Bitcoin Price Analysis

Jeff Ho, Taiyue Mao

Introduction

Bitcoin is a decentralized virtual currency first proposed in "*Bitcoin: A Peer-to-Peer Electronic Cash System*" [1] by Satoshi Nakamoto in 2008 and released as open-source software in 2009. Its price kept wandering little above $0 before 2011 and started to surge up exponentially with its growing popularity in 2013, reaching the peak of $1151 by the end of the year. Then a staggering down trend followed during 2014-2015. As in April 2015, the price is around $200-$260.

Bitcoin was invented and is mined and produced based on a rule and its statistics. It has been playing multiple roles in our society due to the following characteristics:

- First, Bitcoin plays a currency role. Even without mass adoption, Bitcoin is commonly referred to as digital (virtual) currency (cash) or cryptocurrency and it can be used to buy products or services from anyone who accepts it.
- Second, Bitcoin is a monetary system based on block chain, a public ledger that records transactions [2]. Different from the traditional currency issuers, the system is designed, decentralized and transparent.
- Third, Bitcoin is an investment option and some people hold it as digital asset. There are many online platforms where people can trade it publicly.
- Fourth, Bitcoin appeals to tech-savvy people and is now very popular on the social website.

The above characteristics of Bitcoin, its dramatic fluctuation in the history and popularity on the social website are the main reasons that motivate us to do this research.

The goal of our research is to explore some possible features related to Bitcoin price. As a result, we collected data over a broad range of angles, which can be categorized as three kinds, Bitcoin statistics, social media and foreign exchange rates. The sources are *Blockchain*, *Twitter* and *TrueFX* respectively.

We wrote around 350 lines of Python code [3] to connect to APIs of different data providers and built a MySQL database for data storage. Every minute, the code is recursively executed to collect real time data. The data are of 31 features (1 dependent variable and 30 independent variables) and 5000 instances (5000 minutes), from *April 7th 1:00 am* to *April 10th 12:20 pm*. Following table summarizes all the data we have collected:

Source	*Blockchain*	*Twitter*	*TrueFX*
Category	Bitcoin Statistics	Social Website	Foreign Exchange
Number of Features	20	8	3
Features	current_price, total_fees_btc, blocks_size, trade_volume_usd, estimated_btc_sent, timestamp, trade_volume_btc, totalbc, minutes_between_blocks, miners_revenue_usd, estimated_transaction_volume_usd, nextretarget, difficulty, n_blocks_mined, total_btc_sent, n_blocks_total, miners_revenue_btc, hash_rate, n_tx, n_btc_mined	up_tweet, down_tweet, up_ratio, up_difference, love_tweet, hate_tweet, love_ratio, love_difference	EUR_USD, GBP_USD, USD_JPY

Bolckhcain provides clear definitions of Bitcoin statistics. To monitor social website activity, we search over the most recent 1000 tweets on *Twitter* that contains "#Bitcoin". In these 1000 tweets, we count the number of different key words, "up", "down", "love" and "hate", and also calculate the ratios and differences of the positive and negative terms. The foreign exchange data is the exchange rate between two currencies. For example, EUR_USD is the exchange rate of Euro to US dollar.

This report will be divided into two parts. The first one is the regression model analysis, which tries to model the Bitcoin log return from some potential features. The second one is about the basic time series analysis applications, with main focus on ARIMA and GARCH models.

Part One: Regression Model Analysis

Even though we have 30 independent variables at hand, many of them are highly correlated. So our group will drop one variable if a pair of two variables are highly correlated. For example, the "*estimated transaction volume*" and the "*transaction volume*" provided by Blockchain are highly correlated, so we drop the "*estimated transaction volume*" because we prefer real market data. After we finish this process, there are only eight independent variables left:

blocks_size	trade_volume_usd	n_btc_mined	up_tweet	down_tweet	love_tweet	hate_tweet	EUR_USD

To model the Bitcoin log return through minute-by-minute data generates many difficulty for our group. The residual from multiple linear regression model has non-constant variance and autocorrelation problems. After we tried to model the error term as ARMA process, or use the GLS estimate, none of the predictor becomes significant. We found out that unlike minute-by-minute Bitcoin price data, most of other variables keep constant for around half an hour, or even longer period. Therefore, our group adjusts all the minute-by-minute data to hourly data. Then in this part, we will model the Bitcoin log return by hourly data (there are in total 84 hours).

1. Bitcoin log return and social media

There is something interesting between log return and two variables (*down_tweet* & *love_tweet*):

Figure-1

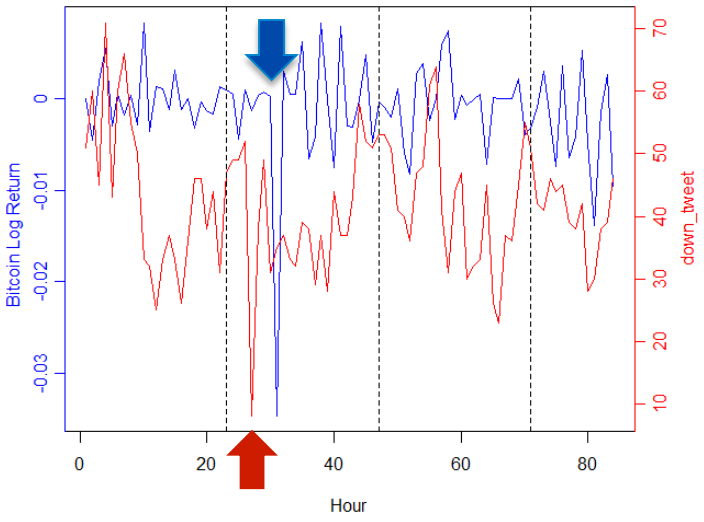
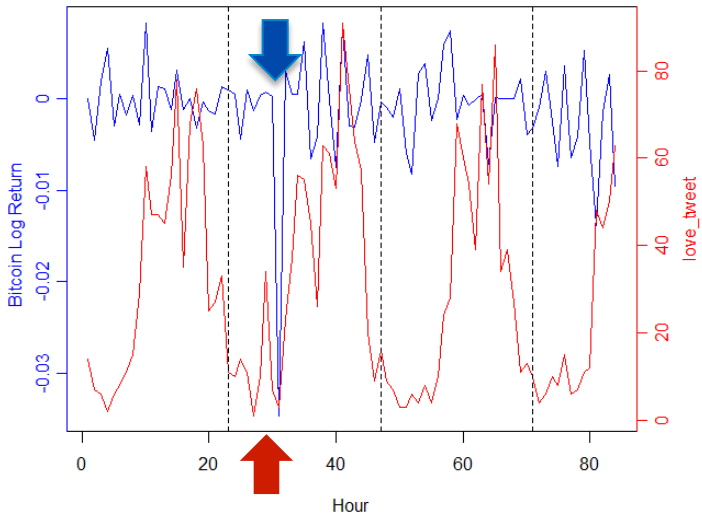


Figure-2



In "Figure-1" and "Figure-2", the vertical black dashed lines show the start of a new day (12 o'clock am). We can see clearly from "Figure-2" that *love_tweet* (red line) shows similar pattern from day to day. It climbs up gradually from morning to noon, and the count is static during night. But in fact, we need more data to verify this pattern.

If we look closer at "Figure-1", we can find out that before the sharp drop of the Bitcoin log return (indicated by a blue solid arrow), there is a abnormal sharp drop (indicated by a red solid arrow) in *down_tweet*. Similarly, in "Figure-2", before the sharp drop of the Bitcoin log return (indicated by a blue solid arrow), there is a abnormal rise (indicated by a red solid arrow) in *love_tweet*. The signal in "Figure-1" is more clear and stronger than that in "Figure-2".

2. Regression model setup

In order to create "clean" indicators from Twitter word counts (*up_tweet*, *down_tweet*, *love_tweet* and *hate_tweet*), we extract the trend components (with the help of R command: *stl*) from them, so we have *up_tweet_trend*, *down_tweet_trend*, *love_tweet_trend* and *hate_tweet_trend*. In order to model the Bitcoin log return, which is a stationary process, we need to take the first difference of the log transformed data of these four trend components to make them stationary. In addition, we need to do this for "*blocks_size*", "*trade_volume_usd*", "*n_btc_mined*" and " *EUR_USD*". Following table summarizes the data for regression:

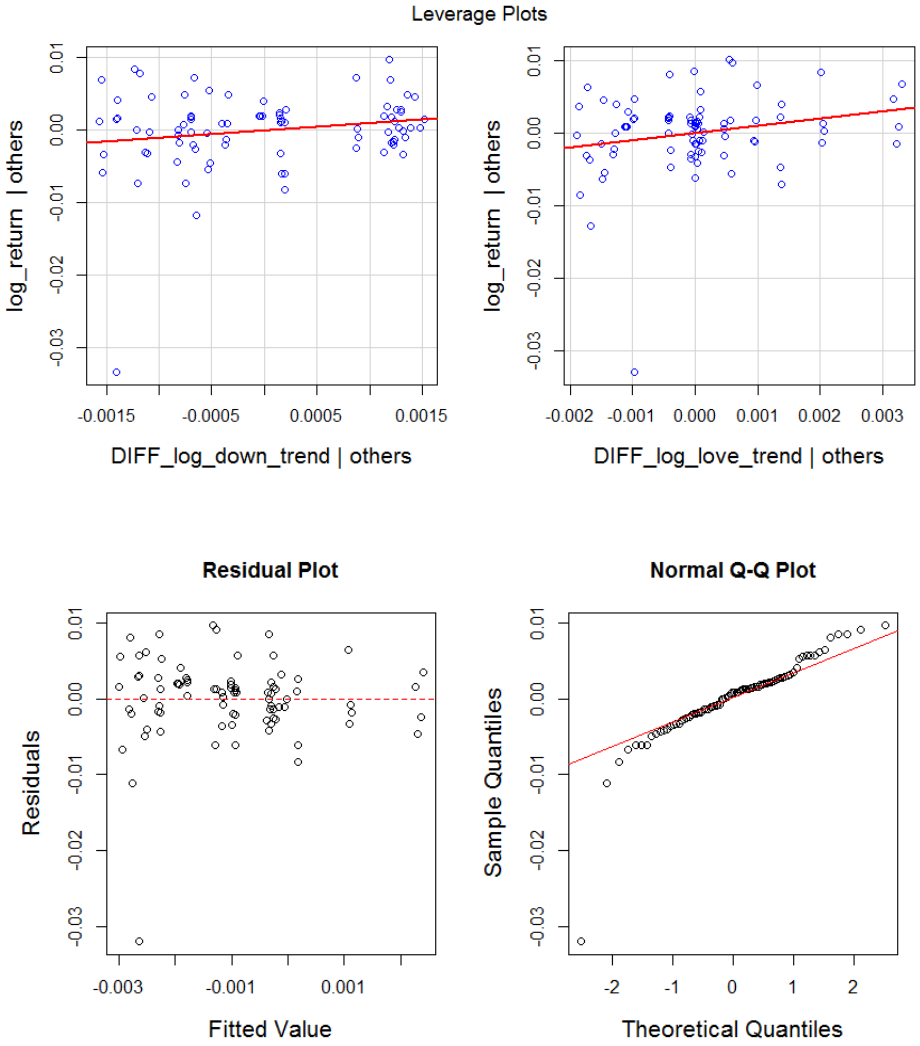
Dependent Variable	Independent Variables (8 in total)	
Bitcoin Log Return = DIFF[log(*current_price*)]	DIFF[log(*blocks_size*)]	DIFF[log(*trade_volume_usd*)]
	DIFF[log(*n_btc_mined*)]	DIFF[log(*EUR_USD*)]
	DIFF[log(*up_tweet_trend*)]	DIFF[log(*down_tweet_trend*)]
	DIFF[log(*love_tweet_trend*)]	DIFF[log(*hate_tweet_trend*)]

We want to model the dependent variable "Bitcoin log return" from 8 potential independent variables above. The variable selection technique is used (the information criterion is AIC). Our group have done forward, backward and stepwise variable selection, the results all indicate that we should use the following model:

Regression Model Summary				
	Estimate	Std. Error	t value	Pr(>|t|)
(Intercept)	-0.0012668	0.0006457	-1.962	0.0532
DIFF[log(down_tweet_trend)]	-0.1192472	0.0714356	-1.669	0.0989
DIFF[log(love_tweet_trend)]	-0.1846986	0.0907961	-2.034	0.0452
Multiple R-squared: 0.05767		Adjusted R-squared: 0.0344		

The above multiple regression model is not fancy at all, which has a quite small adjusted R-squared (around 3.44%). In addition, if we specify a 5% significance level, only the first difference of log transformed *love_tweet_trend* is significant. Only if we loosely specify a 10% level, all coefficients (including intercept) become statistically significant. But given the hourly data at hand, this is the best our group can do. On the other hand, one good thing about this model is that in terms of the model assumption validity, it does not have serious problem.

3. Regression model checking



Except for several outliers, the two leverage plots show no serious problem. In addition, the residual plot shows no serious departure from the constant variance and independence assumptions, except for one outlier there. That outlier is the due to the sharp drop of log return indicated in "Figure-1" and "Figure-2" on page two. The normal probability plot also shows that the normality assumption is not seriously violated.

Part Two: Time Series Analysis

In this part of analysis, since we will focus on Bitcoin log price only, we can utilize the whole data of Bitcoin current price, which consists of 5000 minutes. If we look at "Figure-3" on next page, we can clearly find out that the log price is non-stationary. Because for a stationary process, it will have the property of mean reversion, meaning that the process will frequently revert back to a fixed level. Our time series shows a clear downward trend throughout these 5000 minutes. The result from Augmented Dickey-Fuller (ADF) test tells us that indeed, the log price is non-stationary, with a p-value of 32.63%. We can take the first difference of log price and get *log return*. The ADF test on *log return* tells us that it is stationary, with a p-value of less than 0.1.

Figure-3: Time Series Plot of Bitcoin: log(Price)

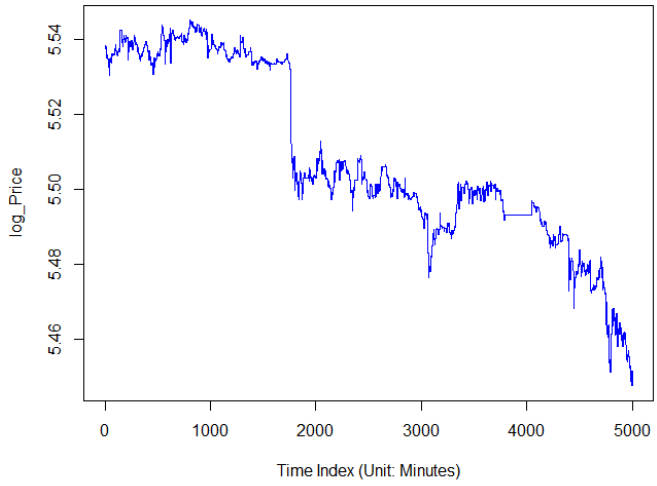
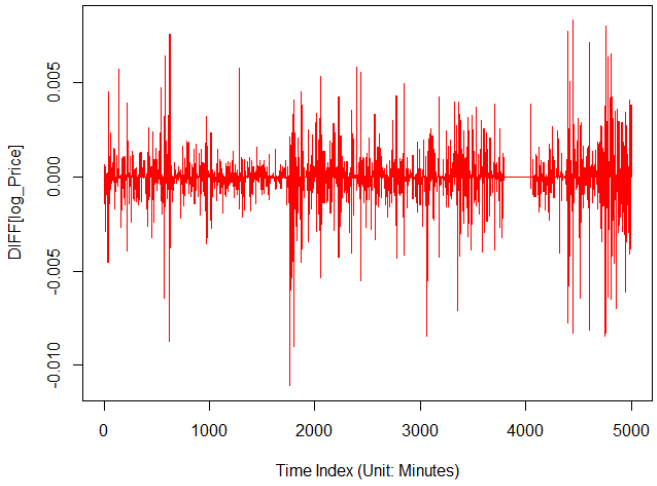


Figure-4: Time Series Plot (Bitcoin): Log Return

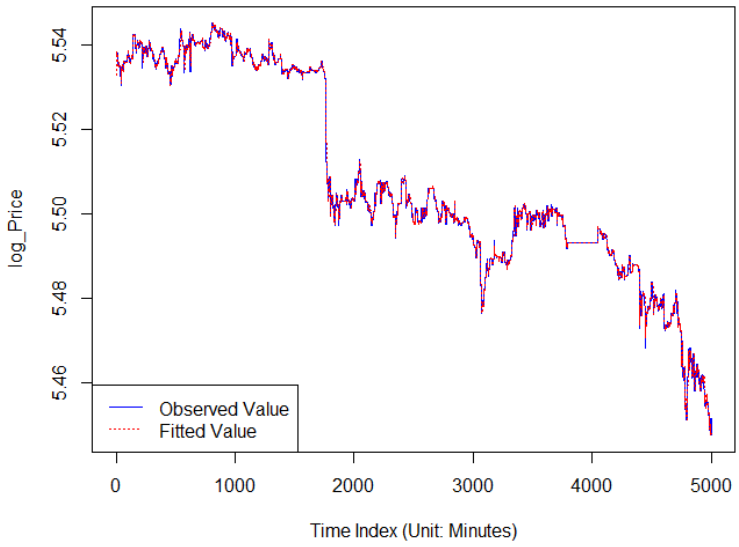


"Figure-3" is the log price and "Figure-4" is the log return. Since log price is non-stationary and its first difference: log return is stationary, we conclude that the log price is a $I(1)$ process. Therefore, we can fit ARIMA (p, d, q) model to the log price and let $d = 1$.

AIC is selected as the information criterion for in-sample model selection because forecasting is the main purpose of the model. If someone prefer simpler model, BIC can be used as the criterion. But for our group, we look at AIC only. We specify the maximum value of p and q to be 5 since we don't want a too complicated model. Then we want to choose the model with combination of p and q $(d = 1)$ that generates the lowest value of AIC.

The most appropriate in-sample model is ARIMA $(p = 3, d = 1, q = 2)$. The following graph and table provide us with some useful information:

Figure-5: Observed Value vs. Fitted Value



ARIMA (3,1,2) Coefficients Summary					
	ar1	ar2	ar3	ma1	ma2
Coefficients	-0.4614	-0.7115	-0.2081	0.1294	0.5777
s.e.	0.1079	0.1102	0.0368	0.1088	0.1043

In "Figure-5", the blue solid line is the observed value and the red dashed line is the fitted value. They are quite close to each other so that it is hard for us to visually tell the difference. It seems that the model fits the in sample data well.

But the model that fits in sample data well may not provide good out-of-sample prediction. One potential reason can be the over-fitting problem. It occurs when we fit the in-sample data too well, leaving less flexibility for the model, so that the out-

of-sample performance is not that satisfactory. In order to tackle this problem, our group came up with two approaches.

<**Approach 1**>

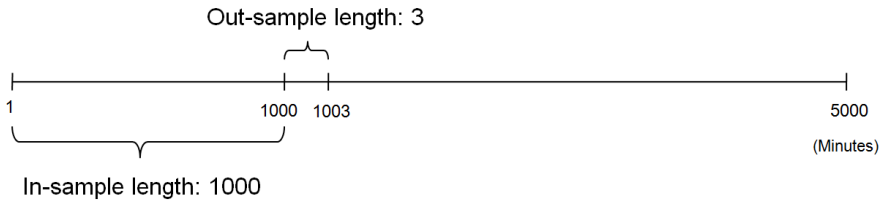
Since we have 5000 observations, we can use the first 4900 observations to fit the in-sample model and leave the rest 100 data as the out-of-sample data. For each combination of p and q, we can fit ARIMA $(p, d = 1, q)$ model to the log price by using the first 4900 data. Then we can produce 100 steps ahead forecast based on this model. Since we have 100 out-of-sample data at hand, we can measure the difference between the observed value and predicted value. By squaring these 100 differences, sum them up and take the average, we can get Mean Squared Forecast Error (*MSFE*) for the specific combination of p and q. We will choose the model with the lowest *MSFE* so that it has best out-of-sample performance.

ARIMA (0,1,4) Coefficients Summary				
	ma1	ma2	ma3	ma4
Coefficients	-0.3334	0.0158	0.0308	0.0303
s.e.	0.0142	0.0148	0.0155	0.0148

Based on this method, ARIMA $(p = 0, d = 1, q = 4)$ is chosen as the most appropriate model. This is different from the previous in-sample (use all 5000 observations) best model, which is ARIMA $(p = 3, d = 1, q = 2)$.

<**Approach 2**>

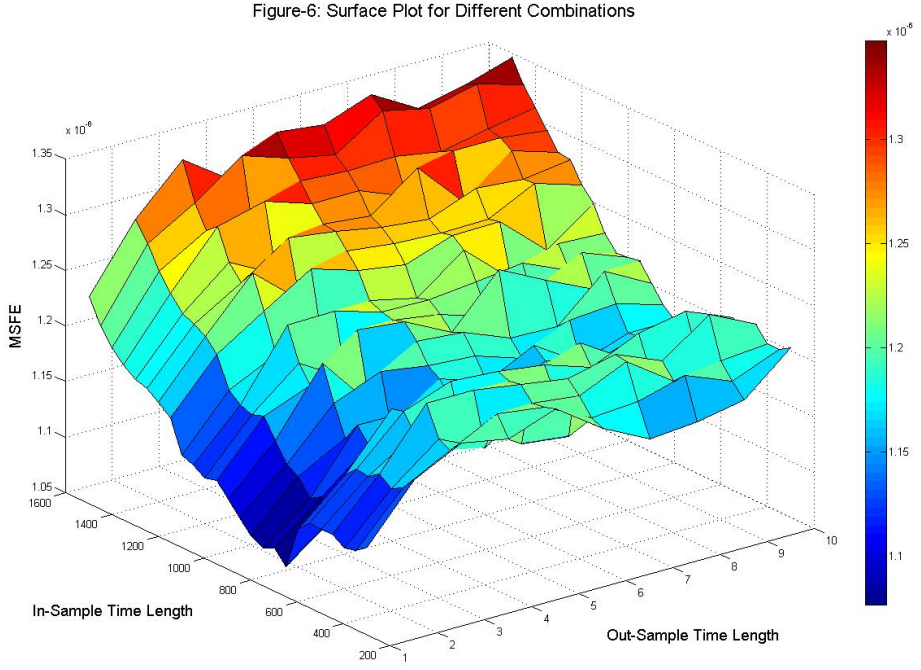
This approach tries to figure out the most appropriate combination of in-sample length and out-sample length, which generates the smallest MSFE.



Here is a simple example about how this method works: suppose now the in-sample length is set at 1000 and out-sample length is set at 3. Then we will use the first 1000 data to find out the most appropriate in-sample ARIMA model for log price, according to the AIC information criterion. After we get the result, we will use this model to predict 3 (out-sample length) steps ahead, then we have the three values of forecast error (difference between observed and predicted value). Afterwards, keeping the in-sample length constant, we move the time frame 3 minutes forward, so we will use observation 4 to 1003 as the in-sample data to find out the best ARIMA, which will be used to produce 3 steps ahead prediction so that we can get another 3 forecast error. We keep doing this until we get all out-sample forecast errors, based on which we can calculate a single value of MSFE.

For different combination of in-sample length and out-sample length, we have a MSFE. We will choose the combination that gives us the smallest value of MSFE.

Figure-6: Surface Plot for Different Combinations



The above surface plot gives us some interesting intuition. The in-sample length runs from 300 to 1500, with an increment of 50, the out-sample length runs from 1 to 10 (increment is 1). The Z-axis is the MSFE value for different combinations. If we fix the in-sample length and focus on the out-sample length, we can find out that the MSFE generally follows an upward trend. This is reasonable because to forecast 10 steps ahead normally will be less precise than only forecast 2 or 3 steps ahead. On the other hand, if we fix the out-sample length and focus on the in-sample length, we can see that it follows a V-shape. The smallest MSFE happens at the in-sample length of 650 and out-sample length of 1.

<**Model Checking**>

Figure-7: Auto-Correlation Plot of Residual

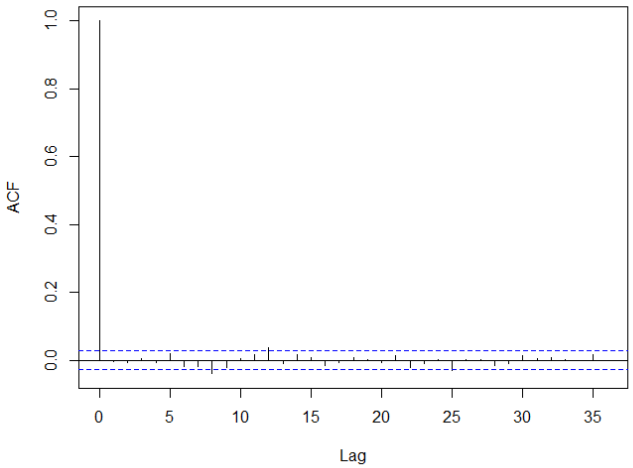
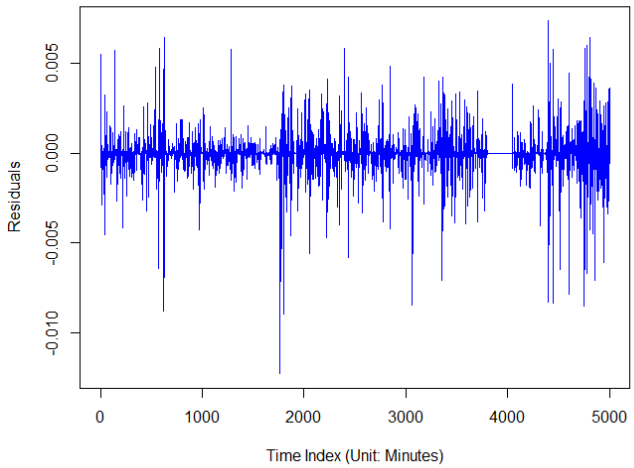


Figure-8: Residual Plot



We focus on checking model assumptions for ARIMA ($p = 3, d = 1, q = 2$), which is the best model for total in-sample 5000 observations (according to AIC). "Figure-7" shows that autocorrelation under different lags are small. In addition, Ljung-Box Test tells us that the first 35 lag autocorrelations among the residuals are zero ($Pvalue = 0.1697$). "Figure-8" shows that residual fluctuates around mean zero, indicating zero mean assumption of White Noise process is satisfied. However, the Breusch–Pagan test gives us the result that the volatility is not constant over time ($Pvalue \approx 0$). Therefore, the constant variance assumption of White Noise process is violated. In addition, the volatility clustering property is obvious. Therefore, we can use GARCH model to depict the error as a random process with zero conditional mean but inconstant conditional variance.

We focus on modeling log return (because it is stationary) by incorporating ARMA(p_A, q_A) model with GARCH(p_G, q_G). The model selection criterion is AIC and the maximum number of p_A, q_A, p_G and q_G are all set to 3. We ends up with a ARMA(0,1)/GARCH(1,1) model:

ARMA (0,1)/GARCH (1,1) Coefficients Summary

	Estimate	Std. Error	t value	Pr(>|t|)	
mu	-1.58E-05	7.31E-06	-2.155	0.0312	*
ma1	-3.20E-01	1.87E-02	-17.081	< 2e-16	***
omega	4.86E-08	6.47E-09	7.506	6.11E-14	***
alpha1	1.10E-01	9.80E-03	11.218	< 2e-16	***
beta1	8.52E-01	1.37E-02	62.201	< 2e-16	***

Conclusion

In this project, our group look at Bitcoin price through multiple regression and time series ways. The main focus is on the log return. In order to find out a suitable multiple linear regression model, data frequency is adjusted to hourly. In time series analysis part, minute-by-minute data is used. log price is fitted to different ARIMA models. Over-fitting problem is considered. At last, log return is modeled by combining ARMA and GARCH models.

References

[1] Satoshi Nakamoto, “Bitcoin: A Peer-to-Peer Electronic Cash System”, 2008
https://bitcoin.org/bitcoin.pdf
[2] Jerry Brito and Andrea Castillo, "Bitcoin: A Primer for Policymakers", Mercatus Center. George Mason University, 2013
http://mercatus.org/sites/default/files/Brito_BitcoinPrimer.pdf
[3] Python code written by our group:
https://github.com/Jeff88Ho/Bitcoin-Analysis/blob/master/Data_collection.py