Comparison of stock price prediction using geometric Brownian motion and multilayer perceptron

Cite as: AIP Conference Proceedings **2242**, 030016 (2020); https://doi.org/10.1063/5.0008066
Published Online: 01 June 2020

M. Azizah, M. I. Irawan, and E. R. M. Putri

Comparison of Stock Price Prediction Using Geometric Brownian Motion and Multilayer Perceptron

M. Azizah[a)], M. I. Irawan and E. R. M. Putri

Department of Mathematics, Faculty of Mathematics, Computing and Data Sciences, Institut Teknologi Sepuluh Nopember, Surabaya 60111, Indonesia

[a)]Corresponding author: muftiyatulazizah@gmail.com

Abstract. The Stock is defined as an investor ownership sign of their investment or the amount of fund invested in a company. In the transaction process of stock exchange, stock is the most instrument traded. Thus, the forecasting of stock price is very important to develop an effective market trading strategy. The forecasting of stock prices can anticipate investment losses and provide optimal benefits for investors. In this paper, Microsoft stock prices will be predicted by the geometric Brownian motion and multilayer perceptron methods. Prediction of stock prices using geometric Brownian motion was begun by calculating the return value of the data. Then, the normality test of the return value is carried out. The value of return must be normally distributed. Then, we do calculation to get the value of drift and volatility. The parameters of geometric Brownian motion are assumed constant. The parameter values will be used as input in prediction process with MATLAB. In the multilayer perceptron method prediction, the data divided into two parts, 70 % for training data and 30 % for validation data. Then, the data must be normalized first. In the prediction process using multilayer perceptron, we will initialize the weight, correct the weight values, correct bias and calculate error value. As the result, the MAPE value from multilayer perceptron method predictions was 0.05266. This value obtained when the number of neurons are 2 in the input and the number of neurons are 3 in the hidden layer. Meanwhile, the MAPE value produced by geometric Brownian motion method were 0.0221 when 10 trajectories and 0.019571689 when 10000 trajectories. Then, the result of geometric Brownian motion method is better than multilayer perceptron method.

Keywords: Stock, Geometric Brownian Motion, Multilayer Perceptron

INTRODUCTION

The Stock is defined as an investor ownership sign of their investment or the amount of fund invested in a company. By publishing a stock, the company is expected to obtain additional capital from each sheet sold. More stocks owned by investors or stockholders show the higher level of company performance. Meanwhile, the investor interest declined in investing funds in the stock form of a company means the lower level of the company performance[1].

In the transaction process of stock exchange, stock price is the most dominant instrument to be traded. Thus, the forecasting of stock price is very important to develop the effective market trading strategy. The forecasting of stock prices can anticipate investment losses and provide optimal benefits for investors [2]. Based on market value in 2018, there are ten largest companies in the world such as Apple, Amazon.com, Alphabet, Microsoft, Facebook, Alibaba, Berkshire Hathaway, Tencent Holdings and JPMorgan Chase [3].

There are several studies related to stock price predictions. Kara, Boyacioglu and Baykan predicted stock prices by taking samples of stock data in Istanbul. The methods used were Artificial Neural Network and Support Vector Machines. The prediction results using the Artificial Neural Network method show the more significant results compared to the prediction results from the Support Vector Machines method [2]. Zahedi and Rounaghi also used artificial neural network methods to predict the changes of stock prices in Tehran [4]. Qiu, Song and Akagi predicted

Proceedings of the 5th International Symposium on Current Progress in Mathematics and Sciences (ISCPMS2019)
AIP Conf. Proc. 2242, 030016-1–030016-12; https://doi.org/10.1063/5.0008066
Published by AIP Publishing. 978-0-7354-2001-4/$30.00

the Japanese Nikkei 225 return index with artificial neural network [5]. Moghaddam, Moghadam and Esfandyari predicted stock prices on the NASDAQ company with artificial neural networks [6]. In 2016, Reddy and Clinton predicted stock prices in Australia using Geometric Brownian Motion (GBM). The results show that the prediction results with GBM are almost the same as the original data [7]. Abidin and Jaffar also predicted stock prices in small companies in the Malaysian stock exchange using the geometric Brownian motion method [8].

Artificial neural network and geometric Brownian motion have used to predict stock prices and produce predictions which are close to the original data. In this paper, the authors will discuss the comparison of the results of geometric Brownian motion and multilayer perceptron in predicting Microsoft stock data.

EXPERIMENTAL

Multilayer Perceptron

In supervised training, there are several number of inputs and output targets used to train the network. At each training, input value is given to the network. The network will process and produce output value. The difference between network output and target value is called an error. The network will modify the weight based on the error. One model that uses supervision training is multilayer perceptron (MLP) [9].

This algorithm is based on a simple relationship, that if output gives a wrong result, then the weight will be corrected. So, errors can be reduced and the next network response is expected to be closer to the targets. Multilayer perceptron capable of correcting weight in hidden layers. When the network is given an input pattern as a training pattern, then the pattern goes to the units in the hidden layer to be passed on to the output layer units. Next, output layer units give a response called network output. When the network output is not the same as the target output, the output will be spread backward in the hidden layer and forwarded to the unit in the input layer [10].

The notations used in the multilayer perceptron algorithm are as follows:

t_k	=	Target to k	b_{2k}	=	The connection weight value in bias for unit y_k
x_i	=	Unit to i in the input layer	w_{jk}	=	The connection weight value from z_{ij} to unit y_k
z_j	=	Unit to j in the hidden layer	Δw_{jk}	=	Difference between $w_{jk}(t)$ and $w_{jk}(t+1)$
z_{in}	=	Output for unit z_j	b_{ij}	=	Weight value in bias for unit y_k
y_k	=	Unit to k in the output layer	v_{ij}	=	The connection weight value from unit x_i to unit z_j
y_{in}	=	Net input for unit y_k	Δv_{ij}	=	Difference between $v_{ij}(t)$ and $v_{ij}(t+1)$
α	=	Learning rate $(0 < \alpha < 1)$	δ_k	=	The setting factor for connection weight value in output layer
			δ_j	=	The setting factor for connection weight value in hidden layer

The steps of the multilayer perceptron are as follows [10]:
Step 0: Random initialization of weights at intervals [0,1].
Step 1: For each training data, do steps 2 until 7.

Feedforward Process.
Step 2: Each input units $(x_i, i = 1,2,3, \dots, n)$ receives the input signal x_i and the signal is transmitted to the hidden layer units.
Step 3: Each units in the hidden layer is multiplied by its weight and added with its bias,

$$z_{in_j} = b_{1j} + \sum_{i=1}^{n} x_i v_{ij} \quad (1)$$

Then, calculated according to the activation function used,

$$z_j = f(z_{in_j}) \quad (2)$$

Then, sending the signal to all output units.

Step 4: Each units of output ($y_k, k = 1,2,3, \dots, m$) is multiplied by its weight and added up,

$$y_{in_k} = b_{2k} + \sum_{j=1}^{p} z_j w_{jk} \tag{3}$$

Recalculate according to the activation function,

$$y_k = f\left(y_{in_k}\right) \tag{4}$$

Backpropagation of error.
Step 5: Each units of output ($y_k, k = 1,2,3, \dots, m$) receive target pattern according to the input pattern during training and the error is calculated.

$$\delta_k = (t_k - y_k) f'\left(y_{in_k}\right) \tag{5}$$

Calculating correction of weight w_{jk},

$$\Delta w_{jk} = \alpha \delta_k z_j \tag{6}$$

Calculating correction of bias,

$$\Delta b_2 = \alpha \delta_k \tag{7}$$

and using the value δ_k on all previous layer units.
Step 6: Each weight that connect units of the output layer to the units in hidden layer $Z_{jk}, j = 1,2,3, \dots, p$; $k = 1,2, \dots, m$) is multiplied by δ and added as input to next layer units.

$$\delta_{in_j} = \sum_{k=1}^{m} \delta_k w_{jk} \tag{8}$$

Next, multiplied by the derivative of the activation function to calculate the error,

$$\delta_j = \delta_{in_j} f'\left(z_{in_j}\right) \tag{9}$$

Then, calculating correction of weight v_{ij},

$$\Delta v_{ij} = \alpha \delta_j x_i \tag{10}$$

Next, calculating correction of bias b_{ij},

$$\delta b_{ij} = \alpha \delta_j \tag{11}$$

Updating weight and bias.
Step 7: Each output units ($y_k, k = 1,2, \dots, m$) is updating bias and weight.

$$w_{jk}(new) = w_{jk}(old) + \Delta w_{jk} \tag{12}$$

$$b_{2k}(new) = b_{2k}(old) + \Delta b_{2k} \tag{13}$$

Each hidden units ($z_j, j = 1,2, \dots, p$) is updating bias and weight.

$$v_{ij}(new) = v_{ij}(old) + \Delta v_{ij} \tag{14}$$

$$b_{1j}(new) = b_{1j}(old) + \Delta b_{1j} \tag{15}$$

Step 8: Checking stop condition. If the stop condition has been met, network training can be stopped. If it has not meet, repeating steps 2 until 7 so that the stop condition being met [10].
Steps to check stop conditions using Mean Absolute Percentage Error as follows:

a) With the current weight, do the feedforward step (step 3 until step 5). The input is taken from training data input if what is counted is training data error. These steps are carried out for all available training data.
b) Finding the difference between targets (t_k) with network output (y_k) and implemented in the Mean Absolute Percentage Error (MAPE). The forecasting method is called good if the MAPE value generated from forecasting method is getting smaller. The formula of MAPE is defined as follows [7].

$$MAPE = \sum_{t=1}^{n} \frac{\frac{|A_t - F_t|}{A_t}}{n} 100\% \tag{16}$$

where A_t : actual data, F_t : prediction result, n : amount of data

Brownian Motion

Brownian motion or also called the Wiener process is a stochastic process of continuous time. The stochastic process Z when $t \leq 0$ is called Brownian motion if it meets three conditions [11]:

1. Z_k is a continuous path where $Z_0 = 0$.
2. For $k_{i+1} > k_i : Z_{k_{i+1}} - Z_{k_i}$ normally distributed with mean 0 and variance t.

For non-negative integer selection $0 = k_0 < k_1 < \cdots < k_m$, then there are random variables,

$$\begin{aligned} Z_1 &= S_1 - S_0 \\ Z_2 &= S_2 - S_1 \\ &\vdots \\ Z_k &= S_k - S_{k-1} \\ Z_{k_{i+1}} &= \sum_{k_{i+1}}^{i=1} S_i \\ Z_{k_i} &= \sum_{k_i}^{i=1} S_i \end{aligned}$$

so obtained $Z_{k_{i+1}} - Z_{k_i} = \sum_{k_{i+1}}^{i=k_i+1} S_i$. The value of $E(Z_{k_{i+1}} - Z_{k_i})$ and $Var(Z_{k_{i+1}} - Z_{k_i})$ as follows,

$$\begin{aligned} E\left(Z_{k_{i+1}} - Z_k\right) &= E\left(\sum_{k_{i+1}}^{i=k_i+1} S_i\right) \\ &= \sum_{k_{i+1}}^{i=k_i+1} E(S_i) \\ &= 0 \end{aligned}$$

$$Var\left(Z_{k_{i+1}} - Z_{k_i}\right) = \sum_{k_{i+1}}^{i=k_i+1} Var(S_i)$$
$$= \sum_{k_{i+1}}^{i=k_i+1} 1$$
$$= k_{i+1} - k_i + 1$$

3. For $k_{i-1} < k_i$: Z_k only be affected by $Z_{k_{i-1}}$.

where Z_t= Wiener process (Brownian motion), S_i= stock price at i.

Geometric Brownian Motion

For the prediction process using geometric Brownian motion, a general solution is needed from the geometric Brownian motion model. This is the process of determining the solution of a geometric Brownian motion model [12]. The stochastic differential equation is given as follows,

$$dX(t) = f\big(t, X(t)\big)dt + g\big(t, X(t)\big)dW(t) \tag{17}$$

For example $Y(t)$ is function from $X(t)$,

$$Y(t) = \phi(t, X(t)) \tag{18}$$

with Lemma Ito, function $\phi(t, X(t))$ will be derived to variabel t and variabel X. Furthermore, by using chain rules, the following results will be obtained,

$$dY(t) = \Big(\phi_t\big(t, X(t)\big) + \phi_X\big(t, X(t)\big) f\big(t, X(t)\big)\Big)\, dt \tag{19}$$

Then, using the Taylor Series obtained as follows,

$$\begin{aligned} dY(t) = {} & \phi_t\big(t, X(t)\big)dt + \frac{1}{2}\phi_{tt}\big(t, X(t)\big)dt^2 + \phi_X\big(t, X(t)\big)dX(t) \\ & + \frac{1}{2}\phi_{XX}\big(t, X(t)\big)\big(dX(t)\big)^2 \end{aligned} \tag{20}$$

Substitute the Equation 17 to Equation 20,

$$\begin{aligned} dY(t) = {} & \phi_t\big(t, X(t)\big)dt + \frac{1}{2}\phi_{tt}\big(t, X(t)\big)dt^2 \\ & + \phi_X\big(t, X(t)\big)\big[f\big(t, X(t)\big)dt + g\big(t, X(t)\big)dW(t)\big] \\ & + \frac{1}{2}\phi_{XX}\big(t, X(t)\big)\big[f^2\big(t, X(t)\big)dt^2 + 2.f\big(t, X(t)\big).g\big(t, X(t)\big)dtdW(t) \\ & \qquad + g^2\big(t, X(t)\big)dW^2(t)\big] \end{aligned} \tag{21}$$

Differentials from high order (dt, dW) headed to zero, so $dt^2 \to 0$ and $dt.dW(t) \to 0$. The term stochastic $dW^2(t)$ according to Brown motion rules is given as follows, $dW^2(t, \omega) = dt$, then $dW^2(t) = dt$.

So, the solution obtained from Equation 17 as follows,

$$dY(t) = \left[\phi_t\big(t,X(t)\big) + \phi_x\big(t,X(t)\big)f\big(t,X(t)\big) + \frac{1}{2}\phi_{xx}\big(t,X(t)\big)g^2\big(t,X(t)\big)\right]dt + \phi_x\big(t,X(t)\big)g\big(t,X(t)\big)dW(t) \quad (22)$$

Furthermore, the differential stochastic equation for the stock price in Brownian motion is [10],

$$\tfrac{1}{S(t)}dS(t) = \mu dt + \sigma dW(t) \quad (23)$$

Equation 23 can be written as,

$$dS(t) = \mu S(t)dt + \sigma S(t)dW(t) \quad (24)$$

Equation 24 has the same form as Equation 17, where:

$$dX(t) = dS(t) \quad (25)$$

$$f\big(t,X(t)\big) = \mu S(t) \quad (26)$$

$$g\big(t,X(t)\big) = \sigma S(t) \quad (27)$$

So the solution of Equation 24 is as follows,

$$dY(t) = \left[\phi_t\big(t,S(t)\big) + \phi_s\big(t,S(t)\big)\mu S(t) + \tfrac{1}{2}\phi_{ss}\big(t,S(t)\big)\sigma^2S^2(t)\right]dt + \phi_s\big(t,S(t)\big)\sigma S(t)dW(t) \quad (28)$$

Then, choose $\ln S(t)$ as the value $Y(t)$ which is [12],

$$Y(t) = \phi\big(t,S(t)\big) = \ln S(t) \quad (29)$$

Substitute Equation 29 to Equation 28, so that,

$$d(\ln S(t)) = \left[\phi_t\big(t,S(t)\big) + \phi_s\big(t,S(t)\big)\mu S(t) + \tfrac{1}{2}\phi_{ss}\big(t,S(t)\big)\sigma^2S^2(t)\right]dt + \phi_s\big(t,S(t)\big)\sigma S(t)dW(t) \quad (30)$$

with,

$$\phi_t\big(t,S(t)\big) = \frac{\delta\phi_t\big(t,S(t)\big)}{\delta t} = \frac{\delta(\ln S(t))}{\delta t} = 0 \quad (31)$$

$$\phi_s\big(t,S(t)\big) = \frac{\delta\phi_s\big(t,S(t)\big)}{\delta t} = \frac{\delta(\ln S(t))}{\delta S(t)} = \frac{1}{S(t)} \quad (32)$$

$$\phi_{ss}\big(t,S(t)\big) = \frac{\delta^2\phi_{ss}\big(t,S(t)\big)}{\delta S^2(t)} = \frac{\delta^2(\ln S(t))}{\delta S^2(t)} = -\frac{1}{S^2(t)} \quad (33)$$

Next, substitute Equation 31, Equation 32 and Equation 33 into Equation 30 so that they are obtained:

$$d(\ln S(t)) = \left[0 + \frac{1}{S(t)}\mu S(t) + \frac{1}{2}\left(-\frac{1}{S^2(t)}\right)\sigma^2S^2(t)\right]dt + \left(\frac{1}{S(t)}\right)\sigma S(t)dW(t) \quad (34)$$

$$d(\ln S(t)) = \left(\mu - \frac{1}{2}\sigma^2\right)dt + \sigma dW(t) \quad (35)$$

Then, do the construction in Equation 35 as follows:

$$\ln S(t) - \ln S(t-1) = \left(\mu - \frac{1}{2}\sigma^2\right)dt + \sigma dW(t) \quad (36)$$

$$\ln \frac{S(t)}{S(t-1)} = \left(\mu - \frac{1}{2}\sigma^2\right)dt + \sigma\varepsilon\sqrt{dt} \quad (37)$$

$$\frac{S(t)}{S(t-1)} = e^{\left(\mu - \frac{1}{2}\sigma^2\right)dt + \sigma\varepsilon\sqrt{dt}} \quad (38)$$

Then, the solution obtained from the geometric Brownian motion model is as follows,

$$S(t) = S(t-1)e^{\left(\mu - \frac{1}{2}\sigma^2\right)dt + \sigma\varepsilon\sqrt{dt}} \quad (39)$$

where $S(t)$ = stock price at t, $S(t-1)$ = stock price at $t-1$, μ = drift, σ = volatility, $dW(t)$ = random walk, ε = random numbers from the standard normal distribution, dt = interval of time.

Research Method

The data of this study are Microsoft stock daily data from October 1, 2018 to March 29, 2019 which obtained from Yahoo Finance [13]. For the prediction process using multilayer perceptron method, the data was divided into two training data and testing data. The data used for training is 70 %, from October 1, 2018 to January 31, 2019, while the testing data is 30 %, from February 1, 2019 to March 29, 2019. Multilayer perceptron is determined by the number neurons of input layer, hidden layer and output layer. Input data will be fed to the input layer. After being processed in the input layer, it will be distributed to the hidden layer and finally it will be distributed to the output layer. In this paper, the number of neurons in the input used are 2 and 3 units. The number of neurons in the hidden layer are 2, 3 and 4. While the number of neurons in the output layer is 1. The number of input neurons and hidden layer neurons used is based on several journals referenced. For each input with 3 variations of the hidden layer will be running five times to get the best MAPE value. From the five times of running results, only one best MAPE value is selected.

In MLP, there are weight values that must be updated to obtain training results similar to the actual data. We use backpropagation algorithm to training process. This technique consists of forward and backward. In the forward, the number neurons of input will be processed and the result will be distributed to the output layer with update weight values. In the backward will be calculated the difference between result of the MLP and the actual data (error values). Before conducting the training in multilayer perceptron, Microsoft stock price data must be normalized, so that the data is in the interval [0, 1]. The formula for normalizing data as follows [14].

$$x = \frac{0.8(x_p - \min x_p)}{(\max x_p - \min x_p)} + 0.1 \quad (40)$$

where x_p = Original data value which normalized yet, $\min x_p$ = Minimum value in the data, $\max x_p$ = Maximum value in the data.

In the training process, weights will be calculated in the hidden layer using Equation 1. At the output layer net, input will be calculated to output the layer with Equation 3. Then each output unit updates bias and weight using Equation 12 and Equation 13. The next process is denormalization. Denormalization is returning the data size which has been normalized before to get the original data. The formulas of denormalization are as follows [14].

$$x = \frac{(x_p - 0.1)(\max(x_p) - \min(x_p))}{0.8} + \min(x_p) \quad (41)$$

where x_p = Original data value which normalized yet, $\min x_p$ = Minimum value in the data and $\max x_p$ = Maximum value in the data.

In the prediction process with the geometric Brownian motion method, the first step is to calculate the return value of the data using the following method [12],

$$\frac{dS(t)}{S(t)} \approx \frac{S_{t+1}-S_t}{S_t} \tag{42}$$

where $\frac{dS(t)}{S(t)}$ = return of stock price, S_{t+1} = Stock price at $t+1$ and S_t = Stock price at t.

Furthermore, a normality test is conducted on the return of Microsoft stock prices on October 1, 2018 until January 31, 2019. A normality test is conducted to determine the return of Microsoft stock prices is normally distributed or not. The normality test used the Kolmogorov-Smirnov method using SPSS. Then, calculate the value of drift and volatility of the geometric Brownian motion model. Before calculating the value of drift and volatility, we calculated the standard deviation value first with the following formula [12],

$$v = \sqrt{\frac{1}{n-1}\sum_{i=1}^{n}(u_i - \bar{u})^2} \tag{43}$$

After getting the standard deviation value, then we are calculating the volatility value with Equation 44 and drift value with Equation 45 [12],

$$\sigma = \frac{v}{\sqrt{\tau}} \tag{44}$$

$$\mu = \frac{\bar{u}}{\tau} + \frac{1}{2}\sigma^2 \tag{45}$$

The equation used to predict Microsoft stock prices with geometric Brownian motion is as follows,

$$F_t = F_{t-1}e^{\left(\mu-\frac{1}{2}\sigma^2\right)dt+\sigma\varepsilon\sqrt{dt}} \tag{46}$$

where F_t = Stock price prediction at t, F_{t-1} = Stock price prediction at $t-1$, μ = drift, σ = volatility, $dW(t)$ = random walk, ε = random numbers from the standard normal distribution, $F_0 = S_0$=initial stock price, v=standard deviation, n = lots of data, u_i= stock price return at i, $\bar{u}$ = average of return, σ = volatility (level of stock price movements), $\tau = t_i - t_{i-1}$ (interval of time) and μ = drift (expectations of the rate stock price movement).

RESULTS AND DISCUSSION

In this section, we will present the results of Microsoft's stock price predictions using multilayer perceptron and geometric Brownian motion. The MAPE value of prediction with multilayer perceptron method as follows.

Based on Table 1, the smallest MAPE value is 0.05266 which is when the number of neurons in the input layer is 2 and the number of neurons in the hidden layer is 3. The MAPE value generated from multilayer perceptron don't depend on the number of neurons in the input layer or hidden layer. The greater number of neurons in the input layer and hidden layer don't affect the MAPE value. We have to do many experiment to get the best MAPE value. The following are graph of training result, graph of MAPE value and comparison graph between stock data and the results of multilayer perceptron predictions.

Based on Fig. 1a, we can see that the training results and stock data are not much different and have the same pattern. The training result MAPE values of multilayer perceptron have decreased in each iteration. In Fig. 1b, the MAPE values of training result decreased which is not much different from 1000th iterations to the 25000th iterations. Prediction results using multilayer perceptron can be seen in Fig. 1c. On the 5th day until 14th day, prediction result have value closed to the stock price, while the prediction results on the 15th day to the 40th day experience quite a big difference.

In the prediction process using the geometric Brownian motion method, the normality test return of Microsoft stock price is conducted. The results of the normality test using the Kolmogorov-Smirnov method with SPSS as follows.

TABLE 1. MAPE Value with multilayer perceptron.

Number of Neuron on the Input Layer	Number of Neuron on the Hidden Layer	α	The Best MAPE	Average of MAPE
2	2	0.2	0.05269	0.052988
		0.5	0.05313	0.053624
	3	0.2	0.05275	0.053844
		0.5	0.05266	0.05284
	4	0.2	0.05278	0.053408
		0.5	0.05306	0.05302
3	2	0.2	0.05349	0.053662
		0.5	0.05424	0.054334
	3	0.2	0.05359	0.053928
		0.5	0.05463	0.05459
	4	0.2	0.05382	0.054072
		0.5	0.05446	0.053978

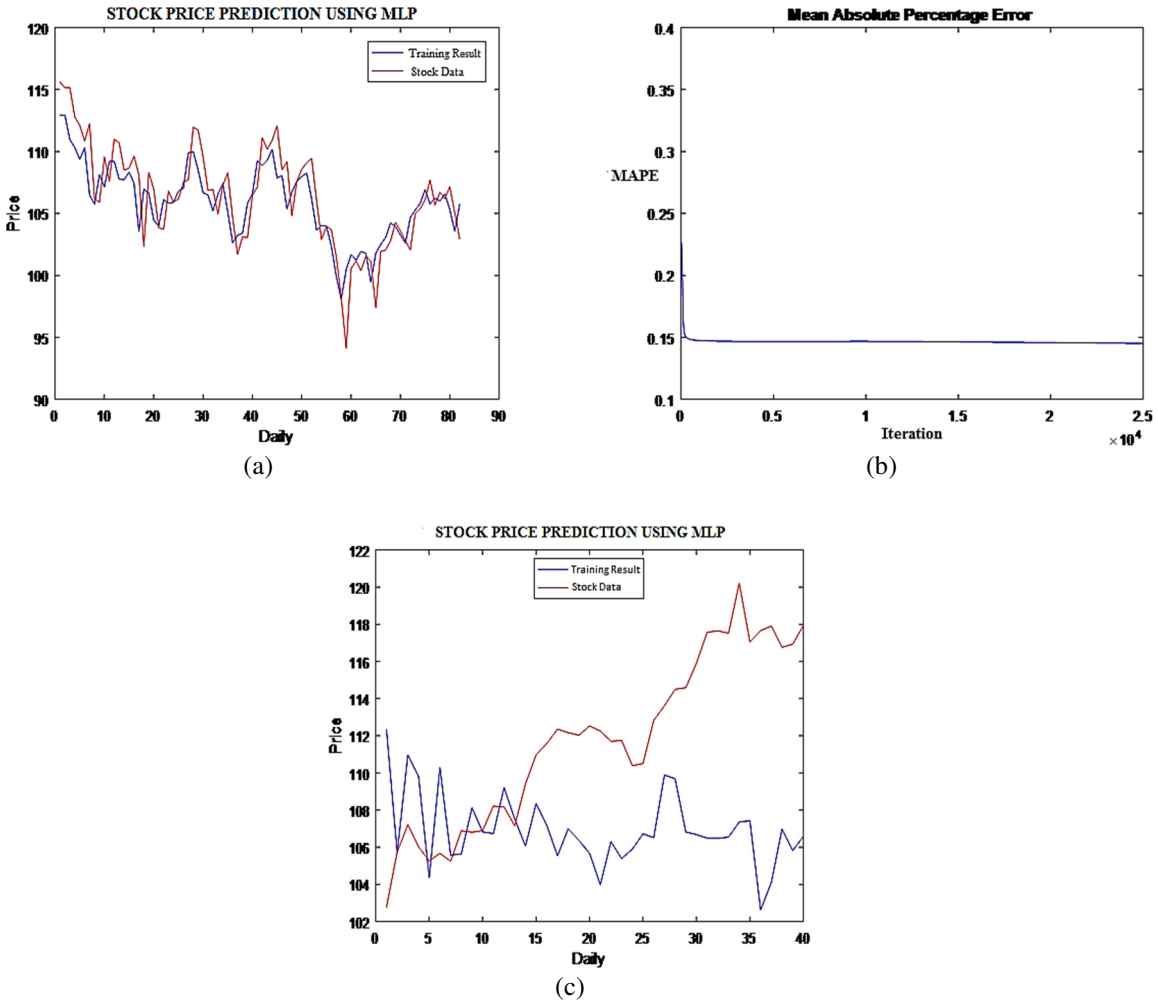


FIGURE 1. (a) Training result, (b) MAPE value and (c) Compariosn between stock price and prediction result.

Based on Fig. 2, it can be seen that P-Value is 0.085 which is greater than 0.05. This means that the return of Microsoft stock prices is normally distributed and forecasting stock prices can be carried out on the data. Furthermore, the standard deviation value, volatility and drift will be calculated using Equation 43, Equation 44 and Equation 45. The results are $v = 0.023426150$, $\sigma = 0.023426150$ and $\mu = -0.000679886$.

In the prediction process with geometric Brownian motion, we used 10 trajectories and 10000 trajectories. We choose 10 trajectories so that each trajectories can be observed and use 10000 trajectories because it's the maximum limit of the trajectories. Each paths have a different MAPE value, so that if more trajectories increase, the MAPE value obtained is likely to be smaller. The graph of 10 trajectories and 10000 trajectories using the geometric Brownian motion method as follows.

In Fig. 3a, there are ten blue trajectories which represent the results of geometric Brownian motion predictions and the red trajectory which represent the stock data. Furthermore, the MAPE value of 10 trajectories of prediction results with geometric Brownian motion as follows.

Based on Table 2, the smallest MAPE value is 0.0221 at the 4th trajectories. In Fig. 3b, there are 10000 trajectories, which is the solution of geometric Brownian motion. The smallest MAPE of 10000 trajectories is 0.019571689. The comparison graph between Microsoft stock prices and the results of predictions using the geometric Brownian motion method with 10 trajectories and 10000 trajectories as follows.

Figure 4a is the result of prediction using geometric Brownian motion with 10 trajectories, while Fig. 4b is the result of prediction with 10000 trajectories. From these results, prediction with 10000 trajectories closer to stock data compared to predictions of 10 trajectories. So that if there are more trajectories, then the predicted solution with GBM will be more varied and closer to the original data.

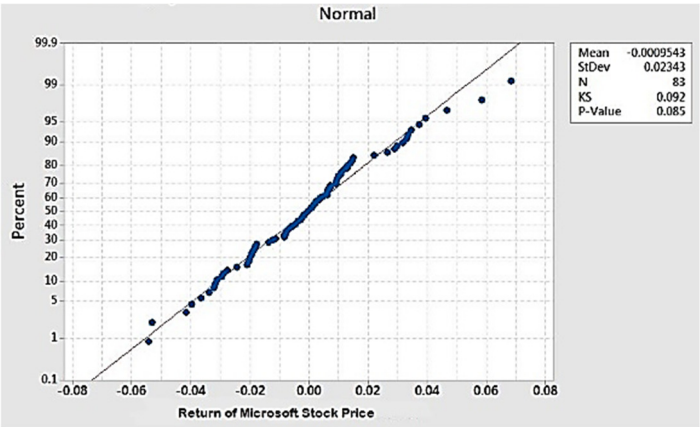
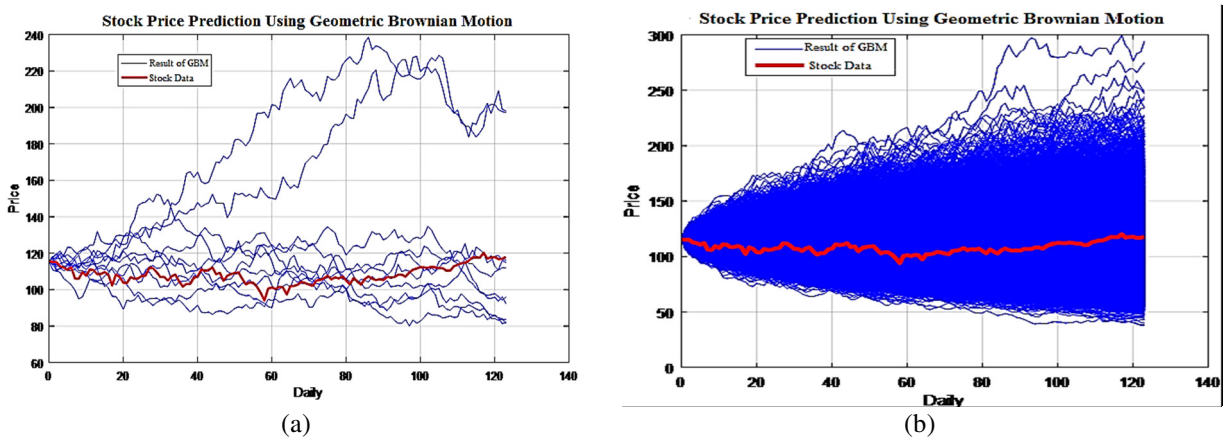


FIGURE 2. The normality test return of Microsoft stock price.

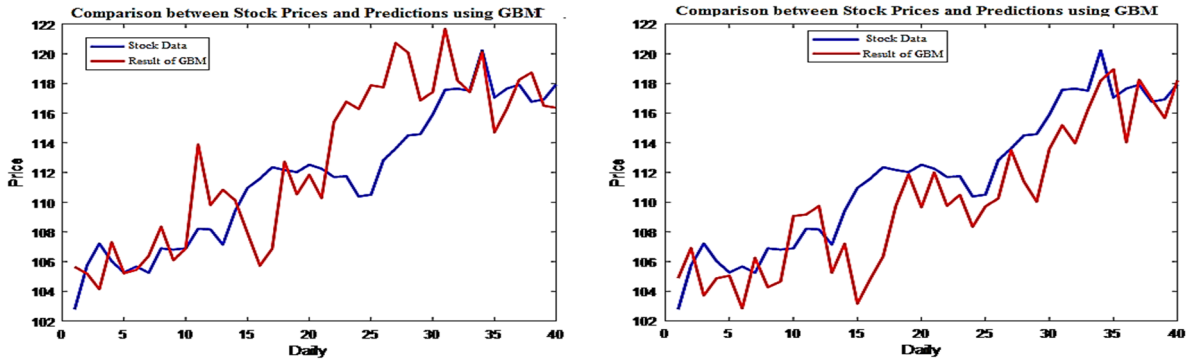


(a) (b)

FIGURE 3. Graph of (a) 10 Iterations and (b) 10000 Iterations.

TABLE 2. MAPE value from prediction result with Geometric Brownian Motion.

Iteration	MAPE Value	Iteration	MAPE Value
1	0.9132	6	0.126
2	0.101	7	0.8689
3	0.1604	8	0.0437
4	0.0221	9	0.2271
5	0.1472	10	0.1174



(a) (b)

FIGURE 4. Comparison between stock data and prediction result using (a) 10 iterations and (b) 10000 iterations.

CONCLUSION

In this paper, the authors predict Microsoft stock price using multilayer perceptron and geometric Brownian motion. The data used are Microsoft stock daily data from October 1, 2018 to March 29, 2019. The stock price prediction of Microsoft using multilayer perceptron method produces a MAPE value of 0.05266. The number of neurons in the input layer and hidden layer don't affect the MAPE value. We run several times to get the best MAPE value. The MAPE value using the geometric Brownian motion method are 0.0221 when 10 trajectories and 0.019571689 when 10000 trajectories. Prediction using geometric Brownian motion is more effective than multilayer perceptron. The prediction results using geometric Brownian motion when 10 trajectories produce MAPE value better than multilayer perceptron. Prediction with multilayer perceptron requires running more than once to get a better MAPE value, whereas prediction with geometric Brownian motion only runs once. From two methods, the writers can be concluded that the geometric Brownian motion method is better than the multilayer perceptron method.

REFERENCES

1. M. Aziz, S. Mintari and M. Nadir, *Manajemen Investasi* (Deepublish, Yogyakarta, 2015).
2. Y. Kara, M. A. Boyacioglu and O. K. Baykan, Expert Syst. Appl. **38**, 5311-5319 (2011).
3. Statista, *Top Companies in The World by Market Value* (2019) available at https://www.statista.com/statistics/263264/top-companies-in-the-world-by-market-value/.
4. J. Zahedi, M. M. Rounaghi, Physica A **438**, 178-187 (2015).
5. M. Qiu, Y. Song and F. Akagi, Chaos Soliton. Fract. **85**, 1-7 (2016).
6. A. H. Moghaddam, M. H. Moghaddam and M. Esfandyari, J. Econ. Financ. Adm. Sci. **21**, 89-93 (2016).
7. K. Reddy and V. Clinton, AABFJ **10**, 23-47 (2016).
8. S. N. Z. Abidin and M. M. Jaffar, Appl. Math. Inf. Sci. **8**, 107-112 (2014).

9. L. V. Fausett, *Fundamentals of Neural Networks: Architectures, Algorithms, and Applications* (Prentice-Hall, London, 1994).
10. S. Kusumadewi, *Artificial Intelligence (Teknik dan Aplikasinya)* (Graha Ilmu, Yogyakarta, 2003).
11. L. Jiang, *Mathematical Modeling and Methods of Option Pricing* (World Scientific Publishing Co, Pte. Ltd, London, 2003).
12. P. Wilmot, *Introduce Quantitative Finance 2nd-Edition* (John Wiley and Son, Ltd Chichester, 2007).
13. Finance Yahoo, *NasdaqGS Real Time Price Currency in USD* available at https://finance.yahoo.com/quote/MSFT/history?p=MSFT/.
14. J. J. Siang, *Jaringan Syaraf Tiruan dan Pemrograman Menggunakan MATLAB 1st Edition* (Andi, Yogyakarta, 2005).