

Cryptocurrency Price Prediction and Forecasting using Machine Learning Algorithm and Long- Short term Memory Mapping

by

Md. Sayeed Ibne Zaman

21141073

Mir Ishraq Kamal

17101459

Samiu Mostafa Ishan

18101452

Shafayat Zamil Khan

17101277

Samiha Hossain

17301185

A thesis submitted to the Department of Computer Science and Engineering
in partial fulfillment of the requirements for the degree of
B.Sc. in Computer Science and Engineering

Department of Computer Science and Engineering
BRAC University
January 2021

© 2021. BRAC University
All rights reserved.

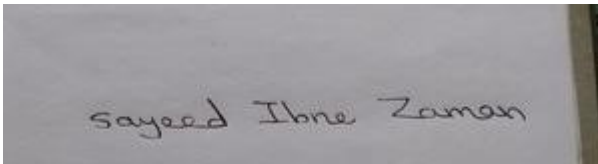
1 Declaration

It is hereby declared that –

1. The thesis submitted is my/our own original work while completing a degree atBrac University.
2. The paper does not contain material previously published or written by a thirdparty, except where this is appropriately cited through full and accurate referencing.
3. The thesis does not contain material that has been accepted or submitted, for anyother degree or diploma at a university or other institution.
4. I/We have acknowledged all main sources of help.

Student's Full Name and Signature:

1:

A photograph of a handwritten signature in black ink on a white surface. The signature reads "sayeed Ibne Zaman".

MD.SAYEED IBNE ZAMAN

2:

A photograph of a handwritten signature in green ink on a white surface. The signature is partially obscured and appears to be "MIR ISHRAQ KAMAL".

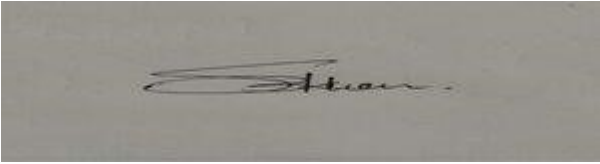
MIR ISHRAQ KAMAL

3:



SAMIU MOSTAFA ISHAN

4:



SHAFAYAT ZAMIL KHAN

5:



SAMIHA HOSSAIN

2 Approval

The thesis/project titled “Cryptocurrency Price Prediction and Forecasting using Machine Learning Algorithm and Long- Short term Memory Mapping” submitted by –

1. MD. SAYEED IBNE ZAMAN – 21141073
2. Mir Ishraq Kamal – 17101459
3. Samiu Mostafa Ishan – 18101452
4. Shafayat Zamil Khan – 17101277
5. Samiha Hossain - 17301185

of Fall, 2022 has been accepted as satisfactory in partial fulfillment of the requirement for the degree of B.Sc. in Computer Science on January 22, 2021.

Examining Committee:

Supervisor:

(Member)



Esfar-E-Alam

Lecturer

Department of Computer Science and Engineering BRAC University

Departmental Head

(Chair)

Mahbubul Alam Majumdar, PhD

**Professor and Dean, School of Data and Sciences Department of Computer
Science and Engineering BRAC University**

Contents

1	Declaration	1
2	Approval	3
3	Abstract	8
4	Keywords	8
5	Acknowledgment	8
6	Introduction	9
6.1	Motivation	9
6.2	Problem Statement	10
6.3	Research Objective	11
7	Literature Review	11
7.1	Literature Review	11
7.2	Cryptocurrency Price Predictions Methods	13
7.3	Cryptocurrency Price Predictions Technique	13
8	Models used to collect Data and Result Analysis	14
8.1	Long Short-Term Memory (LSTM)	14
8.2	Working steps of LSTM	15
8.3	Simple RNN	17
8.4	Simple RNN implementation	18
8.5	Random Forest	18

8.6	Advantages of Random Forest	20
8.7	Disadvantages of Random Forest	20
9	Workplan	21
10	Dataset Collection and Analysis	22
10.1	Dataset Description	22
10.2	Data Processing	24
11	Implementation and Model Setup	25
11.1	For LSTM Model	27
11.2	For SimpleRNN model	29
11.3	For Random Forest Algorithm	32
11.4	Comparison Between Models	34
11.5	Results	34
12	Conclusion and Future Work	35
12.1	Conclusion	35
12.2	Future Work	35
13	References	36

List of Figures

1	Functions of Cryptocurrencies	10
2	Repeating module of a Standard RNN containing a single layer.	14
3	Basic structure of a simple recurrent neural network (RNN)	17
4	Forward pass of a simple recurrent neural network	18
5	Random Forest	19
6	Workplan diagram	22
7	Training data, testing data and validation data diagram	23
8	Dataset Collection and Analysis	26
9	Bitcoin (BTC) output for LSTM	27
10	Litecoin (LTC) output for LSTM	28
11	Ethereum (ETC) output for LSTM	28
12	Bitcoin (BTC) output for SimpleRNN	29
13	Litecoin (LTC) output for SimpleRNN	30
14	Ethereum (ETH) output for SimpleRNN	31
15	Ethereum (ETH) output for SimpleRNN	32
16	Litecoin output for Random Forest	33
17	Ethereum (ETH) output for Random Forest	33

3 Abstract

Cryptocurrency has been a fascinating topic of many over the past few years, as many people are starting to trade these currencies for big cash-outs. It is a decentralized digital currency. It is a cryptocurrency that has used cryptography to manage the trading systems, creation, and management without relying on any third parties. Since the innovation of the first cryptocurrency Bitcoin in 2009, its value has skyrocketed. Starting from 0.09 to 42,226 per bitcoin, it's a very big market. New services, new companies are accepting it as it seems to be the new future. It is an encrypted peer-to-peer network for simplifying digital exchange.

Blockchain-based currency has been the topic of many discussions and is widely popular lately. We decided to analyze the predictability of currency prices by using a machine learning algorithm widely known as the LSTM method. LSTM (Long Short-Term Memory) is a module provided for RNN, later developed and popularized by many researchers; like RNN, the LSTM also consists of modules with recurrent consistency and it works well with short time-sequential datasets. LSTM networks are well-suited for processing and making predictions. This machine-learning algorithm was developed to deal with vanishing gradient points encountered when training traditional RNNs and predict multiple currency prices for a short time interval. It will help traders and buyers to understand more about the price volatility of cryptocurrencies at one-minute intervals for real-life buy and sell.

4 Keywords

Cryptocurrency, LSTM, Bitcoin, Litecoin, Ethereum, Cryptocurrency Price Prediction, Forecasting, Machine learning, Long short-term memory, SimpleRNN, Random Forest

5 Acknowledgment

Firstly, we want to thank our parents who always supported us in every situation of our life. Next, we would like to thank to all those researchers who are continuously working for the betterment of the prediction models Security of Cryptocurrencies. Furthermore, we want to thank our honorable supervisor Mr. Esfar-E-Alam sir for his continuous support and suggestions. We are also thankful to all the respected faculty of CSE Dept, BRAC University.

Chapter 1

6 Introduction

6.1 Motivation

Cryptocurrency is the first and most popular currency used worldwide for advanced installment or speculation purposes. It paves the way as a disruptive innovation to well-established, unchanged financial payment systems that have been in place for numerous decades. It could change the way internet-based global markets interact and sort out the barriers surrounding normative national currency and exchange rate since cryptocurrency is not likely to replace mainstream or traditional fiat currency. Nobody embezzles Cryptocurrencies; they are decentralized as no certain individual controls its flow. Moreover, exchanges made by blockchain-based currency are simple to manage as they are not attached to any nation and don't require any third party, unlike any banking system.

Cryptocurrency is a digital currency that was introduced in 2009 by a pseudonymous author 'Satoshi Nakamoto'. Although it is still not clear if this person(s) exists or not, cryptocurrencies are making their way into big company transactions. It is enabled by blockchain technology which is more secure than any database system. Also, it allows peer-to-peer transactions secured by cryptography.

Prate or visualization can be possible through different commercial centers known as "blockchain-based currency trades". These enable individuals by utilizing various financial forms to sell or buy Blockchain-based currencies. The Biggest Cryptocurrency trade is Mt Gox. These digital currencies are reserved in an advanced wallet similar to a virtual financial balance. Cryptocurrencies are produced as a reward for the solving process known as mining. It can be exchanged for other currencies, products and various services. It was under the hood since its innovation, but the real-world markets are talking about it seriously. It is very profitable; transactions are untraceable and faster than any other transaction system. There is no barrier to transactions. These Cryptocurrencies are based on an elliptic curve and encrypted with the ECDSA algorithm.

According to the European Central Bank, the decentralization of money offered by blockchain-based currency has its theoretical roots in the Austrian school of economics, especially with Friedrich von Hayek in his book *Denationalization of Money: The Argument Refined*, where Hayek states a completely free market in the production, distribution and management of money to end the mainstream monopoly of central banks.

Many more calculations have been utilized on financial exchange information for value forecasting. Moreover, the parameters influencing Blockchain-based currency are pretty

extraordinary. In this economic situation, it is essential to foresee the estimation of Blockchain-based currency so accurate venture choices can be made. Moreover, the cost of Blockchain-based currency doesn't depend on business occasions or intervening government. In a nutshell, to anticipate the worth, we feel it is significant to use ML innovations to foresee the cost of multiple cryptocurrencies.

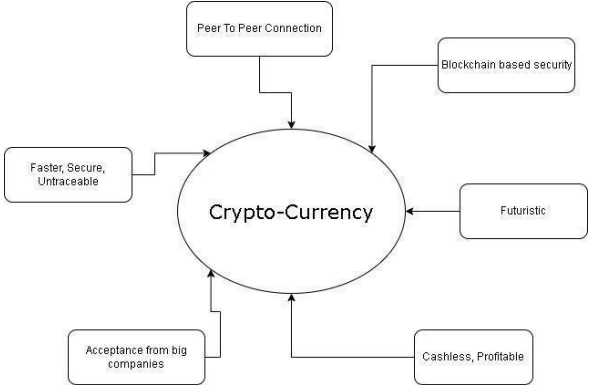


Figure 1: Functions of Cryptocurrencies

6.2 Problem Statement

In recent years the amount of cryptographic currencies has increased. People are now trading and buying virtual goods with this digital payment. It can be produced through mining, but it takes a lot of graphical memory. The currency is given as a reward for solving complex mathematical equations with machine power.

Crypto currency's value is rising. Many believe that one single bitcoin would reach the 1mil mark within 2022. As a result, users are increasing daily for higher profits. In this digital era, half of our reality belongs to the digital world. For buying and trading cryptocurrencies, one must be aware of their current market. It is volatile, more than any currency. It is similar to the stock market, prices can drop or reach high within a minute. Cryptocurrencies are very sensitive to news. But, if someone wants to trade or buy cryptocurrencies they should know what the current price is. What was the price at the last minute? What can be the price of a specific time of the upcoming day? Many software and hardware-based companies are now open for trading cryptocurrencies. Now, blockchain-based currency can be won just by playing NFT based mobile games. Prediction models for such high-demand currency can be helpful for people and companies relying on prices.

The Crypto market is growing faster than ever. It's very easy to make transactions from one point of the world to another. Big companies are now giving more emphasis on online marketing and selling products. Some of them are accepting cryptocurrency as their payment method. As we know, the Facebook company is working on establishing a

digital world where you can buy sell all kinds of services and live a different kind of life, where digital payments would be first priority. Blockchain-based Cryptocurrency is the future of digital payments as it is faster and more secure than other digital payments.

6.3 Research Objective

Real-life traders and buyers of cryptocurrencies use prediction models to get a good idea about the present market value of that exact currency. Price High and Low similar to stock exchanges make it volatile, unlike any other currency. If someone wants to sell their Bitcoin or other currencies, they should ensure about the market status and analyze previous price data of that exact day.

We designed our model for real-life using it as it gives us predictions and forecasts currency prices for upcoming minutes. A whole day is theoretically plenty of time if you consider the price rising or going down. It is beneficial for Cryptocurrency buying, selling or trading for personal or company use. Using highly accurate predictions a buyer may buy digital goods or exclusive hardwires at the lowest possible price and vice versa. Using machine learning algorithms, it is possible to get high accuracy on the price prediction. ML-based algorithms generate more accurate results than other statistical mathematical models. We had to train out data for each currency from the acquired dataset. We divided them in order to train the model for forecasting the upcoming value of the cryptocurrency. We are analyzing data every 1-minute interval for better understanding and real-life price forecasting. With the help of our model and future works, Cryptocurrency users will be able to determine changing currency values for a specific time of the day.

Chapter 2

7 Literature Review

7.1 Literature Review

The primary aim of this part is to talk in reference to previous work on Predicting Blockchain-based currency Price with Machine Learning. Due to the volatile nature of cryptocurrency, the price of Blockchain-based currency has been a subject of major debates all over the world. In [3], we get to know how the LASSO (Least Absolute Shrinkage and Selection Operator) algorithm is quite useful in predicting an accurate outcome for Blockchain-based currency prices. It also tells us about the comparison it makes to other

ML algorithms with LASSO, such as RNN (Recurrent Neural Networks), CNN (Convolution Neural Networks), GLM, etc.

The prediction of cryptocurrency prices has been a topic of research from people all over the world. The decentralization of cryptocurrencies has led to increased popularity and usage by people in the 21st century. [4] also talks about how the value of cryptocurrency fluctuates in an unpredictable manner, being similar to stock exchanges, so if an investment in cryptocurrency has to be made, one has to find an accurate method to predict cryptocurrency prices. Overviews of different prediction techniques, such as Long short-term memory (LSTM), Linear Regression Model, K-NN (K-Nearest Neighbors) and Random Forests, are given briefly in order to put more emphasis on the prediction of an accurate prediction of Blockchain-based currency values.

[4] simply talks about the predictability of the Blockchain-based currency market on a very short timestamp. The short-term market value prediction has not been a topic that has been discussed quite extensively, but it yields results that help to predict and invest in Blockchain-based currencies on a short-term basis. The authors talk about how different machine learning models over a certain period of time can be trained over minutely data of Blockchain-based currencies, giving outcomes that suggest that the models work perfectly with the given data. However, the predictability of cryptocurrency prices is subject to certain market factors, which might cause disruption in the desired output that we expect from these models.

The high-frequency fluctuations of cryptocurrency trading in [5] have made it the most desirable field for both researchers and investors. The implementation and application of DFFNN (Deep Feed-Forward Neural Network) along with Powell-Beale restarts, the resilient and the Levenberg-Marquardt algorithms, results in an outcome where DFFNN, when trained with Levenberg-Marquardt algorithm, turns out to be far superior to the other two algorithms, with the resilient algorithm showing the least convincing results in terms of speed. The results were justified with the help of the root mean of squared errors (RMSEs).

Different cryptocurrency shows different types of results when introduced to varying kinds of machine learning algorithms, thus resulting in dynamical outputs. [6] Algorithms such as Support Vector Machines (SVM), Artificial Neural Networks [ANN], and Deep Learning are explicitly used, in order to predict the desired outcome of the cryptocurrency forecast. Cryptocurrency, such as Blockchain-based currency, Litecoin, Ethereum, Stellar, Ripple and Nem, was used in order to compare results of the output of different algorithms with the cryptocurrency. While forecasting Blockchain-based currency values, SVM turned out to be the most effective algorithm, providing outputs that are close to the accurate results. [7] provides us with the prediction of the Blockchain-based currency price in USD. The data is collected from the Blockchain-based currency Price Index, with the implementation of a Bayesian optimized recurrent neural network (RNN) and a Long

Short-Term Memory (LSTM) network. Alongside these, the Autoregressive Integrated Moving Average (ARIMA) model for time series is used as a reference to the algorithms. This research leads us to the conclusion that ARIMA is of no use compared to the enhanced performance of the machine learning algorithms.

[8] provides us with insights on the forecasting of the Blockchain-based currency price, in-reference to previous data on Blockchain-based currencies. After seeing multiple data and information from the articles, we can infer that the price of Blockchain-based currencies is really volatile, as they tend to fluctuate quite frequently and prices can soar or plummet depending on a number of different factors.

7.2 Cryptocurrency Price Predictions Methods

There is so much research work that is on the prediction of cryptocurrency. Among all of these, Logistic Regression and SVM and analyzed using Graph to predict cryptocurrency price by Greaves et al [9]. Modeling and prediction of cryptocurrency with Bayesian Neural Network by Huisu Jang et al[10], Arief Radityo et al[16] proposed a prediction of cryptocurrency using Artificial Technical indicators that have miserably fallen in their work.

But on the other hand, papers using LSTM are significantly praised and can substantially impact the research field for accuracy. Sean et al [12] propose a method for the price of Cryptocurrency using RNN and combine Using Recurrent Neural networks and Long Short-Term Memory and Ruchi Mittal et al [13] submit an Automated cryptocurrencies prices prediction using machine learning technique based on the historical trend (daily trend). Chih-Hung et al. [14] are created a new forecasting framework for cryptocurrency prices using LSTM, they proposed two various LSTM models (conventional LSTM and LSTM with AR(2) model) with 208 records datasets, compared with MSE, RMSE, MAE, and MAPE. Fei Qian et al.[15] produced a standard stock market prediction model based on LSTM under Different factors that impact the market, in this research they selected three stocks with similar trends. The LSTM prediction model has performed well.

7.3 Cryptocurrency Price Predictions Technique

Technically, Cryptocurrency stock market prediction is the same as a stock market prediction on a conventional stock exchange, but when you use a different approach and strategy, such as sentiment research, you may not obtain different results, and other strategies may not work. Because, in addition to numerous other elements that influence stock exchange forecasts, Cryptocurrency is decentralized and unregulated by any entity, making it distinct from traditional currencies and stock markets.

We can use the same Machine Learning technique for Cryptocurrency prediction (e.g., SVM, Nave Bayes, Regression) or any other advanced machine learning approach, such as Deep Learning using Neural Networks (e.g., LSTM, ANN and RNN) to enhance the results. On the issue of prediction, we may use a strategy on a particular subject to forecast Cryptocurrency. For example, we may only want to predict based on the signal or the price. Also, we may only want to predict based on the current day or next day relative value based on Long Short-Term Memory (LSTM),[13] historical price, and other techniques such as regime prediction to detect the current day’s trend on the market, to assist investors in making investment decisions.

We can combine the prediction algorithm with another prediction method or methodology of Cryptocurrency for gaining more accuracy and richness in output.

Chapter 3

8 Models used to collect Data and Result Analysis

8.1 Long Short-Term Memory (LSTM)

Long Short-Term Memory networks, or "LSTMs," are a kind of RNN that can learn long-term dependencies. Hochreiter Schmidhuber (1997) introduced them, and numerous individuals developed and popularized them in subsequent work. As a result, they are frequently utilized and function exceptionally effectively in a wide range of situations.

LSTMs are mainly developed to prevent the problem of long-term reliance. They don’t have to work hard to remember knowledge for lengthy periods; it’s nearly second nature to them. All recurrent neural networks or RNN are repeated neural network modules. This repeating module in ordinary RNNs will have a relatively basic structure.

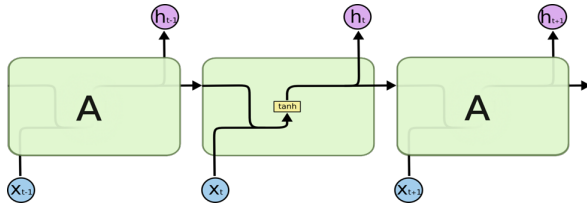
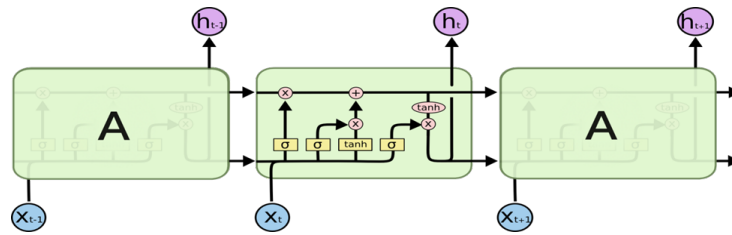


Figure 2: Repeating module of a Standard RNN containing a single layer.

LSTMs have a chain-like structure, but the repeating module is different. There are four neural network layers, each interacting uniquely in LSTM.



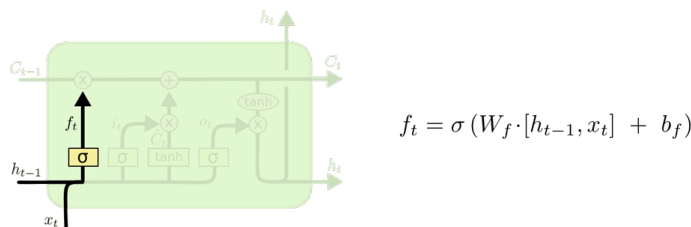
Cell state is the key to LSTMs. It is a horizontal line that runs through top of diagram. The cell state is kind of a conveyor belt that runs straight down the entire chain, with only minor linear interactions. So it's elementary for information to flow along with it unchanged.

The LSTM can remove or add data to the cell state, and gates can regulate them carefully. Gates are a structure that lets information through optionally. They are composed by a sigmoid neural net layer and a point-wise multiplication operation. Gates are a structure that lets information through optionally. They are composed of a sigmoid neural net layer and a point-wise multiplication operation.

8.2 Working steps of LSTM

The first stage in our LSTM is to decide which information from the cell state will be discarded. The "forget gate layer," a sigmoid layer, makes this judgment. It examines values in h_{t-1} and x_t and returns a number between 0 and 1 for each number in the cell state C_{t-1} . A 1 indicates "totally keep this," while 0 shows "entirely discard this."

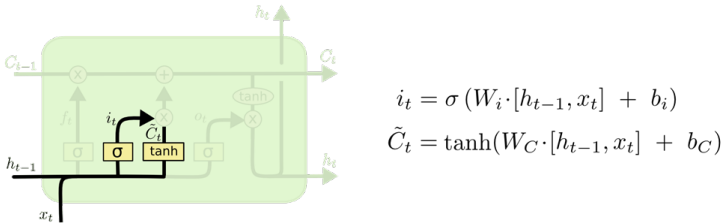
Let's come to our previous example of a language model attempting to anticipate the next word based on the preceding ones. In this case, the cell state may include the gender of the current subject, allowing the appropriate pronouns to be used. When we observe a new topic, we desire to forget the previous subject's gender.



The next stage is to figure out what new data we'll put in the cell state. This is broken down into two sections. First, the "input gate layer," a sigmoid layer, determines which values will be updated first. Following that, a tanh layer generates a C_t vector of new

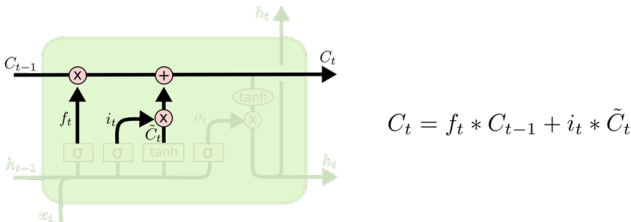
candidate values that might be added to the state. We'll combine these two in the following step to update a state.

We'd like to replace the old subject's gender with the new subject's gender in the cell state in our language model.



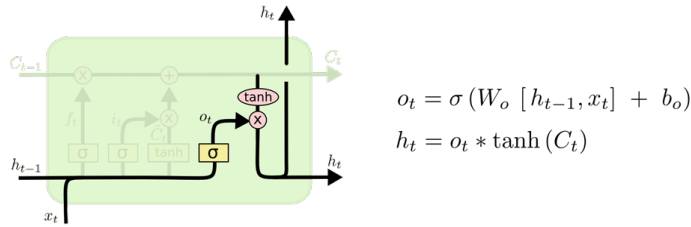
It's time to switch from the old cell state, C_{t-1} , to the new cell state, C_t . The preceding phases have already determined what we should do, and now we must accomplish it. We magnify the previous state by foot, forgetting the items we had previously agreed to ignore. Then we put it C_t . These are the new candidate values, scaled according to how much each state value was updated.

In the language model instance, we'd remove the information about the old subject's gender and replace it with the new information we agreed on in the previous phases.



Finally, we must determine what we will produce. This output will be dependent on the condition of our cells, but it will be filtered. First, we run a sigmoid layer to determine which aspects of the cell state will be output. The cell state is then passed through the (to force the values to be between -1 and 1) and multiplied by the output of the sigmoid gate, resulting in just the parts we choose to output.

Because it just observed a subject, the language model might wish to output information relevant to a verb if that's what comes next. It might, for example, output whether the subject is singular or plural so that we know how to conjugate a verb if that's what comes next.



8.3 Simple RNN

Recurrent neural networks (RNN) are a type of neural network that is particularly effective at representing sequence data such as time series or natural language.

The most sophisticated approach for sequential data is recurrent neural networks (RNN). It is an algorithm that remembers any input data because its internal memory aids in the solution of machine learning issues involving any sequential data. It is one of the finest deep learning algorithms with notable outcomes.

Keras includes three RNN layers by default. One of them is Simple RNN. It is a fully-connected RNN in which the output of the previous timestep is passed into the following timestep. RNN is a deep learning network that uses intrinsic input from neurons. Internal feedback allows for the remembering of major previous events and the incorporation of prior experiences. In contrast to a standard fully connected feedforward network, RNN shares parameters across all sections of a model, allowing it to generalize to sequence lengths not observed during training. Here, figure 1 illustrates an example of an RNN architecture that generates an output at each time step and features recurrent connections between hidden neurons.

The basic structure of a simple Recurrent neural network is given below:

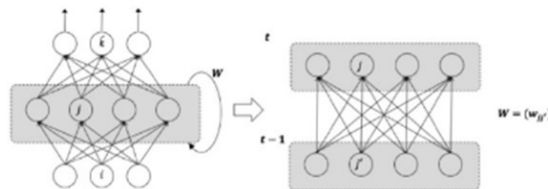


Figure 3: Basic structure of a simple recurrent neural network (RNN)

The simple RNN may be expressed mathematically as follows:

$$h(t) = f_H(W_{IH}x(t) + W_{HH}h(t-1))$$

$$y(t) = f_O(W_{HO}h(t))$$

Here, $x(t)$ and $y(t)$ indicate the input and output vectors, W_{IH} , W_{HH} , and W_{HO} are the weight matrices and the hidden and output unit activation functions are f_H and f_O .

8.4 Simple RNN implementation

Preparing the data before we do anything else is an unavoidable initial step in any data science effort.

1ST step: In this step, our task is to define all of the variables and functions that will be used in the RNN model. Our model will accept the input sequence, run it through a hidden layer of 100 units, and output a single-valued output.

2nd step: Now we have to train the model. Since we've constructed our model, we can begin training it on our sequence data. In this step we will check the loss on training data. To determine the loss value, we will do a forward pass over our RNN model and calculate the squared error for all predictions. We'll keep doing the same thing when it comes to computing the loss on validation data. Then, we shall begin the network's real training. In this case, we'll execute a forward pass to compute the errors, followed by a backward pass to calculate and update the gradients. Then we will do the next steps forward pass, backpropagate error and then update weights.

3rd step: To obtain our predictions, we shall do a forward pass through the training weights.

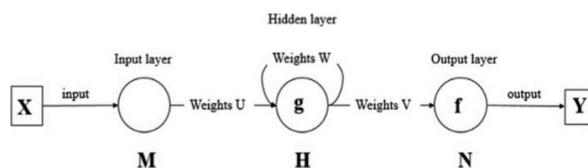


Figure 4: Forward pass of a simple recurrent neural network

8.5 Random Forest

A random forest is a machine learning approach for solving regression and classification issues. It makes use of ensemble methods, which is a technique that combines multiple

classifiers to solve complicated problems. A random forest method is made up of a large number of decision trees. The random forest algorithm's 'forest' is trained via bagging or bootstrap aggregation. Bagging is a meta-algorithm ensemble that increases the accuracy of machine learning algorithms. The outcome is determined by the random forest algorithm based on the estimates of the decision trees. It predicts by averaging the output of various trees. The precision of the outcome improves as the number of trees increases. In a nutshell,

It outperforms the decision tree algorithm in terms of accuracy.

It gives an efficient method of dealing with missing data.

It is capable of producing a fair forecast without the need of hyper-parameter tweaking.

It eliminates the problem of overfitting in decision trees.

At the node's splitting point in every random forest tree, a subset of characteristics is chosen at random.

Random Forest is a classifier that comprises a number of decision trees on various subsets of the provided dataset and takes the average to enhance the predicted accuracy of that dataset, as the name implies. Instead of depending on a single decision tree, the random forest collects the forecasts from each tree and predicts the final output based on the majority vote of predictions. The bigger the number of trees in the forest, the higher the accuracy and the lower the risk of overfitting.

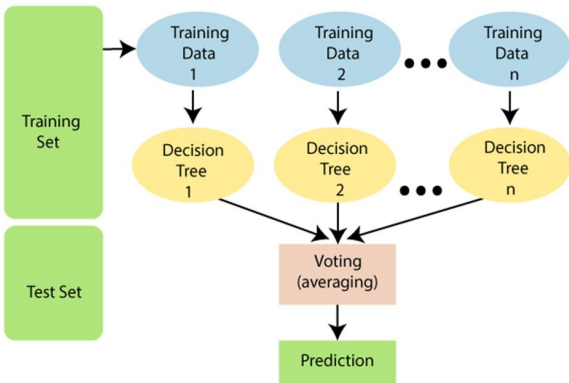


Figure 5: Random Forest

The diagram above explains how the Random Forest algorithm works. Random Forest operates in two stages: (1) to generate the random forest by mixing N decision trees, and (2) to make predictions for each tree generated in the first phase.

1st Step: Choose K data points at random from the training set.

2nd Step: Create the decision trees linked with the data points you've chosen (Subsets).

3rd Step: Decide on the number N for the number of decision trees you wish to create.

4th Step: Repetition of Steps 1-2.

5TH Step: Find the forecasts of each decision tree for new data points and allocate the new data points to the category with the most votes.

8.6 Advantages of Random Forest

1. It is capable of performing both regression and classification tasks.
2. A random forest generates accurate forecasts that are simple to understand.
3. It is capable of effectively handling huge datasets. The random forest method outperforms the decision tree algorithm in terms of accuracy in forecasting outcomes.
4. The random forest approach may be used for both classification and regression tasks.
5. Cross-validation improves accuracy.
6. The random forest classifier will manage missing values while maintaining the accuracy of a substantial part of the data.

8.7 Disadvantages of Random Forest

1. When dealing with categorical variables, random forests are proven to be biased.
2. Slow and steady training.
3. Linear methods with a lot of sparse features are not suited for this model.

Chapter 4

9 Workplan

We start by assembling the dataset we would be using in our experiments. In total, we have collected datasets for 3 different types of cryptocurrencies, which we were able to collect from websites such as Yahoo Finance, Kaggle and Google Analytics. After assembling the data, we start to preprocess our dataset with scaling and normalization, which we were able to achieve with the sklearn preprocess module.

After preprocessing our dataset, we classify it into two different categories: training dataset and testing dataset. We divide our dataset into 90 percent training data and 10 percent testing data, of which we take a further 10 percent of the dataset as validation data, which is taken from the training dataset. After dataset classification, we implement the models we will use to predict the future cryptocurrency values. We would be implementing 3 different models, namely Long Short-Term Memory (LSTM), Simple Recurrent Neural Networks (SimpleRNN) and Random Forest.

After the implementation of our models, we start to train the models using our training dataset. We run a total of 5 iterations on our dataset with each model, since training it with more iterations may result in overfitting. To make sure the models are running correctly, we use the validation data. After we are done with training the models, we move towards testing the models to see whether or not our models are working properly.

With the testing dataset, we move towards predicting output using the models we have trained. After predicting the outputs, we represent them visually using graphs, with which we analyze the outputs we predicted using our models. After analyzing the outputs, we give our final verdict of whether or not there should be a buy or a sell of the cryptocurrency.

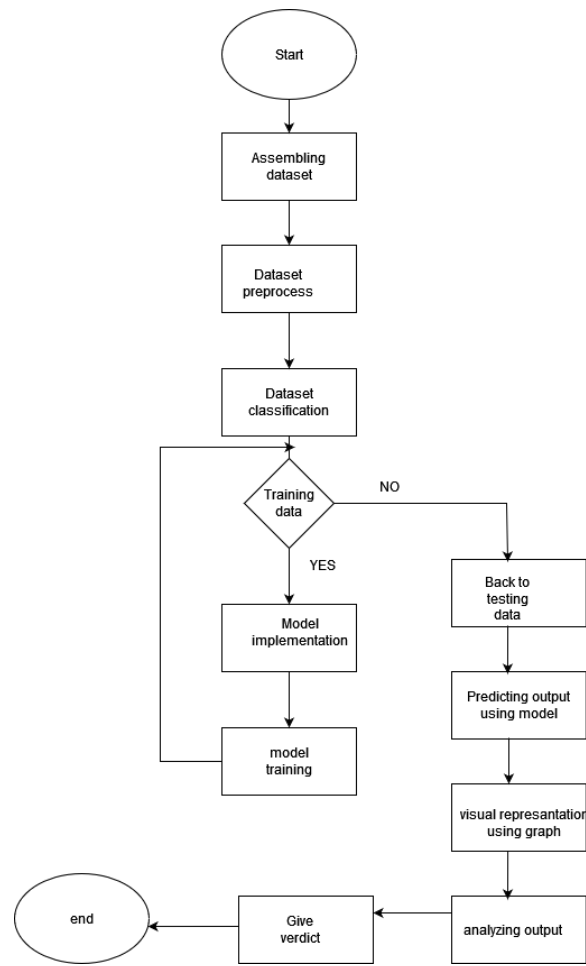


Figure 6: Workplan diagram

Chapter 5

10 Dataset Collection and Analysis

10.1 Dataset Description

In machine learning, a dataset is a collection of data elements that can be processed as a single unit by a computer for analytic and predictive purposes. This means that the data gathered should be homogeneous and understandable to a machine because it does not see data in the same manner that people do.

At first, we need two types of datasets. They are the training dataset and testing dataset.

Training Dataset: The data used to train machine learning models is only as good as the data used to train them. A training set, a training dataset, or a learning set are all terms used to describe training data. Even the most efficient machine learning algorithms will fail to execute without high-quality training data.

The initial dataset used to train machine learning algorithms is known as training data. This data is used by models to generate and refine their rules. It's a collection of data samples that are used to fit the parameters of a machine learning model in order to train it by example.

Testing Data: The sample of data used to offer an unbiased evaluation of a final model fit on the training dataset is referred to as testing data. The sample of data used to offer an unbiased evaluation of a final model fit on the training dataset is referred to as testing data. A test data set is a separate data set from the training data set that has the same probability distribution as the training data set. If a model that fits the training data set well also fits the test data set, there has been minimal overfitting.

Then there's validation data. Validation data, unlike training data, provides an initial check that the model can make valuable predictions in a real-world environment. The machine learning algorithm can evaluate both training and validation data at the same time. This is a dataset that is evaluated frequently during the training phase. Despite the fact that the model encounters this dataset on a regular basis, it does not learn from it. The development set, or dev set, is another name for the validation set. It helps in the prevention of overfitting and underfitting in models.

In our model, we have put 80 percent of the data in the training set, 10 percent in the validation set, and 10 percent in the test set.

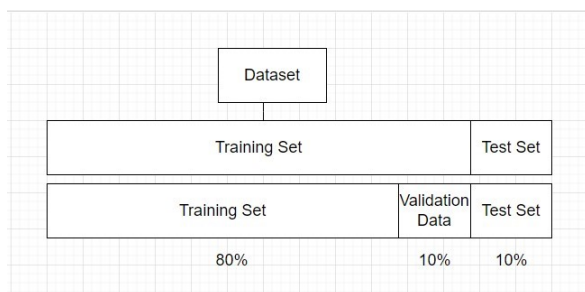


Figure 7: Training data, testing data and validation data diagram

There are several functional ways to download cryptocurrency historical data. Some sources are:

1. Yahoo Finance
2. Investing.com

3. Kaggle

4. Google analytics and so on.

We have collected our dataset from Kaggle.com, which is very common for any machine learning enthusiast.

We have downloaded 3 separate datasets for 3 coins (Bitcoin or BTC, Litecoin or LTC and Ethereum or ETH). In each dataset, there is 100000 or one lakh data with one-minute or 60 seconds intervals. So, for each coin, we have one lakh minutes of data.

In each dataset, there are 5 columns. They are open, high, low, close and volume data. In cryptocurrency trading, the maximum and minimum prices in a specific time period are referred to as high and low. The prices at which a stock began and concluded trading in the same period are known as the open and close. The overall amount of trading activity is referred to as volume.

10.2 Data Processing

We have to process our data to test or analyze the data sets through algorithms. Unfortunately, real-world data tends to be incomplete, noisy, and sometimes inconsistent or also irrelevant.

Data Pre-Processing methods help fill in missing values, smooth out noise while identifying outliers, and correct inconsistencies in the data. Data can be noisy, having incorrect, misleading attribute values. As a result, the predicted price of the existing data sets may be faulty. Human or computer errors can occur at data entry. Errors in data transmission can also occur. For example, a considerable spike in cryptocurrency prices may occur due to Sentimental posts, acceptance in the market or other inconvenient issues. A considerable increment or decrement of values may cause “Noise” for the dataset. Noisy data is meaningless data that machines can’t interpret. It also can be generated due to faulty data collection, data entry errors etc. To smoothen the data and avoid noisy data sets, we are using the following methods for our Crypto-currency Prediction Model.

The resulting datasets are preprocessed by normalization. It allows us to bring the data to a better scale, while also maintaining the robustness of the dataset. To acquire prediction of the coin prices, we had to train our dataset first and test the model with the primary set firstly. We need to

1. Find a dataset for training the model.
2. We can train a model from scratch web data, but that is not suitable for our purpose here. That’s why we need to select our dataset.

3. Load the dataset as a data frame using pandas.
4. The Data we will be using is- Open, High, Low, Close, Volume data for Bitcoin, Ethereum and Litecoin.

For our purposes here, we will only be focused on the Close and Volume columns because the close column measures the final price at the end of each interval. In this case, these are 1-minute intervals. So, at the end of each minute, we will get the asset's price. The Volume column is how much of the asset was traded per each interval, in this case, per 1 minute. And here, close is the price of the thing and volume means how much of that thing.

Then in this model, we need to handle missing values if needed. Then we will encode categorical features if needed. After that scaling all the values between 0-1 using the proper scaling technique. Split the dataset into features and labels, use intuition to determine which column indicates the labels. Then we need to perform classification and calculate accuracy using logistic regression (use SK learn library) and pre-processing the dataset before classification. Our next step is to Perform classification and calculate accuracy using a decision tree (use SK-learn library) and pre-processing the dataset before sorting. We know that Support Vector Machine (SVM), Neural Network and popular machine learning classifiers drive the dataset into two steps and then perform support vector machine, Neural Network, and Random Forest using SK-learn library. Then we will apply support vector machine, neural network and random forest again on the reduced dataset. Finally, we will compare the accuracy of the pre-PCA and post-PCA results.

Now our dataset is ready to apply to the machine learning model.

Chapter 6

11 Implementation and Model Setup

For our work, We used the Python programming language. we used python version 3.6 for prediction and analysis cryptocurrency as python has the most libraries and packages and for machine learning python language is more powerful and better than any other programming language. The python libraries we used are discussed below:

Pandas: pandas is a Python package providing fast, flexible, and expressive data structures designed to make working with “relational” or “labeled” data both easy and intuitive. It aims to be the fundamental high-level building block for doing practical, real-world data analysis in Python.

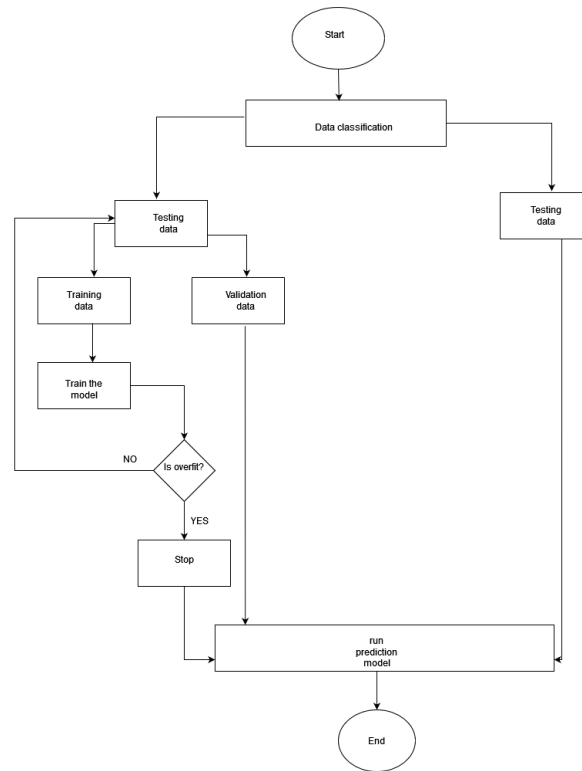


Figure 8: Dataset Collection and Analysis

NumPy: NumPy is a Python package. NumPy stands for Numerical Python, is a library consisting of multidimensional array objects and a collection of routines for processing those arrays. Using NumPy, mathematical and logical operations on arrays can be performed.

Sklearn: Sklearn(Scikit-learn) is the most useful and robust library for machine learning in Python. It provides a selection of efficient tools for machine learning and statistical modeling including classification, regression, clustering and dimensionality reduction via a consistent interface in Python. We used sklearn for the classification of the data model.

TensorFlow: TensorFlow is a Python library, It is a foundation library that can be used to create Deep Learning models directly or by using wrapper libraries that simplify the process built on top of TensorFlow.

Keras: Keras is a Python-based deep learning API that runs on top of TensorFlow, a machine learning platform. It was created with the goal of allowing for quick experimentation. Keras' ability to travel from idea to outcome as quickly as feasible is crucial to conducting effective research. Keras is used to create deep models as well as to train deep learning models in a distributed manner. It supports all neural network models very well.

Tensorboard: TensorFlow comes with a visualization tool called TensorBoard. TensorBoard is a platform that provides the measurements and visualizations required for machine learning. It allows us to keep track of experimental parameters such as loss and accuracy, visualize the model graph, project embeddings to a lower-dimensional space, and much more.

We have implemented the LSTM and SimpleRNN layers from Tensorflow and Keras modules. After implementing the models, we get to see the outcomes of our prediction with the aid of Tensorboard. After we train our models multiple times, we get to justify our claims based on the following outputs:

11.1 For LSTM Model

For Bitcoin(BTC):

```
Epoch 5/5
1300/1300 [=====] - ETA: 0s - loss: 0.1836 - accuracy: 0.9095WARNING:absl:Found untraced functions such as lstm
INFO:tensorflow:Assets written to: models/RNN_Final-05-0.917.model/assets
INFO:tensorflow:Assets written to: models/RNN_Final-05-0.917.model/assets
1300/1300 [=====] - 540s 416ms/step - loss: 0.1836 - accuracy: 0.9095 - val_loss: 0.1634 - val_accuracy: 0.9175
Loosing Probability: 0.16339579224586487
Accuracy Probability: 0.9174866080284119
```

Figure 9: Bitcoin (BTC) output for LSTM

From the output terminal of Google Colab, we get the following results-

```
Epoch 1/5 1300/1300 [=====] - ETA: 0s -
loss: 0.2798 accuracy: 0.8610 1300/1300 [=====]
- 570s 433ms/step - loss: 0.2798 accuracy: 0.8610 - val_loss : 0.2648 - val_accuracy :
0.8712Epoch2/51300/1300[=====]-ETA :
0s-loss : 0.2077accuracy : 0.89741300/1300[=====]
] - 562s432ms/step - loss : 0.2077accuracy : 0.8974 - val_loss : 0.1830 - val_accuracy :
0.9036Epoch3/51300/1300[=====]-ETA :
0s-loss : 0.1934accuracy : 0.90511300/1300[=====]
] - 564s434ms/step - loss : 0.1934accuracy : 0.9051 - val_loss : 0.1718 - val_accuracy :
0.9155Epoch4/51300/1300[=====]-ETA :
0s-loss : 0.1871accuracy : 0.90801300/1300[=====]
] - 565s435ms/step - loss : 0.1871accuracy : 0.9080 - val_loss : 0.1741 - val_accuracy :
0.9188Epoch5/51300/1300[=====]-ETA :
0s-loss : 0.1820accuracy : 0.91031300/1300[=====]
]-578s444ms/step-loss : 0.1820accuracy : 0.9103-val_loss : 0.1734-val_accuracy : 0.9175

Loosing Probability: 0.1734219491481781

Accuracy Probability: 0.9174866080284119
```

For Litecoin(LTC):

```
Epoch 5/5
1218/1218 [=====] - ETA: 0s - loss: 0.0380 - accuracy: 0.9857WARNING:absl:Found untraced functions such as lstm
INFO:tensorflow:Assets written to: models/RNN_Final-05-0.992.model/assets
INFO:tensorflow:Assets written to: models/RNN_Final-05-0.992.model/assets
1218/1218 [=====] - 511s 420ms/step - loss: 0.0380 - accuracy: 0.9857 - val_loss: 0.0440 - val_accuracy: 0.9920
Loosing Probability: 0.0440407432615757
Accuracy Probability: 0.9919689297676086
```

Figure 10: Litecoin (LTC) output for LSTM

From the output terminal of Google Colab, we get the following results-

```
Epoch 1/5 1218/1218 [=====] - ETA: 0s -
loss: 0.2017 accuracy: 0.9085 1218/1218 [=====]
- 516s 418ms/step - loss: 0.2017 accuracy: 0.9085 - val_loss : 0.0807 - val_accuracy :
0.9759Epoch2/51218/1218[=====]-ETA :
0s-loss : 0.0814accuracy : 0.96721218/1218[=====]
] - 508s417ms/step - loss : 0.0814accuracy : 0.9672 - val_loss : 0.0519 - val_accuracy :
0.9845Epoch3/51218/1218[=====]-ETA :
0s-loss : 0.0560accuracy : 0.97861218/1218[=====]
] - 509s418ms/step - loss : 0.0560accuracy : 0.9786 - val_loss : 0.0533 - val_accuracy :
0.9788Epoch4/51218/1218[=====]-ETA :
0s-loss : 0.0464accuracy : 0.98291218/1218[=====]
] - 514s422ms/step - loss : 0.0464accuracy : 0.9829 - val_loss : 0.0459 - val_accuracy :
0.9813Epoch5/51218/1218[=====]-ETA :
0s-loss : 0.0380accuracy : 0.98571218/1218[=====]
]-511s420ms/step-loss : 0.0380accuracy : 0.9857-val_loss : 0.0440-val_accuracy : 0.9920
```

Loosing Probability: 0.0440407432615757

Accuracy Probability: 0.9919689297676086

For Ethereum (ETH):

```
Epoch 5/5
1304/1304 [=====] - ETA: 0s - loss: 0.0759 - accuracy: 0.9679WARNING:absl:Found untraced functions such as lstm
INFO:tensorflow:Assets written to: models/RNN_Final-05-0.968.model/assets
INFO:tensorflow:Assets written to: models/RNN_Final-05-0.968.model/assets
1304/1304 [=====] - 542s 416ms/step - loss: 0.0759 - accuracy: 0.9679 - val_loss: 0.0819 - val_accuracy: 0.9684
Loosing Probability: 0.0819472899150848
Accuracy Probability: 0.9684005975723267
```

Figure 11: Ethereum (ETC) output for LSTM

From the output terminal of Google Colab, we get the following output.

```
Epoch 1/5 1304/1304 [=====] - ETA: 0s -
loss: 0.1022 accuracy: 0.9545 1304/1304 [=====]
- 545s 418ms/step - loss: 0.1022 accuracy: 0.9545 - val_loss : 0.1594 - val_accuracy :
```

```

0.9307Epoch2/51304/1304[=====]-ETA :
0s-loss : 0.0958accuracy : 0.95921304/1304[=====]
] - 547s419ms/step - loss : 0.0958accuracy : 0.9592 - val_loss : 0.1284 - val_accuracy :
0.9487Epoch3/51304/1304[=====]-ETA :
0s-loss : 0.0871accuracy : 0.96261304/1304[=====]
] - 540s414ms/step - loss : 0.0871accuracy : 0.9626 - val_loss : 0.0945 - val_accuracy :
0.9633Epoch4/51304/1304[=====]-ETA :
0s-loss : 0.0788accuracy : 0.96681304/1304[=====]
] - 545s418ms/step - loss : 0.0788accuracy : 0.9668 - val_loss : 0.1110 - val_accuracy :
0.9507Epoch5/51304/1304[=====]-ETA :
0s-loss : 0.0759accuracy : 0.96791304/1304[=====]
]-542s416ms/step-loss : 0.0759accuracy : 0.9679-val_loss : 0.0819-val_accuracy : 0.9684

```

Loosing Probability: 0.08194772899150848

Accuracy Probability: 0.9684005975723267

11.2 For SimpleRNN model

For Bitcoin (BTC):

```

Epoch 1/5
1300/1300 [=====] - ETA: 0s - loss: 0.3259 - accuracy: 0.8408INFO:tensorflow:Assets written to: models/RNN_Final
1300/1300 [=====] - 161s 122ms/step - loss: 0.3259 - accuracy: 0.8408 - val_loss: 0.2130 - val_accuracy: 0.8951
Epoch 2/5
1300/1300 [=====] - ETA: 0s - loss: 0.2518 - accuracy: 0.8759INFO:tensorflow:Assets written to: models/RNN_Final
1300/1300 [=====] - 158s 122ms/step - loss: 0.2518 - accuracy: 0.8759 - val_loss: 0.2158 - val_accuracy: 0.8877
Epoch 3/5
1300/1300 [=====] - ETA: 0s - loss: 0.2562 - accuracy: 0.8739INFO:tensorflow:Assets written to: models/RNN_Final
1300/1300 [=====] - 159s 123ms/step - loss: 0.2562 - accuracy: 0.8739 - val_loss: 0.2027 - val_accuracy: 0.8953
Epoch 4/5
1300/1300 [=====] - ETA: 0s - loss: 0.2319 - accuracy: 0.8860INFO:tensorflow:Assets written to: models/RNN_Final
1300/1300 [=====] - 161s 124ms/step - loss: 0.2319 - accuracy: 0.8860 - val_loss: 0.1961 - val_accuracy: 0.9076
Epoch 5/5
1300/1300 [=====] - ETA: 0s - loss: 0.2250 - accuracy: 0.8894INFO:tensorflow:Assets written to: models/RNN_Final
1300/1300 [=====] - 158s 121ms/step - loss: 0.2250 - accuracy: 0.8894 - val_loss: 0.1972 - val_accuracy: 0.9076
Loosing Probability: 0.1971551775932312
Accuracy Probability: 0.9076476097106934
INFO:tensorflow:Assets written to: models/BTC-USD-60-SEQ-3-PRED-1642676419/assets

```

Figure 12: Bitcoin (BTC) output for SimpleRNN

From the output terminal of Google Colab, we get the following output

```

Epoch 1/5 1300/1300 [=====] - ETA: 0s
- loss: 0.3259 accuracy: 0.8408INFO:tensorflow:Assets written to: models/RNN_Final -
01 - 0.895.model/assets1300/1300[=====]
] - 161s122ms/step - loss : 0.3259accuracy : 0.8408 - val_loss : 0.2130 - val_accuracy :
0.8951Epoch2/51300/1300[=====]-ETA :
0s-loss : 0.2518accuracy : 0.8759INFO : tensorflow : Assetswrittento : models/RNN_Final-
02 - 0.888.model/assets1300/1300[=====]
] - 158s122ms/step - loss : 0.2518accuracy : 0.8759 - val_loss : 0.2158 - val_accuracy :

```

```

0.8877Epoch3/51300/1300[=====] - ETA :
0s - loss : 0.2562 accuracy : 0.8739 INFO : tensorflow : Assets written to : models/RNN_Final-
03 - 0.895.model/assets1300/1300[=====
] - 159s123ms/step - loss : 0.2562 accuracy : 0.8739 - val_loss : 0.2027 - val_accuracy :
0.8953Epoch4/51300/1300[=====] - ETA :
0s - loss : 0.2319 accuracy : 0.8860 INFO : tensorflow : Assets written to : models/RNN_Final-
04 - 0.908.model/assets1300/1300[=====
] - 161s124ms/step - loss : 0.2319 accuracy : 0.8860 - val_loss : 0.1961 - val_accuracy :
0.9076Epoch5/51300/1300[=====] - ETA :
0s - loss : 0.2250 accuracy : 0.8894 INFO : tensorflow : Assets written to : models/RNN_Final-
05 - 0.908.model/assets1300/1300[=====
] - 158s121ms/step - loss : 0.2250 accuracy : 0.8894 - val_loss : 0.1972 - val_accuracy : 0.9076

```

Loosing Probability: 0.1971551775932312

Accuracy Probability: 0.9076476097106934

For Litecoin (LTC):

```

Epoch 1/5
1218/1218 [=====] - ETA: 0s - loss: 0.2459 - accuracy: 0.8883 INFO:tensorflow:Assets written to: models/RNN_Final-
1218/1218 [=====] - 168s 136ms/step - loss: 0.2459 - accuracy: 0.8883 - val_loss: 0.0935 - val_accuracy: 0.9632
Epoch 2/5
1218/1218 [=====] - ETA: 0s - loss: 0.1183 - accuracy: 0.9524 INFO:tensorflow:Assets written to: models/RNN_Final-
1218/1218 [=====] - 166s 137ms/step - loss: 0.1183 - accuracy: 0.9524 - val_loss: 0.0999 - val_accuracy: 0.9588
Epoch 3/5
1218/1218 [=====] - ETA: 0s - loss: 0.0909 - accuracy: 0.9643 INFO:tensorflow:Assets written to: models/RNN_Final-
1218/1218 [=====] - 163s 134ms/step - loss: 0.0909 - accuracy: 0.9643 - val_loss: 0.0531 - val_accuracy: 0.9842
Epoch 4/5
1218/1218 [=====] - ETA: 0s - loss: 0.0780 - accuracy: 0.9697 INFO:tensorflow:Assets written to: models/RNN_Final-
1218/1218 [=====] - 165s 135ms/step - loss: 0.0780 - accuracy: 0.9697 - val_loss: 0.0486 - val_accuracy: 0.9837
Epoch 5/5
1218/1218 [=====] - ETA: 0s - loss: 0.0680 - accuracy: 0.9735 INFO:tensorflow:Assets written to: models/RNN_Final-
1218/1218 [=====] - 163s 134ms/step - loss: 0.0680 - accuracy: 0.9735 - val_loss: 0.0375 - val_accuracy: 0.9889
Loosing Probability: 0.037480201572179794
Accuracy Probability: 0.9888601303100586
INFO:tensorflow:Assets written to: models/LTC-USD-60-SEQ-3-PRED-1642761258/assets

```

Figure 13: Litecoin (LTC) output for SimpleRNN

From the output terminal of Google Colab, we get the following output

```

Epoch 1/5 1218/1218 [=====] - ETA: 0s
- loss: 0.2459 accuracy: 0.8883 INFO:tensorflow:Assets written to: models/RNN_Final -
01 - 0.963.model/assets1218/1218[=====
] - 168s136ms/step - loss : 0.2459 accuracy : 0.8883 - val_loss : 0.0935 - val_accuracy :
0.9632Epoch2/51218/1218[=====] - ETA :
0s - loss : 0.1183 accuracy : 0.9524 INFO : tensorflow : Assets written to : models/RNN_Final-
02 - 0.959.model/assets1218/1218[=====
] - 166s137ms/step - loss : 0.1183 accuracy : 0.9524 - val_loss : 0.0999 - val_accuracy :
0.9588Epoch3/51218/1218[=====] - ETA :
0s - loss : 0.0909 accuracy : 0.9643 INFO : tensorflow : Assets written to : models/RNN_Final-
03 - 0.984.model/assets1218/1218[=====
] - 163s134ms/step - loss : 0.0909 accuracy : 0.9643 - val_loss : 0.0531 - val_accuracy :

```

0.9842Epoch4/51218/1218[=====]-ETA :
 0s-loss : 0.0780accuracy : 0.9697INFO : tensorflow : Assetswrittento : models/RNN_{Final}-
 04 - 0.984.model/assets1218/1218[=====]
] - 165s135ms/step - loss : 0.0780accuracy : 0.9697 - val_{loss} : 0.0486 - val_{accuracy} :
 0.9837Epoch5/51218/1218[=====]-ETA :
 0s-loss : 0.0680accuracy : 0.9735INFO : tensorflow : Assetswrittento : models/RNN_{Final}-
 05 - 0.989.model/assets1218/1218[=====]
] - 163s134ms/step-loss : 0.0680accuracy : 0.9735-val_{loss} : 0.0375-val_{accuracy} : 0.9889

Loosing Probability: 0.037480201572179794

Accuracy Probability: 0.9888601303100586

For Ethereum (ETH):

```
Epoch 1/5
1304/1304 [=====] - ETA: 0s - loss: 0.2776 - accuracy: 0.8678INFO:tensorflow:Assets written to: models/RNN_Final-
1304/1304 [=====] - 179s 135ms/step - loss: 0.2776 - accuracy: 0.8678 - val_loss: 0.2049 - val_accuracy: 0.8979
Epoch 2/5
1304/1304 [=====] - ETA: 0s - loss: 0.1984 - accuracy: 0.9080INFO:tensorflow:Assets written to: models/RNN_Final-
1304/1304 [=====] - 175s 134ms/step - loss: 0.1984 - accuracy: 0.9080 - val_loss: 0.1942 - val_accuracy: 0.9074
Epoch 3/5
1304/1304 [=====] - ETA: 0s - loss: 0.1797 - accuracy: 0.9169INFO:tensorflow:Assets written to: models/RNN_Final-
1304/1304 [=====] - 175s 134ms/step - loss: 0.1797 - accuracy: 0.9169 - val_loss: 0.2385 - val_accuracy: 0.8935
Epoch 4/5
1304/1304 [=====] - ETA: 0s - loss: 0.1624 - accuracy: 0.9245INFO:tensorflow:Assets written to: models/RNN_Final-
1304/1304 [=====] - 176s 135ms/step - loss: 0.1624 - accuracy: 0.9245 - val_loss: 0.1370 - val_accuracy: 0.9351
Epoch 5/5
1304/1304 [=====] - ETA: 0s - loss: 0.1475 - accuracy: 0.9329INFO:tensorflow:Assets written to: models/RNN_Final-
1304/1304 [=====] - 178s 137ms/step - loss: 0.1475 - accuracy: 0.9329 - val_loss: 0.1493 - val_accuracy: 0.9305
Loosing Probability: 0.14929451048374176
Accuracy Probability: 0.9304812550544739
INFO:tensorflow:Assets written to: models/ETH-USD-60-SEQ-3-PRED-1642762601/assets
```

Figure 14: Ethereum (ETH) output for SimpleRNN

From the output terminal of Google Colab, we get the following output

Epoch 1/5 1304/1304 [=====] - ETA: 0s
 - loss: 0.2776 accuracy: 0.8678INFO:tensorflow:Assets written to: models/RNN_{Final}-
 01 - 0.898.model/assets1304/1304[=====]
] - 179s135ms/step - loss : 0.2776accuracy : 0.8678 - val_{loss} : 0.2049 - val_{accuracy} :
 0.8979Epoch2/51304/1304[=====]-ETA :
 0s-loss : 0.1984accuracy : 0.9080INFO : tensorflow : Assetswrittento : models/RNN_{Final}-
 02 - 0.907.model/assets1304/1304[=====]
] - 175s134ms/step - loss : 0.1984accuracy : 0.9080 - val_{loss} : 0.1942 - val_{accuracy} :
 0.9074Epoch3/51304/1304[=====]-ETA :
 0s-loss : 0.1797accuracy : 0.9169INFO : tensorflow : Assetswrittento : models/RNN_{Final}-
 03 - 0.894.model/assets1304/1304[=====]
] - 175s134ms/step - loss : 0.1797accuracy : 0.9169 - val_{loss} : 0.2385 - val_{accuracy} :
 0.8935Epoch4/51304/1304[=====]-ETA :
 0s-loss : 0.1624accuracy : 0.9245INFO : tensorflow : Assetswrittento : models/RNN_{Final}-
 04 - 0.935.model/assets1304/1304[=====]
] - 176s135ms/step - loss : 0.1624accuracy : 0.9245 - val_{loss} : 0.1370 - val_{accuracy} :


```
0.9351Epoch5/51304/1304[=====]-ETA :
0s-loss : 0.1475accuracy : 0.9329INFO : tensorflow : Assetswrittento : models/RNN_Final-
05 - 0.930.model/assets1304/1304[=====]
]-178s137ms/step-loss : 0.1475accuracy : 0.9329-val_loss : 0.1493-val_accuracy : 0.9305
```

Loosing Probability: 0.14929451048374176

Accuracy Probability: 0.9304812550544739

11.3 For Random Forest Algorithm

For this model, we have taken the R-squared error in order to make the evaluation. It shows how near the regression line (that is, the plotted projected values) is to the actual data values. The R squared value ranges from 0 to 1, with 0 indicating that the model does not match the data and 1 indicating that the model fits the dataset completely.

For example, An R-squared error of 0.6 suggests that the model cannot explain 60 percent of the variability in the result data.

For Bitcoin(BTC):

From the output terminal of Google Colab, we get the following output

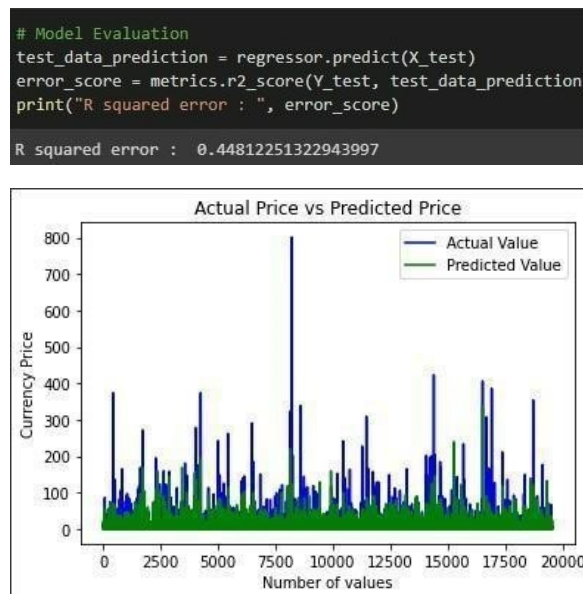


Figure 15: Ethereum (ETH) output for SimpleRNN

For Litecoin (LTC): From the output terminal of Google Colab, we get the following output

```
# Model Evaluation
test_data_prediction = regressor.predict(X_test)
error_score = metrics.r2_score(Y_test, test_data_prediction)
print("R squared error : ", error_score)

R squared error : 0.42108963626992946
```

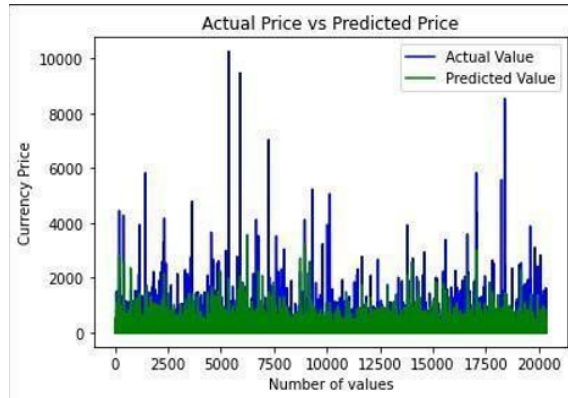


Figure 16: Litecoin output for Random Forest

For Ethereum (ETH):

From the output terminal of Google Colab, we get the following output:

```
# Model Evaluation
test_data_prediction = regressor.predict(X_test)
error_score = metrics.r2_score(Y_test, test_data_prediction)
print("R squared error : ", error_score)

R squared error : 0.4153062771384275
```

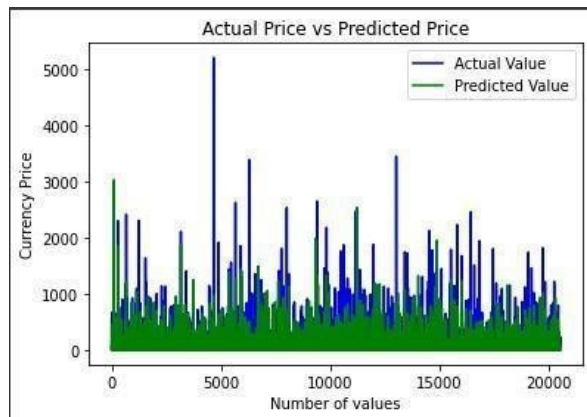


Figure 17: Ethereum (ETH) output for Random Forest

11.4 Comparison Between Models

Comparison between LSTM and SimpleRNN model:

Cryptocurrency Name	Accuracy Probability(LSTM)	Accuracy Probability(SimpleRNN)
Bitcoin (BTC)	0.917486	0.907647
Litecoin (LTC)	0.991968	0.988860
Ethereum (ETH)	0.968400	0.930481

For Random Forest model:

Cryptocurrency Name	R-squared error value
Bitcoin (BTC)	0.448122
Litecoin (LTC)	0.421089
Ethereum (ETH)	0.415306

11.5 Results

From the tables stated above, we find that

In the LSTM model,

Bitcoin or BTC shows 91 percent accuracy.

Litecoin or LTC shows 99 percent accuracy.

Ethereum or ETH shows 96 percent accuracy.

In the SimpleRNN model,

Bitcoin or BTC shows 90 percent accuracy.

Litecoin or LTC shows 98 percent accuracy.

Ethereum or ETH shows 93 percent accuracy.

So, LSTM works better than simpleRNN in our dataset.

For the random forest model, the R-squared error value is much high. Our Bitcoin R-squared value is 0.44 which means the model can't explain 44 percent of the variability in the result data. For Litecoin it can't explain 42 percent and for Ethereum it can't explain 41 percent of the model data as the r-squared data for Litecoin and Ethereum is 0.42 and 0.41 respectively.

So, the random forest model doesn't fit in our dataset at all.

From the discussion above, we can state that Long short-term memory or LSTM shows the best accuracy among all others. SimpleRNN model also shows good accuracy but less than the LSTM model. The Random Forest model is ineffective for real-time predictions because of its much high R-squared error.

Chapter 7

12 Conclusion and Future Work

12.1 Conclusion

The main purpose of this study is to predict the price of different cryptocurrencies using various machine learning models. We selected Bitcoin (BTC), Litecoin (LTC) and Ethereum (ETC) as our cryptocurrencies and choose Long Short-Term Memory or LSTM, SimpleRNN and Random Forest as our machine learning models. We choose Python as our programming language and used its various libraries and api in our work. We can say our research is worked out and we implemented all resources which were required for the implementation of the models. Every model worked very well and among them, Long Short-Term Memory or LSTM showed the best accuracy. SimpleRNN became second and Random Forest showed poor performance compared to LSTM and SimpleRNN.

12.2 Future Work

The cryptocurrency market is booming. New currencies are getting into the market and prices have gone up much in recent years. New countries are accepting cryptocurrency every year and many take giants like Tesla and Facebook are also working with cryptocurrencies. In the future, we will implement more neural and deep learning models like CNN, cuDRNN and GRU models. Implementing GRU with the existing LSTM model may give us better results for short datasets. For implementing CuDRNN our pc requires more processing power (High GPU) for running the algorithms. Also, we have plans to measure the volatility of the Cryptocurrency market using GARCH-MIDAS analysis [14]. Alternatively, the number of datasets used to train our model can be expanded and we can work with a new bigger dataset later for more reliable predictions for an exact time of the whole year.

13 References

- [1] "Blockchain based currency Historical Prices". OfficialData.org. Archived from the original on 4 July 2018. Retrieved 3 July 2018.
- [2] "WSJ Markets Blockchain based currency USD". Retrieved 8 February 2021
- [3] L.S. Reddy, Dr. P. Sriramya, A Research On Blockchain based currency Price Prediction Using Machine Learning Algorithms. (2020)
- [4] Patrick Jaquart, David Dann, Christof Weinhardt, Short-term blockchain based currency market prediction via machine learning, *The Journal of Finance and Data Science*, Volume 7, 2021, Pages 45-66, ISSN 2405-9188, <https://doi.org/10.1016/j.jfds.2021.03.001>.
- [5] Lahmiri, S., Bekiros, S. Deep Learning Forecasting in Cryptocurrency High-Frequency Trading. *Cogn Comput* 13, 485–487 (2021).
- [6] Hitam, Nor Azizah Ismail, Amelia Ritahani. (2018). Comparative Performance of Machine Learning Algorithms for Cryptocurrency Forecasting. *Indonesian Journal of Electrical Engineering and Computer Science*. 11. 1121-1128. [10.11591/ijeecs.v11.i3.pp1121-1128](https://doi.org/10.11591/ijeecs.v11.i3.pp1121-1128).
- [7] S. McNally, J. Roche and S. Caton, "Predicting the Price of Blockchain based currency Using Machine Learning," 2018 26th Euromicro International Conference on Parallel, Distributed and Network-based Processing (PDP), 2018.
- [8] S. Roy, S. Nanjiba, Dr. A. Chakrabarty, Blockchain based currency Price Forecasting Based on Historical Data, (2018)
- [9] A. Greaves and B. Au, "Using the bitcoin transaction graph to predict the price of bitcoin," ,2015
- [10] H. Jang and J. Lee, "An Empirical Study on Modeling and Prediction of Bitcoin Prices With Bayesian Neural Networks Based on Blockchain Information," *IEEE ACCESS*, vol. 6, pp. 5427–5437, 2018
- [11] E. Sin and L. Wang, "Bitcoin Price Prediction Using Ensembles of Neural Networks," in 2017 13TH INTERNATIONAL CONFERENCE ON NATURAL COMPUTATION, FUZZY SYSTEMS AND KNOWLEDGE DISCOVERY (ICNC-FSKD), 2017, pp. 666–671
- [12] S. McNally, J. Roche, and S. Caton, "Predicting the price of Bitcoin using Machine Learning," in *Parallel, Distributed and Network-based Processing (PDP)*, 2018 26th Euromicro International Conference on, 2018, pp. 339–343.

[13] R. Mittal, S. Arora, and M. P. S. Bhatia, "AUTOMATED CRYPTOCURRENCIES PRICESPREDICTION USING MACHINE LEARNING," 2018.

[14] Asgharian, Hossein; Hou, Ai Jun; Javed, Farrukh, "The importance of the macroeconomicvariables in forecasting stock return variance: A GARCH-MIDAS approach."