Bitcoins Stock Return Analysis Based on ARMA and GARCH Models

Shirley Zhang

University of Wisconsin Madison

2019.12.09

Content

Summary

Virtual currency is getting increasingly more attention of the era. The news about releasing digital currency is endless, led by China's Central Bank and Facebook. One of the most popular virtual currency is Bitcoin, which is generated by the calculation of network nodes and is a virtual encrypted digital currency in the form of P2P. However, with the decentralization goal of the Bitcoin itself and the emergence of enormous speculators in the Bitcoin stock market, its stock price is fluctuating like a roller coaster, ranging from the largest 19497.2, which happened on Dec 16th, 2017 and all the way down to the lowest trough on Feb 7th,2019, when the stock price is only 3399.47. Hence, whether can find an adequate model and give prediction to the Bitcoin stock price is also the goal of our article.

The dataset I use is from https://finance.yahoo.com/quote/BTC-USD/history. The data contains daily data of open price, close price, the high price and low price, the adjusted close price and the volume of the Bitcoin stock. I only employ the adjusted close price variable. The time lag is from 2014-10-31 to 2019-10-31. Transforming the adjusted close price into log returns r_t, I first give basic statistic summary of the $r_t$and test the stationarity of them.

After identifying the log returns to be stationary via ADF,PP and KPSS tests but not white noise via ACF,PACF plots and Ljung-Box test, the first procedure is to fit ARMA model and filter out the autocorrelation. However because the complexity of the stock price, the order of the ARMA model is high, selected by AIC criterion and lead to model simplification. After adding and deleting nonsignificant coefficients and sorted by AIC criterion, finally gain the mean model ARMA((1,3,5),(1,3,5,6,10)). However, the residuals have ARCH effect and therefore different GARCH(1,1) models, including IGARCH, TGARCH, EGARCH and GARCH-M with different distributions of the standardized residuals are considered.

Then, considering diagnostic performance based on some diagnostic plots , Ljung-Box and ARCH-LM tests, AIC criterion and simplicity these aspects, I finally get the best the model is ARMA((1,3,5),(1,3,5,6,10))+GARCH(1,1) with standardized residuals following student's t distribution. Drawing some fitting plots, I confirm that this model gives fantastic performance.

Next, based on the final model, I make dynamic prediction out-of-sample or forward forecasting for 60 days. To begin with I only use the ARMA((1,3,5),(1,3,5,6,10)) model to do forecasting but find the result is not perfect. Therefore, I also add the volatility effect estimated from GARCH(1,1) when constructing the predictive confidence interval.

Finally, according to the forecast results I give some trading strategies separately to investors and speculators. The strategies are little different because of the variance in the aversion of risk. However, the main idea is to first consider the positive or negative sign predicted from ARMA((1,3,5),(1,3,5,6,10)) model the next day and then consider the volatility predicted from GARCH(1,1) the next day. As the day moves on, the model should be updated after each prediction. Besides, I also give insights to the last 60 day's out-of-sample forecast and get some important aspects that might influence and cause great plunge to the Bitcoin price via comparing the VaR and the log returns. The two particular points did have great events happening on those two days. The big event "Libra", the officially releasing of the quantum computer related papers from Google, the futures' exchanging rate and many other aspects will make a variance to the return of the investors. Thus, keep abreast of the news from those distinguished Internet companies because their news will usually have a great impact on the market behavior for digital currencies.

Because there are so many aspects that could influence the stock price of Bitcoins, I try to find how my model will be applied in other related fields such as machine learning and deep learning, which are really hot topics these days and will give great performance for dataset involved complexed factors. I discover that these models might consider more specifically of the above influence and could be improved by importing and optimizing the features. Besides, the volatilities could be used when constructing the labels and here the volatilities fitted from our GARCH model could be applied. This also show the great meaning of my model from another aspect.

Bitcoins Stock Return Analysis Based on ARMA and GARCH Models

I. Introduction

These days, Virtual currency is getting more and more attention of the times. In August this year, the People 's Bank of China began to frequently announce the launch of digital currencies. The digital currency of the People 's Bank of China is rapidly advancing. Besides, Facebook has published a statement of releasing the digital currency Libra, which will be linked to US currency and Treasury bonds.

One of the most popular virtual currencies is Bitcoin. The concept of Bitcoin was originally proposed by Satoshi Nakamoto on November 1, 2008, and was formally born on January 3, 2009. [1] Designed and released open source software and constructed a P2P network based on Satoshi Nakamoto's ideas, bitcoin does not have a centralized issuer, but is generated by the calculation of network nodes. Bitcoin is a virtual encrypted digital currency in the form of P2P. Peer-to-peer transmission means a decentralized payment system.

Bitcoin's natural nature is decentralized, comparing to the digital currency central banks, which must adhere to a centralized management model when implementing decentralization. Hence, the price of Bitcoin is highly volatile. Due to the involvement of a large number of speculators, the price of bitcoin for cash has fluctuated like a roller coaster, which makes Bitcoin more suitable for speculation rather than anonymous transactions.

In this article I try to research on the stock price of Bitcoin. Since 2014, the digital currency has remained relatively stable and hence I extracted the Bitcoin stock data from October 31st ,2014 to October 31st ,2019 totally 6 years and do time series analysis.

The digital currencies of mankind have experienced the earliest credit era of commodities, evolved into the era of electronic money later, and then to the era of decentralized money. In the era of Bitcoin, mining is used to generate confirmation of the POW consensus mechanism. It's interesting to catch the pace of the era and give finance insights of Bitcoin stock price.

II. Preliminary Analysis

A. Stock Price and the Log Returns Stat Summary.

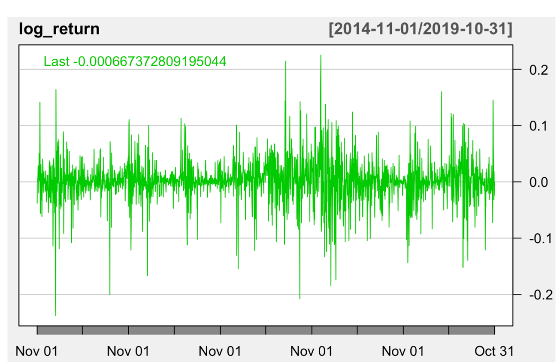

Figure 1: (a) Adjusted closing prices S_t for Bitcoin stock from Oct 31st ,2014 to Oct 31st,2019. (b) Log returns r_t for the Bitcoin stock from Nov1st,2014 to Oct 31st,2019.

	Length	*mean*	*sd*	*median*	*min*	*max*	*range*	*skew*	*kurtosis*
S_t	1827	3709.78	3965.06	1421.6	178.1	19497.2	19319.3	1.05	0.38
r_t	1826	0	0.04	0	-0.24	0.23	0.46	-0.3	5.28

Table 1: The statistical summary of the Bitcoin stock prices S_t and the log returns r_t.

To begin with, log return usually gives better interpretation when analyzing finance time series. Hence, I computed the return $R_t = \frac{P_t}{P_{t-1}}$ and changed the return into log return scale $r_t =$

log (R_t). From the Bitcoin stock price , it could be easily identified the price has great fluctuation. There is an obvious peak at around the mid December in 2017. The largest stock price is 19497.2, which happens on Dec 16th, 2017. However, after this peak, the stock price goes all the way down to the lowest trough on Feb 7th,2019, when the stock price is only 3399.47. In the whole picture, we could find the range of the price (19319.3) is very large in these 6 years.

As for the log return, we could find although the mean is 0 but the kurtosis is very large, even greater than 5, indicating the great leptokurtosis and fat-tail feature. This also illustrates there are many extreme return values concentrating at the lowest and highest sides.

B. Log Returns Stationary Analysis.

There are three common tests for testing the stationarity of the series. Augmented Dickey-Fuller (ADF) Test, Phillips-Perron Unit Root (PP) Test and Kwiatkowski-Phillips-Schmidt-Shin (KPSS) Test. The first two null hypothesis is not stationary and the last one is stationary. If any one of the test indicates non-stationarity, the series has unit root.

	ADF	*PP*	*KPSS*
p-value	0.01	0.01	0.1

Table 2: ADF, PP and KPSS stationarity tests for log returns r_t.

From Table 2, the first two p-values are smaller than the significance level 0.05 and therefore reject the null hypothesis that the series is not stationary. The p-value of KPSS test is greater than 0.05 won't reject the null hypothesis that the series is stationary. Hence, all the three tests suggest that the series is stationary. Next, I construct ACF and PACF plot of the log returns.

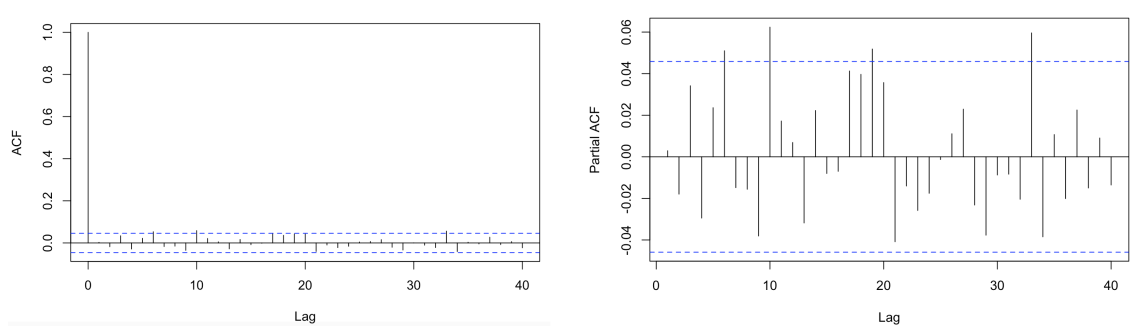

Figure 2: (a) ACF plot for log returns. (b) PACF plot for log returns.

Ljung-Box Test for r_t

Lag	1	8	9	10	11	12
p-value	0.9006	0.1837	0.1368	0.0300	0.0361	0.0534

Table 3: Ljung-Box test for log returns r_t.

From Figure 2, at some lag points, the correlation value exceeds the dash line, especially in the PACF plot. Therefore, autocorrelation might exist in this series. This could be confirmed by Ljung-Box(LB) test. The null hypothesis of LB test is that there is no correlation in the series. In Table 3, when the lag equals 10 or 11, p-values are around 0.03 and smaller than 0.05. Therefore, we could reject the null hypothesis and there is some autocorrelation in the log returns. Next, I construct some ARMA model to squeeze out the correlation.

III. Model Fitting and Model Selection

A. ARMA Model for Log Returns

Autoregressive moving average (ARMA) model , is an important method for studying time series. It is composed of an autoregressive (AR) model and a moving average (MA) model. It has the formula:

$$r_t = \sum_{i=1}^{p} \phi_i r_{t-i} + a_t - \sum_{i=1}^{p} \theta_i a_{t-i} \qquad where\ \ a_t \sim i.i.d\ WN(0,\sigma^2)$$

Because Bitcoins series have great fluctuation and there is a complexed pattern in the log returns series, I try many different orders for ARMA(p,q) model. When the order or AR and MA component are smaller than 5, the series would still have some correlation. Besides, it's hard to extract out all the information especially when later constructing GARCH model for the residuals. There is always great autocorrelation in the standardized residuals. This might because each time point, it will have great impact to the later reaction of the investors and speculators even after a very long period. Hence, I first consider high order ARMA model and then reduce the insignificant coefficients to simplify the model.

AR / MA	0	1	2	3	4	**5**	6	7	8	9	10
0	-6661.8	-6659.8	-6658.4	-6658.6	-6658.1	-6657.2	-6660.0	-6658.3	-6656.8	-6657.5	-6662.6
1	-6659.8	-6657.8	-6658.2	-6657.7	-6656.4	-6656.6	-6660.3	-6656.3	-6655.1	-6659.8	-6661.2
2	-6658.5	-6658.1	-6659.4	-6655.3	-6662.3	-6663.8	-6662.5	-6660.7	-6658.7	-6659.0	-6659.5
3	-6658.5	-6657.3	-6655.0	-6657.6	-6658.3	-6662.6	-6663.0	-6663.1	-6661.4	-6660.8	-6657.6
4	-6657.8	-6655.9	-6662.2	-6654.2	-6656.4	-6660.6	-6661.0	-6660.0	-6657.5	-6658.7	-6658.0
5	-6656.4	-6656.0	-6663.8	-6662.5	-6663.5	-6658.8	-6661.7	-6659.6	-6658.0	-6656.7	-6664.5
6	-6660.7	-6660.8	-6662.8	-6664.6	-6662.7	-6661.0	-6659.7	-6657.1	-6656.0	-6656.1	-6665.4
7	-6659.3	-6657.7	-6660.9	-6664.8	-6660.6	-6659.9	-6658.0	-6663.4	-6663.7	-6653.6	-6665.1
8	-6657.7	-6655.3	-6670.0	-6662.8	-6658.7	-6658.3	-6654.8	-6661.4	-6663.1	-6657.6	-6663.9
9	-6658.0	-6659.7	-6659.7	-6660.8	-6658.8	-6656.6	-6658.6	-6657.3	-6657.6	-6652.7	-6661.5
10	-6664.0	-6663.1	-6661.9	-6667.6	-6666.5	**-6670.3**	-6662.2	-6665.1	-6664.0	-6665.8	-6660.4

Table 4: AIC values for ARMA(p,q) model where p,q are integers from 0 to 10.

Considering the p and q from 1 to 10 and compute AIC for each combination of p and q. We could get 100 AIC value and make it into a 10x10 matrix. The AIC values' table are shown in Table 4. From the table we could identify that when p=5,q=10, the ARMA(5,10) fits the dataset best with the smallest AIC,-6670.305. However not all the coefficients are significant. I then delete those most nonsignificant coefficients and compute their AIC. If the AIC goes smaller this variable won't be added back and if AIC goes larger I will add the coefficient back. Then I find the even smaller AIC. The Sparse coefficient ARMA models construction process and AICs are in Table 5.

Sparse coefficient ARMA models		
Model	Procedure	AIC
ARMA((1,2,3,4,5),(1,2,3,4,5,6,7,8,9,10))	Delete ar4,ma4,ma9 (3 most nonsignificant)	-6673.9
ARMA((1,2,3,5),,(1,2,3,5,6,7,8,10))	Smaller AIC, delete ma8	-6674.7
ARMA((1,2,3,5),(1,2,3,5,6,7,10))	Smaller AIC, delete ma7	-6676.0
ARMA((1,2,3,5),(1,2,3,5,6,10))	Smaller AIC, delete ma10	-6674.1
ARMA((1,2,3,5),(1,2,3,5,6))	Larger AIC, add ma10 back and delete ar2	-6665.0
ARMA((1,3,5),(1,2,3,5,6,10))	Smaller AIC, delete ma2	-6675.2
ARMA((1,3,5),(1,3,5,6,10))	**Smaller AIC,(smallest), delete ma10**	**-6677.1**
ARMA((1,3,5),(1,3,5,6))	Larger AIC , add ma10 back, delete ma6	-6663.7
ARMA((1,3,5),(1,3,5,10))	Larger AIC compare to not delete ma6	-6673.4

Table 5: Steps and the corresponding AIC values for constructing Sparse coefficient ARMA models beginning from ARMA(5,10) model.

From Table 5, the best model is ARMA((1,3,5),(1,3,5,6,10)). After this model, deleting either the nonsignificant coefficient ma6 or ma10 will lead to the increasing of AIC. The model is:

$$\begin{aligned} r_t - 0.0018 &= 0.3211(r_{t-1} - 0.0018) - 0.5999(r_{t-3} - 0.0018) - 0.3256(r_{t-5} - 0.0018) \\ &+ a_t - 0.309a_{t-1} + 0.6340a_{t-3} + 0.3467a_{t-5} + 0.0617a_{t-6} \\ &+ 0.0924a_{t-10} \qquad where\ \ a_t \sim i.i.d\ WN(0,\sigma^2) \end{aligned}$$

Then again I check the ACF, PACF plots for the residuals a_t of ARMA((1,3,5),(1,3,5,6,10)).

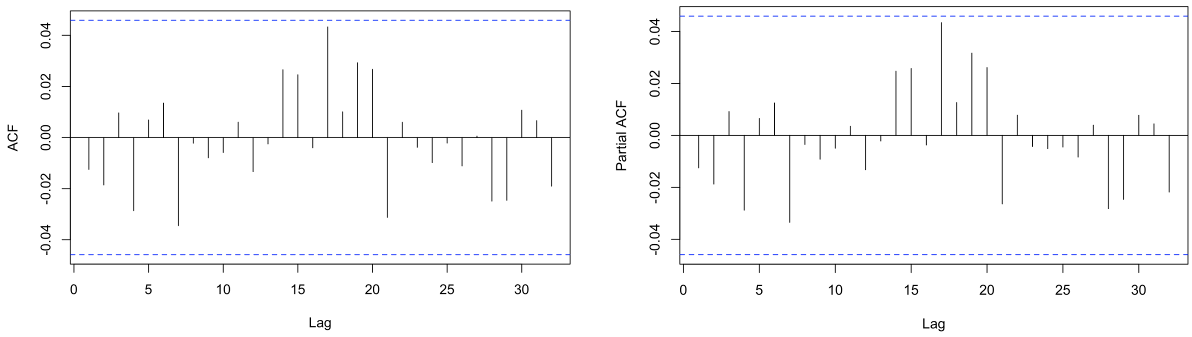

Figure 3: (a) ACF plot for residuals a_t of ARMA((1,3,5),(1,3,5,6,10)) model. (b) PACF plot for residuals a_t of ARMA((1,3,5),(1,3,5,6,10)) model.

Ljung-Box Test for a_t

Lag	1	8	9	10	11	12
p-value	0.5935	0.7362	0.8061	0.8646	0.9078	0.9272

Table 6: Ljung-Box test for residuals a_t of ARMA((1,3,5),(1,3,5,6,10)) model.

This time, after using the sparse coefficient ARMA model, from Figure 3 we could guarantee that there’ no much autocorrelation in the residuals now. Also for the Ljung-Box Test (Table 6), all the p-values are now greater than 0.05 and also confirms the assumption. Now, the next step is to check whether there is ARCH effect in the residuals.

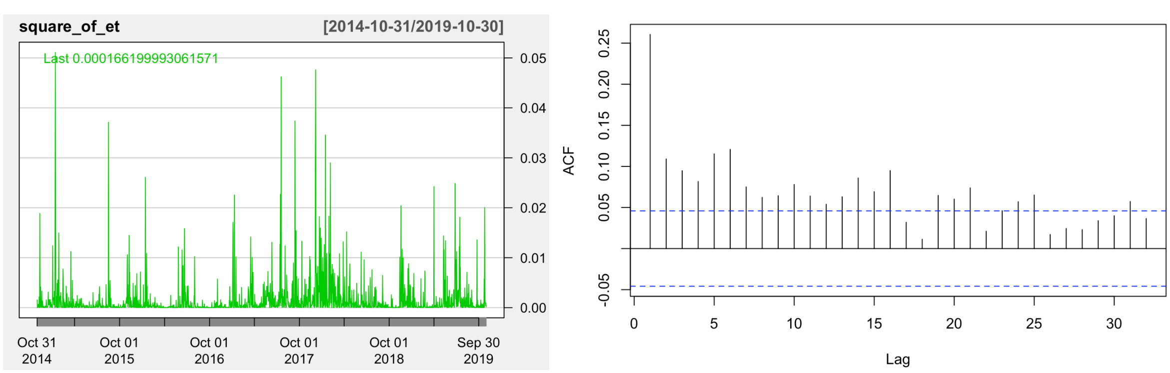

Figure 4: (a) The time series plot of ${a_t}^2$, square of the residuals of ARMA((1,3,5),(1,3,5,6,10)) model. (b) ACF plot for ${a_t}^2$.

Ljung-Box Test for ${a_t}^2$

Lag	1	8	9	10	11	12
p-value	<0.0001	<0.0001	<0.0001	<0.0001	<0.0001	<0.0001

Table 7: Ljung-Box test for ${a_t}^2$, square of the residuals of ARMA((1,3,5),(1,3,5,6,10)) model.

To test whether there’s ARCH effect in residuals, I plot the series of the square of the resiudals. From Figure 4, the time series plot has great clustering phenomenon and therefore it seems that after fitting the ARMA model the residuals are not independent. Also, from the ACF plot we could find at many lag points the autocorrelation value is beyond the dash line. In Table 7, all the p-values of the Ljung-Box test for the square of the residuals are much smaller than 0.05. There must be great ARCH effect in the residuals. Next, I fit different GARCH(1,1) models to the residuals.

B. GARCH models for residuals of ARMA model

1. GARCH(1,1) with Normal Standardized Residuals

In many practical problems, as time t changes, the conditional variance of the random perturbation term of the sequence $\{a_t\}$ also changes, that is, the sequence has the characteristic of variable variance. Engel first proposed the ARCH model for variance modeling in 1982 to describe the volatility clustering and persistence of the stock market.[2]

For the generalization, I choose GARCH (1,1) model, which performs well to the residuals of ARMA model. The formula of my model becomes:

$$\begin{cases} r_t = \mu_t + a_t \\ a_t = \sigma_t \varepsilon_t \\ \sigma_t^2 = \alpha_0 + \alpha_1 \varepsilon_{t-1}^2 + \beta_1 \sigma_{t-1}^2 \end{cases}$$

where μ_t is our ARMA model and a_t is the residual series. $\varepsilon_t \sim i.i.d\ N(0,1)$. $\alpha_1 + \beta_1 < 1$.

After fitting the GARCH (1,1) model I make the some diagnostic plots to check the model assumptions.

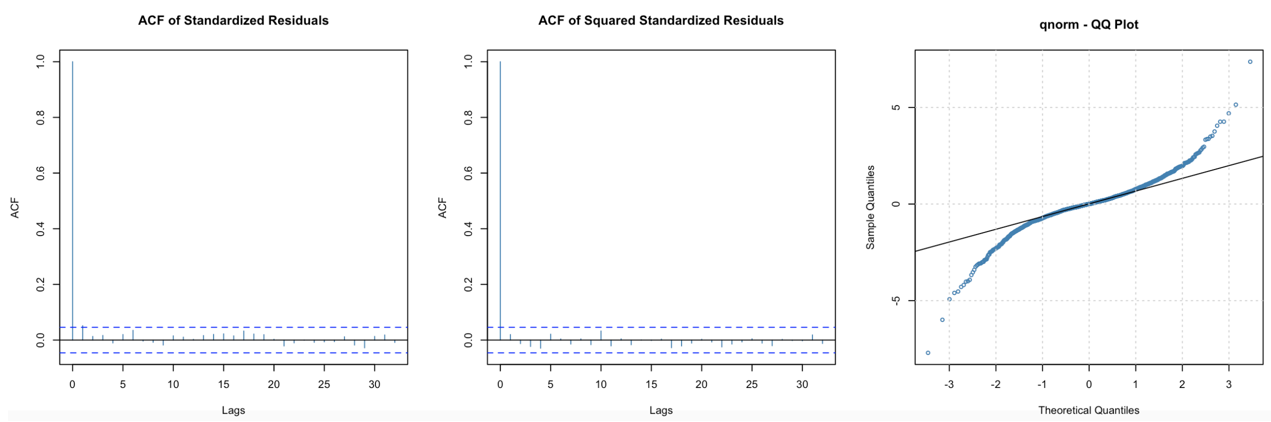

Figure 5: (a) The ACF plot of standardized residuals ε_t from GARCH(1,1) model with normal distribution assumption. (b) The ACF plot of the square of standardized residuals, ${\varepsilon_t}^2$. (c) Normal Q-Q plot for the standardized residuals ε_t.

From Figure 5, the ACF plot of the standardized residuals and the squared standardized residuals are appropriate and indicates that ε_t are independent. However, in the the Normal-QQ plot the points are in a curvature shape, this is due to the fact that the distribution of r_t has the feature of leptokurtosis and thick-tail. Hence, I change the assumption that the ε_t follows student t distribution.

2. GARCH(1,1) with Student's t Standardized Residuals

After changing the assumption of distribution for standardized residuals to be t distribution, the diagnostic plots are shown in Figure 6.

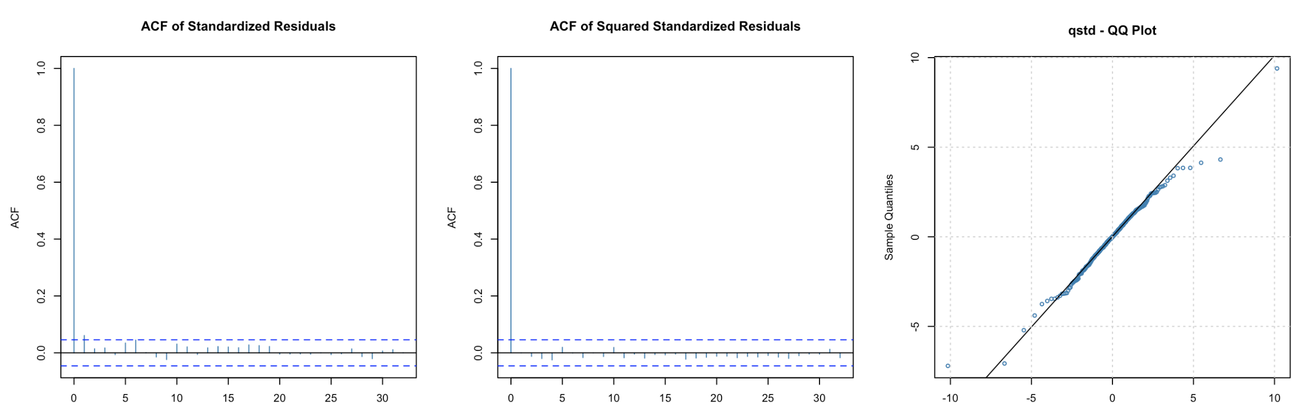

Figure 6: (a) The ACF plot of standardized residuals ε_t from GARCH(1,1) model with student's t distribution assumption. (b) The ACF plot of the square of standardized residuals, ${\varepsilon_t}^2$. (c) Student's t distribution Q-Q plot for the standardized residuals ε_t.

This time the two ACF plots are still appropriate. As for the Student's t Q-Q plot, the points are almost in a line and therefore GARCH (1,1) with student's t distribution standardized residuals is better.

3. Other Extended GARCH(1,1) Models

These days, different kinds of GARCH models appeared and are used by scholars, such as IGARCH, TGARCH, EGARCH, and GARCH-M. The sum of the coefficients for GARCH(1,1) model is very close to 1 (Table 8) and therefore I first try IGARCH(1,1) model to see if this model fits better. After that I also fit other extended GARCH(1,1) models and compare their AIC and BIC to choose the best model. According to the data property, the distribution of the standardized residuals for all these models I choose is student t distribution.

	AIC	BIC
GARCH(1,1)	-4.1675	-4.1524
IGARCH(1,1)	-4.1689	-4.1568
TGARCH(1,1)	-4.1828	-4.1647
EGARCH(1,1)	-4.1828	-4.1647
GARCH-M(1,1)	Failure to converge	

Table 8: AIC and BIC values for different types of GARCH(1,1) models.

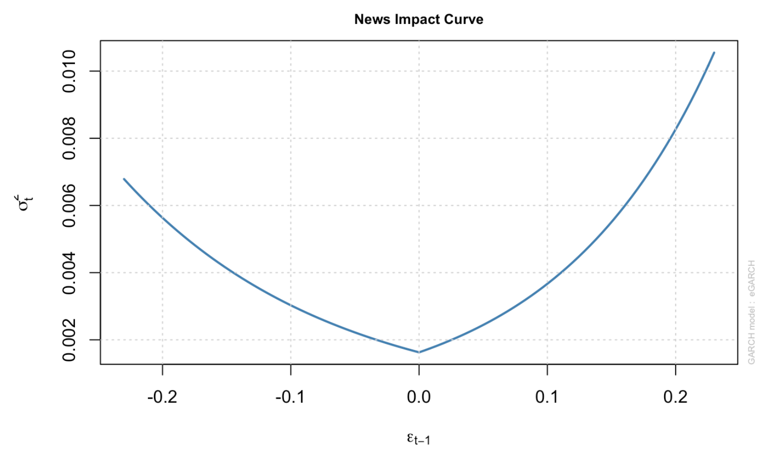

Figure 7: The news impact curve for TGARCH(1,1)model.

According to the AIC and BIC table we could identify the difference between these AIC and BIC values are smaller than 0.02 . Besides the news impact curve (Figure 7) shows little difference between the effect of the positive return news and negative return news. The good news has litter larger impact on the volatility. Hence, considering the model simplicity I choose GARCH(1,1) model.

4. More Statistical Tests for Diagnostics

To ensure this model is a good candidate, after selecting GARCH (1,1) model with student t standardized residuals, I use some other common tests to check the model accuracy. The most common one is the Eagle's ARCH test, also known as the ARCH-LM test to test whether the residuals still have ARCH effect. The null hypothesis is that $H_0: \alpha_0 = \alpha_1 = \cdots = \alpha_m = 0$. The alternative hypothesis is that $H_a: \varepsilon_t^2 = \alpha_0 + \alpha_1\varepsilon_{t-1}^2 + \cdots + \alpha_m\varepsilon_{t-m}^2$. [3] That is to say there is no auto correlation in the squared residual. I do this test and the p-value is 0.8109>0.05 and therefore could not reject H_0. That is to say there's no ARCH effect then.

Besides Ljung-Box test can still be used to test the autocorrelation of residuals and the p value for lag 10,15 20 are 0.0802, 0.1780 and 0.2417, all greater than 0.05. Hence our model indeed squeezes out the ARCH effect.

Thus my final model ARMA((1,3,5),(1,3,5,6,10))+GARCH(1,1) is :

$$r_t - 0.0018 = 0.3211(r_{t-1} - 0.0018) - 0.5999(r_{t-3} - 0.0018) - 0.3256(r_{t-5} - 0.0018) + a_t - 0.309a_{t-1} + 0.6340a_{t-3} + 0.3467a_{t-5} + 0.0617a_{t-6} + 0.0924a_{t-10}$$

$$a_t = \sigma_t \varepsilon_t$$
$$\sigma_t^2 = 0.000014 + 0.1284\varepsilon_{t-1}^2 + 0.8706\sigma_{t-1}^2$$

where $a_t \sim i.i.d\ WN(0, \sigma^2)$, $\varepsilon_t \sim i.i.d\ t_{3.359}$.

5. Model fitting

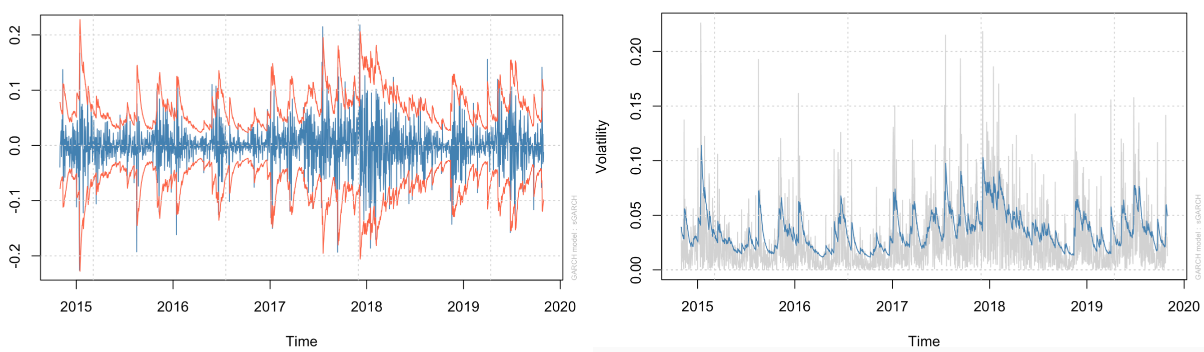

Figure 8: (a) Time series plot of residuals of ARMA model a_t (log return filtered out the mean part). The two red lines indicate 95% predictive interval based on the t model. (b) The blue line represents the estimated daily volatilities, the grey lines indicate the absolute value of residuals of ARMA model a_t.

Based on the model ARMA((1,3,5),(1,3,5,6,10))+GARCH(1,1) model, the mean ARMA part won't influence the volatility part and therefore I filter out the ARMA mean part and then based on the residuals a_t I get the two plots shown in Figure 8. Because the residuals almost lie in between these red lines, meaning the model is appropriate. From the volatilities vs the absolute value a_t plot, because the estimated daily volatilities are almost have the same trend as a_t. When a_t are at the peak, the estimated volatilities are also at the peak. When a_t fall down, the estimated volatilities also go down. That is to say, our model could give adequate fit to the volatility of the log return.

IV. Model Forecasting and Strategies Providing

A. Model Forecasting for mean ARMA part

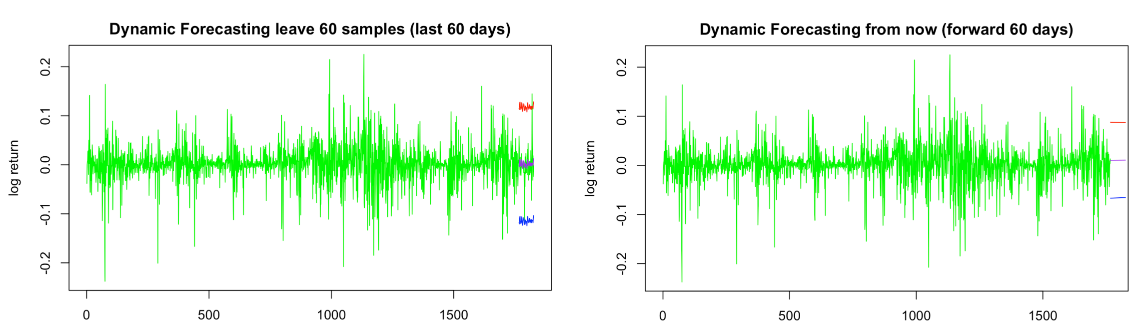

Figure 9: (a) Out sample dynamic forecasting for log returns r_t leaving last 60 days based on only ARMA ((1,3,5),(1,3,5,6,10)) model. The two red line and the blue line indicate upper bound and the lower bound of the 95% predictive interval. (b) Dynamic Forecasting for r_t from now to the forward 60 days based on ARMA ((1,3,5),(1,3,5,6,10)) model. The red line and blue line still indicate the 95% predictive interval band.

Out sample num	*MSFE*	*MAFE*	*Bias*
60	0.0011	0.0210	-0.0024

Table 9: MSFE, MAFE and Bias for the out-of-sample comparison. (leave 60 days)

For the mean ARMA model part, I make the dynamic forecasting. That is to say the coefficient of the ARMA model will change based on every prediction. The quantile here I choose 3 for computing the predict confidence interval because the data almost follow $t_{3.359}$

based on the GARCH model construction. The 95% quantile of distribution $t_{3.359}$ is 2.998418 , and round to 3. From Figure 9, the forward 60 days forecasting shows most log returns are in the confidence interval and therefore the ARMA model is appropriate. However, the volatilities are really small for these prediction and still some points might lie outside the predictive confidence interval bands. In Table 9, Mean square of forecast errors (MSFE), mean absolute forecast errors (MAFE)and bias are all very small, also illustrates great predicting performance of the ARMA model for the mean of our return. Next, I consider the volatilities of the log returns and add the effect to the ARMA model.

B. Model Forecasting for ARMA+GARCH

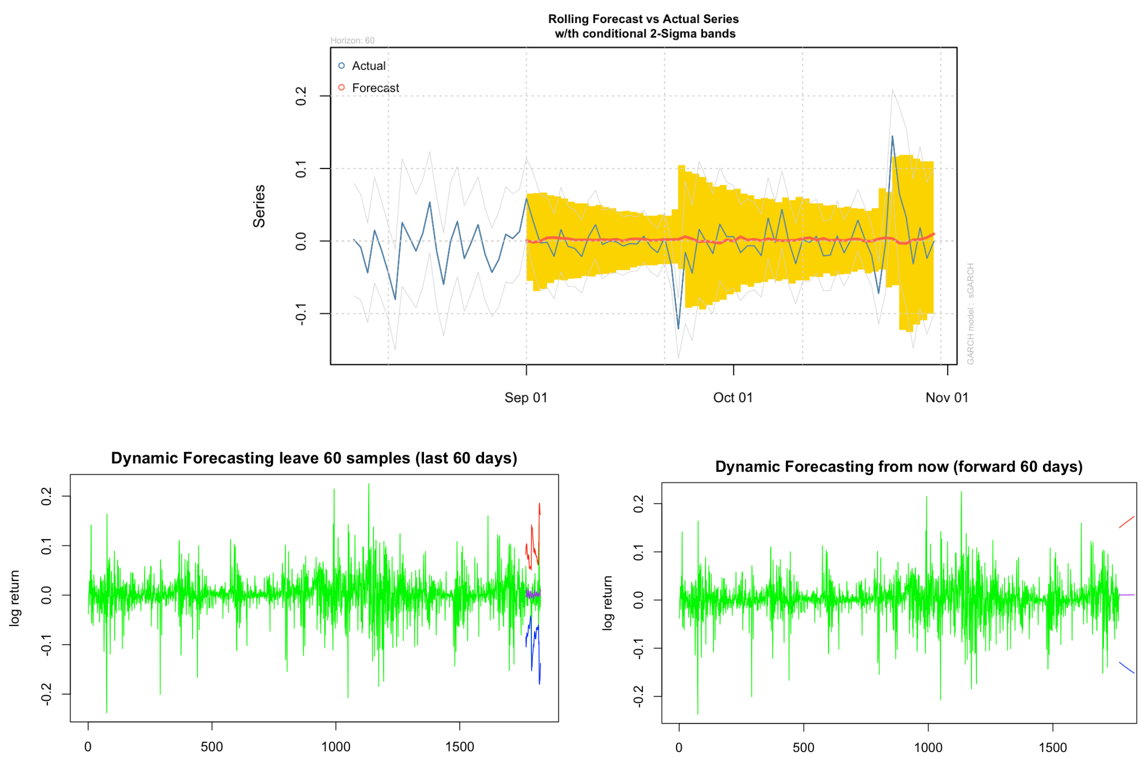

Figure 10: (a) Rolling Forecast vs Actual Series with conditional two standard deviation band. The red line indicates the forecast volatilities and the blue line represents the actual a_t series. The grey line indicates the two standard deviation band. (b). Out sample dynamic forecasting log returns $r_t$leaving last 60 days based on only ARMA ((1,3,5),(1,3,5,6,10)) +GARCH(1,1) model with t distribution standardized residuals. The two red line and the blue line indicate upper bound and the lower bound of the 95% predictive interval. (c) Dynamic Forecasting for $r_t$from now to the forward 60 days based on ARMA ((1,3,5),(1,3,5,6,10))+GARCH(1,1) model. The red line and blue line still indicate the 95% predictive interval band.

From the Rolling forecast for the volatilities σ_t, most of the log returns are in the 2-sigma bands and therefore the GARCH(1,1) model works well for predicting the volatilities. This time when constructing the predictive confidence interval, the prediction standard error I choose is from the prediction result of the volatilities from GARCH(1,1) model. From the forward forecasting, we could see a trend of increasing volatilities. For the last 60 days out of sample forecasting, this time the points that lie outside the prediction interval before are lying in the band now. There is a trend of increasing volatility at some time points and a decreasing volatility trend at other time points. Next, I will give trading strategies based on the ARMA+GARCH model.

C. Trading Strategies based on ARMA+GARCH model

1. Basic trading strategies for conservative investors.

1. First use the ARMA((1,3,5),(1,3,5,6,10)) +GARCH(1,1) model to fit the original dataset and make the prediction for the next day log return.

2. If the log return is below 0, it means the predictive return will below 1 and therefore will be at a losing money level tomorrow. The best way is to short your stock at today's closing time.

3. If the log return is above 0, it means the predictive return will be greater than 1 and therefore will be at a winning money level tomorrow. However, this time we could consider the volatility as well. If the volatility is small and within the investor's acceptability, he or she could long this stock. If the volatility is very large the investor could still consider short the stock.

4. After getting a new data (next day's data), we could fit again for the ARMA((1,3,5),(1,3,5,6,10))+GARCH(1,1) model and get new coefficients and update the model for next day's prediction and repeat the steps from 1 to 3.

2. Basic trading strategies for speculators.

1. First use the ARMA((1,3,5),(1,3,5,6,10)) +GARCH(1,1) model to fit the original dataset and make the prediction for the next day log return.

2. If the log return is below 0, it means the predictive return will below 1 and therefore will be at a losing money level tomorrow. The best way is to short your stock at today's closing time.

3. If the log return is above 0, it means the predictive return will be greater than 1 and therefore will be at a winning money level tomorrow. This time we could consider the volatility as well. If the volatilitys is within the investor's acceptability, he or she could long this stock. If the volatility is large the stock holder could even consider buy in more stocks because he will have a large chance to gain a high profit and a great return tomorrow. Besides from the news impact curve before we have known that the good news usually has a better impact than the bad news and therefore this could be consider a good time for spectators to take a try because if tomorrow the stock has a great performance it will quickly attract more investors and spectators to buy in and crowd in to the market.

4. After getting a new data (next day's data), we could fit again for the ARMA((1,3,5),(1,3,5,6,10)) +GARCH(1,1) model and get new coefficients and update the model for next day's prediction and repeat the steps from 1 to 3.

3. The insights based on the 60 rolling forecast.

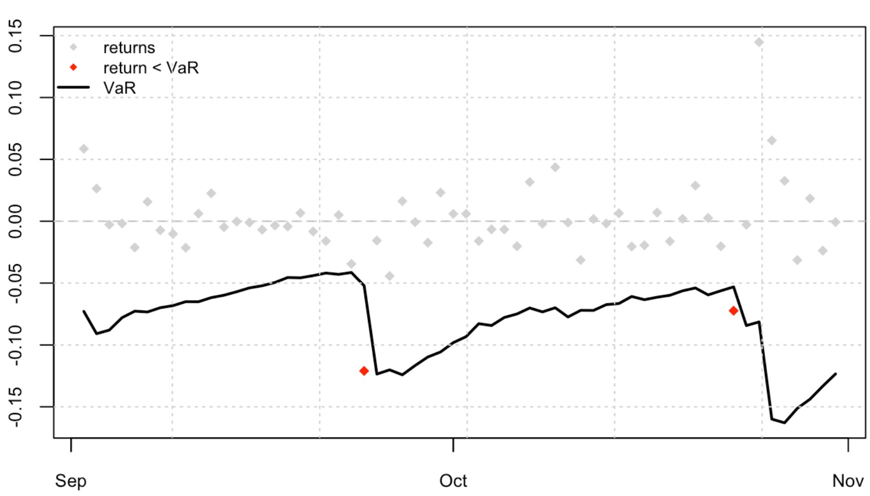

Figure 10: The plot of the log returns r_t versus VaR curve for the last 60 days rolling forecast.

From the 60 days forecast (last 60 days) based on our ARMA+GARCH model, there are 2 days, where the returns are below the VaR values, which should be paid great attention because this means that it's really dangerous to get into the market at these 2 days. These two days are September 24th and October 23rd.

On September 24, 2019, one day before the BTC crash, Emin G · n Sirer, a professor of computer science at Cornell University and a digital token expert, posted on Twitter that the BTC hash rate plummeted from over 100EH / s to 67.38EH / s. This data caused panic in the market, and people were worried that a problem in the BTC network led to the loss of a large number of miners. There were even rumors in China that the mines in Sichuan and other places will be inspected as a reason for the sharp decline in BTC computing power. [4]

On October 23rd ,2019, it was a miserable sorrow again. Bitcoin, represented by digital currencies, fell across the board, reaching a recent low. At around 8:40 last night, Bitcoin crashed. In just a few minutes, it plunged from around $ 7,900 to below $ 7,500, dropped to nearly $ 500, a drop of more than 9%, and hit a price low since June.

Four major reasons that could cause the plunge. Many people will think that the first reaction will be related to last night 's "big event" Libra. The second reason everyone guessed was that Google's quantum computer related papers were officially released. Last month, Google researchers published a quantum hegemonic paper on the NASA website, which was quickly withdrawn at that time. On October 23rd Google formally published the paper in the journal Nature, announcing a major breakthrough in the field of quantum computing, and officially achieving "quantum hegemony." Earlier, there were many remarks that quantum hegemony would pose a threat to Bitcoin, so some people think that this plunge has something to do with Google 's paper published yesterday. The third reason is the performance of Bitcoin and the Bitcoin network itself: on the one hand, the price performance of Bitcoin itself is not good enough to meet expectations; on the other hand, the liquidity of Bitcoin caused by it has reached a new low. Byte Tree CEO James Bennet also gave a more objective interpretation. He believes that the decline in the price of bitcoin is caused by the decrease in transaction speed. [5]

Hence, here I could give the suggestions that keeping abreast of the news from the well-known Internet companies because their news will influence the market behavior to a large extent.

Besides, the news impact and many other aspects such as the exchange rates, futures delivery day effect might also influence the price of the Bitcoin stock. Therefore, considering these aspects, some other related field that can give better performance of the prediction should be considered.

D. Other application to related field

The volatilities could be used in many other related fields. For instance, machine learning and deep learning are hot topics these days. Let's see how my fitted and predictive volatilities could be applied in the machine learning trading strategies.

We could use t+1, t+2,..., t+N, the average of the returns of these days, and based on the feature whether it's positive or negative, to construct labels y_t. (positive 1 and negative 0). This seems to be reliable. But if you think about it, if the average of these days' income is 0.02, but the standard deviation reaches 0.04, it seems to be a bit flustered. And if the average return on these days is 0.01, but the standard deviation is only 0.005, it seems that this is reliable. This reminds us that as a trend investor, in the turmoil, we are generally unreliable as duckweed (of course, turmoil is another opportunity for other traders to make a fortune). When we build y_t., we can't just think about the average of the benefits, but also consider its fluctuations. [6]

With this concept, we can rethink the setting of y_t. We record the average of the daily returns of t+1, t+2,..., t+N as r_t, and the standard deviation is σ_t. We consider the positive and negative of such a quantity $r_t - \lambda\sigma_t$.Here, λ is a pre-set constant that has a good financial perspective: it reflects the investor's aversion to risk. The bigger the λ, the stronger the aversion.

Then, here the estimated volatilities fitted from my GARCH(1,1) model could be applied here. The researchers could use these volatilities as σ_t and get the value of $r_t - \lambda\sigma_t$. Based on this new quantity we could again get y_t labels based on their positive and negative sign of $r_t - \lambda\sigma_t$.

Different kinds of signals such as such as MACD, RSI [7] and other features like exchange rate in the technical analysis of stocks can be regarded as X feature matrix. Then, throw the X and y_t into the neural networks to train a model and use the model for prediction. If the prediction is 1 then tomorrow will have positive return and investors could long the stock or even buy in more stocks. It the prediction is 0, then the investors could consider short the stock to limit the loss. This could be more reliable because the machine learning model might involve more complexed information and situations.

Reference

[1] "Statement of Jennifer Shasky Calvery, Director Financial Crimes Enforcement Network United States Department of the Treasury Before the United States Senate Committee on Banking, Housing, and Urban Affairs Subcommittee on National Security and International Trade and Finance Subcommittee on Economic Policy" (PDF). fincen.gov. Financial Crimes Enforcement Network. 19 November 2013. Archived (PDF) from the original on 9 October 2016. Retrieved 1 June 2014.

[2] https://en.wikipedia.org/wiki/Autoregressive%E2%80%93moving-average_model

[3] Engle, Robert F. "Autoregressive Conditional Heteroskedasticity with Estimates of the Variance of United Kingdom Inflation." *Econometrica*. Vol. 50, 1982, pp. 987–1007.

[4] https://www.chainnews.com/articles/156282387434.htm

[5] https://www.beekuaibao.com/article/628103526631923712

[6] https://medium.com/auquan/https-medium-com-auquan-machine-learning-techniques-trading-b7120cee4f05

[7] https://www.investopedia.com/ask/answers/122214/what-are-main-differences-between-moving-average-convergence-divergence-macd-relative-strength-index.asp

Appendix (Partial important R codes)

```
library(quantmod)
getSymbols("BTC-USD",from="2014-10-31",to="2019-10-31")
head(`BTC-USD`)
tail(`BTC-USD`)
BTC_adj=`BTC-USD`$`BTC-USD.Adjusted`
chartSeries(BTC_adj,theme="white")
log_return=diff(log(BTC_adj))[-1,]
chartSeries(log_return,theme="white")
return=log_return
library(psych)
describe(return)
describe(BTC_adj)
library(tseries)
adf.test(return)    ##daily return doesn't have a unit root
pp.test(return)          ## daily return doesn't have a unit root
kpss.test(return)
par(mfrow=c(1,2))
acf(return,lag=40)
pacf(return,lag=40)
Box.test(return,type="Ljung-Box")
Box.test(return,lag=8,type="Ljung-Box")
Box.test(return,lag=9,type="Ljung-Box")
Box.test(return,lag=10,type="Ljung-Box")
Box.test(return,lag=11,type="Ljung-Box")
Box.test(return,lag=12,type="Ljung-Box")
X<-NULL              ##Construct AIC table
for (i in 0:10){
  {for (j in 0:10)
    X=rbind(X,c(arima(return,order=c(i,0,j),include.mean = T)$aic,i,j))
  }
}
X
A=X[which.min(X[,1]),]
A
return.arima1=arima(return,order=c(5,0,10),transform.pars=F,fixed=c(NA,NA,NA,0,NA,NA,NA
,NA,0,NA,NA,NA,NA,0,NA,0))
return.arima1
var1=return.arima1$var.coef
sd1=sqrt(var1[row(var1)==col(var1)])
coef(return.arima1)/sd1
##ARIMA((1,2,3,5),0,(1,2,3,5,6,7,10))
return.arima2=arima(return,order=c(5,0,10),transform.pars=F,fixed=c(NA,NA,NA,0,NA,NA,NA
,NA,0,NA,NA,NA,0,0,NA,0))
return.arima2
var2=return.arima2$var.coef
sd2=sqrt(var2[row(var2)==col(var2)])
coef(return.arima2)/sd2
##ARIMA((1,2,3,5),0,(1,2,3,5,6,10))
return.arima3=arima(return,order=c(5,0,10),transform.pars=F,fixed=c(NA,NA,NA,0,NA,NA,NA
,NA,0,NA,NA,0,0,0,NA,0))
return.arima3
var3=return.arima3$var.coef
sd3=sqrt(var3[row(var3)==col(var3)])
coef(return.arima3)/sd3
##ARIMA((1,2,3,5),0,(1,2,3,5,10))
return.arima4=arima(return,order=c(5,0,10),transform.pars=F,fixed=c(NA,NA,NA,0,NA,NA,NA
,NA,0,NA,0,0,0,0,NA,0))
return.arima4
```

```
##ARIMA((1,2,3,5),0,(1,2,3,5,6))
return.arima5=arima(return,order=c(5,0,10),transform.pars=F,fixed=c(NA,NA,NA,0,NA,NA,NA
,NA,0,NA,NA,0,0,0,0,NA))                    #aic = -6664.99
##ARIMA((1,2,3,5),0,(1,2,3,5,6,10))
return.arima6=arima(return,order=c(5,0,10),transform.pars=F,fixed=c(NA,NA,NA,0,NA,NA,NA
,NA,0,NA,NA,0,0,0,NA,NA))                   #aic = -6675.19
##ARIMA((1,3,5),0,(1,2,3,5,6,10))
return.arima7=arima(return,order=c(5,0,10),transform.pars=F,fixed=c(NA,0,NA,0,NA,NA,NA,
NA,0,NA,NA,0,0,0,NA,NA))                    #aic = -6675.21
##ARIMA((1,3,5),0,(1,3,5,6,10))
return.arima8=arima(return,order=c(5,0,10),transform.pars=F,fixed=c(NA,0,NA,0,NA,NA,0,NA
,0,NA,NA,0,0,0,NA,NA))                    #aic = -6677.08  ##The smallest
##ARIMA((1,3,5),0,(1,3,5,6))
return.arima9=arima(return,order=c(5,0,10),transform.pars=F,fixed=c(NA,0,NA,0,NA,NA,0,NA
,0,NA,NA,0,0,0,0,NA))                    #aic = -6665.65
##ARIMA((1,3,5),0,(1,3,5,10))
return.arima10=arima(return,order=c(5,0,10),transform.pars=F,fixed=c(NA,0,NA,0,NA,NA,0,N
A,0,NA,0,0,0,0,NA,NA))                #aic = -6673.37
var8=return.arima8$var.coef
sd8=sqrt(var8[row(var8)==col(var8)])
coef(return.arima8)/sd8
at=residuals(return.arima8)
square_of_et=at^2
chartSeries(square_of_et,theme="white")
par(mfrow=c(1,2))
acf(at,lag=50)
pacf(at,lag=50)
Box.test(at,lag=1,type="Ljung-Box")
Box.test(at,lag=8,type="Ljung-Box")
Box.test(at,lag=9,type="Ljung-Box")
Box.test(at,lag=10,type="Ljung-Box")
Box.test(at,lag=11,type="Ljung-Box")
Box.test(at,lag=12,type="Ljung-Box")
acf(at^2)
pacf(at^2)
Box.test(at^2,type="Ljung-Box")
Box.test(at^2,lag=8,type="Ljung-Box")
Box.test(at^2,lag=9,type="Ljung-Box")
Box.test(at^2,lag=10,type="Ljung-Box")
Box.test(at^2,lag=11,type="Ljung-Box")
Box.test(at^2,lag=12,type="Ljung-Box")
library(fGarch)
model1=garchFit(formula = ~garch(1,1), data = at, trace = F,include.mean = F)
model2=garchFit(formula = ~garch(1,1), data = at, cond.dist="std",trace = F,include.mean = F)
model3=garchFit(formula = ~garch(1,1), data = at, cond.dist="sstd",trace = F,include.mean = F)
model4=garchFit(formula = ~garch(1,1), data = at, cond.dist="ged",trace = F,include.mean = F)
model5=garchFit(formula = ~garch(1,1), data = at, cond.dist="sged",trace = F,include.mean = F)
summary(model1)   #AIC=-3.895133 BIC=-3.886081
summary(model2)  #AIC=-4.177915 BIC=-4.165845  #student t distribution is the best
summary(model3)  #AIC=-4.176910 BIC=-4.161823
summary(model4)  #AIC=-4.159771 BIC=-4.147701
summary(model5)  #AIC=-4.158789 BIC=-4.143702
#GARCH(1,1) with Normal distribtion for at
par(mfrow=c(1,3))
plot(model1,which=10)
plot(model1,which=11)
plot(model1,which=13)
#GARCH(1,1) for at with t distribution
par(mfrow=c(1,3))
plot(model2,which=10)
plot(model2,which=11)
plot(model2,which=13)
```

```
##GARCH(1,1) with t distribution for at
t=ugarchspec(variance.model = list(model = "sGARCH",garchOrder = c(1, 1)), mean.model =
list(armaOrder = c(0, 0)),distribution.model ="std")
modelt=ugarchfit(at,spec=t)  #AIC=-4.1675  #BIC=-4.1524
plot(modelt,which=1,main="Conditional SD (vs |at|)")
plot(modelt,which=2,main="Conditional SD (vs |at|)")
plot(modelt,which=3,main="Conditional SD (vs |at|)")
# IGARCH model for at
library(rugarch)
spec1=ugarchspec(
mean.model = list(armaOrder = c(0, 0)),
variance.model = list(model = "iGARCH", garchOrder = c(1, 1)), distribution.model = "std")
m1=ugarchfit(at,spec=spec1)   #AIC=-4.1689 BIC=-4.1568
par(mfrow=c(1,3))
plot(m1,which=10)
plot(m1,which=11)
plot(m1,which=9)
##TGARCH(1,1) for at
spec3=ugarchspec(variance.model = list(model = "eGARCH",garchOrder = c(1, 1)), mean.model
= list(armaOrder = c(0, 0)),distribution.model ="std")
m3=ugarchfit(at,spec=spec3)   #AIC=-4.1828      BIC=-4.1647
m3
plot(m3,which=12)  ##news impact curve
##EGARCH(1,1) for at
spec4=ugarchspec(variance.model = list(model = "eGARCH",garchOrder = c(1, 1)), mean.model
= list(armaOrder = c(0, 0)),distribution.model ="std")
m4=ugarchfit(at,spec=spec4)   #AIC=-4.1828      BIC=-4.1647
##GARCH-M(1,1) for at
spec5=ugarchspec(variance.model=list(model="sGARCH", garchOrder=c(1,1)),
mean.model=list(armaOrder=c(0,0), archm=T, archpow=2),distribution.model = "std")
m5=ugarchfit(at,spec=spec5)   #AIC=-4.1828      BIC=-4.1647
##Model Forecasting
##For ARIMA((1,3,5),0,(1,3,5,6,10)) model
T1=length(return)
N=60
n=T1-N
pre=rep(0,N)
pre.se=rep(0,N)
for (i in 1:N){
m=arima(return[1:(n-
1+i)],order=c(5,0,10),transform.pars=F,fixed=c(NA,0,NA,0,NA,NA,0,NA,0,NA,NA,0,0,0,NA,N
A))
pre[i]<-predict(m)$pred
pre.se[i]<-predict(m)$se
}
plot(as.numeric(return)[1:T1],ylab="log return",type="l",col="green")
title("Dynamic Forecasting leave 60 samples (last 60 days)")
lines(x=(n+1):T1,y=pre,col="purple")
lines(x=(n+1):T1,y=pre+3.00*pre.se,col="red")
lines(x=(n+1):T1,y=pre-3.00*pre.se,col="blue")
error=as.numeric(return)[(T1-59):T1]-pre
MSFE=sum(error^2)/60
MAFE=sum(abs(error))/60
Bias=sum(error)/60
MSFE;MAFE;Bias
##Forward 60 days forecasting
T1=length(return)
N=60
pre2=rep(0,N)
pre2.se=rep(0,N)
return2=return
for (i in 1:N){
```

```
m=arima(return2[1:(T1-
1+i)],order=c(5,0,10),transform.pars=F,fixed=c(NA,0,NA,0,NA,NA,0,NA,0,NA,NA,0,0,0,NA,N
A))
pre2[i]<-predict(m)$pred
pre2.se[i]<-predict(m)$se
return2=c(return2,predict(m)$pred)
}
plot(as.numeric(return)[1:n],ylab="log return",type="l",col="green")
title("Dynamic Forecasting from now (forward 60 days)")
lines(x=(n+1):T1,y=pre2,col="purple")
lines(x=(n+1):T1,y=pre2+3.00*pre2.se,col="red")
lines(x=(n+1):T1,y=pre2-3.00*pre2.se,col="blue")

##ARIMA((1,3,5),0,(1,3,5,6,10))+GARCH(1,1) with t distribution model forecasting
final1=ugarchspec(mean.model = list(armaOrder = c(5, 10)),fixed.pars =
list(ar2=0,ar4=0,ma1=0,ma4=0,ma7=0,ma8=0,ma9=0),
variance.model = list(model = "sGARCH", garchOrder = c(1, 1)), distribution.model = "std")
modelfinal1=ugarchfit(return,spec=final1)
par(mfrow=c(3,1))
plot(modelfinal1,which=1)
plot(modelfinal1,which=2)
plot(modelfinal1,which=3)
N=60
modelfinal1=ugarchfit(return,spec=final1,out.sample=N)
forecast1=ugarchforecast(modelfinal1,n.ahead=1,n.roll=N,out.sample=N)
deltat1=as.numeric(sigma(forecast1))
T1=length(return)
N=60
n=T1-N
pre=rep(0,N)
pre.se=rep(0,N)
for (i in 1:N){
m=arima(return[1:(n-
1+i)],order=c(5,0,10),transform.pars=F,fixed=c(NA,0,NA,0,NA,NA,0,NA,0,NA,NA,0,0,0,NA,N
A))
pre[i]<-predict(m)$pred
pre.se[i]<-deltat1[i]
}
plot(as.numeric(return)[1:T1],ylab="log return",type="l",col="green")
title("Dynamic Forecasting leave 60 samples (last 60 days)")
lines(x=(n+1):T1,y=pre,col="purple")
lines(x=(n+1):T1,y=pre+3.00*pre.se,col="red")
lines(x=(n+1):T1,y=pre-3.00*pre.se,col="blue")
N=60
modelfinal1=ugarchfit(return,spec=final1,out.sample=N)
forecast1=ugarchforecast(modelfinal1,n.ahead=1,n.roll=N,out.sample=N)
plot(forecast1,which=2)
predict1=sigma(forecast1)
predict1
returnsample=return[(length(return)-N):length(return)]
MSFE1=sum(returnsample-predict1)^2/N
MAFE1=sum(abs(returnsample-predict1))^2/N
Bias1=sum(returnsample-predict1)/N
MSFE1;MAFE1;Bias1
roll1=ugarchroll(final1,data=return, n.ahead=1,forecast.length=60,refit.every=10,
refit.window = c("moving"))
plot(roll1, which=4)
```