**Introduction**

Bitcoin is a popular topic recently. Bitcoin's price is determined by supply and demand. The specific exchange rates are formed in the process of Bitcoin trading on various online exchanges. Just as with any other currency, Bitcoin's price is ever-changing and depends on a multitude of factors, including but not limited to the number and size of businesses which accept bitcoins as payment, general sentiment regarding the cryptocurrency's future and pure speculation.

Last month, the price of Bitcoin increased exponentially which reached to nearly $20,000. This unexpected phenomenon really interested us a lot. So that we would like to have a try to make the prediction of bitcoin price.

In order to verify the accuracy of our prediction, we also compared the predicted price, which time period is from 2017.10.9 to 2017.12.7, with actual price we found on official website.

**Data Analysis Process & Results**

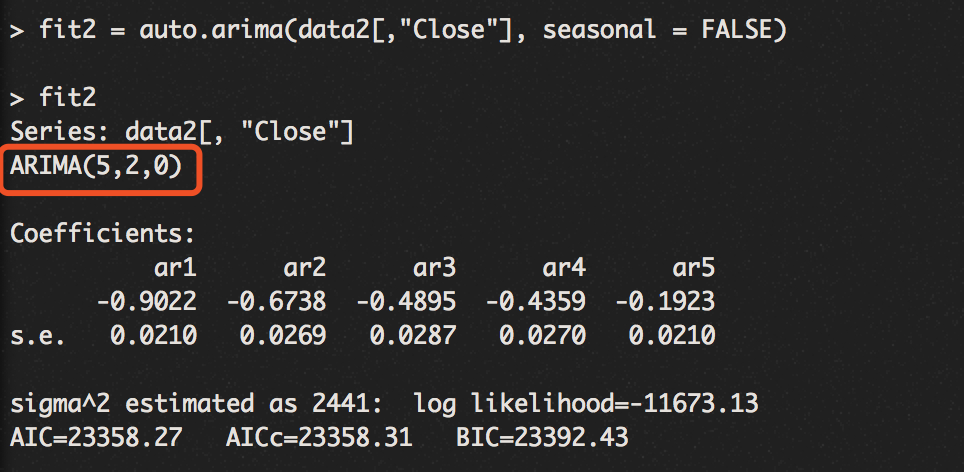
* **Research Questions**

Using the data from 2011 to 2017.10.8, we tried to figure out how bitcoin price fluctuated based on USD currency. We also wanted to make prediction for bitcoin price in 45 days and compared with actual price.

* **Analytical Process & Results**

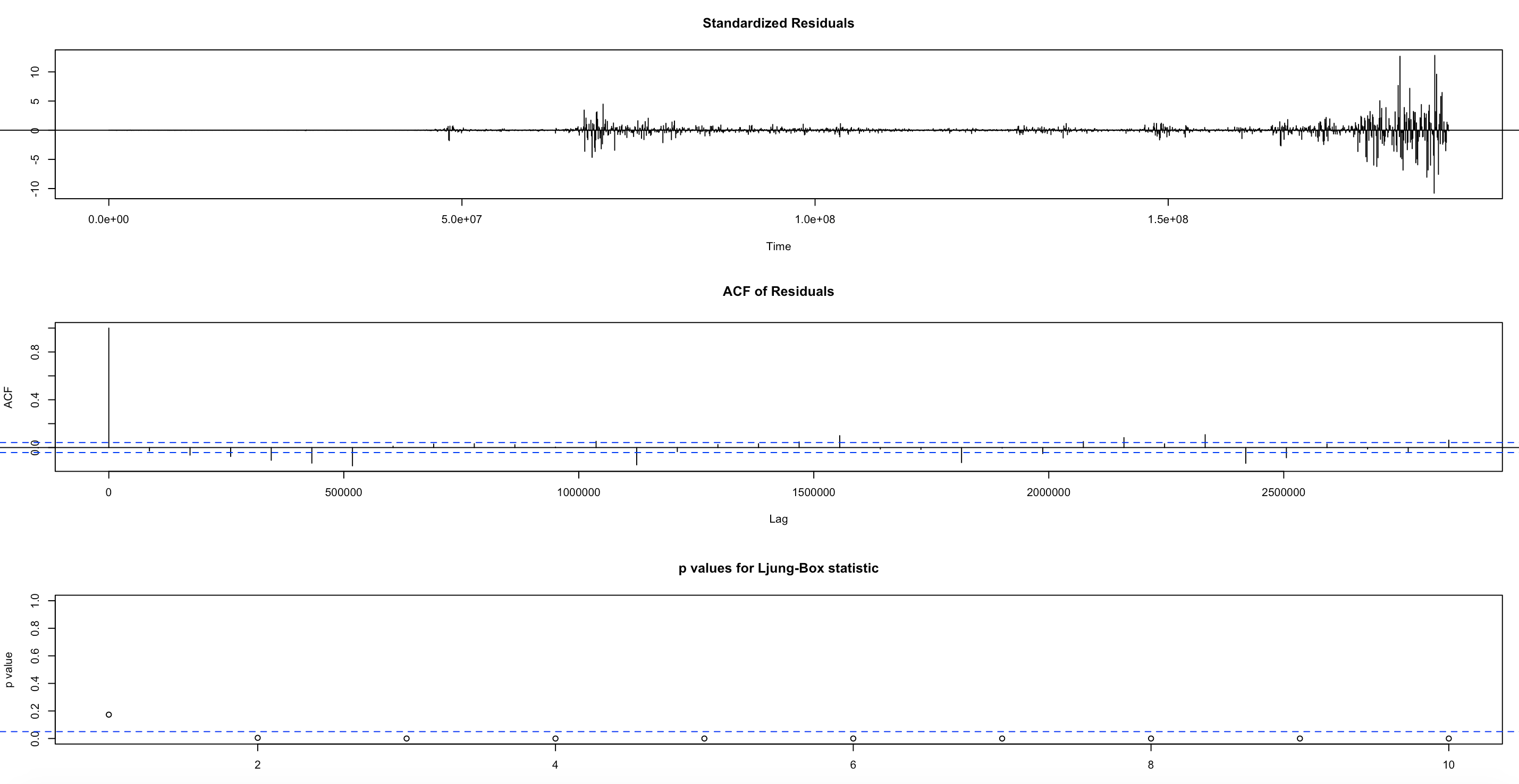
**Part I ARIMA**

ARIMA models aim to describe the autocorrelations in the data. Let's model our ARIMA using *auto.arima* function that selects the best fit for ARIMA. The result of *auto.arima* in Figure 1 suggests that we could use ARIMA(5,2,0) to describe the autocorrelations in the data, which means the model will differencing 2 times.



*Figure 1 Result of auto.arima*

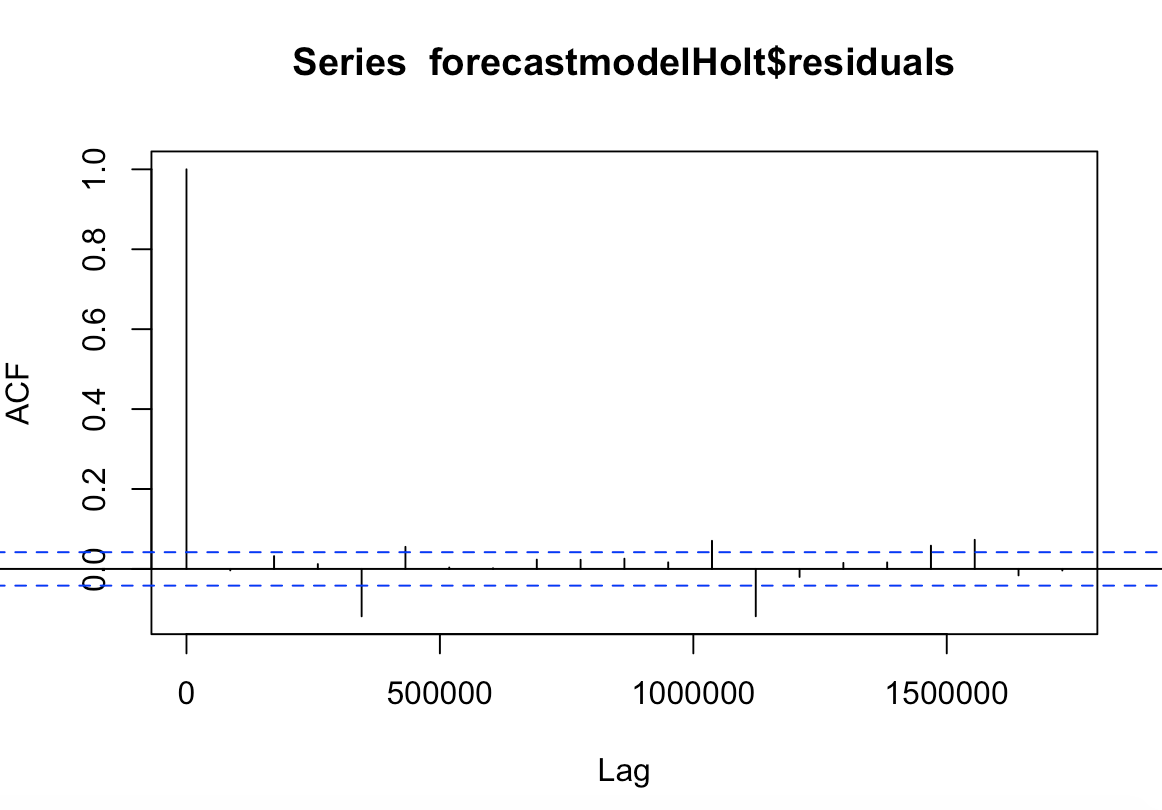
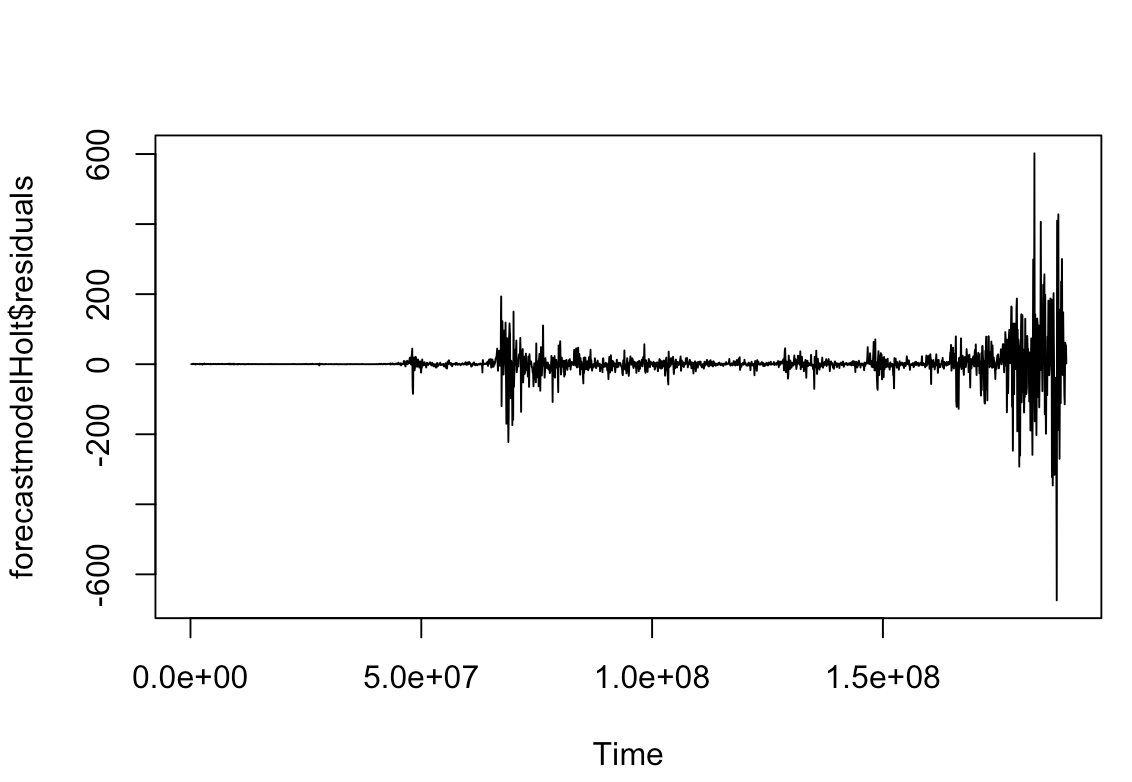
Then we diagnosed the ARIMA model to verify whether the model fitted and the results were shown in Figure 2. The time series plot of the standardized residuals mostly indicated that there was a trend in the residuals, and in general, there was changing variance across time. The ACF of the residuals showed there were significant autocorrelations – a not good result. The bottom plot gave p-values for the Ljung-Box statistics for each lag up to 10. These statistics considered the accumulated residual autocorrelation from lag 1 up to and including the lag on the horizontal axis. The dashed blue line is at 0.05. All p-values except the first one was below it. That is not a good result. We want non-significant values for this statistic when looking at residuals. All in all, the fit looks not good. So, we cannot use this model to predict price.



*Figure 2 Diagnose results of ARIMA(5,2,0)*

**Part II Holt's linear trend method**

While ARIMA models aim to describe the autocorrelations in the data, exponential smoothing models were based on a description of trend and seasonality in the data. We used the exponential smoothening--Holt's linear trend method to do next step analysis. After diagnosing the Holt’s model, we found that this model also cannot be used to predict price. The results could be found in Figure 3.

*Figure 3 Diagnose results of Holt’s Model*

The ACF of the residuals showed there were significant autocorrelations. These statistics considered the accumulated residual autocorrelation from lag 1 up to and including the lag on the horizontal axis. We also wanted non-significant values for this statistic when looking at residuals. All in all, the fit looks not good. So, we cannot use this model to predict price.

**Part II Random walk**

The underlying assumption of random walk is the daily change rate of the price is lognormal, thus, we can predict price on basis of passive data.

Here are three ways to predict future price:

1. Predict the probability of whether the future will exceed or below expected figure during the set future period.

2. Predict the probability of whether the future will exceed or below expected figure on the set day in the future.

3. Predict the price of a certain day in the future.

Limitation:

1.The underlying assumption supposes the change rate of the price is lognormal, however, the real change rate may be not fit lognormal distribution.

2.Based on the feature of normal distribution, upward probability is symmetric as downward, in reality, the price of bitcoin is more likely to increase exponentially.

Conclusion:

Based on the assumption and outcome of the random walk model, the price of bitcoin will increase linearly. And the probability of getting to a specific price can be predicted from the model. The test data reflects that the forecast price is quite as the same as the true price.

**Recommendations**

* **Recommendations**
* **Compared with actual price**

**About analysis methodology**

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Concept** | **Strengths** | **Weaknesses** |
| Holt's linear trend method | Holt’s Linear Trend computes an evolving trend equation through the data using a special weighting function that places the greatest emphasis on the most recent time periods. Instead of the global trend equation of the least squares trend algorithm, this technique uses a local trend equation. The trend equation is modified from period to period. The forecasting equation changes from period to period. | -It produces accurate forecasts.  -It gives more significance to recent observations.  -It is easy to learn and apply. | -It produces forecasts that lag behind the actual trend.  -It cannot handle trends well. |
| ARIMA(5,2,0) | The acronym ARIMA stands for Auto-Regressive Integrated Moving Average. Lags of the stationarized series in the forecasting equation are called "autoregressive" terms, lags of the forecast errors are called "moving average" terms, and a time series which needs to be differenced to be made stationary is said to be an "integrated" version of a stationary series. | It can increase the forecast accuracy while keeping the number of parameters to a minimum | -Some of the traditional model identification techniques for identifying the correct model from the class of possible models are difficult to understand and usually computationallyexpensive  - Subjective and the reliability of the chosen model can depend on the skill and experience of the forecaster |
| Random Walk | An agent which traverses a graph randomly. Each step randomly goes from node A to a random neighbour A’ | -Easy to implement  -No knowledge of underlying graph required  -Little memory footprint | -Unpredictable cover and hitting time  -Worst case of infinite |

**#random walk**

**#days is the parameter determining the sample size, which means how many random rates to generate within one sample group.**

days <- 100

**#Here, we set days of investment**

predictday <- 60

**#calculate mean and variance of DailychangeRate**

rate\_mean <- mean(DailyChangeRate)

rate\_var <- var(DailyChangeRate)

**#we predict the price range from 3500 to 5000**

begin <- 3500

end <- 5500

rang <- end -begin +1

**#the close price of the last day is 4432.56**

new\_Close <- 4432.56

**#niter is the times we do experiment, also indicate the number of experiment groups, bigger niter means more groups, and the result is more accurate, but if it’s too big, it costs more time to do simulation.**

niter <- 40

result <- rep(0,niter)

probdistribute<- data.frame(price= numeric(rang),prob= numeric(rang))

dayfluc <- data.frame(day= numeric(predictday),price= numeric(predictday))

set.seed(2009)

**(1) 1. Predict the probability of whether the future will exceed or below expected figure during the set future period.**

**#plot the graph indicating the probability lower or higher than certain price.**

for (re in begin:end) {

for (i in 1:niter)

{

r <- rnorm(days,mean=rate\_mean,sd=sqrt(rate\_var))

logPrice <- log(new\_Close) + cumsum(r)

maxlogP <- max(logPrice)

minlogP <- min(logPrice)

if (re < new\_Close) {

result[i] <- as.numeric(minlogP < log(re))

}else {

result[i] <- as.numeric(maxlogP > log(re))

}

}

row1<- c(re, mean(result))

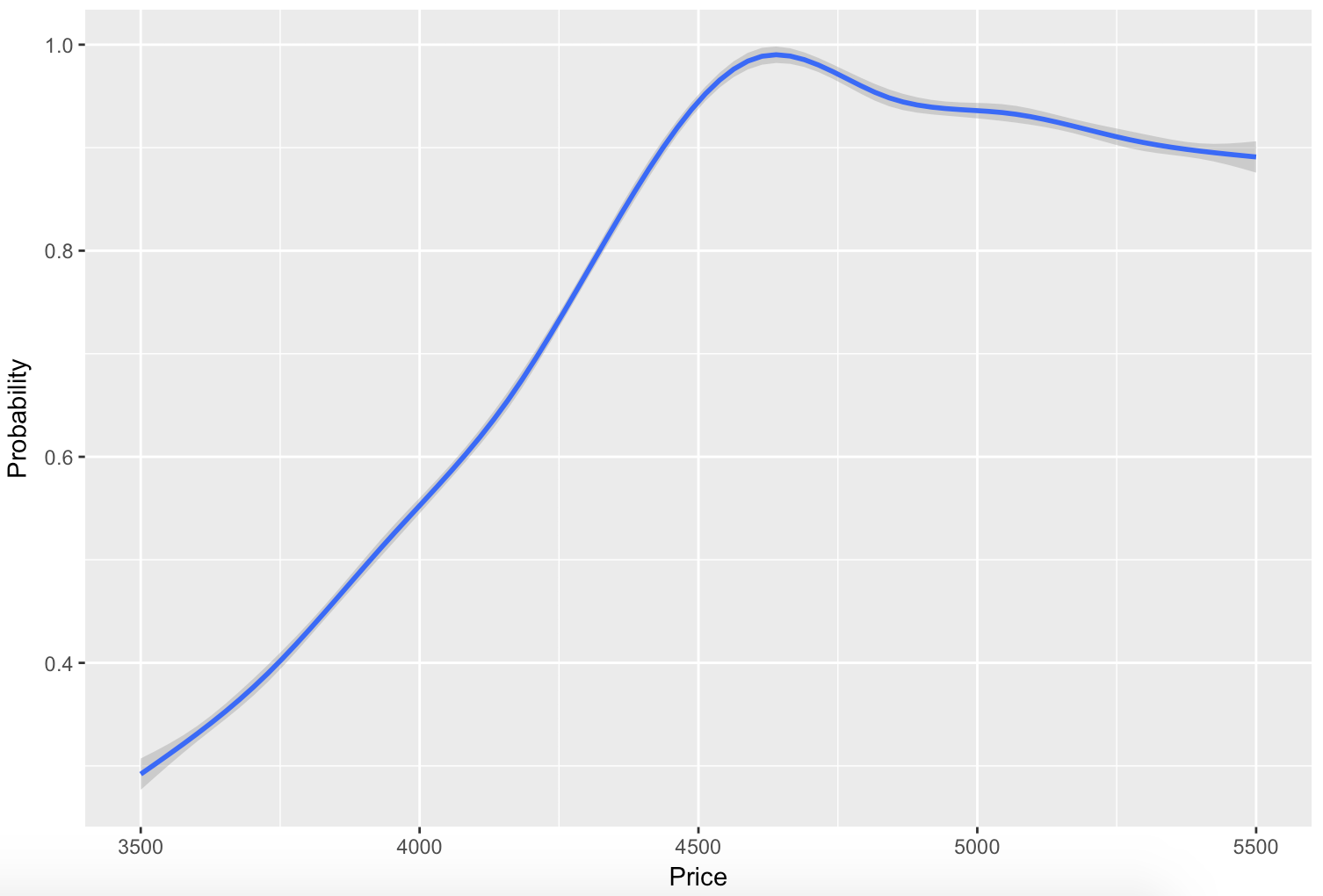
probdistribute[re-begin+1,] <- row1

}

ggplot(probdistribute, aes(x=probdistribute$price, y=probdistribute$prob)) + geom\_smooth()+

xlab("Price") +

ylab('Probability')



**(2) Predict the probability of whether the future will exceed or below expected figure on the set day in the future.**

# the price lower than specific price in the last day

for (re in begin:end) {

for (i in 1:niter)

{

r <- rnorm(days,mean=rate\_mean,sd=sqrt(rate\_var))

logPrice <- log(new\_Close) + sum(r)

if (re < new\_Close) {

result[i] <- as.numeric(logPrice < log(re))

}else {

result[i] <- as.numeric(logPrice > log(re))

}

}

row1<- c(re, mean(result))

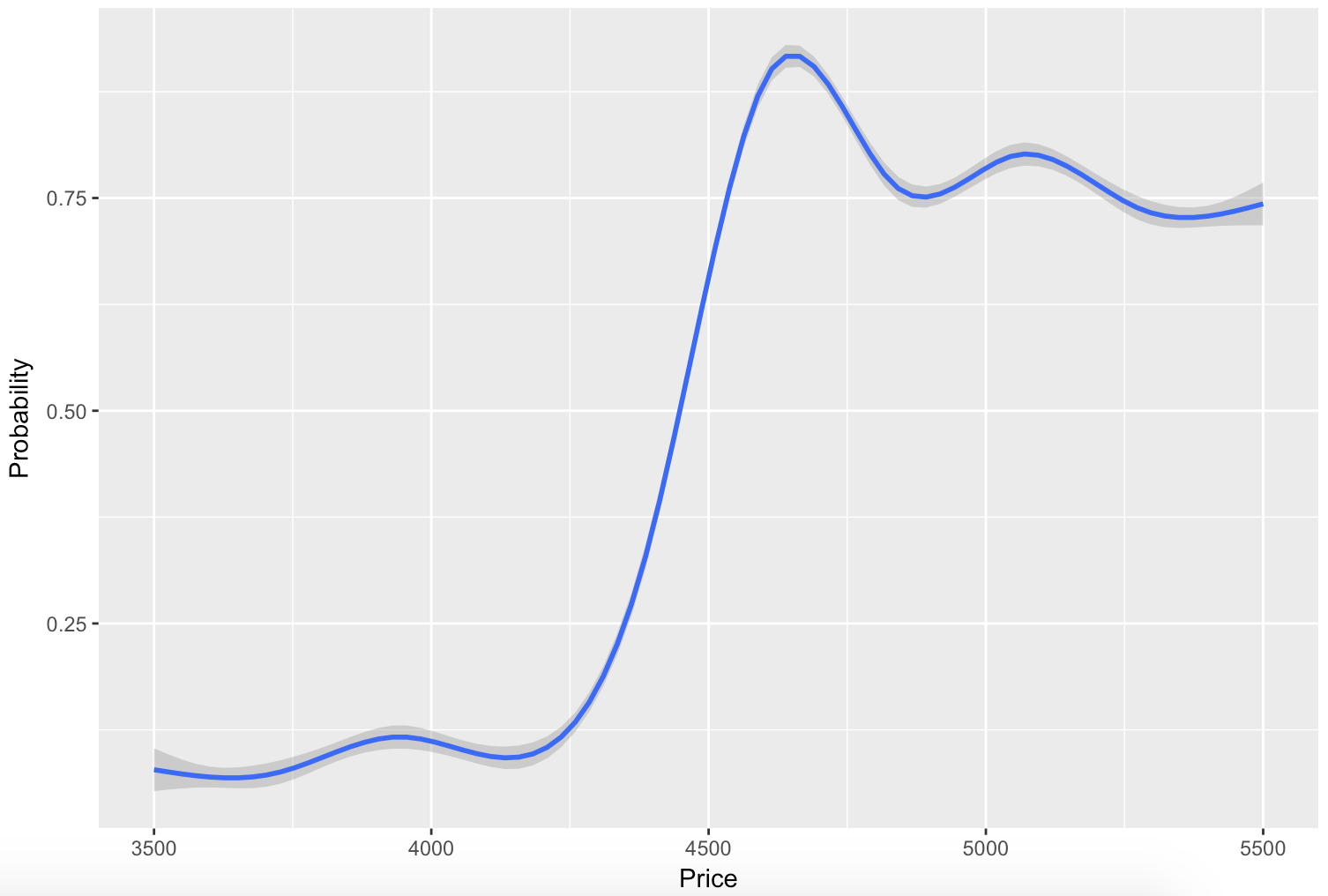
probdistribute[re-begin+1,] <- row1

}

ggplot(probdistribute, aes(x=probdistribute$price, y=probdistribute$prob)) + geom\_smooth()+

xlab("Price") +

ylab('Probability')



**(3) Predict the price of a certain day in the future.**

for (one in 1:predictday) {

days <- one

for (re in begin:end) {

for (i in 1:niter)

{

r <- rnorm(days,mean=rate\_mean,sd=sqrt(rate\_var))

logPrice <- log(new\_Close) + sum(r)

# if (re < new\_Close) {

# result[i] <- as.numeric(logPrice < log(re))

# }else {

# result[i] <- as.numeric(logPrice > log(re))

# }

result[i] <- exp(logPrice)

}

row1<- c(re, mean(result))

probdistribute[re-begin+1,] <- row1

}

dayfluc[one,]<- c(one, mean(probdistribute$prob))

}

attach(dayfluc)

ggplot(dayfluc, aes( dayfluc$day, dayfluc$price,colour="red")) + geom\_line(size=1.5) +

xlab("Day") +

ylab('Price')

