Multi-factor Time-series Predication of Bitcoin Price

Yafeng GUO
Illinois Institute of Technology
10 W 35th St, Chicago, IL 60616
yguo82@hawk.iit.edu

Abstract

Bitcoin is a cryptocurrency invented in 2008 by an unknown person or group of people using the name Satoshi Nakamoto [12]. It is now the most popular form of cryptocurrency in the world, and is now being included in institutional investment portfolios. How to predicate bitcoin price precisly is an interesting and valuable question. This article explored several critical drivers behind bitcoin price and also validated different predication approaches, like ARIMA and RNN.

1. Introduction

Bitcoin is a cryptocurrency invented in 2008 by an unknown person or group of people using the name Satoshi Nakamoto. The currency began use in 2009 when its implementation was released as open-source software. Bitcoin is a decentralized digital currency, without a central bank or single administrator, that can be sent from user to user on the peer-to-peer bitcoin network without the need for intermediaries. Transactions are verified by network nodes through cryptography and recorded in a public distributed ledger called a blockchain. Bitcoins are created as a reward for a process known as mining. [12] [5].

Bitcoin price has grown more than 5 times since beginning of 2020 (from 9367.4 USD to a new high 61,468.0 USD in Mar 2021). It attracted more and more attentions in the world. However, not all people have consensus on its value. The ex-president of US Donald Trump said digital currencies are "not money" and the value of bitcoin and cryptocurrencies are based on "thin air". Whereas Elon Musk said he is "a Bitcoin Supporter" and accept "buy a Tesla with bitcoin". Those conflicted statements somehow can explain the price volatility of bitcoin, which bring challanges to bitcoin price predication.

Since outbreak of Covid-19 pandemic, the bitcoin price surged (Figure 1). A research conducted in [11] reflected that the portion of "investor-held Bitcoin" raised to 77% since 2017. The institutional investors treat bitcoin not so much as a currency, but as a hedge against macroeconomic uncertainty. According to analysis in [8], the trend of considering bitcoin as a hedge in the long run emerged already in 2017. Bitcoin has become a new type of safe haven asset. From this point of view, prices of other assets categories should have correlations with Bitcoin price. In this article, gold price, SP500 Index, Shanghai Stock Market Index, major currencies exchange rate are included as factors of macroeconomic.

Personal investors contributed major transactions in terms of transaction numbers. Personal investors shared the drivers behind institutional investors. Besides that, the public media impact has to be evaluated as well. In this article, google trend are included to represent public media impact. Baidu Index is also included as Google service is not available in China.

1.1. Research Questions

In short, the research question of this article is: **With what accuracy level can the next day price of Bitcoin be predicted using which machine learning method?**

1.2. Evaluation Method

The question can be defined as a regression question, e.g. what is the exact price of next day (I will use close price of every day in China Standard Time) . In this case the root mean squared error (RMSE) can be used as measurement. Besides the exact price, investors may also have intersting to know if the predicated trend is correct, e.g. it will go up, go down, or be flat tomorrow. The revert of growing trend or the reverting of decreasing trend are even more critical. We will use both RMSE and trend predication accuracy (TPA in short) to evaluate our models.

The formula of RMSE is defined as below:

$$RMSE = \sqrt{\frac{\sum_{i=1}^{n}(Y_i - \hat{Y}_i)^2}{n}}$$

Here n is the number of samples, and Y_i is the true value in test set, and $\hat{Y}_i$ is the predicated value.

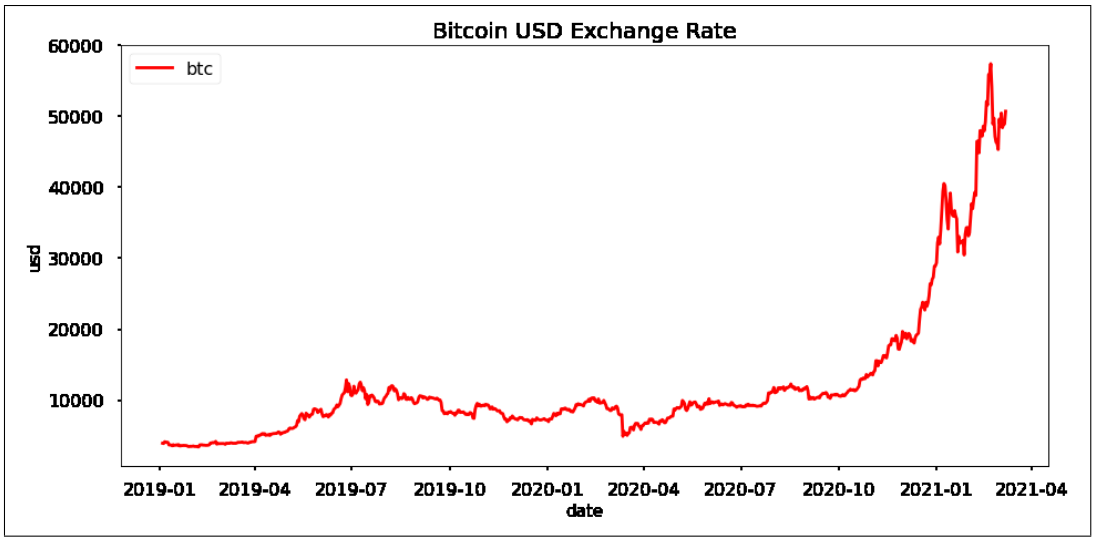

Figure 1. Bitcoin USD exchang rate raised more than 6 times, since Jan 2019 to Mar 2021.

The formula of TPA is defined as below:

$$TPA = \frac{correct\ predication\ of\ peaks\ or\ troughs}{number\ of\ peaks\ +\ number\ of\ troughs}$$

2. Related Work

Although bitcoin has more than 10 years history, many pepole only become familiar with that since 2018 after bitcoin exceeded 10,000 USD. And most articles I found on bitcoin price predication were around 2018. As machine learning got very fast development in those years, leveraging latest machine learning technology to do such research will be interesting.

In [9], Obryan Poyser explored determinants of bitcoin price. Obryan separated bitcoin price drivers to internal drivers and external drivers. For internal drivers, it is mainly about transaction volume, hash rate etc. Obryan actually spend a considerable research effort on external factors. Obryan found bitcoin price has correlation with gold price, stock market, currency exchange rate, investors' sentiment etc. He used traditional shallow machine learning algorithms, and get an error level at 3.146%.

As bitcoin prices changed very frequently and sharply in 2021, it will be hard to achieve the error level in case we want to predict bitcoin prices for 2021. However, the bitcoin prices drivers won't change dramatically. So our predicating factors are mostly inspired by Obryan's work.

H. Jang et al. used rolling window LSTM model to predicate bitcoin prices in 2018 [3]. They mainly include blockchain information and macroeconomics information like exchange rate to do the predication. They compared different machine learning algorithm like SVR, LR etc. They don't have a comparison with typical time-series algorithm like ARIMA, VAR etc. From their experiment, rolling window LSTM is a promising model for bitcoin price predication.

H. Jang et al. used a modern NN approach to do the predication, however they don't include traditional time-series algorithm as a benchmark. Another thing we want to mention is they didn't consider emotional drivers. Speculative investors are sensitive to public emotion. Including search index, social media emotional analysis could be helpful for accurate predication.

3. Predication Models

Here we will introduce our work on bitcoin predication. In the first section, we will introduce the data sources we will use and also data pre-processing we adopted. We build two models to do the predication. One is ARIMA model, which used only history of bitcoin prices to do the predication. ARIMA is a popular model to do the time-series predication. We will use it as a benchmark. Another model we will build is LSTM. We will include multiple factors to do the predication with LSTM. We will prove LSTM is a better method to do the predication.

3.1. Dataset and Data Processing

We used following dataset to do the predication:

- Bitcoin price data - www.investing.com
- Gold price data - www.investing.com
- SP500 data - www.investing.com
- Shanghai Strock data - www.investing.com

- USD to CNH exchange rate data - www.investing.com
- EUR to USD exchange rate data - www.investing.com
- EUR to CNH exchange rate data - www.investing.com
- Baidu index data with keywords "Bitcoin" (in Chinese) - index.baidu.com
- Google search trend with keywords "Bitcoin" - trends.google.com

As we are familiar with SQL language, we decided to import all data we got into a SQLite database. Our models will retrieve data from SQLite database and we could filter out date range we are interesting in with SQL language. A free database manage tool DBeaver is leveraged to explore and manage data in the SQLite database.

As we used stock market index as our modeling features, and stock market is closed at the weekend. We will simply fill the weekend stock market index using the close index of last Friday.

Google trends will provide only monthly data if the time range crossed more than two years. A tool [2] were used to retrieve daily data.

3.2. ARIMA

In this section we will implement a seasonal ARIMA model to predicate bitcoin price. We will first give a brief introduction on seasonal ARIMA. Later we will cover details of seasonal ARIMA modeling, and then give a short summary.

3.2.1 ARIMA Introduction

The ARIMA model was introduced by George Box and Gwilym Jenkins in their textbook Time Series Analysis: Forecasting and Control in 1970. It also referred to as Box-Jenkins methodology, and is a very popular method to do time-series predication, especially for finical forecasting. You can refer to an online book [4] by Rob and George for more details.

ARIMA stands for AutoRegressive Integrated Moving Average. It is a generalization of the simpler AutoRegressive model and Moving Average model with integration.

AutoRegressive model AutoRegressive model leveraged historical observations to predict future value. It assume the data it is observing are stationary, e.g the mean and variance of data has no dependency on time. In this case, it can forecast the future value using following formula:

$$y_t = \mu + \sum_{i=1}^{p} \gamma_i y_{t-i} + \epsilon_t$$

Here y_t is the forecast value, μ is a constant value, p is called AR order, meaning how many preceeding observances have impact on current value. γ_i is coefficients and ϵ_t is residual error.

Moving Average If we look at AutoRegressive model, we will find the residual error is something we can not ignore. If in time step $t-1$ there is a white noise, which will impact time step t, and it will propagate to time step $t+1$, the white noise will have very significant impact on predications. Moving average model is used to handle residual errors. It assume the future value we want to predicate is impacted by previous residual errors.

$$y_t = \mu + \epsilon_t + \sum_{i=1}^{q} \theta_i \epsilon_{t-i}$$

Here y_t is the forecast value, μ is a constant value, q is called MA order, meaning how many preceding residual errors have impact on current value. θ_i is coefficients and ϵ_t is residual error.

Moving average model is taking residual error into predication consideration, which will improve predication accuracy.

Integration AutoRegressive model assume data are stationary, but which is not the case in most time. We need some tricks to make data stationary. Integration assume if we subtract previous observations from value of current time step, the data will become stationary. We usually use d to notate the order of integration (e.g. the number of times doing differencing).

ARIMA ARIMA is a combination of above three. ARIMA model can be (almost) completely summarized by three numbers:

$$\begin{aligned} p &= \#\, of\ autoregressive\ terms \\ d &= \#\, of\ nonseasonal\ differences \\ q &= \#\, of\ moving-average\ terms \end{aligned}$$

This is called an "ARIMA(p,d,q)" model.
For example, ARIMA(1,1,2) means

$$y_t = \mu + \epsilon_t + \gamma y_{t-i} + \phi(y_{t-1} - y_{t-2}) + \sum_{i=1}^{2} \theta_i \epsilon_{t-i}$$

Seasonal ARIMA Seasonal ARIMA is an ARIMA model which include seasonal factors into consideration further. For example, personal bank account balance may have monthly changes because he or she receive salaries periodically. When we do account balance predicating, include

seasonal factors into consideration will be very helpful. We can use exactly same parameters to describe seasonal components of the time-series data. e.g. We can add three additional numbers:

$$P = \#\ of\ seasonal\ autoregressive\ terms$$
$$D = \#\ of\ seasonal\ differences$$
$$Q = \#\ of\ seasonal\ moving - average\ terms$$

The complete model is called an "ARIMA(p,d,q) $\times$ (P,D,Q)" model.

3.2.2 Seasonal ARIMA model implementation

As we said above, to build a seasonal ARIMA model, we have 6 critical hyper-parameters to identify. In the first step below, we will verify the seasonal component of bitcoin price (from Jan 2019 to Mar 2021). Through seasonal components analysis, we will identify 3 parameters related to seasonal components. After that, we will do necessary transformation and differences to make the time-series data stationary. During this step, we will find a right value for parameter d. After that, we will search through the hyper-parameters spaces to right find p and q, and find a best model with right coefficients for each term.

Seasonal components analysis We used $statsmodels.api.tsa.stattools.adfuller$ to do a Dickey-Fuller stationary test. The test result is 1.000, which indicate the data is not stationary at all. We also used $statsmodels.api.tsa.seasonal\_decompose$ to do the seasonal components analysis. Figure 2 illustrated the analysis result. From the chart we can find bitcoin prices contains very few seasonal components, it is even less significant than residual errors. Hence we could ignore seasonal components. In this case, we can simply set (P, D, Q) to $(0, 0, 0)$

Stationary check and transformation As the time-series data is not stationary by default, and the growth of bitcoin price is near to an exponential growth, we do a $boxcox$ transformation. After that do a one order difference to make data stationary. Figure 3 show the result after transformation. We did another round Dickey-Fuller stationary test and its result is 0.000, which means the data is stationary.

Searching best model Based on previous analysis, after a boxcox transformation, the parameter d will be 1. And parameters (P, D, Q) will be $(0, 0, 0)$. Hence when searching models, we only need consider (p, q). Supposing those two parameters can take values 0, 1, 2 respectively, we will have 9 combinations to search. Figure 4 illustrate the whole searching process. We will use AIC (Akaike Information Criterion) as the criteria to select models.

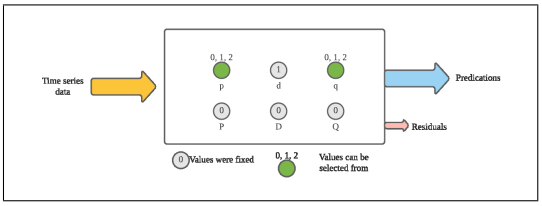

Figure 4. Models Search [6]. We will use AIC as criteria to choose models.

When we have a model fitted, we will use that predicate the value of next day only to improve predication accuracy. This is a kind of rolling window predication. When we finish a predication for a date, we will add the real data to train dataset and train a new model. We start the predication from Jan 12 2021, and predicated 45 days in total, which means we will train 45 models independently to do the predication. Figure 5 is the predication result.

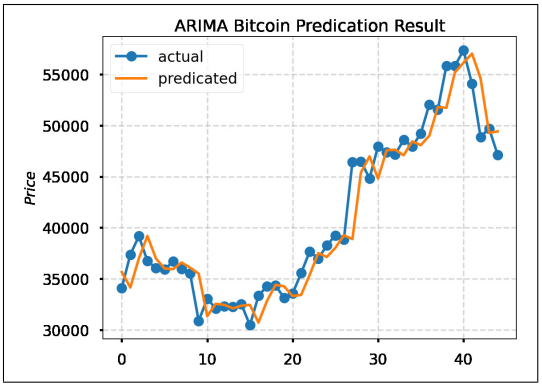

Figure 5. Predication result, from Jan 12 2021 to Feb 25 2021.

Summary The average RMSE of predication 50.21. Which is 0.1% of price value, it is very good. However, when we look at the TPA (Trend Predication Accuracy), it is 3.57% only (1 of 28). The predication by ARIMA is almost just shift the real time-series data right by one time step, which makes it almost not useful for any investors.

We will continuously explore LSTM modeling technology.

3.3. Long short-term memory, LSTM

In this section we will implement a LSTM model to predicate bitcoin price. We will first give a brief introduction on RNNs and LSTM networks. Later we will cover details of LSTM modeling, and then give a short summary.

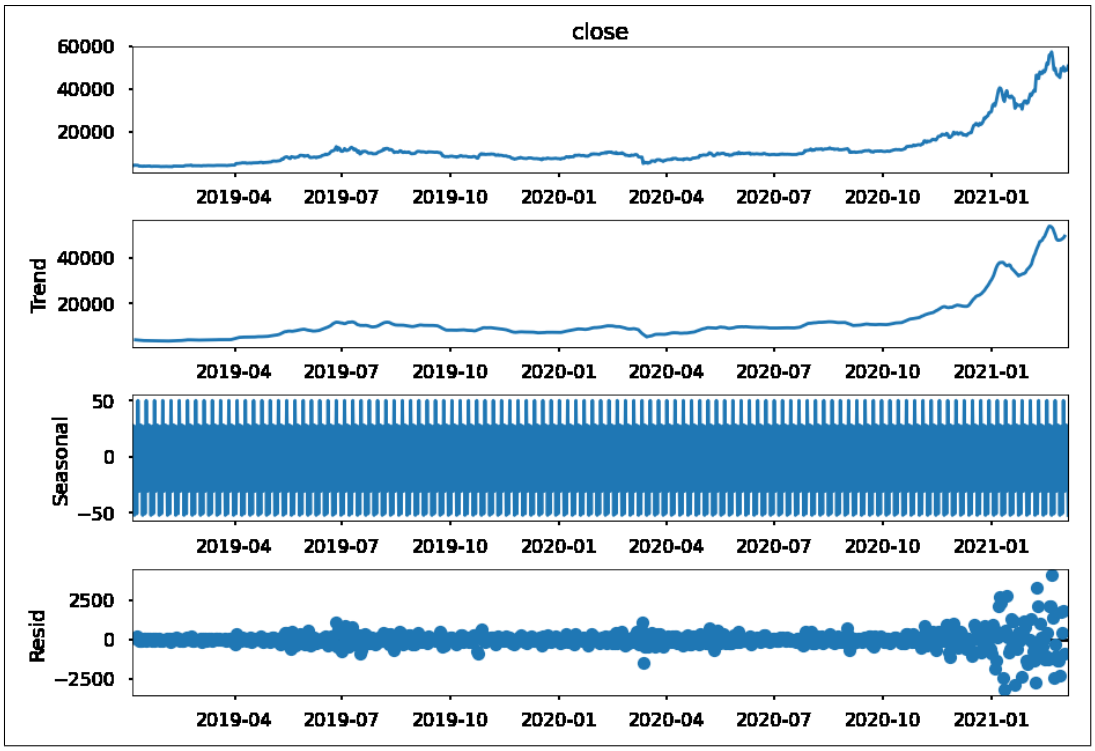

Figure 2. Seasonal components analysis of bitcoin price. Seasonal components are not significant.

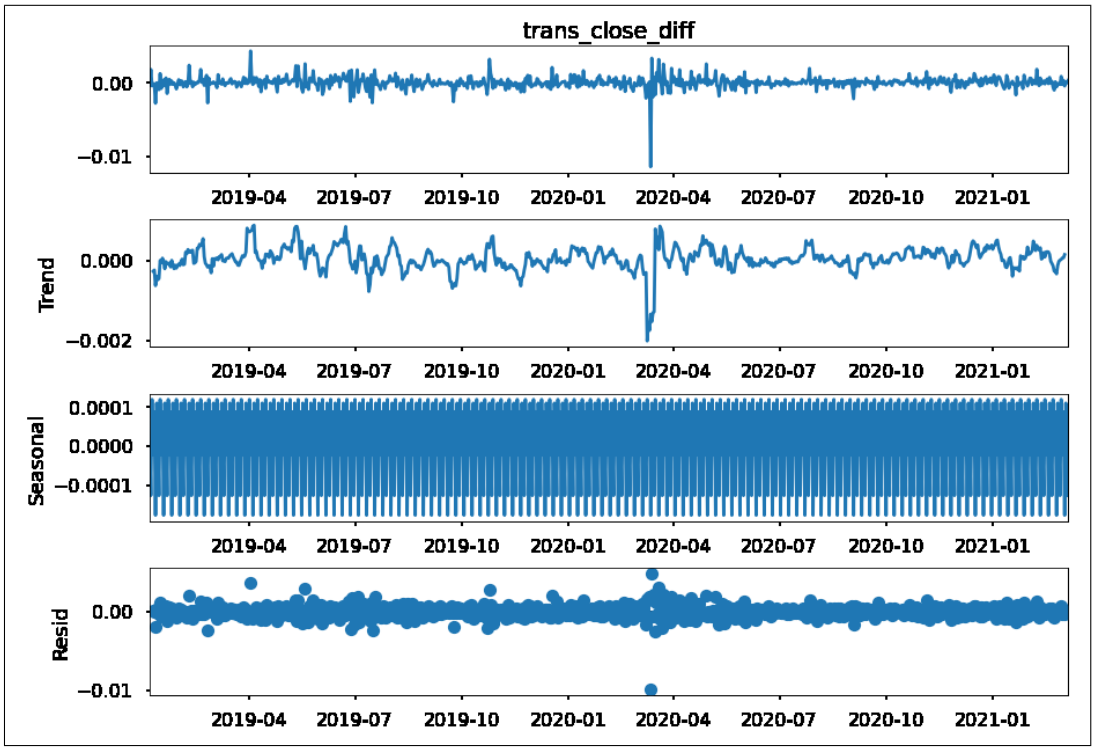

Figure 3. Bitcoin price after boxcox transformation and one order difference. It becomes stationary after transformation.

3.3.1 Recurrent Neural Networks

Recurrent Neural Networks or RNNs in short, are a type of neural network used to process and predicate sequential data. Time series data is a type of sequential data, hence RNNs are good network structures to handle them.

Vanilla RNNs Review Recurrent neural networks were based on David Rumelhart's work in 1986 [10]. Both fully-connected network and CNN didn't capture the sequence information in the dataset. To precess sequential data, a straightforward thinking is we need feed previous time step output into current time step input. And that is Vanilla RNNs basic structure. Please refer to figure 6. The left side of figure 6 is the most simple RNN. Input X got a transformation (with weight U) then we have the hidden state s, from one perspective, s got a transformation (with weight V) and generate output o of current time step, and from another perspective, s was feed into next time step after a transformation (with weight W). the right side of figure 6 is the same RNN network which unfolded to all time steps.

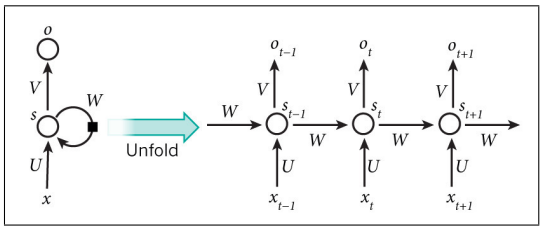

Figure 6. Vanilla Recurrent Neural Networks[7]

Vanilla RNNs are not able to handle long range dependencies and also hard to train because the vanishing gradient and the exploding gradient problem. In practice, few people are using vanilla RNNs now.

LSTM Review Long short-term memory (LSTM) networks were invented by Hochreiter and Schmidhuber in 1997[1]. Unlike vanilla RNNs cell can only pass one hidden state to next time step, LSTM cell can pass a hidden state and also a cell state. How can LSTM track long term dependency?

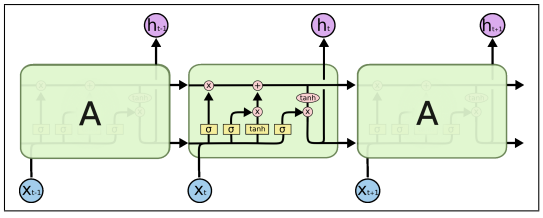

Figure 7. Long short-term memory (LSTM) networks [7]

Please refer to figure 8. In one LSTM cell, there are three major components, called three "gates".

- The yellow σ function accept current time step input X_t and previous time step hidden state h_{t-1}. Its output will do an element-wise product with previous time step cell state C_{t-1}. This is called **"forget gate"**. The forget gate will control whether or not the previous cell state C_{t-1} should be considered in current time step. If the output of forget gate is zero, that means the current time step will forget previous cell state.

- The orange $tanh$ function will take current time step input X_t and previous time step hidden state h_{t-1} and generate an output $\tilde{C}_t$. The output $\tilde{C}_t$ is only a candidate. Whether or not it can be added into cell state, is controlled by the output of green σ function. If the output of green σ function is one, then $\tilde{C}_t$ will be added into cell state, combined together with portion of C_{t-1} which passed forget gate. After the combination, we will get the cell state C_t for this time step, The green σ is called **"input gate"**.

- The cell state C_t will be passed to next time step, which will be controlled by forget gate of next time step. The red σ function accept current time step input X_t and previous time step hidden state h_{t-1}. Its output will do an element-wise product with result of $tanh$ function, which take current step cell state as input. The result of this element-wise product will become hidden state h_t of current time step, and will be feed into next time step. The red σ is called **"output gate"** as it control whether or not output cell state as hidden state.

All the parameters of those gate functions, activation functions will be trained through back-propagation. It is proven that it can overcome most challenges vanilla RNNs have. The drawback is it can be computationally expensive. Although we will not cover here, but GRUs discovered by Cho, et al. in 2014 relieved some computation cost issues.

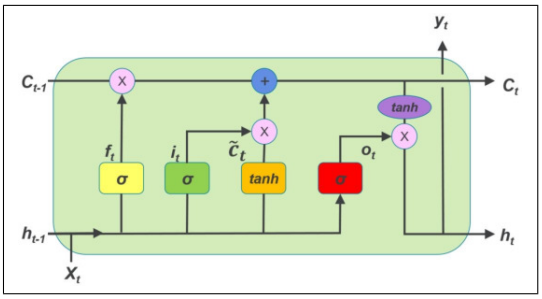

Figure 8. Long short-term memory (LSTM) networks cell structure [7]

3.3.2 LSTM model implementation

In this section, we will first introduce the network structure we will use for bitcoin price predication. Then we will explain the data preparation and dataset split. Then we will demonstrate how we adjust several key hyper-parameters, such as rolling window size, batch size, iteration number etc. And finally we will give a summary on LSTM model performance.

Network structure To build a LSTM model, we need define the network structure. We will use one LSTM layer, plus two fully-connected layers. please refer to figure 9.

Layer (type)	Output Shape	Param #
lstm_59 (LSTM)	(None, 50)	11800
dense_118 (Dense)	(None, 25)	1275
dense_119 (Dense)	(None, 1)	26

Total params: 13,101
Trainable params: 13,101
Non-trainable params: 0

Figure 9. Our LSTM network Structure

Dataset split and Data pre-processing First of all, we need separate data into train dataset, validation dataset and test dataset. We will keep 45 records for test, to keep consistent with ARIMA model. We will leave 30 records for validation and other dataset for training. After hyper-parameters were selected, we will combine the training dataset and validation dataset together to re-train the model, and then do the predication.

For RNNs model, we need split data using rolling window mode, using past several days features (called window size) to predicate next day bitcoin price. for example in figure 10, we split 6 days data into 3 training data pairs, using past 3 days data to predicate the 4th day bitcoin price.

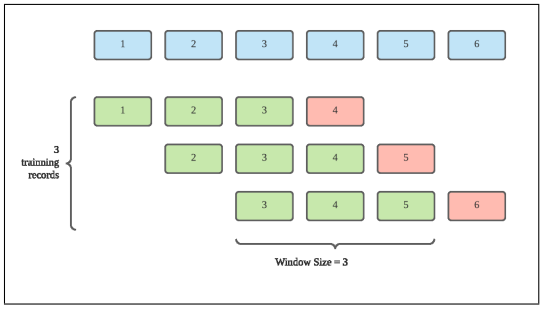

Figure 10. Rolling window data pre-processing

Hyper-parameters adjust We will pickup hyper-parameters through validation dataset. The hyper-parameters we need optimize are:

sw_width: rolling window size.
$layer1\_unit$: LSTM layer unit number.
$layer2\_unit$: Dense layer unit number.
$batch\_set$: training data batch size.
$epochs\_num$: epoch numbers.

The hyper-parameters selected are: rolling window size=3, LSTM layer unit number=50, Dense layer unit number=25, batch size=3, epoch numbers=50.

Summary Please refer to figure 11 for the final model result. The RMSE of LSTM model is 78.65, which is a bit higher than ARIMA model. The TPA of LSTM model is 50% (14 of 28), which is much higher than ARIMA model. Especially it almost predicated every peaks and troughs before day 30 (which is from Jan 11 2021 to Feb 9 2021).

In general LSTM based RNNs are very promising method to do time-series predication. However, how to improve the trend predication accuracy is remaining an open question. We will deep dive on this direction in later research. For example, building a classification model to predicate the trend, instead of regression model to predicate the price.

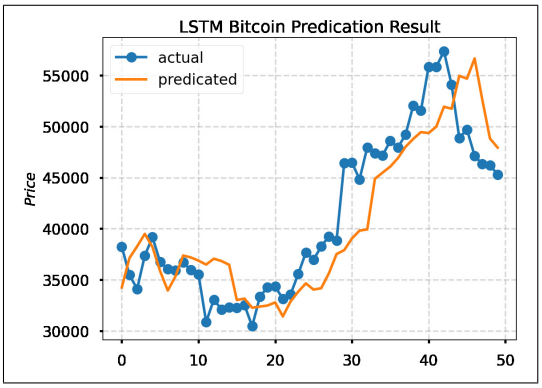

Figure 11. LSTM model predication result

4. Conclusion and Future works

This study focuses on how to predicate bitcoin closing price leveraging macro-economy indexes and past bitcoin prices. We used both ARIMA and LSTM based models to do the predication. ARIMA can only ingest history bitcoin prices as the input, whereas LSTM can ingest all features as its input. As bitcoin prices contains very few seasonal components, we used ARIMA, not SARIMA. The result of ARIMA model is disappointed. Although the RMSE of ARIMA model is not too high, the predication is just lag one time step compare to real bitcoin price, this gives no value to bitcoin investors.

The LSTM model has a bit higher RMSE, but it can give more accurate trend predication. This is because we include

more features into the LSTM model, hence LSTM model can predicate the changes of prices trend.

In summary, in our experiment, LSTM model is better than ARIMA in bitcoin price predication.

In the future we will improve the prediction from three perspectives:

- Introduce more data dimensions. For example, introduce sentiment analysis on public media like Twitter, Weibo and open forums. We will try if adding more dimensions will improve prediction accuracy.
- Try the predication with the latest time-series predication algorithms, like DeepAR etc.
- Try the predication with vector based time-series algorithms, like VAR etc, so we can leverage all features on hand.

The financial assets valuation predication is remaining a challenging problem and worth continous research on that.

5. Appendeix

The source code and dataset used in this paper can be found here:

https://github.com/yafengguo/cs584_term_paper.

References

[1] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9:1735–80, 12 1997.

[2] https://github.com/qztseng/googletrends daily. google trends daily.

[3] Jang Huisu. Predicting bitcoin prices by using rolling window lstm model. 2018.

[4] Rob J Hyndman and George Athanasopoulos. Forecasting: Principles and practice, 2nd. 2018.

[5] Satoshi Nakamoto. Bitcoin: A peer-to-peer electronic cash system. (4), 2008.

[6] Robert Nau. Introduction to arima models, 2014.

[7] Christopher Olah. Lstm walkthrough.

[8] O. Poyser. Exploring the determinants of bitcoin's price: an application of bayesian structural time series. *Papers*, (2), 2017.

[9] Obryan Poyser. Exploring the determinants of bitcoin's price: an application of bayesian structural time series, 2017.

[10] DE Rumelhart, G. E. Hinton, and R. J. Williams. Learning representations by back propagating errors. *Nature*, 323(6088):533–536, 1986.

[11] Chainalysis Team. Why bitcoin is surging and how this rally is different from 2017, 2020. chainalysis.com.

[12] Wikipedia. Bitcoin.