Bitcoin Price Time Series Prediction Using ARMA & LSTM Model

Xiangyuan Ren, A53249047, xir010@eng.ucsd.edu
Shiyi Wang, A53258363, shw333@ucsd.edu

Abstract— **The bitcoin price is now dramatically increasing which make the prediction of its price a hot topic nationwide. This paper concludes how we use ARMA and LSTM model separately to predict the bitcoin price in short and long term, and compare the performance of each model by MSE (mean squared error).**

I. INTRODUCTION

Since the price of bitcoin receives more and more attention, we try to analyze the trend of bitcoin price and predict the future price based on the previous dataset. Time series prediction is not a new phenomenon. Prediction of mature financial markets such as the stock market has been researched for decades. Bitcoin presents an interesting parallel to this as it is a time series prediction problem in a market still in its transient stage. As a result, there is high volatility in the market and this provides an opportunity in terms of prediction. In addition, Bitcoin is the leading cryptocurrency in the world with adoption growing consistently over time. Due to the open nature of bitcoin it also poses another paradigm as opposed to traditional financial markets. It operates on a decentralized, peer-to-peer and trustless system in which all transactions are posted to an open ledger called the Blockchain. This type of transparency is unheard of in other financial markets.

Nowadays the main trend of bitcoin price is obvious, it is exponentially increasing. However, the price is still fluctuating within days. For example, the prices decreased 40% when China banned bitcoin, and then the prices gradually bounced back. Thus if the prediction is valid for a short or long term, there are huge profits from bitcoin market.

The main predictive process is as follow, we first get the time series of the bitcoin prices using the bitfinex API, and then we design several functions to automatically get bitcoin price with one minute interval of a whole day. After that, we use ARMA and LSTM model to analyze the data and predict respectively. Finally, we analyze the result from the two model and propose some insight into the result.

II. DATASET ANALYSIS

A. Obtain the price data from Bitfinex

The dataset of the bitcoin prices that we use is obtained from the Bitfinex website. Bitfinex provides an overview of the state of the market. It shows the current best bid and ask, as well as the open price, close price, highest price and lowest price within the trade time frame. It also includes information such as daily volume and how much the price has moved over the last day.

In order to get the bitcoin trade prices of every minute during a long time frame, We use pycurl toolbox in python to deal with the dataset and make the analysis on it. The maximum number of one request set by the website is 1000, which is almost 16.7 hours. So we first define a getprice function, which takes the millisecond time stamp and the coin symbol as inputs, and returns the trade prices and volume of the following 720 minutes from Bitfinex. Then we define a oneday function to change the start time we want from numerical date into the standard time stamp format, and call the getprice function two times to get a whole day's trading information per minute and write the final dataset to a csv file.

From what we have done above, we can get a whole day's bitcoin price dataset simply by calling the oneday function, and we use panda toolbox to get the dataset in python notebook in order to analyze it easily. The head of the dataset looks as follows.

	OPEN	CLOSE	HIGH	LOW	VOLUME
MTS					
2017-12-14 00:00:00	16648.0	16651.0	16680.0	16613.0	98.959339
2017-12-14 00:01:00	16650.0	16676.0	16701.0	16650.0	81.457980
2017-12-14 00:02:00	16676.0	16688.0	16770.0	16676.0	124.565816
2017-12-14 00:03:00	16689.0	16770.0	16770.0	16686.0	35.940528
2017-12-14 00:04:00	16767.0	16768.0	16780.0	16767.0	71.088120

Fig. 1. dataset of bitcoin price per minute

B. Analysis of the price data

In order to get the price of every minute conveniently for the following work, we set the column of date as the index. From what we have learned from class, the large number theorem and the central theorem all require samples with the same distribution, which is equivalent to the stability of the time series, and many of our modeling processes are based on the large number theorem and the central limit theorem preconditions. If it is not satisfied, many of the conclusions obtained are unreliable. Therefore, we define some function to analyze the stationary stability of the data series.

First, we define a draw_trend function to calculate the rolling mean averages and weighted mean averages of in order to smooth out the data to see the underlying trend of it. We then calculate the p-value of the dataset to be 0.614873, which is not big.

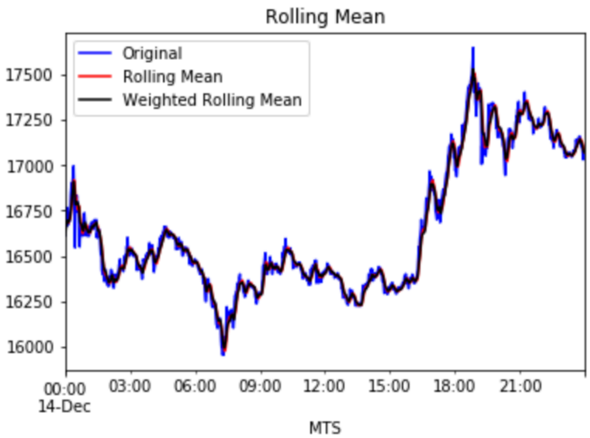

Fig. 2. rolling mean of dataset

In order to quantify the correlation between the point at t and point at $t+k$ (k is the lag) with respect to expectation, autocorrelation is used. And we also use partial autocorrelation to describe the direct effect of a value at t on value at $t+k$, ignoring the values between them. So we define a draw_acf_pacf function to get the autocorrelation and partial correlation.

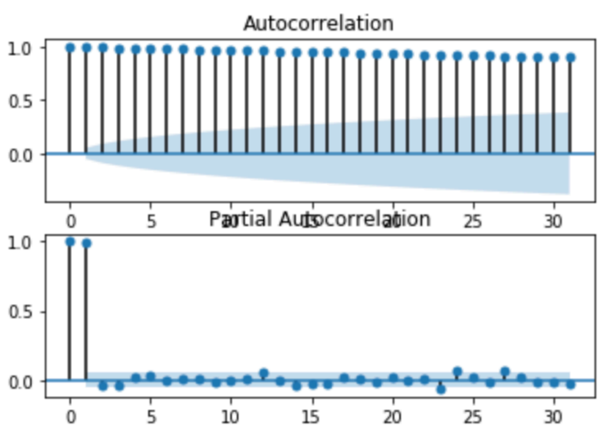

Fig. 3. autocorrelation and partial correlation of dataset

From the figure above, we find out that both the auto-correlation and partial correlation coefficients have trailing characteristics, and they all have obvious first-order correlations, so we set $p = 1$ and $q = 2$. Below we can use the ARMA model for data fitting.

III. ARMA MODEL

Autoregressive moving average (ARMA) model is a popular method that is widely used for time series forecasting. It is based on the combination of the autoregressive (AR) model and the moving average (MA) model. It predicts the future price from the prices of several past steps. Specifically, its prediction is made as a linear combination of the past prices subject to a Gaussian noise term.

A. Training set

We import ARMA model from statsmodels.tsa.arima_model toolbox, and choose the parameters of $p = 1$ and $q = 2$ from previous data analysis. We choose the bitcoin price dataset of every minute on the date of 12-14-2017, and use it to train the ARMA model and get the prediction on this training set with a MSE=720.98.

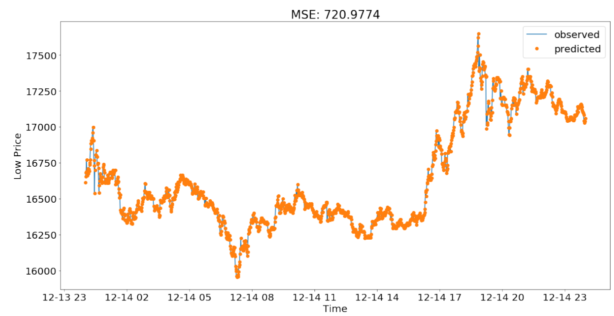

Fig. 4. ARMA model on the training set

B. Testing set

Then we use the model in the previous part to make prediction of the next two hours on the date of 12-15-2017. We put all the previous data in a list called history, and for every next minute, we use the ARMA model.fit to get the model, and then use model.forecast to get the next prediction. After the forecast, we add the true value of the current minute to the history list to help for the next prediction. Below is a comparison of the prediction label and true label, and the MSE we obtained from this model achieves 351.58.

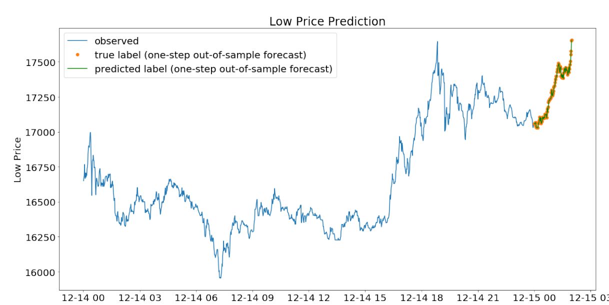

Fig. 5. ARMA model on the training and testing set

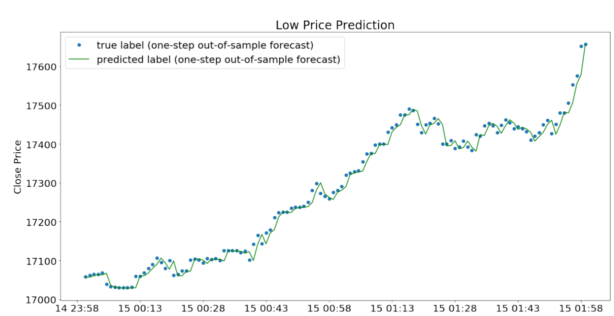

Fig. 6. ARMA model on the testing set

C. Analysis

From the result above, we can see the model fits well on short time prediction especially in one minute. However, it cannot predict on long term prediction because of the limit of linear stationary hypothesis. As a consequence, we implemented another machine learning method of LSTM model to realize long term prediction.

IV. LSTM PREDICTION

In this section, we present how we use LSTM model to predict Bitcoin price. LSTM(Long short-term memory) is an improved version of Recurrent Neural Network, which solves the long-term dependence of RNN. In the model, we use the prices and volumes of previous 7 days to predict the next day's price.

A. Data

Before defining the model, we first get one year's bitcoin prices and volumes using the bitfinex API, and then we preprocess data using pandas to get the 10 dimensional data shown in Fig.

	Unnamed: 0	MTS	VOLUME	OPEN2	OPEN3	OPEN4	OPEN5	OPEN6	OPEN7	OPEN8	OPEN1
395	4	2017-12-13 16:00:00	45731.174456	0.744284	0.732666	0.634640	0.620590	0.683491	0.724830	0.557366	0.701486
396	3	2017-12-14 16:00:00	63123.657646	0.701486	0.744284	0.732666	0.634640	0.620590	0.683491	0.724830	0.712726
397	2	2017-12-15 16:00:00	48267.950664	0.712726	0.701486	0.744284	0.732666	0.634640	0.620590	0.683491	0.776761
398	1	2017-12-16 16:00:00	51803.972129	0.776761	0.712726	0.701486	0.744284	0.732666	0.634640	0.620590	0.865384
399	0	2017-12-17 16:00:00	16886.784072	0.865384	0.776761	0.712726	0.701486	0.744284	0.732666	0.634640	0.851712

Fig. 7. Input Data Example

B. Model

The LSTM model has parameters including time lengths, input dimensions, layers, units. In our model, we set time lengths to be 7 to include the previous price and volume. We set input dimensions to be 2 to include both the volume and price. The output is one dimensional. Since we are building a simple LSTM model, we set the layers to be 4 including two dense networks and we use dropout to prevent overfitting. Finally, we compile the model using MSE lost function and adam optimizer. After one hundred epochs we get our model trained. The result is plotted in Fig. The training error is 0.01, while the test error is 0.17

C. Analysis

From the result we can see that as time increases, the prediction get worse. Though we have are normalized data, the output is still bad, looking like this simple model overfit the training data. There are two main reasons. The first is that bitcoin prices are more than time series, the trend is changing from time to time. The second is that there are a lot of preprocessing need doing, like logarithm.

V. CONCLUSIONS

In this paper, we present ARMA and LSTM models to make short and long term prediction of bitcoin price. Both models are evidently effective learners on training data, and the LSTM is more capable of recognizing longer-term dependencies. However, from the result of two model we can

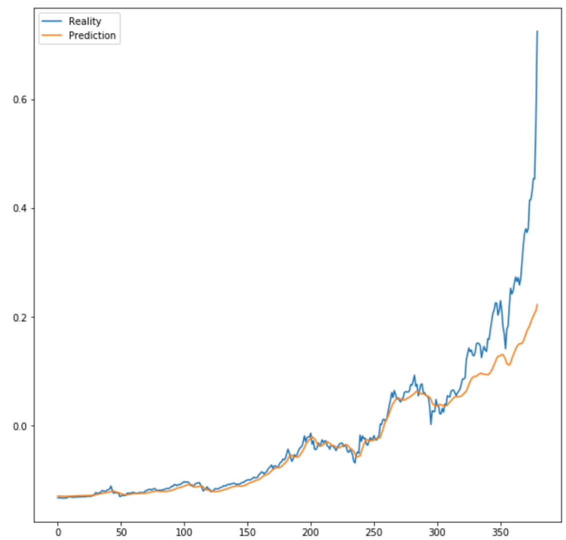

Fig. 8. LSTM prediction

tell that even though LSTM is said to perform better than ARMA, the truth is that it will need complex parameterization, which will cost lots of time. Besides, although both models can yield a similar prediction in term of MSE, they do not provide really helpful information because they both cannot predict the turning point of the prices. Therefore, there are some measures that can be done to improve the model. First, the bitcoin price is not just a stationary trend, we should make some adjustment to it to ensure the p-value is below 0.01. Besides, the LSTM can only include the information provided by the training set, which is the historical data. We may need to incorporate reinforcement learning to update the model so the model can respond to the future changes.