UNIVERSITY OF CALIFORNIA

Los Angeles

Bitcoin Price Forecast Using LSTM and GRU Recurrent networks, and Hidden Markov Model

A thesis submitted in partial satisfaction
of the requirements for the degree
Master of Science in Statistics

by

Yike Xu

2020

ABSTRACT OF THE THESIS

Bitcoin Price Forecast Using
LSTM and GRU Recurrent networks,
and Hidden Markov Model

by

Yike Xu
Master of Science in Statistics
University of California, Los Angeles, 2020
Professor Yingnian Wu, Chair

Bitcoin, the first decentralized cryptocurrency, has become popular not only because a growing size of merchants accepts it in transactions, but also because people buy it as an investment. This study focuses on the Bitcoin price forecast using Hidden Markov Model and two machine learning methods, LSTM and GRU Recurrent networks. Evaluated by MAPE and RMSE, the results indicate that the Hidden Markov Model with the Gaussian Mixture Models has the best performance among all methods. The GRU model outperforms the LSTM model, though sometimes it might have a more extreme result. When the price remains constant or changes steadily, the predictions are more precise than the fluctuation period.

The thesis of Yike Xu is approved.

Frederic R. Paik Schoenberg

Robert Gould

Yingnian Wu, Committee Chair

University of California, Los Angeles

2020

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1

Introduction

Cryptocurrencies, or digit assets, have gained public attention over the last decade. According to a study by the University of Cambridge in 2018, there are exceeding 139 million global user accounts at cryptocurrency service providers with at least 35 million identity-verified users [RBK18]. Another survey showed that approximately 7.95% of the United States adults, about 20 million users, have purchased the digit assets [Ng18]. There is no doubt that this industry is rapidly growing.

As of April 2020, there are more than five thousand cryptocurrencies in the market [Bag20]. Among those, the most famous and outstanding one is Bitcoin. Its inventor registered the domain name bitcoin.org on August 18, 2008, and published a whitepaper titled *Bitcoin: A Peer-to-Peer Electronic Cash System* under the pseudonym Satoshi Nakamoto later that year [con20]. Bitcoin uses a peer-to-peer network, relying on Blockchain technology to verify the transaction instead of the central authorities [Nak19]. Since published, Bitcoin has become the most influential cryptocurrency. As of July 16, 2020, Bitcoin Market Capitalization, which is calculated by multiplying bitcoin's price by its current circulating supply, is 168 billion U.S. dollars (USD) [Coi20b, Jay20]. It occupies 62.5% of the market compared to the second popular cryptocurrency, Ethereum, which takes only around 10% of the total Market Capitalization [Coi20b].

One reason that Bitcoins rises to fame is that it can be exchanged and spent conveniently and globally with a low transaction fee. Users can buy and sell bitcoins through both online exchanges and offline bitcoin ATMs. As of July 2020, there are about 8,677

Bitcoin ATMs, three-quarters of which locate in the United States [Rad20]. There are estimated more than 500 cryptocurrency exchanges worldwide, 314 of which are tracked on CoinMarketCap [Zam20, Coi20c]. These exchanges accept diverse currencies, which makes the Bitcoin transaction accessible. With all exchanges tracked on Bitcoincharts, U.S. Dollars (USD) is the most popular currency that takes 34% volume in the market, followed by Japanese Yen (27%) and Euro (22%) [Bit20a]. Moreover, there is an increasing number of merchants around the world that allow Bitcoin in transactions. For example, Bitcoins are accepted for donations on Wikipedia, cell phone bill (AT&T), services (Microsoft account), and other local businesses [Fou20, AT19, Smi14]. Bitcoin users can find such local stores and restaurants by searching on `Coinmap.org` [Coi20a]. Besides, many people purchase Bitcoin as an investment, which makes the prediction of Bitcoin attractive. A successful forecast of the future price or the price trend can provide abundant profits since investors can use such information to assist in decision-making.

This study aims to predict the next time step's Bitcoin closing price based on the previous 300-minute (20-time steps) information and the trained models with the optimal hyperparameters. We use 15-minute Bitcoin data from Bitstamp Exchange, a European-based exchange that supports U.S. Dollars (USD) [Bitte]. In the past 30 days, the Bitstamp exchange is a popular exchange which occupies 16% of the market volume [Bit20a]. It was established and had the first transaction in September 2011 [Bitte]. The data is separated by timestamp into a training set and a testing set that are analyzed and discussed before model fitting.

We implemented three forecasting methods, including Long Short-term Memory Model (LSTM), Gated Recurrent Units Model (GRU), and Hidden Markov Model (HMM) with two different emission probabilities. The LSTM model and the GRU model are popular deep learning methods for predicting Time Series data. They are well-known for learning from long-term sequence dependency, though increasing the computing complexity by introducing extra parameters [Ola15]. The HMM is another suitable and recognized method in time series forecast, strongly depending on Markov assumptions [GD12]. The reason to use this method is that the price of a cryptocurrency is determined by some underlying stochastic process that is invisible to investors, which matches the properties of the HMM [GD12, Zha04]. Shi and Weigend successfully used this model in their previous work on financial time series data [SW97].

In this study, each model's performance is evaluated by MAPE and RMSE of the testing set. We will discuss these three methods' details and results and conclude the best model in the later chapter.

CHAPTER 2

Dataset

2.1 Data Overview

The raw dataset includes the Bitcoin information from a European-based exchange, BitStamp, since its first transaction in September 2011. The data was extracted using an API from the website Bitcoincharts.com, which is recommended by Bitcoin.org. [Bit20b, Bit20c]. It includes eight variables, shown in Table 2.1, and 301,803 observations that are in chronological order. Each observation includes price information and volume information for every 15-minute time step. The first-ever record on Bitstamp was on September 13, 2011, at 13:45 UTC, which includes bitcoin transactions between 13:45 and 14:00 UTC. The last observation in the dataset was made on April 25, when this dataset was downloaded.

Within more than 300 thousand time steps in this raw dataset, about 12% of them lack price information or volume information. The Fig. 2.1 plots the time interval between non-missing observations. Since the time step is every 15 minutes, we ignore the observations that are 15 minutes apart as expected. From this figure, it is evident that most of these missing values occurred before 2014. At the early stage, both Bitcoin and the Bitstamp Exchange platform are less known by the public. It could be possible that there was no trade within 15 minutes, which led to the empty record of such information in the dataset.

Since the missing observations before 2014 do not provide much useful information, and the rest of the data is sufficiently large, we eliminated the time to a narrower range from January 1, 2014, to April 25, 2020. The subset dataset only has 534 out of total 221,476

Table 2.1: Variables Description

Variables	Meaning
Open	Price (USD) of the first trade at the current time step
High	Highest Price (USD) of trades at the current time step
Low	Lowest Price (USD) of trades at the current time step
Close	Price (USD) of the last trade before the next time step
Volume_BTC	Trade volume in Bitcoins
Volume_Currency	Trade volume in USD
Weighted_Price	Weighted Bitcoin price (USD)
Timestamp	Ten digits integer that represents time

Note:
1. Each time step is a period from the current timestamp to the next timestamp which last for 15 minutes.
2. Timestamp uses Coordinated Universal Time standard (UTC).
3. All prices are in U.S. dollars (USD).

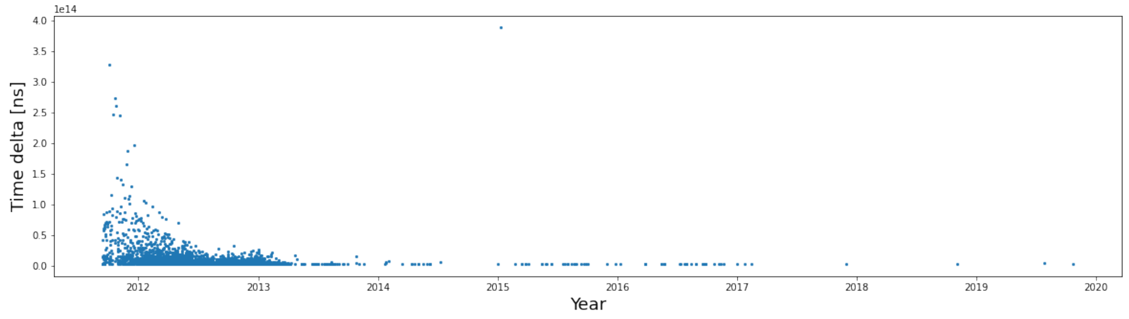

Figure 2.1: Detect Missing Observations

missing observations (0.24%), compared to the original 12% missing observations. In Fig. 2.1, one point in early 2015 is distinguishable from others, which matches the most prolonged missing observations in Table 2.3. The data was missing for a long time due to the Bitstamp Exchange reporting a hack and losing less than 19 thousand coins in January 2015 [Whi15]. The exchange suspended the service from January 05 to January 09, 2015, for investigation, which is the longest termination since the establishment of the Bitstamp Exchange. From the summary of the subset dataset in Table 2.3, the rest of the missing observations do not

Table 2.2: Raw Dataset Distribution

Variables (Units)	Min	Max	Mean	Standard Deviation
Open (USD)	2.2200	19626.3000	3021.7472	3757.15
High (USD)	2.2300	19666.0000	3030.7827	3770.21
Low (USD)	1.5000	19590.0100	3011.8442	3742.88
Close (USD)	2.2300	19614.0000	3021.7077	3757.05
Volume_BTC	0	11166.6908	116.1701	223.84
Volume_Currency	0	29973282.78	340448.26	877447.72
Weighted_Price (USD)	0	19640.6236	3021.3428	3756.62
Timestamp (UTC)	2011-09-13 13:45:00	2020-04-25 06:15:00	N/A	N/A

persist longer than 2 hours.

Table 2.3: Time Duration of Missing Observations in Descending Order Since 2014

Rank	Date Start	Date End	Duration
1.	2015-01-05 09:00:00	2015-01-09 21:00:00	4.5 days
2.	2014-02-02 09:00:00	2014-02-02 11:00:00	2 hours
3.	2014-07-11 04:15:00	2014-07-11 06:00:00	1.75 hours
4.	2014-01-25 08:00:00	2014-01-25 09:30:00	1.5 hours
5.	2014-01-25 09:45:00	2014-01-25 10:45:00	1 hour
6.	2019-06-25 10:15:00	2019-07-25 10:15:00	1 hour

2.2 Exploratory Data Analysis

We have seen Bitcoin becoming popular during the last decade. Its closing price rose from a few dollars at the beginning to nearly 20,000 dollars in late 2017 and then dropped back

Table 2.4: Subset Dataset Summary Since Jan 1, 2014

Variables (Units)	Min	Max	Mean	Standard Deviation	Median
Open (USD)	159.0000	19626.3000	3602.9968	3866.8889	1060.5350
High (USD)	168.5900	19666.0000	3613.6810	3880.6429	1062.6850
Low (USD)	152.4000	19590.0100	3591.3041	3851.8403	1058.8550
Close (USD)	157.0500	19614.0000	3602.9475	3866.7783	1060.5450
Volume_BTC	0.0021	11166.6908	114.2812	213.7527	52.8168
Volume_Currency	1.3390	29973282.78	402261.18	948208.91	86467.44
Weighted_Price (USD)	166.5863	19640.6236	3602.5257	3866.3268	1060.7420

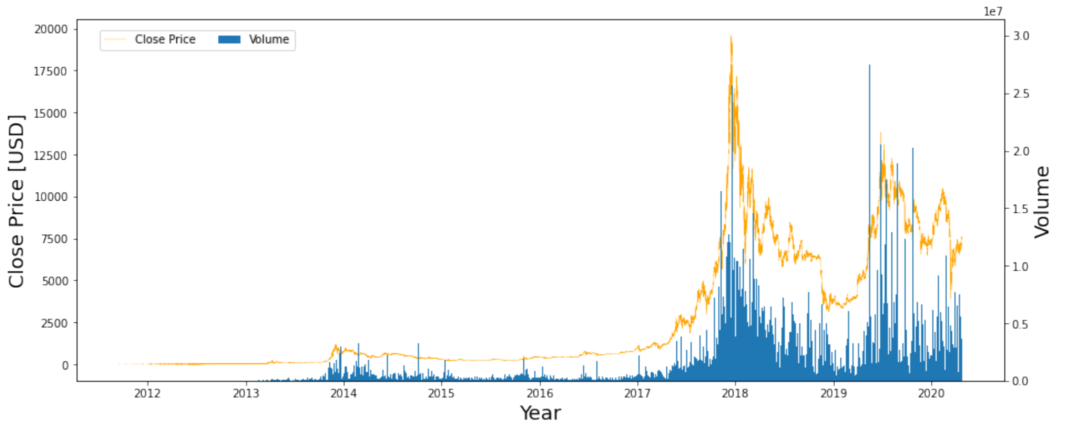

Figure 2.2: Price and Volume of Bitcoin on Bitstamp

to about 10,000 dollars in the first quarter of 2020. At the early stage, the closing price was relatively stable. There was one peak in late November and early December in 2013 when the price achieved 1,000 dollars. Since 2017, the Bitcoin market has fluctuated within the range from 3,000 to 20,000 USD. According to Table 2.2 and Fig. 2.2, the maximum Closing price 19614 USD occurred on December 17, 2017, which is more than 8,500 times of its minimum. After the second peak, the price dropped back to the short-term lowest 3135

USD on December 15, 2018. The price increased again to the recent highest price of 13837 USD on June 26, 2019. Besides these, there are always small fluctuations in price.

Another sign that Bitcoin has become famous is its trading volume. Generally, the volume of Bitcoin is associated with its price. There were a few transactions when the price was below 1,000 USD. The most massive volume in currency was 29,973,282 USD on June 26, 2019, which occurred on the same date as the third peak. The second-largest volume in currency was 29,213,880 USD on December 22, 2017, during the highest Bitcoin price period. There was a dramatic decline in the bitcoin price on March 12 and 13, 2020, shown in Fig. 2.3, when the price dropped 46.8% of its original price, from 7342 USD to 3902 USD, in 17 hours.

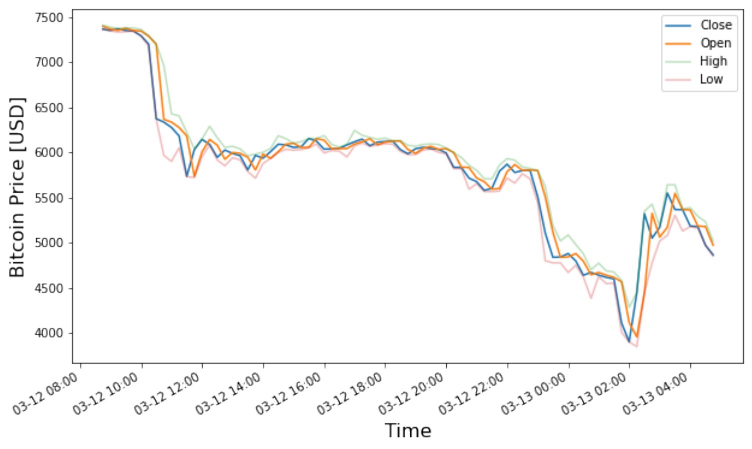

Figure 2.3: Historical Bitcion Price on 2020-03-12/13

Besides closing price, other variables also play a crucial role in the Bitcoin forecast. In Fig. 2.5, the difference between the closing price and opening price over the same 15-minute period is almost symmetric along zero axes. In Table 2.5, the median and the mean of the difference are close to 0. Both of these indicate that the Bitcoin price increases or decreases about the same amount within a small time interval. The summary Table 2.5 shows that 90% of such difference is within the range [-27, 27], meaning that within 15 minutes, the price is relatively stable most of the time. The difference is negligible when the overall price is low

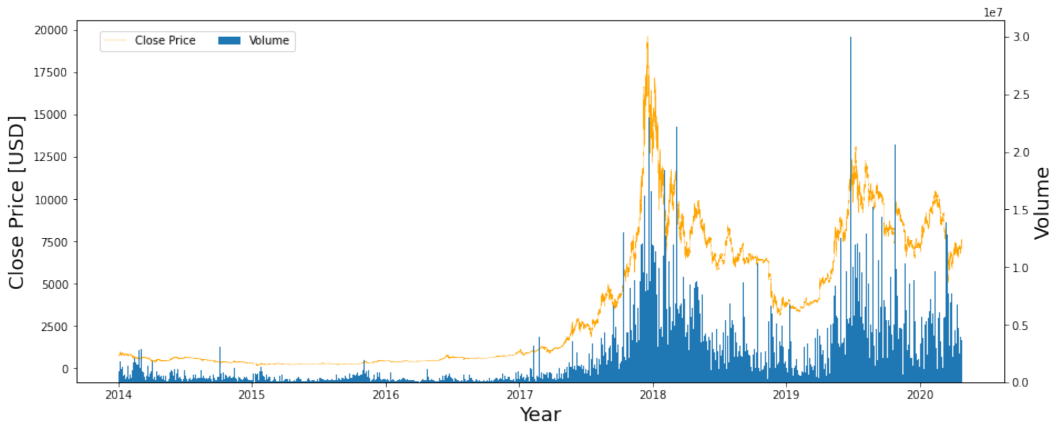

Figure 2.4: Price and Volume of Bitcoin on Bitstamp Since 2014

or at the local minimum, such as before 2017 and in early 2019. The difference is noteworthy from late 2017 to early 2018 and after the middle of 2019, which are coincident with periods that the Bitcoin price reaches its local maximum points, as we mentioned earlier. The most remarkable price movement occurred on June 26, 2019, at 20:30 UTC when the opening price of 13675.74 USD dropped to 11954.34 USD in 15 minutes. There are other observations with a massive difference of about 800 USD in absolute value.

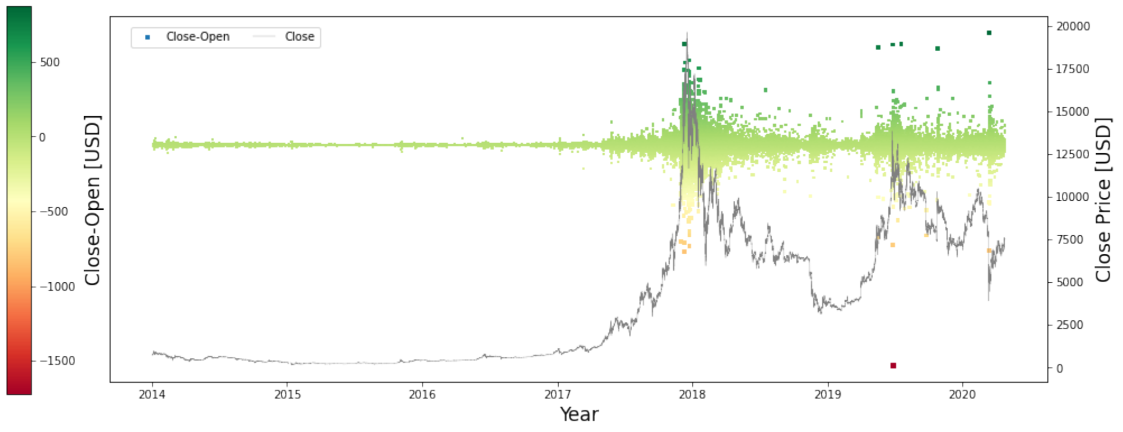

Figure 2.5: Price Difference between Closing Price and Opening Price

Besides the closing price and the opening price, the dataset includes the highest price and lowest price information. There is no doubt that the difference between the highest price

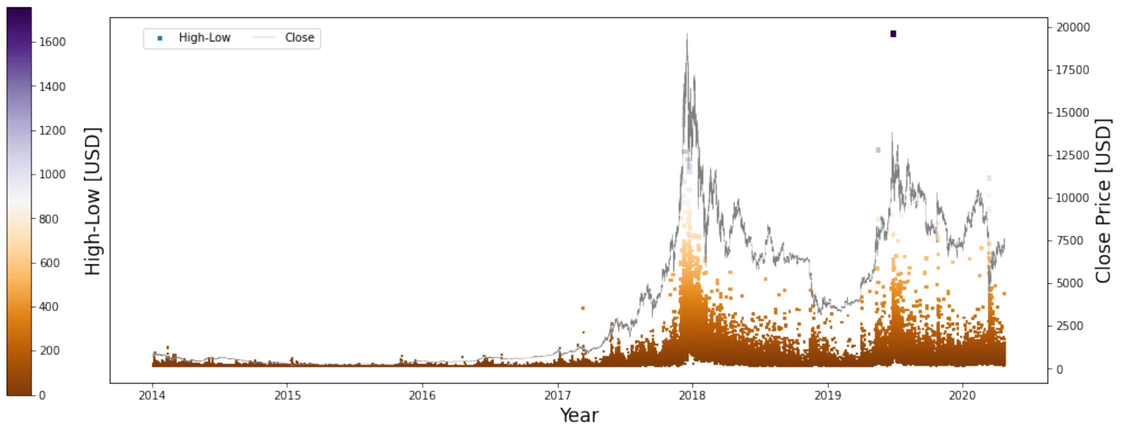

Figure 2.6: Price Difference between Highest Price and Lowest Price

and lowest price at the same time step is associated with the trend of the overall Bitcoin price. The highest price is likely to be markedly different from the lowest price before and after the price reaching the local maximum, which coincides with the period that the opening price is prominently different from the closing price. Contrasting to the difference between the closing price and the opening price that could be any rational numbers, the difference between the highest price and the lowest price is always positive. According to Table 2.5, the mean difference is 22.38 USD, and the median is 5.53 USD, which indicates that the distribution is skewed to the right, and potential outliers exist. The most remarkable difference over the same period occurred on June 26, 2019, at 20:30 UTC, when the closing price and the opening price also has the maximum gap. During these 15 minutes, the highest prices reached 13699.99 USD, and the lowest price dropped to 11942.24 USD. In Fig. 2.6, few other points have a High and Low gap larger than 1000 USD.

Since the fluctuation within the same time step is associated with the Bitcoin price itself, it is worth taking a look at the fractional change based on the opening price. The price movement within 15 minutes with respect to the opening price (**FC**) is defined as the following equation:

$$\text{FC} = \frac{\text{Close} - \text{Open}}{\text{Open}} \tag{2.1}$$

Table 2.5: Price Movements within One Time Step

	Close-Open (USD)	High-Low (USD)	FC	FH	FL
Mean	-0.493	22.3769	0.0000	0.0025	0.0029
std	30.1927	44.2150	0.0047	0.0039	0.0043
Min	-1721.4000	0.0000	-0.1441	0.0000	0.0000
5%	-27.0100	0.4000	-0.0060	0.0000	0.0000
Median	0.0000	5.5300	0.0000	0.0014	0.0018
95%	27.1600	93.4895	0.0059	0.0086	0.0096
Max	874.6900	1757.7500	0.1968	0.2033	0.2407

Then, the difference between the maximum price and the opening price over the same period compared to the opening price (**FH**) and the difference between the opening price and the minimum price over the same period with respect to the opening price (**FL**) are defined as the following equations:

$$\text{FH} = \frac{\text{High} - \text{Open}}{\text{Open}} \tag{2.2}$$

$$\text{FL} = \frac{\text{Open} - \text{Low}}{\text{Open}} \tag{2.3}$$

From Fig. 2.7 and Table 2.5, the distribution of FC is symmetric at the center of 0, indicating that the mean and the median are also around 0. Although the closing price is always different from the opening price, the fractional price variation within 15 minutes is likely to be trivial since 84.5% observations have the FC values within the range [-0.005,0.005] and 94% of them are within the range [-0.01,0.01]. Only 11 observations out of the sample size of 220,942 have absolute FC values beyond 0.1, most of which happened in January 2015, middle 2019, and March 2020.

The distributions of the FH values and the FL values are similar, shown in Fig. 2.8. Since the numerator of equation 2.3 is the opening price minus the lowest price, all FL values are non-negative. Like FC, most observations have small fractional differences between the opening price and the maximum price, and between it and the minimum price. About 84.5%

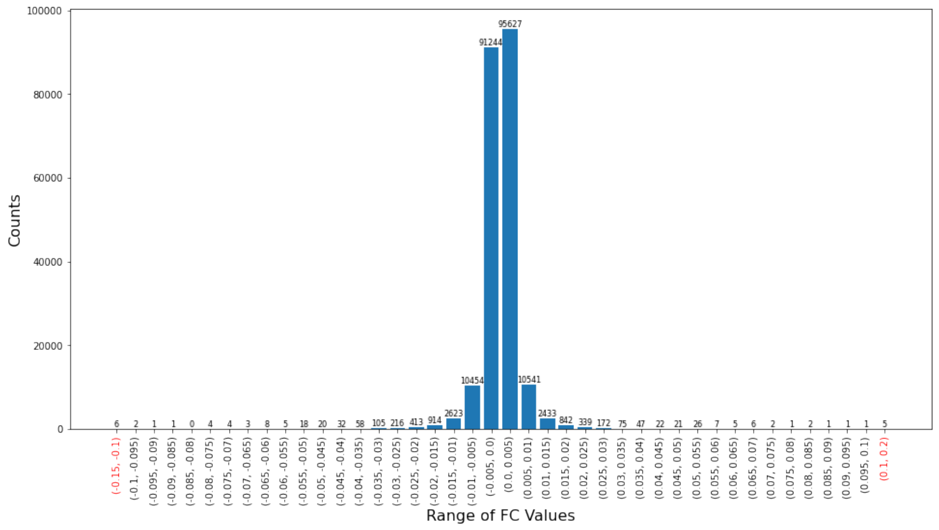
Figure 2.7: Distribution of FC Values

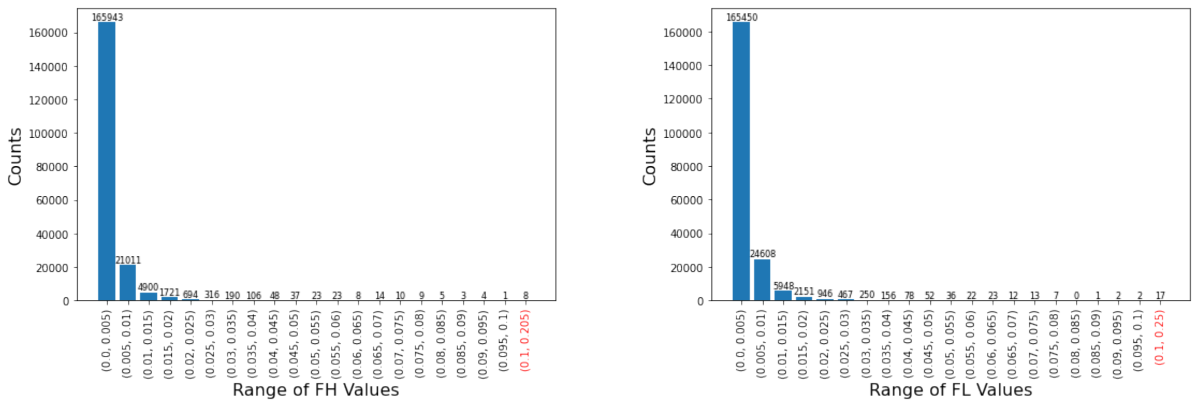
Figure 2.8: Distribution of FH (left) and Distribution of FL (right)

FH values and 86% FL values are within range [0,0.01]. A small number of FH values and FL values are beyond 0.1, most of which also occurred in January 2015, middle 2019, and March 2020 as the large FC values. As mentioned earlier, the price variation on March 12 and 13, 2020, was dramatic, shown in Fig. 2.3. The largest fractional change of price (FC) occurs on March 13 at 02:30 UTC when the opening price of 4445.41 USD increased 19.68% to 5320.1 USD. Simultaneously, the maximum price of 5349 USD was 20.33% higher than the opening price. In a nutshell, we can limit the fractional of High, Low, Close compared to Open to the range [0, 0.1] and [-0.1, 0.1].

2.3 Data Preprocessing

The dataset including information from January 1, 2014, to April 25, 2020, was separated into a training set and a testing set by timestamp, shown in Table 2.6.

It is necessary to validate both datasets. The observation on June 23, 2016, at 12:30 UTC has the lowest price of 1.5 USD while its opening price, maximum price, and closing price are around 580 to 590 USD. After checking the information from the exchange's official trade terminal, this price entry seems to be a mistake during the data collecting process. Since this observation was not in a fluctuation period, the actual lowest price is probably closer to 580 USD, which can be replaced by the closing price of 580.74 USD. Other information in both the training set and the testing set is reasonable.

Table 2.6: Training Set and Testing Set

Datasets	Start Date	End Date	Dimensions
Train	January 1, 2014	December 31, 2018	(175355, 8)
Test	January 1, 2019	April 25, 2020	(46141, 8)

As mentioned in Section 2.1, there are 0.24% missing information in the training set and the testing set combined. Since the dataset's sample size is sufficiently large, the missing values are filled by the last observation available to avoid any model fitting issues.

We need to prepossess the data before applying it to the forecasting methods. To reduce the run time and follow the suggestion in [Aun17], Long Short-term Memory Model (LSTM) and Gated Recurrent Units Model (GRU) forecast the next time step's Bitcoin closing price based on Open, Close, Volume_BTC, and Volume_Currency from the previous 20 time steps (300 minutes). We use the sliding window approach to read the data into an x-window that includes the past 20 time steps and a y-window that needs to be predicted. To prevent any correlation between any rows within x-window and y-window, we standardize the data in both windows by

$$p_i^* = \left(\frac{p_i}{p_0} - 1\right) \tag{2.4}$$

where p_0 is the first row of x-window, and p_i is any row in x-window or y-window. Since the Bitcoin dataset in total includes more than 200 thousand observations, we applied Python generators to create data windows and stored them in HDF5 data format using the h5py library to save the memory and time [Col13]. Both LSTM and GRU take the three-dimensional input in the shape of [batch size, time steps, number of features], or [batch_size, 20,4] in our case.

Following the Hidden Markov Model (HMM) method in [GD12], we used four different features (Close, High, Low, and Close) from LSTM and GRU. The observations are the 15-minute fractional change of the Bitcoin price that is discussed in Section 2.2:

$$(\text{FC},\text{FH},\text{FL}) = \left(\frac{\text{Close} - \text{Open}}{\text{Open}}, \frac{\text{High} - \text{Open}}{\text{Open}}, \frac{\text{Open} - \text{Low}}{\text{Open}}\right). \tag{2.5}$$

They are stored as two-dimensional NumPy array.

CHAPTER 3

Methodology

3.1 Long Short-Term Memory (LSTM)

Long short-term memory or LSTM network is a popular deep learning method in time series forecasting that was first introduced by Hochreiter and Schmidhuber in 1997 [HS97]. The LSTM has a Recurrent neural network (RNN) architecture, which uses a loop to pass information from one step of the network to the next [Ola15]. These loops can be unfolded or unrolled into a chain-like structure of the same network, shown in Fig. 3.1 [Ola15]. The LSTM is known for the capability to learn from sequence dependency while solving the traditional RNN's memory problem [Phi18]. For the RNN, if the necessary information is further back of the sequence, the gradients, which update neural network weights, can be minimal and leads to a short-term memory problem [Phi18]. In this case, the RNN struggles to fetch the relevant information, and then the LSTM was introduced to tackle this problem.

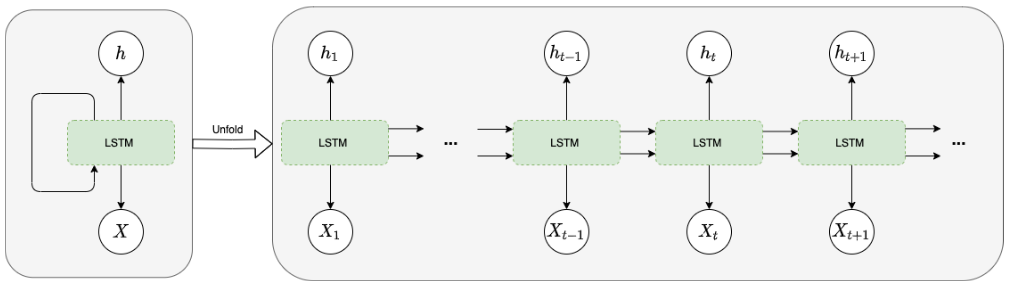

Figure 3.1: Unfold the LSTM Network to A Chain-like Structure

The LSTM is a variation of the RNN with a similar recurrent or chain-like structure but more layers. The key of the LSTM is the cell C_t that stories memory and three gates (input gate, output gate, and forget gate) that control the information added to or removed from the cell. Two activation functions play an essential role in the LSTM network. The first one is a sigmoid function, also known as a logistic function. It regulates how much information to be passed through in these gates, which is defined as:

$$\sigma(x) = \frac{1}{1+e^{-x}} = \frac{e^x}{1+e^x} \tag{3.1}$$

The outputs of this function are the numbers within range 0 to 1, shown in Fig. 3.2. If the result is close to 0, less information is allowed to be let through. In contrast, if the result is close to 1, more information can be involved in the next step. The second one is a hyperbolic tangent or a tanh function that outputs the values within range -1 to 1, shown in Fig. 3.3, which is defined as:

$$\tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}} \tag{3.2}$$

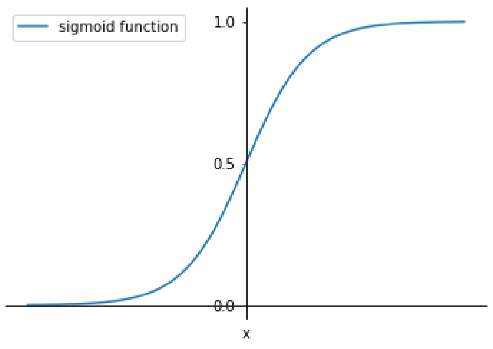

Figure 3.2: Sigmoid Function

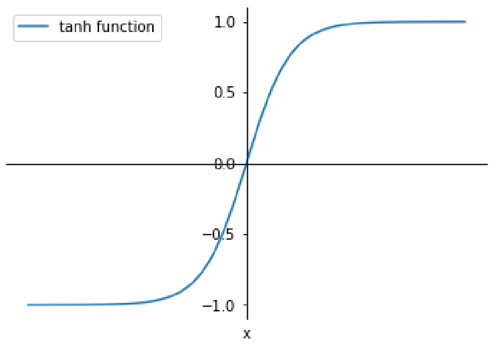

Figure 3.3: Tanh Function

A LSTM unit involves four parts, illustrated in Fig. 3.4. The first part is called a forget gate layer that decides which previous information is discarded from the cell. It passes the hidden vector from the previous step h_{t-1} and new input X_t into the sigmoid function, and

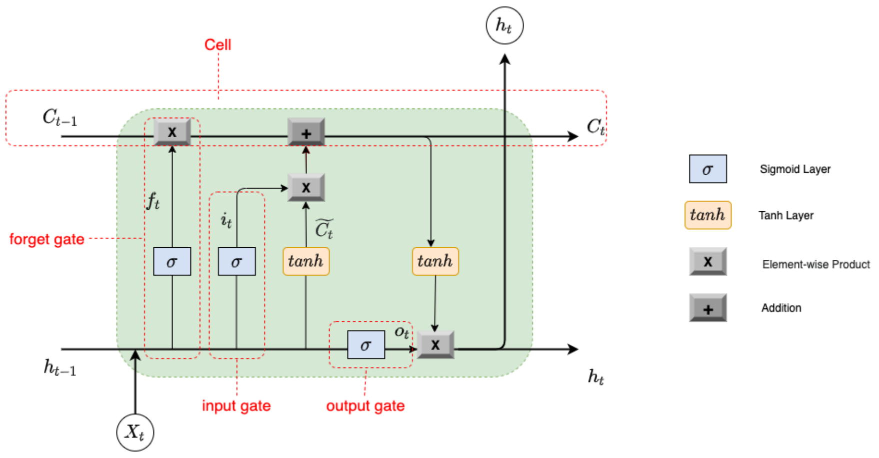

Figure 3.4: The LSTM Unit

produce a forget gate vector f_t from the function:

$$f_t = \sigma\left(\mathbf{W}_f x_t + \mathbf{U}_f h_{t-1} + b_f\right) \tag{3.3}$$

where $\mathbf{W}_f \in \mathbb{R}^{h\times d}$ and $\boldsymbol{U}_f \in \mathbb{R}^{h\times h}$ are the neural network weight matrices and $b_f \in \mathbb{R}^h$ is the neural network bias vector. They are updated through the learning process. The subscript f denotes the forget gate layer, and h and d in $\mathbb{R}^{h\times d}$ represent the number of hidden units and the number of input features.

The second part creates new information that involves an input gate layer and a tanh layer, shown in Fig. 3.4. The input gate layer decides how much new information is allowed. It takes the the previous hidden vector h_{t-1} and new input X_t into the sigmoid function and outputs a input gate vector i_t. Besides, a new candidate vector for updating the cell $\widetilde{C}_t$ is created by the tanh layer. The formulas are:

$$i_t = \sigma\left(\mathbf{W}_i x_t + \mathbf{U}_i h_{t-1} + b_i\right) \tag{3.4}$$

$$\widetilde{C}_t = \tanh\left(\mathbf{W}_c x_t + \mathbf{U}_c h_{t-1} + b_c\right) \tag{3.5}$$

where $\mathbf{W}_i, \mathbf{W}_c \in \mathbb{R}^{h\times d}$, $\boldsymbol{U}_i, \boldsymbol{U}_c \in \mathbb{R}^{h\times h}$, $b_i, b_c \in \mathbb{R}^h$ are the weight matrices and bias vectors.

The next step is to modify the former cell by combining the new information from the previous two parts. First, multiplying the old cell C_{t-1} by the forget gate vector f_t determines the old information passed through. Then, the new information added to the cell is determined by the input gate vector i_t multiplied by the candidate $\widetilde{C}_t$. Then, the new cell is formed by:

$$C_t = f_t \circ C_{t-1} + i_t \circ \widetilde{C}_t \tag{3.6}$$

where $\circ$ is the element-wise product.

The last part is to create the output for the current state. It contains an output gate with the output vector o_t and a tanh layer with the current cell's input C_t, which decides what information from the cell that the output should include. Multiplying each element in these vectors results in a new hidden vector h_t as a final output for this state:

$$o_t = \sigma\left(\mathbf{W}_o x_t + \mathbf{U}_o h_{t-1} + b_o\right) \tag{3.7}$$

$$h_t = o_t \circ \tanh\left(C_t\right) \tag{3.8}$$

The cell C_t and the hidden vector h_t are passed through the chain to the next step.

In this study, we increased the models' depth by implementing a stacked LSTM architecture with two hidden layers. We will discuss the details in the Implementation section.

3.2 Gated Recurrent Unit (GRU)

Gated Recurrent Unit or GRU is another variation of the RNN, which is introduced in 2014 [CVG14]. Like LSTM, the GRU is proposed to solve the RNN's shrinking gradient issue, and it also includes the sigmoid layer, the tanh layer, and the hidden state. However, the GRU does not depend on the cell to store memory.

Simpler than the LSTM, the GRU has two gates, update gate and reset gate, shown in Fig. 3.5. The reset gate is similar to the LSTM's forget gate, which decides how much past information is ignored. It takes the previous hidden state h_{t-1} and a new input X_t into the

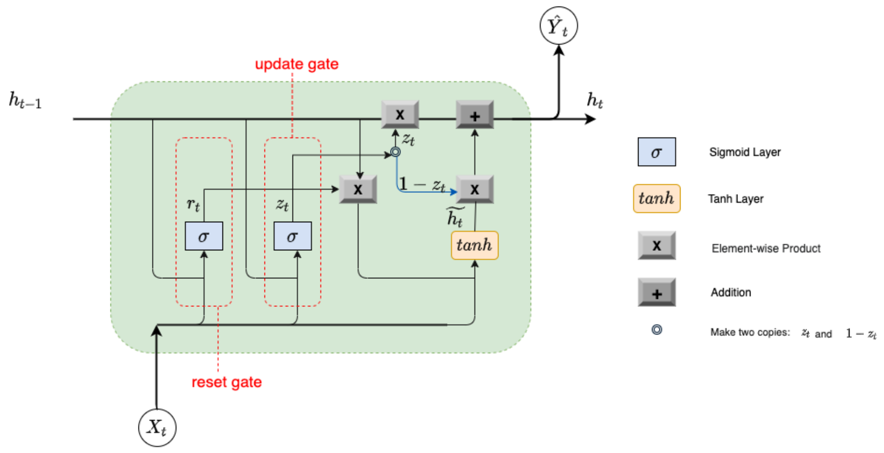

Figure 3.5: The GRU Unit

sigmoid function:

$$r_t = \sigma\left(\mathbf{W}_r x_t + \mathbf{U}_r h_{t-1} + b_r\right) \tag{3.9}$$

where $\mathbf{W}_r$ and $\mathbf{U}_r$ are weight matrices that learn through the training process, and $\mathbf{U}_r$ is a bias parameter. Similar to the LSTM's forget gate, if the output vector's values r_t is close to 0, little information from the previous hidden state is maintained, and the vector resets with the current input.

The update gate determines what past information is used in the new hidden state, which serves as the similar function as the cell in the LSTM [CGC14]. It is computed by

$$z_t = \sigma\left(\mathbf{W}_z x_t + \mathbf{U}_z h_{t-1} + b_z\right) \tag{3.10}$$

where the definition of $\mathbf{W}_z$, $\mathbf{U}_z$ and b_z are similar as the reset gate.

The next step is to create a new candidate $\widetilde{h}_t$ that store the memory content:

$$\widetilde{h}_t = tanh\left(\mathbf{W}_h x_t + \mathbf{U}_h\left(r_t \circ h_{t-1}\right) + b_h\right) \tag{3.11}$$

where $\circ$ denotes the element-wise product.

The final step is to update the hidden state. After the update gate, there are two copies of the update gate vector, z_l and $1 - z_l$. The former one is used with the previous hidden state h_{t-1}, while the later one is used with the new candidate vector $\widetilde{h}_t$. The purpose is that if the values of z_l is close to 1, we keep more past information than the new information. Combining all information, the current hidden state is

$$h_t = z_t \circ h_{t-1} + (1 - z_l) \circ \widetilde{h}_t. \tag{3.12}$$

Similar to the LSTM, we applied hierarchical learning capacity by adding a second GRU layer. Details will be discussed in a later section.

3.3 Hidden Markov Model (HMM)

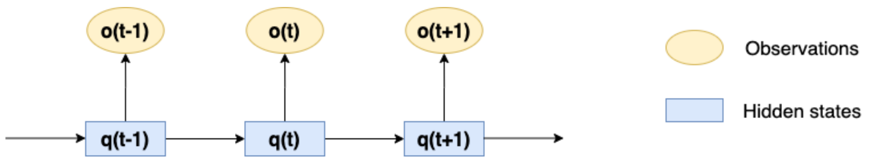

Figure 3.6: The Unfolded Markov Chain Over Time

Hidden Markov Model or HMM is a statistical model that is built on the Markov Chain, shown in Fig. 3.6 [JM19]. It relies on Markov process assumption and Observation Independence assumption. The Markov process assumption claims that the probability of a state is independent on other states given the previous state [SR17]:

$$P(q(t) \| q(1), ..., q(t-1)) = P(q(t) \| q(t-1)) \tag{3.13}$$

where $q(t)$ is the hidden state at time t that belongs to the hidden states set $Q = \{q_1, q_2, ..., q_N\}$. The Observation Independence assumption believes that the observation $o(t)$ at time t depends only on the the current hidden state $q(t)$ [SR17]:

$$P(o(t) \| q(1), ..., q(t), o(1), ..., o(t-1)) = P(o(t) \| q(t)) \tag{3.14}$$

where $o(t)$ belongs to an observation in a sequence of L observations $O = \{o_1, o_2, ..., o_L\}$.

An HMM can be represented by

$$\lambda = (A, B) \tag{3.15}$$

where $A = (a_{ij})_{N\times N}$ represents the state transition matrix and $B = b_j(o_t)$ represents the emission probabilities. The probabilities a_{ij} and $b_j(o_t)$ are defined by:

$$a_{ij} = P(Q = j \| Q = i), s.t. \sum_{j=1}^{N} a_{ij} = 1 \tag{3.16}$$

$$b_j(o_t) = P(o_t \| Q = j). \tag{3.17}$$

The emission probabilities are modelled as multivariate Gaussian distribution or Gaussian mixture models. Both parameters are trained by the Baum-Welch algorithm. The HMM also requires an initial state distribution $\pi = \{\pi_1, ..., \pi_N\}$ such that $\pi_j = P(q(1) = j)$ and $\sum_{j=1}^{N} \pi_j = 1$. Then, Maximum a Posteriori (MAP) approach is used to forecast the next observation based on the previous d observations [GD12]:

$$\hat{o}_{d+1} = argmax_{o_{d+1}} P(o_{d+1} \| o_1, ..., o_d, \lambda) \tag{3.18}$$

$$= argmax_{o_{d+1}} \frac{P(o_1, ..., o_d, o_{d+1} \| \lambda)}{P(o_1, ..., o_d, \lambda)} \tag{3.19}$$

$$= argmax_{o_{d+1}} P(o_1, ..., o_d, o_{d+1} \| \lambda). \tag{3.20}$$

The probability $P(o_1, ..., o_d, o_{d+1})$ is the maximum value over the discrete set of potential o_{d+1}, shown in next section (Table 3.2).

3.4 Implementation

3.4.1 LSTM

We implemented the stacked LSTM model in the Keras Python library. The first layer takes the input in the shape of [batch size, time steps, number of features], followed by a second LSTM layer and a fully connected output layer with a tanh function. Between two LSTM layers, there is a dropout layer, which randomly sets 20% of the input to 0, to slow down the learning rate and prevent overfitting. The model is trained by fit_generator() function with the reshaped training set discussed in the Section 2.3 as an input, and it is tested by predict_generator() function with the reshaped testing set as an input.

Before training the final LSTM model, it is necessary to find the optimal set of hyperparameters, including epoch size, the number of neurons, batch size, and optimizer. The training set is separated into four 15-month subset datasets, as shown in Table 3.1. They include different months across a year to eliminate possible seasonal influence. Then, the subset datasets are also split into a 12-month training set and a 3-month testing set. Because of the randomness in the LSTM network, each experimental scenario with candidate hyperparameters is repeated five times for every subset dataset. The optimal set of hyperparameters is the combination that always has the small Mean Absolute Percentage Error, which is defined in equation 4.2, of all subset testing sets.

Table 3.1: Subset datasets for Tuning Hyperparameters

Datasets	**Train**		**Test**	
	Start	End	Start	End
1	Jan 1, 2014	Dec 31, 2014	Jan 1, 2015	Mar 31, 2015
2	Apr 1, 2015	Mar 31, 2016	Apr 1, 2016	Jun 30, 2016
3	Jul 1, 2016	Jun 30, 2017	Jul 1, 2017	Sept 31, 2017
4	Oct 1, 2017	Sept 31, 2018	Oct 1, 2018	Dec 31, 2018

We trained the final model using the best tuning hyperparameters. Then, the model forecasted the return percentage based on the testing set. We de-standardized the predictions into prices corresponding to the inverse of the equation 2.4:

$$p_i = p_0 * (\hat{p}_i + 1) . \tag{3.21}$$

where $\hat{p}_i$ is the prediction from the model, and p_0 is the first row in the x_window. Since some randomness exists in the LSTM network, we run the whole process 30 times and store the de-standardized prediction in one data frame.

3.4.2 GRU

The GRU model has an identical structure as the LSTM model, which is implemented using the GRU class from the Keras library. Similar to the LSTM model, a dropout layer is used in the GRU model to prevent overfitting. We tuned the same hyperparameters using four subset datasets described in Table 3.1. Moreover, the final model with the optimal hyperparameters is repeated 30 times, and the prediction is stored in a data frame after de-standardizing using the equation 3.21.

3.4.3 HMM

In this study, the observation is:

$$o_t = (FC, FH, FL) = \left(\frac{close - open}{open}, \frac{high - open}{open}, \frac{open - low}{open} \right). \tag{3.22}$$

Following the previous two methods, the HMM model forecasts the closing price based on the previous 20-time step information. We tried two emission probabilities in the HMM model. The first one is a single multivariate Gaussian distribution with mean μ_j and covariance matrix Σ_j [SR17]:

$$b_j(o_t) = N(o_t; \mu_j, \Sigma_j) \tag{3.23}$$

We implemented it by using the GaussianHMM function from the hmmlearn package [Leb19]. The number of hidden states (n) is chosen by the number that gives the minimum error. The second emission probability is M-Component Gaussian mixture models:

$$b_j(o_t) = \sum_{m=1}^{M} c_{jm} N(o_t; \mu_{jm}, \Sigma_{jm}) \tag{3.24}$$

where c_{jm} is the weight, μ_{jm} is the mean, and Σ_{jm} is the covariance of the m^{th} mixture component in state j. The number of hidden states (n) and the number of mixture components for each state (M) are determined by the combination that has a small error. We implemented this emission probability using the GMMHMM function from the same package [Leb19].

We computed the probabilities of a candidate set of potential values to find the MAP estimate $\hat{o}_{d+1}$ in the HMM model. Table 3.2 includes information about the range of values and the number of points. As discussed in section 2.2, this range covers most of the fractional change in the dataset, with a few negligible outsiders.

Table 3.2: Range of Potential Values of MAP Estimate

dataset	Min	Max	Number of Points
FC	-0.1	0.1	60
FH	0	0.1	10
FL	0	0.1	10

CHAPTER 4

Results

The metrics used to evaluate the performance of the models are Root Mean Squared Error (RMSE) and Mean Absolute Percentage Error (MAPE). The RMSE measures the squared root of the squared errors between the predicted price p_i and the actual price a_i:

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(p_i - a_i)^2}. \tag{4.1}$$

The MAPE measures the absolute difference between the predicted price p_i and the actual price a_i in percentage:

$$MAPE = \frac{1}{n}\sum_{i=1}^{n}\frac{|p_i - a_i|}{|a_i|}. \tag{4.2}$$

A smaller RMSE and MAPE indicate a better result that the prediction is close to the actual price. Besides evaluating the performance of three methods, the MAPE is also used to decide the tuning hyperparameters.

In the following sections, we will discuss the tuning results, predicted results from each method separately, and then compare the predictions in the last section.

4.1 LSTM

To find the optimal hyperparameters of the LSTM model, we ran each experimental scenario five times and recorded the RMSE results. We summarized the results in box plots for epoch size, the number of neurons, batch size, and optimizer, shown in Appendix A, respectively.

Every figure includes four box plots of the RMSE values, one for each subset dataset. The optimal set is defined as the hyperparameters that always have small MAPE values across all subset datasets.

After evaluating the results from the Figs. A.1-A.4, the final LSTM configured with 32 neurons, a batch size of 10, and trained for 500 epochs. The best optimization algorithm is the Nesterov-accelerated adaptive moment estimation (Nadam), which is Adam with Nesterov momentum [Cho15].

The final model's evaluation is based on 30 repetitions due to the randomness during the model training. The average prediction, which is the average predicted prices of 30 reruns, is shown in Fig. C.1. For better visualization, we used a log scale. The overall performance of the LSTM model is as good as expected, while the prediction is more conservative and less extreme than the actual price. Specifically, the predicted price is higher than the actual minimum price in mid-May 2019 and mid-March 2020, while lower than the actual maximum price in late June and late October 2019.

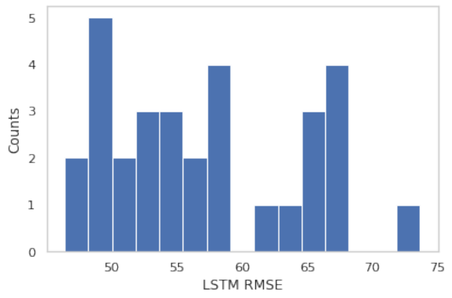

Figure 4.1: LSTM RMSE Results

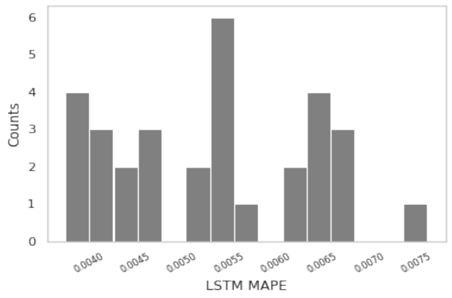

Figure 4.2: LSTM MAPE Results

The LSTM model has an almost identical performance evaluated by RMSE and MAPE. It has average RMSE value at 57 and average MAPE value at 0.0053 or 0.53%, shown in Table 4.1. For both evaluations, the medians are slightly lower than the means due to the maximum RMSE value 73.65, shown in Fig. 4.1, and the maximum MAPE value 0.0076 or 0.76%, shown in Fig. 4.2, that are much higher than other values. The overall performance

Table 4.1: LSTM Testing Results

	RMSE	MAPE
Mean	57.2310	0.0053
Standard Deviation	7.4317	0.0011
Min	46.3745	0.0037
50% (Median)	57.0664	0.0053
Max	73.6512	0.0076

indicates a distinguishable difference within repetitions' results, and the predictions from some repetitions are less accurate than others, resulting in higher RMSE and MAPE values.

4.2 GRU

Like the LSTM model, we tuned the hyperparameters in the GRU model following the earlier methods. According to the tuning results summarized in Appendix B, the final GRU model configured with 32 neurons, a batch size of 32, and trained for 500 epochs. The best optimization algorithm is RMSprop with a gradient norm clipping of 1.0 to avoid exploding gradient problem, meaning that the L2 vector norm for a gradient is limited to a threshold of 1.0 [Bro19].

The results of the GRU model repetitions are summarized in Table 4.2, Fig. 4.3 and Fig. 4.4. The average prediction of GRU, shown in Fig. C.2, has comparable performance as the LSTM model. The mean RMSE value is 44.77, which is higher than the median 43.64. Furthermore, the mean MAPE value is 0.0035 or 0.35%, which is higher than the median 0.0033 or 0.33%. This result indicates that the GRU model is likely to have small errors, while the prediction is sometimes undesirable.

The RMSE values and the MAPE values are associated with each other. There are two potential outliers in RMSE results (Fig. 4.3) and one potential outlier in MAPE results

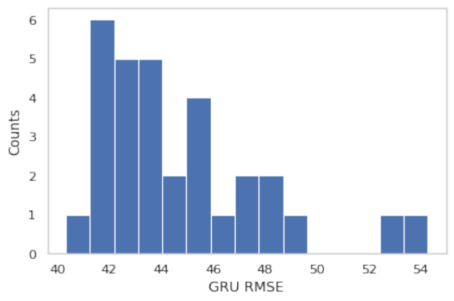

Figure 4.3: GRU RMSE Results

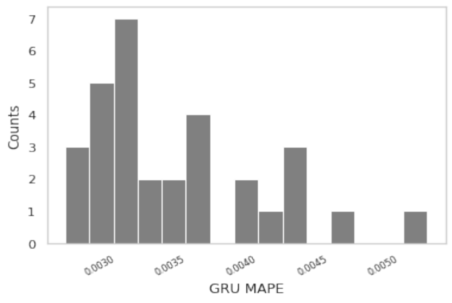

Figure 4.4: GRU MAPE Results

(Fig. 4.4). The outlier prediction in MAPE has a value of 0.0053, and it is also the outlier measured by RMSE with a value of 54.27. Another outlier RMSE value is 52.50, while its corresponding MAPE value is 0.0046, which is considered the maximum of MAPE results.

Table 4.2: GRU Testing Results

	RMSE	MAPE
mean	44.7693	0.0035
std	3.2983	0.0006
min	40.3035	0.0028
50%	43.6422	0.0033
max	54.2775	0.0053

4.3 HMM

There are two choices of emission probabilities for the HMM model. Before implementing the single multivariate Gaussian distribution (GaussianHMM), we determined the number of hidden states (n). The previous work [GD12] suggested to use four hidden states as the dimension of the dataset is 4. By tuning on the number of hidden states based on the subset datasets in Table 3.1, $n = 4$ has a good performance. Another emission probabilities are the

M-Component Gaussian mixture models (GMMHMM). By tuning on the number of mixture components for each state, the combination of $n = 4$ and $M = 5$ has a good performance.

The HMM model with two different emission probabilities has similar predictions. The difference between GaussianHMM and GMMHMM is imperceptible from the Fig. C.3 and Fig. C.4. They have good performance even in the fluctuation period, such as the remarkable drop in mid-March 2020. The RMSE values and the MAPE values in Table 4.3 indicate that the prediction from GMMHMM is slightly better than that of GaussianHMM, though the contrast is negligible.

Table 4.3: HMM Testing Results

Methods	RMSE	MAPE
GaussianHMM	37.8054	0.002814
GMMHMM	37.4856	0.002803

4.4 Model Comparison

Back to this study's goal, we want to find the best model to predict the Bitcoin price. First of all, it is necessary to compare the repetitions results from two variations of the traditional RNN. Table 4.4 summarizes the distribution of the RMSE and the MAPE results from the LSTM repetitions and the GRU repetitions, which is also visualized through box plots in Figs. 4.5-4.6. It is clear from the box plots that the GRU outperforms the LSTM evaluated by both RMSE and MAPE. The GRU model has a smaller mean and a smaller median of errors, and its prediction is less varied than the LSTM model. However, the GRU model might lead to a more extreme prediction.

We measured the performance of four models by comparing the details of prediction errors. Due to the randomness, the LSTM model and the GRU model are evaluated by the average predictions. Fig. 4.7 plots the average forecast errors from four methods compared

Table 4.4: Comparing LSTM and GRU Reruns Results

Statistics	**RMSE**		**MAPE**	
	LSTM	GRU	LSTM	GRU
mean	57.2310	44.7693	0.0053	0.0035
std	7.4317	3.2983	0.0011	0.0006
min	46.3745	40.3035	0.0037	0.0028
50%	57.0664	43.6422	0.0053	0.0033
max	73.6512	54.2775	0.0076	0.0053

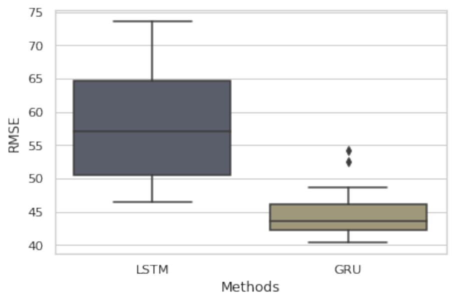

Figure 4.5: Box Plots of LSTM and GRU RMSE Results

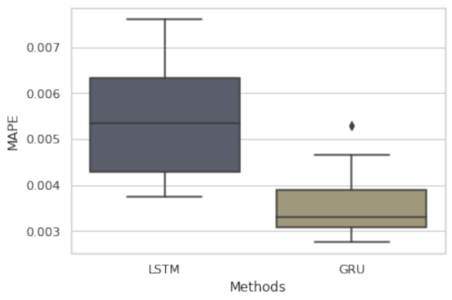

Figure 4.6: Box Plots of LSTM and GRU MAPE Results

to the actual Bitcoin price, which shows that the overall trend of absolute errors from all Bitcoin forecast methods is similar and indicates an association between price movement and the errors. When the price remains relatively constant or changes steadily, all models have good predictions that the absolute errors are close to 0. During the fluctuation period, the price is difficult to be predicted as accurate as the normal period such that the absolute errors tend to be large for all models.

Interestingly, all models have a notable prediction error on June 26, 2019, at 20:30 UTC, corresponding to the most remarkable Bitcoin price declining in history, as mentioned in Section 2.2. At the beginning of that day, the price gradually increased from 11751 USD to

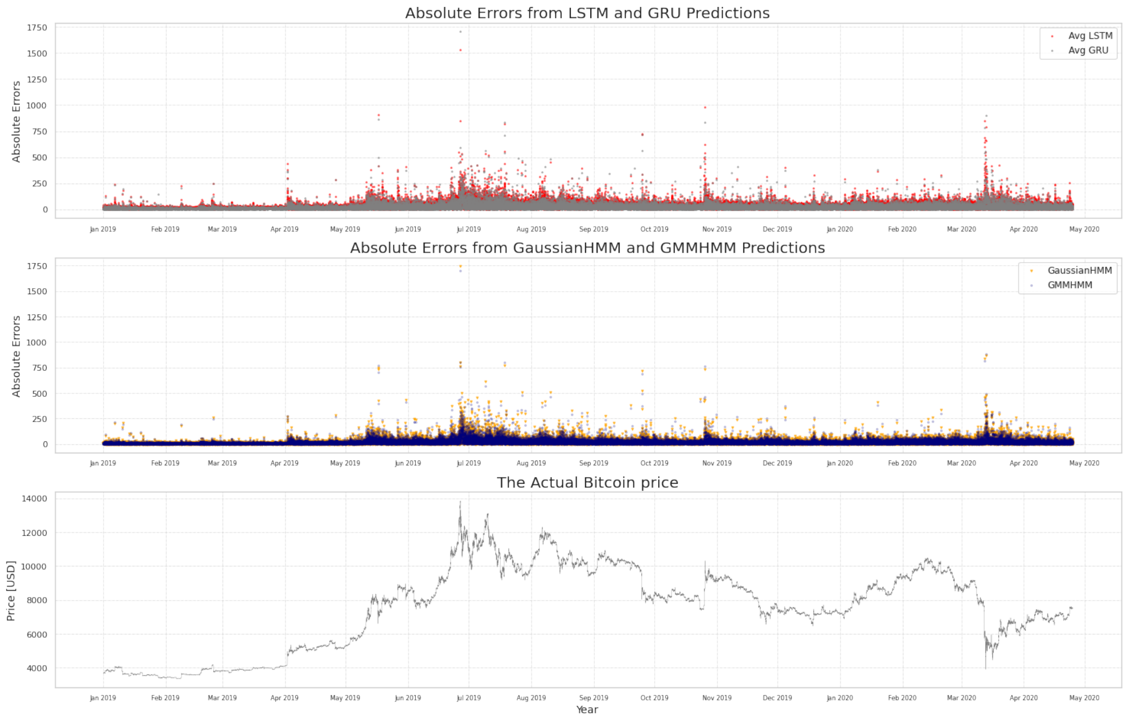

Figure 4.7: Absolute Errors

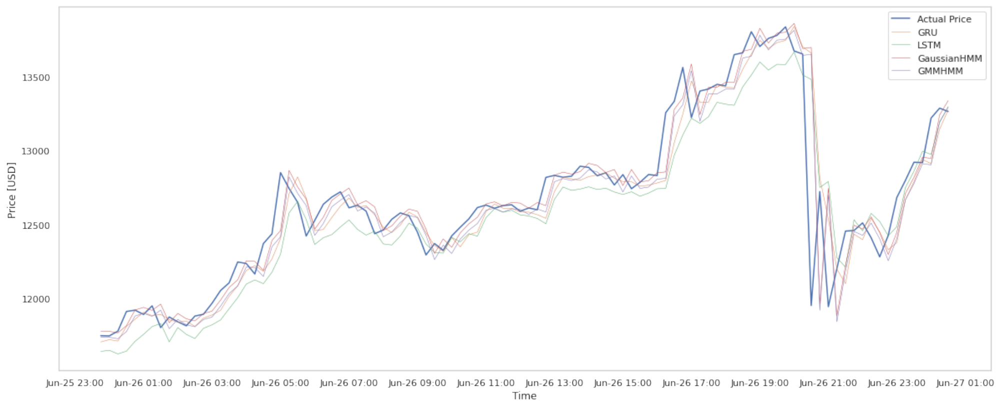

Figure 4.8: The Actual Price and The Predictions on June 26, 2019.

more than 13837.96 USD. Then, it suddenly dropped to 11954.34 USD in 15 minutes, shown in Fig. 4.8. Since the models are based on the previous 300-minute information, predicting this abnormal behavior is difficult, and thus resulting in the maximum errors for all models, shown in Table 4.5.

Table 4.5: The Distribution of Absolute Prediction Errors

	LSTM	GRU	GMMHMM	GaussianHMM
Mean	39.2422	25.8095	21.6315	21.8164
Standard Deviation	38.4952	34.3891	30.6148	30.8758
Min	0.0086	0.0025	0.0003	0.0013
25%	17.4294	8.1889	6.6154	6.4539
50%	30.6605	16.7220	13.6135	13.9382
75%	49.5708	31.9816	26.7994	27.3862
95%	97.2273	76.3918	64.0733	63.4393
97.5%	128.4093	104.0899	88.6742	87.1146
Max	1528.5816	1706.9937	1698.2208	1744.5792

Table 4.5 shows the distribution of the prediction errors, which indicates the Hidden Markov Model has the best performance among all methods. There is a subtle difference between GMMHMM and GsussianHMM, though GMMHMM has a slightly better prediction. The GMMHMM has the RMSE value 37.4856 and the MAPE value 0.0028 or 0.28% (Table 4.3) which are even smaller than the minimum RMSE values and MAPE values of all LSTM and GRU predictions (Table 4.4). The overall errors of the GMMHMM model are small, though the extreme case exists, such as the dramatic price declining on June 26, 2019.

CHAPTER 5

Conclusion

Since Bitcoin and the Blockchain technology were introduced in 2008, it has taken a predominant place in the cryptocurrency field. There are millions of users around the world, especially in the United States. Predicting the price of cryptocurrencies has been a popular topic, from which we can take advantage of the arbitrage for an investment.

In this study, we used a 15-minute Bitcoin price from Bitstamp Exchange and took 20 time steps, in total 300-minute information, to forecast the closing price of the next time step. Three methods were implemented: Long Short-term Memory Model, Gated Recurrent Units Model, and Hidden Markov Model with two different emission probabilities.

We conclude that the Hidden Markov Model with the Gaussian Mixture Models has the best performance in price forecast among all methods, evaluated by the RMSE function and the MAPE function. The Hidden Markov Model with single multivariate Gaussian distribution has comparable performance with the best model. The Gated Recurrent Units model's prediction is more precise than that from the Long Short-term Memory model, although the former might have a more extreme error. Additionally, there is an association between price movement and model accuracy. The prediction is more precise during the normal period than the fluctuation period.

One major challenge we are currently facing is the model efficiency, especially the HMM. It can take up to a day to obtain the results. Therefore, we only include a few features in the dataset at the current step and assume it provides sufficient information, but in reality, Bitcoin might be more complex and correlated with other markets. As future work, we can search for an alternative algorithm or package that is more efficient, and include additional information in the current dataset, such as the market index, to improve the performance.

APPENDIX A

LSTM Hyperparameters Tuning Results

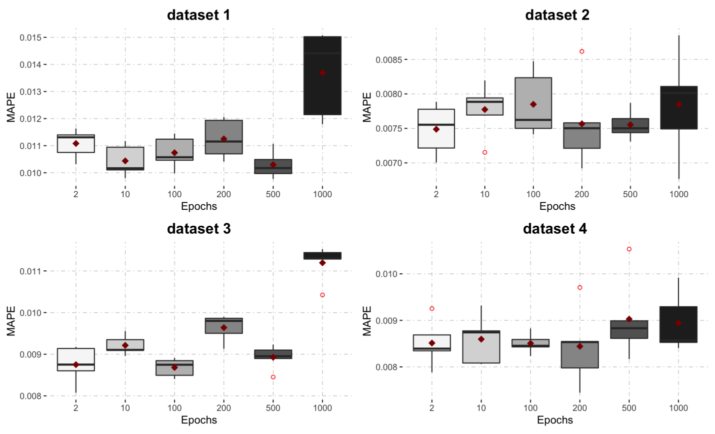

Figure A.1: Box Plot Summarizing LSTM's Epoch MAPE Results

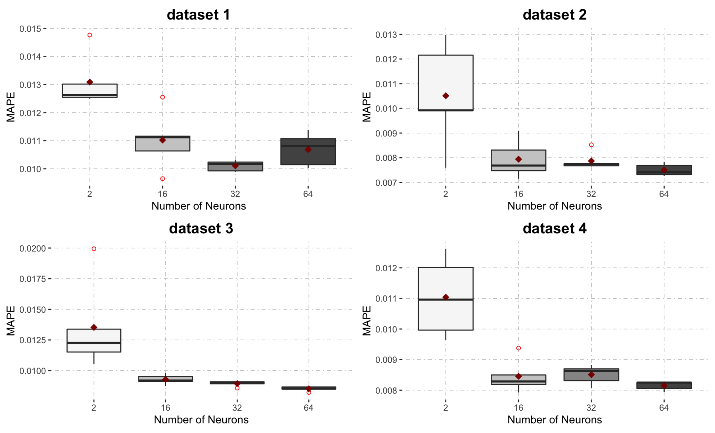

Figure A.2: Box Plot Summarizing LSTM's Neurons MAPE Results

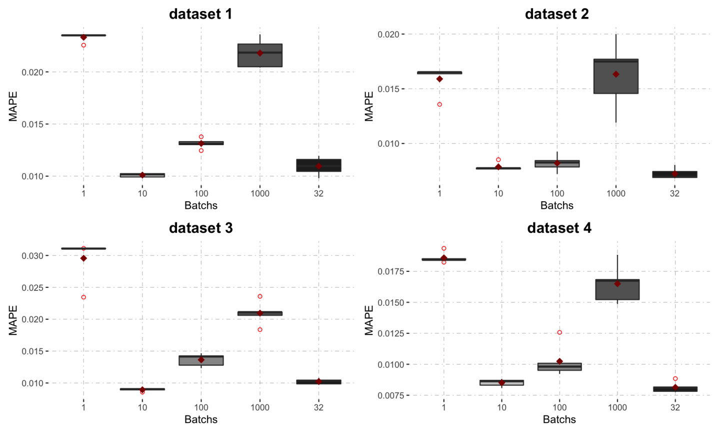

Figure A.3: Box Plot Summarizing LSTM's Batchs MAPE Results

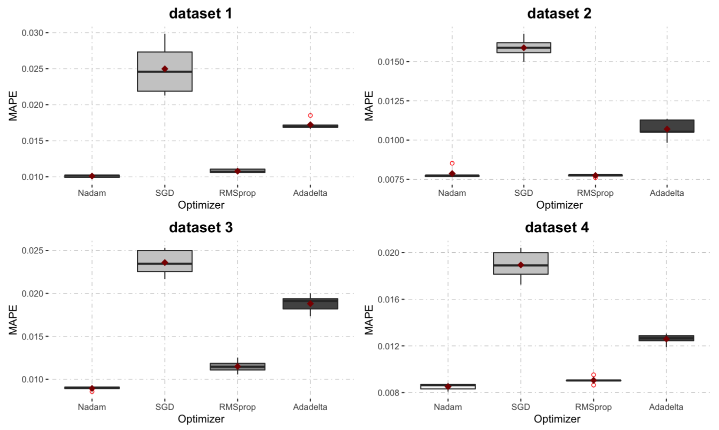

Figure A.4: Box Plot Summarizing LSTM's Optimizer MAPE Results

APPENDIX B

GRU Hyperparameters Tuning Results

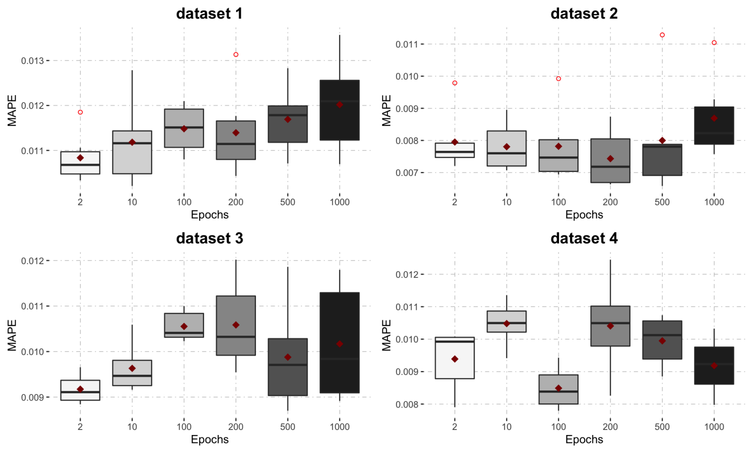

Figure B.1: Box Plot Summarizing GRU's Epoch MAPE Results

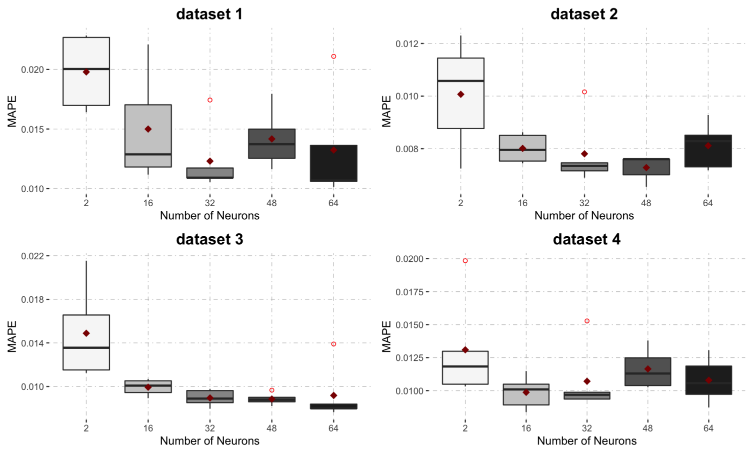

Figure B.2: Box Plot Summarizing GRU's Neurons MAPE Results

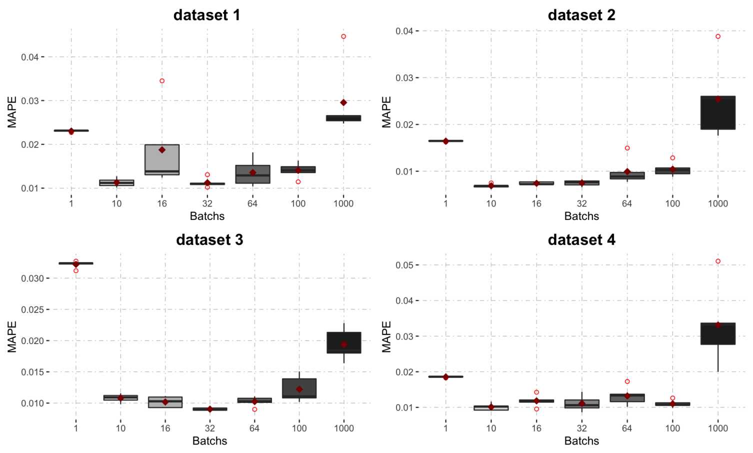

Figure B.3: Box Plot Summarizing GRU's Batchs MAPE Results

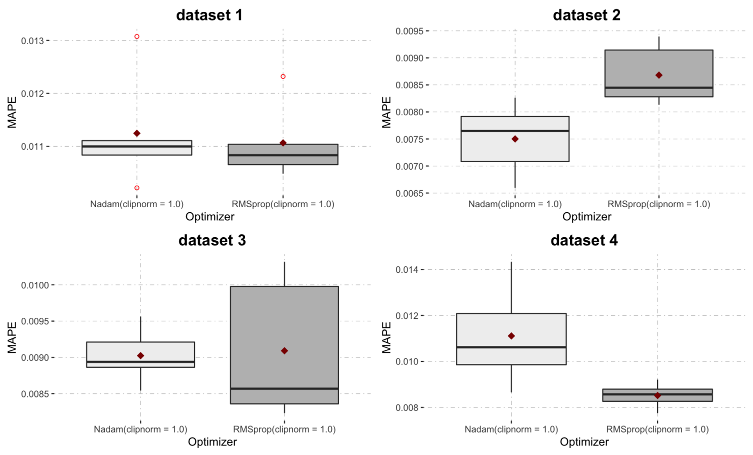

Figure B.4: Box Plot Summarizing GRU's Optimizer MAPE Results

APPENDIX C

Predictions and Errors

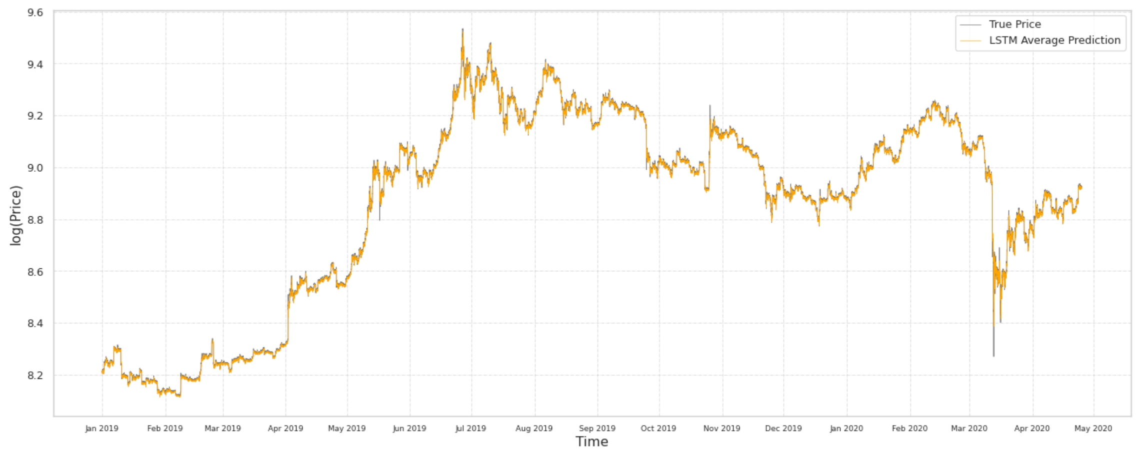

Figure C.1: LSTM Predictions

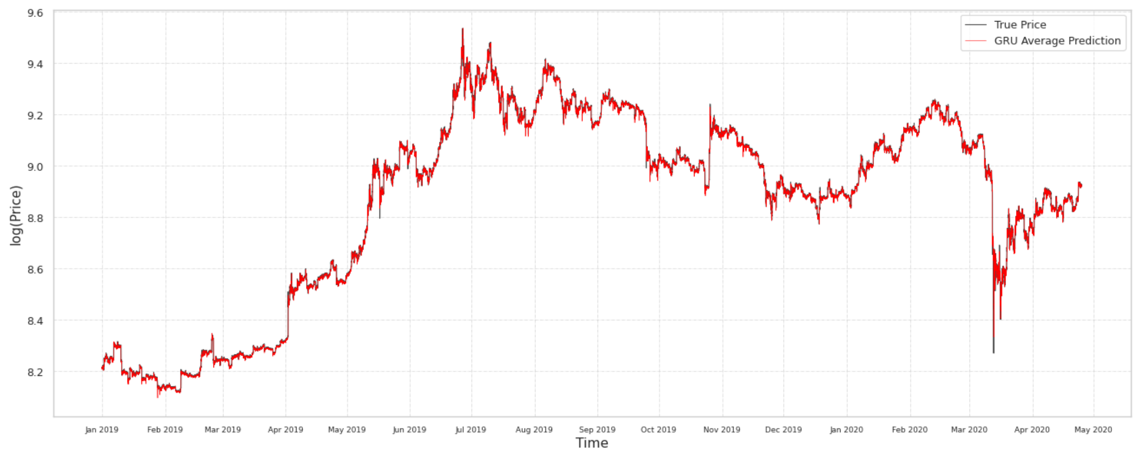

Figure C.2: GRU Predictions

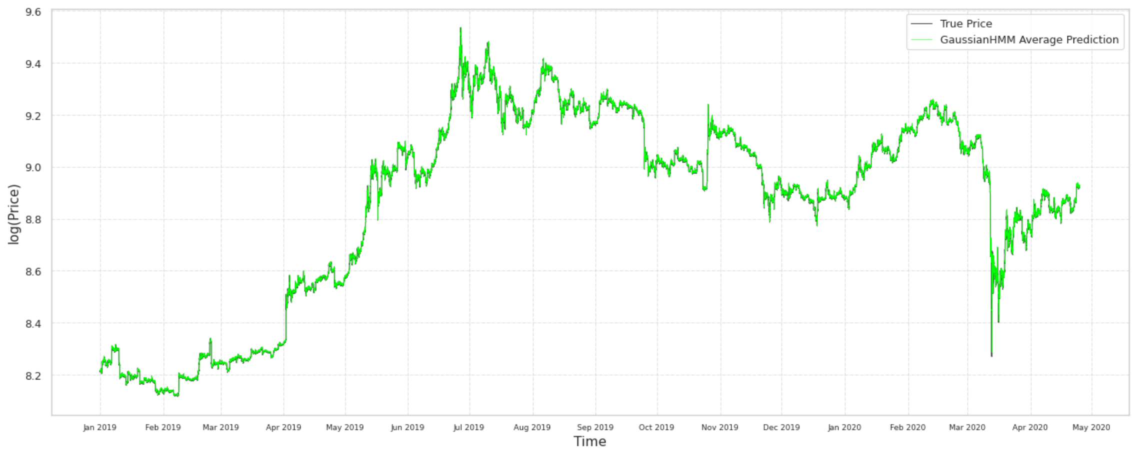

Figure C.3: GaussianHMM Predictions

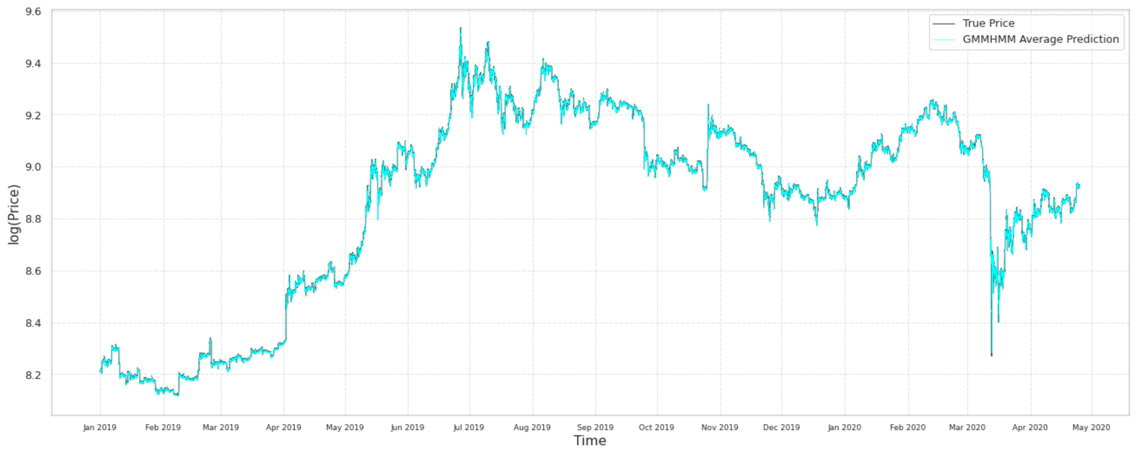

Figure C.4: GMMHMM Predictions

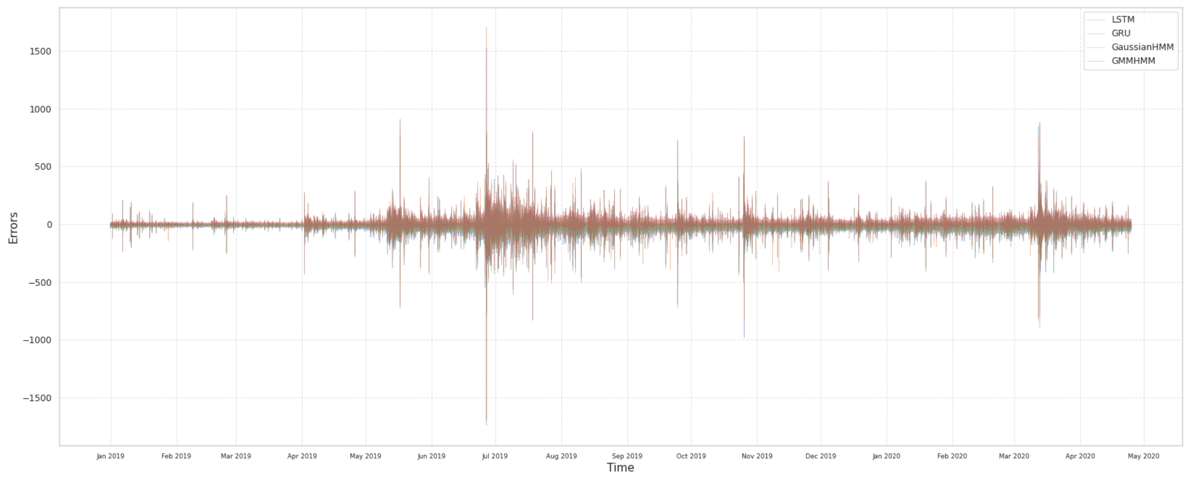

Figure C.5: Prediction Errors

REFERENCES

[AT19] AT&T. “AT&T Now Accepts BitPay.” `https://about.att.com/story/2019/att_bitpay.html`, 2019. [Online; accessed 15-July-2020].

[Aun17] Jakob Aungiers. “Multidimensional LSTM Networks to Predict Bitcoin Price.” `https://www.jakob-aungiers.com/articles/a/Multidimensional-LSTM-Networks-to-Predict-Bitcoin-Price`, 2017. [Online; accessed 10-July-2020].

[Bag20] Rick Bagshaw. “Top 10 cryptocurrencies by market capitalisation.” `https://finance.yahoo.com/news/top-10-cryptocurrencies-market-capitalisation-160046487.html`, Apr 2020. [Online; accessed 15-July-2020].

[Bit20a] Bitcoincharts. “Exchange volume distribution.” `https://bitcoincharts.com/charts/volumepie/`, 2020. [Online; accessed 9-August-2020].

[Bit20b] Bitcoincharts. “Pricechart.”, 2020. data retrieved from Bitcoincharts, `https://bitcoincharts.com/`.

[Bit20c] Bitcoin.org. “”Bitcoin Resources”.” `https://bitcoin.org/en/resources`, 2020. [Online; accessed 2-November-2020].

[Bitte] Bitstamp. “What is Bitstamp?” `https://www.bitstamp.net/faq/what-is-bitstamp/`, no date. [Online; accessed 30-July-2020].

[Bro19] Jason Brownlee. “How to Avoid Exploding Gradients With Gradient Clipping.” `https://machinelearningmastery.com/how-to-avoid-exploding-gradients-in-neural-networks-with-gradient-clipping/`, 2019. [Online; accessed 10-June-2020].

[CGC14] Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. “Empirical evaluation of gated recurrent neural networks on sequence modeling.” *arXiv preprint arXiv:1412.3555*, 2014.

[Cho15] François Chollet et al. “Keras.” `https://keras.io`, 2015.

[Coi20a] CoinMap. “All the cryptocurrency merchants and ATMs of the world in one map.” `Coinmap.org`, 2020. [Online; accessed 15-July-2020].

[Coi20b] CoinMarketCap. “Top 100 Cryptocurrencies by Market Capitalization.” `https://coinmarketcap.com/`, 2020. [Online; accessed 16-July-2020].

[Coi20c] CoinMarketCap. “Top Cryptocurrency Spot Exchanges.” `https://coinmarketcap.com/rankings/exchanges/`, 2020. [Online; accessed 23-July-2020].

[Col13] Andrew Collette. *Python and HDF5: unlocking scientific data.* " O'Reilly Media, Inc.", 2013.

[con20] Wikipedia contributors. "History of bitcoin — Wikipedia, The Free Encyclopedia." `https://en.wikipedia.org/w/index.php?title=History_of_bitcoin&oldid=965407737`, 2020. [Online; accessed 15-July-2020].

[CVG14] Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. "Learning phrase representations using RNN encoder-decoder for statistical machine translation." *arXiv preprint arXiv:1406.1078*, 2014.

[Fou20] Wikimedia Foundation. "Other ways to give." `https://donate.wikimedia.org/wiki/Ways_to_Give#Cryptocurrency`, 2020. [Online; accessed 15-July-2020].

[GD12] Aditya Gupta and Bhuwan Dhingra. "Stock market prediction using Hidden Markov Models." In *2012 Students Conference on Engineering and Systems*, pp. 1–4, 2012.

[HS97] Sepp Hochreiter and Jürgen Schmidhuber. "Long Short-Term Memory." *Neural Computation*, **9**(8):1735–1780, 1997.

[Jay20] Jay. "Market Capitalization (Cryptoasset, Aggregate)." `https://support.coinmarketcap.com/hc/en-us/articles/360043836811-Market-Capitalization-Cryptoasset-Aggregate-`, 2020. [Online; accessed 30-July-2020].

[JM19] Dan Jurafsky and James H. Martin. "Speech and Language Processing." Third Edition draft, 2019.

[Leb19] Sergei Lebedev. "hmmlearn (version 0.2.3)." `https://github.com/hmmlearn/hmmlearn`, December 2019.

[Nak19] Satoshi Nakamoto. "Bitcoin: A peer-to-peer electronic cash system." Technical report, Manubot, 2019.

[Ng18] Vicky Ng. "Why haven't we all bought cryptocurrency yet?" `https://www.finder.com/why-people-arent-buying-cryptocurrency`, 2018. [Online; accessed 14-July-2020].

[Ola15] Christopher Olah. "Understanding LSTM Networks." `https://colah.github.io/posts/2015-08-Understanding-LSTMs/`, 2015. [Online; accessed 10-August-2020].

[Phi18] Michael Phi. "Illustrated Guide to LSTM's and GRU's: A step by step explanation." `https://towardsdatascience.com/illustrated-guide-to-lstms-and-gru-s-a-step-by-step-explanation-44e9eb85bf21`, 2018. [Online; accessed 10-August-2020].

[Rad20] Coin ATM Radar. "Bitcoin ATM Map." `https://coinatmradar.com`, 2020. [Online; accessed 15-July-2020].

[RBK18] Michel Rauchs, Apolline Blandin, Kristina Klein, Gina C Pieters, Martino Recanatini, and Bryan Zheng Zhang. "2nd global cryptoasset benchmarking study." *Available at SSRN 3306125*, 2018.

[Smi14] Aaron Smith. "Microsoft begins accepting Bitcoin." `https://money.cnn.com/2014/12/11/technology/microsoft-bitcoin/`, 2014. [Online; accessed 15-July-2020].

[SR17] Hiroshi Shimodaira and Steve Renals. "Hidden Markov Models and Gaussian Mixture Models." `http://www.inf.ed.ac.uk/teaching/courses/asr/2013-14/asr03-hmmgmm.pdf`, 2017. [Online; accessed 10-August-2020].

[SW97] Shanming Shi and A. S. Weigend. "Taking time seriously: hidden Markov experts applied to financial engineering." In *Proceedings of the IEEE/IAFE 1997 Computational Intelligence for Financial Engineering (CIFEr)*, pp. 244–252, 1997.

[Whi15] Zack Whittaker. "Bitstamp exchange hacked, $5M worth of bitcoin stolen." `https://www.zdnet.com/article/bitstamp-bitcoin-exchange-suspended-amid-hack-concerns-heres-what-we-know/`, 2015. [Online; accessed 31-August-2020].

[Zam20] Sergio Zammit. "How Many Cryptocurrency Exchanges are there?" `https://www.cryptimi.com/guides/how-many-cryptocurrency-exchanges-are-there`, July 2020. [Online; accessed 23-July-2020].

[Zha04] Yingjian Zhang. *Prediction of financial time series with Hidden Markov Models.* PhD thesis, Applied Sciences: School of Computing Science, 2004.