verify that:

• The data cleaning process produces a dataset suitable for the model

• The model training can handle the data given to it and produces results (ensuring that code can be refactored in the future)

• The data is consumable by the model (a label exists for every input, and the types of the data are accepted by the type of model chosen)

• Code refactor can ve done in the future, without breaking the end to end

# CHAPTER 5

# 5.1. Introduction

This chapter presents the proposed model evaluation results together with comparison with existing models studied.

# 5.2: Evaluation and comparison results

# 5.2.1: Proposed model evaluation results

The proposed model used Long Short-Term Memory algorithm and Adam as the learning algorithm. Evaluation results of the model are outlined as follows in the table;

|  |  |
| --- | --- |
|  | **Proposed Model (LSTM)** |
| Accuracy | 96.2% |
| Validation Loss | 0.00042 |
| Learning Rate | 0.00001 |
| Learning Algorithm | Adam |
| RMSE | 0.05 |
| MAE | 2416.8 |

**Table 2: Proposed model evaluation results**

# 5.2.2: Comparison between the proposed method and existing methods

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Algorithm** | **Learning Algorithm** | **Accuracy (% )** |
| Proposed Model | LSTM | Adam | 96.2% |
| Paper(Albariqi & Winarko, 2020) | MLP |  | 81.3% |
| RNN | 77.3% |
| Paper(McNally et al., 2018) | ARIMA |  | 50.05% |
| LSTM | 52.78% |
| RNN | 50.25% |

**Table 3: Comparison between the proposed method and existing methods**

# 5.2.3 Limitations of the proposed model and potential areas for improvement

The limitations of using Recurrent Neural Network with Long Short-Term Memory algorithm model for bitcoin price prediction:

**(i) Dependance on historical data:** RNN-LSTM models heavily depend on historical data to make predictions, which can limit their ability to predict sudden changes or events that have not occurred in the past.

**(ii)Sensitivity to hyperparameters:** The performance of RNN-LSTM model is highly dependant on the choice of hyperparameters, which can make it difficult to optimize the model for the best performance.

Potential areas for improvement include;

**(i)Incorporating external data sources:** By incorporating external data sources such as news articles or social media sentiment, the model maybe able to better capture the impact of exogenous variables on bitcoin prices.

**(ii)Hyperparameter tuning:** The performance of the RNN-LSTM model can be further improved by fine-tuning the hyperparameters. This involves adjusting the number of layers, number of neurons, learning rate, batch size, and other parameters to optimize the model's performance.

# 5.3: User Documentation

Bitcoin price prediction model is a system that predicts future prices of bitcoin crypto currency using bitcoin historical price obtained from yahoo. The project uses the following tools;

**(i)Anaconda**- it is a scientific python distribution that comes with all necessary packages needed to build the model. The packages include pandas, numpy, sklearn, keras, tensorflow and Jupyter notebook which is an interactive, open source web application for creating and sharing documents that integrate live code.

**(ii)Jupyter notebook**- used to perform task such as data cleaning, data transformation, exploratory data analysis, statistical modelling, machine learning and data visualizations.

**(iv)Visual studio code**- it is a code editor redefined and optimized for building and debugging modern web and cloud applications. In this project the user interface design has been designed using the flask framework.

**(v) Chrome web browser**- allows to load the python web application. The user interface allows the user to enter date and number of future days to predict. After entering the inputs and using predict button to predict, a loop is created to run and display the output for next number of days and the system outputs the result. The user can reset the predictions and enter new inputs. The model has been trained using the Long-Short Term Memory algorithm

# CHAPTER 6

# Conclusions and Future Works

# 6.1: Conclusion

In this study, the researcher developed a long-short memory recurrent neural network architecture for predicting Bitcoin cryptocurrency prices. For the purpose of fine-tuning the model, the network was assessed using several parameters. It has been discovered that applying the Adam learning algorithm significantly improved the final outcome of the prediction process and had an impact on training time performance. The prediction is performed using the real-time Bitcoin dataset, and after training it achieved an accuracy of 96.2% with error losses of 0.00042 and 0.05 RMSE, which is fairly satisfactory for predicting time series data. The researcher evaluated the performance and accuracy of this study in comparison to other approaches, including RNN, MLP, ARIMA, and other LSTM models. The study showed that one of the best techniques for forecasting bitcoin values is to use LSTM with appropriate hyperparameter tuning.

# 6.2: Future Works

Bitcoin price prediction using RNN-LSTM models and the Adam optimizer is an exciting area of research in the field of cryptocurrency. There are several potential future works that can be explored to improve the accuracy of these models:

**(i)Feature selection**: Bitcoin price prediction models can benefit from including additional features besides the historical price data. For example, incorporating data on trading volumes, market sentiment, and other external factors that influence cryptocurrency prices can help the model make better predictions.

**(ii)Multiple models:** It is possible to train and compare multiple RNN-LSTM models to find the one that works best for Bitcoin price prediction. Different models can have different architectures and use different types of optimization algorithms.

**(iii)Ensemble models:** Ensemble models can be used to improve the accuracy of Bitcoin price predictions. This involves combining the predictions of multiple RNN-LSTM models to create a final prediction that is more accurate than any individual model.

**(iv)Real-time prediction:** Real-time prediction of Bitcoin prices is essential for traders and investors. Future work can explore the use of RNN-LSTM models to provide real-time predictions of Bitcoin prices and incorporate them into trading algorithms.

Overall, there are many potential areas of future work in Bitcoin price prediction using RNN-LSTM models and the Adam optimizer. By exploring these areas, researchers can improve the accuracy of the models and create more effective trading strategies for investors.

# **APPENDICES**

# APP 1: Budget

|  |  |  |
| --- | --- | --- |
| ITEM | QUANTITY | UNIT PRICE |
| Printing and binding |  | 1000 |
| Laptop | 1 | 27000 |
| Software | 3 | 2500 |
| Internet |  | 2000 |
| USB Flash Disk |  | 1500 |
| TOTAL (k.sh) |  | 34000 |

**Table 4: Budget**

# APP2: Schedule

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| ACTIVITY | OCT | NOV | DEC | JAN | FEB | MARCH | APRIL |
| Project identification |  |  |  |  |  |  |  |
| System analysis |  |  |  |  |  |  |  |
| System Design |  |  |  |  |  |  |  |
| Coding and Testing |  |  |  |  |  |  |  |
| Implementation |  |  |  |  |  |  |  |
| Documentation |  |  |  |  |  |  |  |
| Project submission |  |  |  |  |  |  |  |

**Table 5: Schedule**

# References

[1]Nakamoto S. Bitcoin: a peer-to-peer electronic monetary system. https://bitcoin.org/bitcoin.pdf. Work Pap; 2008.

[2]Sin E, Wang L. Using neural network ensembles to predict the price of bitcoin. 2017. IEEE. 2017

[3] Shankhdhar A, Singh AK, Naugraiya S, and P.K. Saini. Various models are used in the Bitcoin Price Alert and Prediction System. 2021 Obtainable from: (dx.doi.org).j

[4]Mittal R, Arora S, Bhatia MP. Automated cryptocurrencies prices prediction using machine learning. ICTACT Journal on Soft Computing. 2018

[5] S. Nakamoto. Bitcoin: A Peer-to-Peer Electronic Cash System. The Bitcoin Whitepaper is accessible at https://git.dhimmel.com

[6]Sebastian H. and Godinho P. utilizing machine learning to predict and trade cryptocurrency in dynamic market environments. Available from: (dx.doi.org).

Phladisailoed T, Numnonda T.

[7] Comparison of machine learning models for predicting the price of bitcoin. 2018

[8]Jaquart P, Dann D, Weinhardt C. Machine learning-based short-term market prediction for bitcoin. 2021 The Journal of Finance and Data Science.

[9] Rane PV and SN Dhage machine learning strategies for systematic education in bitcoin price prediction. 2019

[10].Hossain MN, Alam MZ, Nazmul N, Roy R, Roy S Utilizing the MSE approach, do research on nonlinear partial differential equations. 2020 Global Scientific Journals.

[11]You would have had the following sums of money if you had invested $1,000 in bitcoin 1, 5, and 10 years ago. Published July 26 at 20:14:31 EDT Mon, July 26 202:17:26 EDT updated 2021;7:45-66 is available at www.cnbc.com

[12]Huang X, Zhang W, Huang Y, Tang X, Zhang M, Surbiryala J, et al.. 2021.

[13] Muniye TA, Satapathy S, and Mohanty MR Deep learning models for predicting and analyzing the price of bitcoin. Published 2020Read more at: 10.1007/978-981-15-5397-4 63.

[14] What Is Bitcoin And How Does It Operate? Accessible at www.Forbes.com, by Kate Ashford and Farran Powell

[15]Bitstamp. (2022). (2022). Bitcoin USD/BTC. https://api.cryptocompare.com

[16]Neural networks are widely used to forecast time series data (accessed Nov. 21, 2022).

[17]The authors are Patel, M. M., Tanwar, S., Gupta, and Kumar (2020). A Deep Learning-based Scheme for Predicting Cryptocurrency Prices for Financial Institutions 55(May), 102583;

[18]S. McNally, J. Roche, & S. Caton (2018). Machine learning for Bitcoin Price Prediction. PDP 2018, the proceedings of the 26th Euromicro International Conference on Parallel, Distributed, and Network-Based Processing, were published. https://doi.org/10.1109/PDP2018.2018.00060

[19] (2021). Lahmiri, S., and Bekiros, S. Deep Learning Forecasting in High-Frequency Trading for Cryptocurrencies. 13(2) of Cognitive Computation, 485-487. https://doi.org/10.1007/s12559-021-09841

[20]Godinho, P., Sebastio, H. (2021). utilizing machine learning to predict and trade cryptocurrency in dynamic market environments. Innovation in Finance, 7 (1). https://doi.org/

[21]Ferdiansyah, Zahilah Raja Md Radzi, R., Stiawan, D., Y. Sazaki, & U. Ependi (2019). A Case Study Using the LSTM Method to Predict the Price of Bitcoin on the Stock Market at Yahoo Finance.

[22]Albariqi, R., & Winarko, E. (2020). Prediction of Bitcoin Price Change using Neural Networks. *Proceeding - ICoSTA 2020*

[23] Alkaya, A. EVOLUTIONARY ARTIFICIAL NEURAL NETWORKS ARE NEURON OPTIMIZED FOR STOCK PRICE INDEX PREDICTION. (2013). Available

Dergipark.org.tr

[24]Turkish Stock Market's BIST 30 Index. International Journal of Engineering Research and Development

[25] B. K. O. and Kemalbay G. (2021). Genetic programming vs. SARIMA-ARCH for stock price prediction. 39(2), 110–122, Sigma Journal of Engineering and Natural Sciences.

[26]M. Sun, X. Wu, and W. Wang, "A Hybrid Model for Bitcoin Price Prediction Based on Deep Learning," in IEEE Access, vol. 8, pp. 200905-200915, 2020, doi: 10.1109/ACCESS.2020.3035834.

[27]P. Tiwari and S. K. Jena, "Bitcoin Price Prediction Using Long Short-Term Memory Based Recurrent Neural Networks," in IEEE Access, vol. 6, pp. 10717-10730, 2018, doi: 10.1109/ACCESS.2018.2804710.

[28]X. Wang and L. Liu, "Bitcoin Price Prediction Based on LSTM Neural Network," in 2019 International Conference on Computational Science and Computational Intelligence (CSCI), Las Vegas, NV, USA, 2019, pp. 877-881, doi: 10.1109/CSCI49370.2019.00150.

[29]A guide for faculty and students seeking research funds 2021 Accessed at libguide.wits.ac.za.

[30] Wikipedia - County of Murang'a population 2022. Accessible at en.m.wikipedia.org

[31] Data collecting technique using interviews 2022. Available at https://www.mbaknol.com

[32] Survey template for cryptocurrencies 2021. Available at www.surveymonkey.com

[33] Records and documents Data collection method 2022. Disclosed at https://www.utwente.nl

[34] Advantages and downsides of the waterfall paradigm in software development life cycles.2021. Accessible at https://www.guru99.com

[35] Advantages and disadvantages of the agile model 2021 Accessible at https://www.javapoint.com

[36] Prototype model's benefits and drawbacks are listed at https://www.geeksforgeeks.org.