Optimal Trading Strategy Based on BP Neural Network and Dynamic Programming Model

Xinyi Chen*

Beijing Jiaotong University, Beijing, 100044 China

*Corresponding author. Email: chenxinyi220324@163.com

Abstract. In order to avoid the risks brought by blind investment, it is necessary to forecast the trading price of gold and bitcoin on that trading day before investment. To aim at this prediction problem, this paper puts forward the price prediction model of the BP neural network. The experimental results show that the model can reasonably predict the trading price of gold and bitcoin on the next trading day, and the relative error between them is less than 10%.However, predicting the trading price of gold and bitcoin can roughly determine the future direction, and it cannot provide scientific and reasonable trading strategies for traders, so it is necessary to design a trading strategy planning model. Considering that every trading activity except holding financial products will charge additional trade costs, the trade costs of gold and bitcoin are 1% and 2% of the trade amount, respectively. By comparing various classic trading models, we propose to use a dynamic programming model to provide traders with effective trading strategies, which takes the trading changes of gold and bitcoin in the next three trading days as decision variables. Meanwhile, it takes the highest value of the Sharp ratio in the next trading day as the objective function. It takes the proportion of each part as equal to or greater than 0 as the constraint condition to establish a trading strategy planning model to obtain the maximum profit.

Keywords: Quantitative trading, Price projection, BP neural network, Dynamic programming mode.

1. Introduction

More and more people choose to invest in various economic products to expand personal assets in today's society. Currently, the more well-known financial products are gold and bitcoin. In order to maximize returns, quantitative trading is required to develop the most appropriate trading strategy.

Investing in financial products has become an emerging investment method. Bitcoin is a digital currency consisting of a series of complex codes generated by computers and can be circulated around the world. By 2021, the total market value of bitcoin has exceeded the one trillion dollars mark. It should be noted that the trading price of bitcoin fluctuates greatly, while high returns bring high risks. As a recognized financial asset, gold has shown a high degree of activity in different markets, so the fluctuation of the trading price is not large. Quantitative trading is based on specific financial knowledge. With the help of scientific statistics and mathematical tools, computer technology is used to select various "high probability" events that can bring excess returns from massive historical data to formulate a strategy.

2. Model Overview

So far, there are two widely used methods for forecasting trading prices: a forecasting model based on statistical methods, and the other is a forecasting model based on machine learning algorithms. However, after drawing the trading price data into a graph, it can be found that the trading prices of bitcoin and gold have greater volatility over time, and the change in the trading price of bitcoin is particularly obvious. Therefore, the prediction model using statistical methods will not be very accurate in predicting the trading price of bitcoin with large fluctuations in data. At the same time, through the corresponding experiments, it is found that this kind of prediction can only predict the general trend of the price in a long time when the data is relatively sufficient. It cannot predict the specific price in the short-term future. Machine learning algorithms are widely used in financial time

series and have been proven to have better forecasting effects. Therefore, we finally choose to use the deep learning method to make predictions for this kind of volatile data.

3. BP Neural Network Model Principle

BP neural network is one of the most classic and mature prediction methods in deep learning. The advantage is that there is no need to determine the mathematical equation of the mapping relationship between the input and output in advance, and only through its training, it can learn a certain rule, and when the input value is given, the result that is closest to the expected output value can be obtained. BP neural network is a multi-layer feedforward network trained by error backpropagation (referred to as error backpropagation). Its algorithm is called the BP algorithm. Its basic idea is the gradient descent method, which uses gradient search technology to make the network. The error means square error between the actual output value and the expected output value is the smallest.

Specifically, we generally randomly initialize the weights (the coefficients by which the results of the previous neuron are passed to the next neuron) and the bias (the role of the bias will be described in detail below). Pass the sample data into the network, and compare the output with the actual value to see how big the model error is. Since the parameters are randomly initialized, it can be said that the parameter setting is very arbitrary, and it is impossible to determine the model for the first time. There must be a "learning mechanism" to continuously optimize the parameters of the model to the best state. The concept of the gradient is involved here. The neural network tends to choose the fastest "path" to reduce the error every time, that is, to find the minimum value of the error according to the negative gradient direction.

In order to achieve the goal of reducing the error, the neural network needs to be trained many times, and a large number of matrix operations realize the training process. We denote the weight between the ith neuron in the previous layer and the jth neuron in the next layer as w(ij). The three inputs passed in from the previous layer are recorded as Input(1), Input(2), and Input(3), respectively, and the outputs are recorded as Out(1), Out(2), and Out(3). The output in Out(1) naturally comes from Input(1), Input (2), and Input (3) are multiplied by weights, respectively:

$Out(1) = Input(1) \times w(11) + Input(2) \times w(21) + Input(3) \times w(31)$ With the output, the error can be calculated, but there are two or more outputs that must be considered when calculating the error. Taking two outputs as an example here, the calculation formula of the error is:

$$Cost = \frac{1}{2}\left(target_{O_1} - Out_{O_1}\right)^2 + \frac{1}{2}\left(target_{O_2} - Out_{O_2}\right)^2$$

Where target refers to the actual value (target value), and out refers to the output value of the neural network. There is a 1/2 coefficient in front because when the derivative is obtained later, the quadratic power of the parentheses cancels out 1/2. Since there are two outputs here, the errors of the two outputs need to be added up.

In order to reduce the error just calculated, we need to adjust the values of weights and biases between neurons. We need to take the derivative of the loss function concerning the weights and biases. According to the chain rule of derivation, the intermediate variables can be written out, and then the intermediate variables can be solved to obtain the final solution.

$$\frac{\partial Cost}{\partial w_5} = \frac{\partial Cost}{Out_{O_1}} \times \frac{Out_{O_1}}{\partial Input_1} \times \frac{\partial Input_1}{\partial w_5}$$

$$\frac{\partial Cost}{\partial w_5} = \partial_{O_1} Out_{h_1}$$

The derivation of the bias is relatively simple. It is first propagated to the node of $O1$, and when it is propagated to the bias, the coefficient before the bias is 1 during the weighted summation, so the result is still itself. For the previous weights, the recursive idea can be combined with the above derivation process to obtain the final calculation formula:

$$\frac{\partial Cost}{\partial w_1} = \partial_{h_1} \times \frac{\partial Input_{h_1}}{\partial w_1} = \left(\partial_{O_1} \times w_5 + \partial_{O_2} \times w_7\right) \times Input_{h_1}\left(1 - Input_{h_1}\right) \times Out_{i_1}$$

After the above methods repeatedly train the neural network, it can finally achieve the effect of learning from a large amount of data, finding out the rules, and then making predictions for the future.

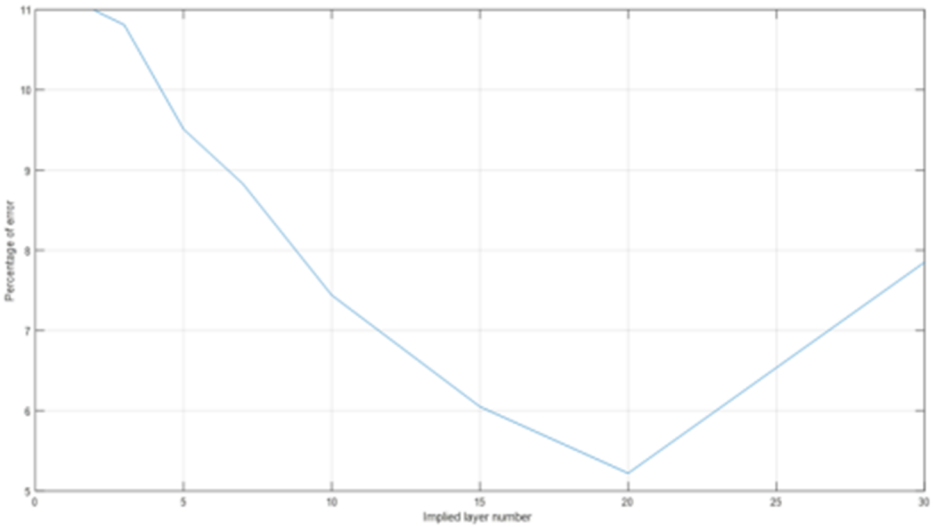

Figure 1 Hidden Layers of BP Neural Network

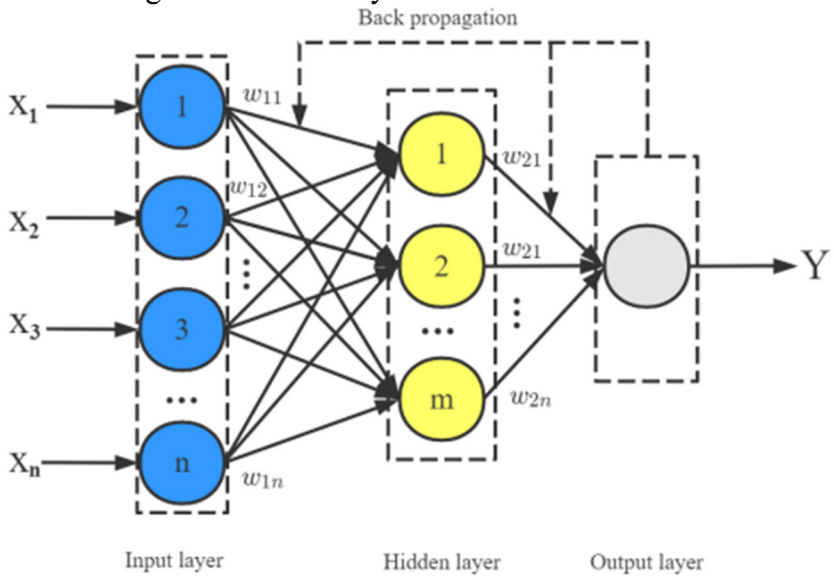

Figure 2 Schematic Diagram of Neural Network Principle

3.1 Data Processing

Before training a neural network, the data needs to be processed first. It can be found that there are some null values in the data. In view of a large amount of data, eliminating a small part of the data has little effect, so we delete these data directly.

Different dimensions are used among gold, bitcoin, and US dollars, which will affect the prediction of data. In order to eliminate this influencing factor, each variable needs to be standardized. It simplifies the data while retaining the original characteristics of the data and improves the training efficiency without affecting the neural network training effect.

3.2 Train the Neural Network

In this prediction, we used a three-layer neural network: an input layer, a hidden layer, and an output layer. The input layer receives the time information, and the output layer outputs the predicted price. The number of hidden layers is continuously adjusted according to the actual learning situation. Finally, when the hidden layer is above 20, the prediction result does not change much and almost

reaches the relatively best situation, so we think that the number of hidden layers can be selected as 20 layers.

To train the network, 80% of all data is used to train the model, which remains 20% to test the model. The comparison between the predicted value and the actual value of the obtained model is as follows. Figure 1 shows the comparison between the predicted value and the actual value of bitcoin price, and Figure 2 shows the comparison between the predicted value and the actual value of gold price.

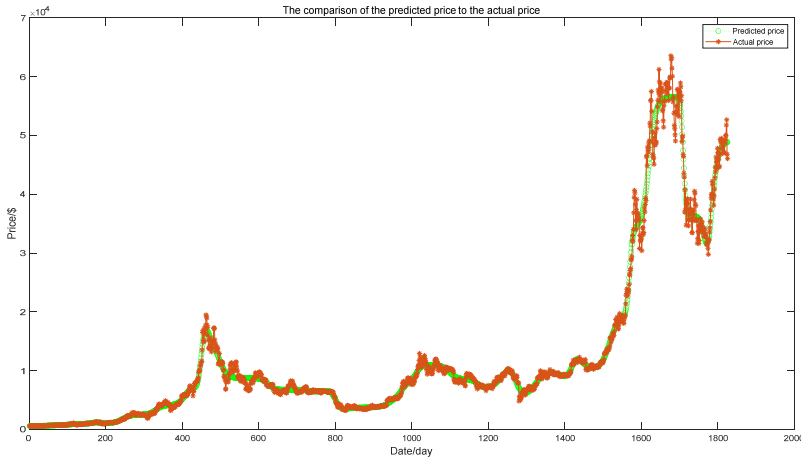

Figure 3 The Comparison Between The Predicted Value And The Actual Value of Bitcoin Price

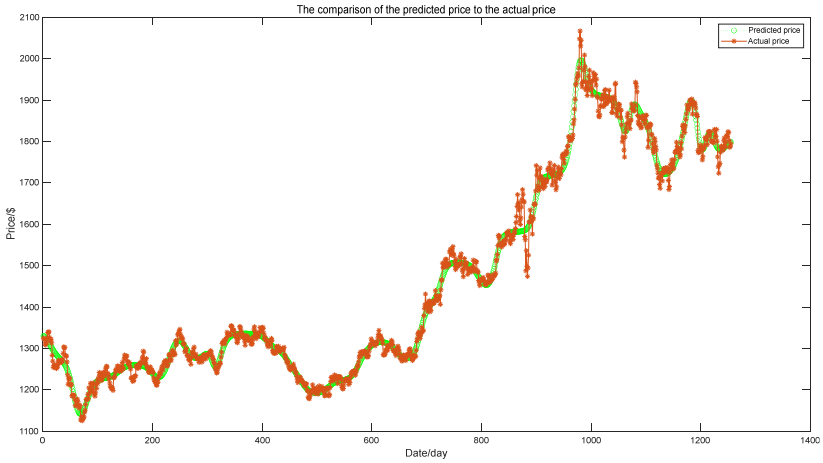

Figure 4 The Comparison Between The Predicted Value And The Actual Value of Gold Price

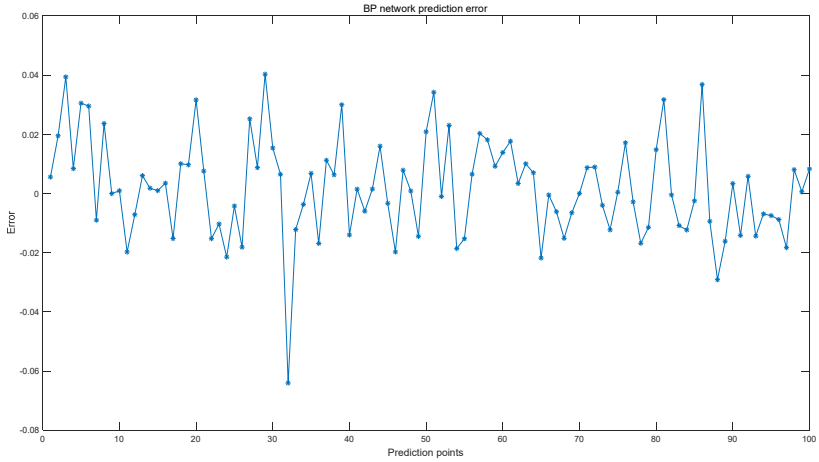

Figure 5 Error Between Prediction and Actual Value Based on BP Neural Network

From the above figures, it can be found that the BP neural network has a good degree of fitting to the prices of Bitcoin and gold and is also very accurate with short-term trend prediction. Although

the short-term mutation value is not handled well, it meets the forecasting requirements and can be used to assist the next investment decision model.

3.3 Quantitative Trading Strategy Based on Dynamic Programming

3.3.1 Analysis of Dynamic Programming Problems

According to the meaning of the question, it can be concluded that on this trading day, our known information is the previous transaction price information to find the optimal trading strategy to obtain the maximum profit. On the next trading day, the trading price data of the trading day will be added to the historical data again and based on this, the prediction for the next trading day will be made. Such a prediction process belongs to the dynamic programming process. Based on the previous forecast results, the trading strategy for the trading day can be formulated, and the recursion can be continued until the last day of the trading day according to the above description.

3.3.2 Model Assumptions

In order to simplify the subsequent model establishment, the following assumptions are made based on the known information. The total assets held on the *N*th day are divided into gold, bitcoin, and cash, and each proportion can be recorded as $[\mathrm{G_N}, \mathrm{B_N}, \mathrm{C_N}]$, and the ratio between the three. The relationship will satisfy the following relationship:

$\mathrm{G_N}+\mathrm{B_N}+\mathrm{C_N}=1$

Record the change in the proportion of gold and bitcoin in total assets as $[\Delta \mathrm{g_N}, \Delta \mathrm{b_N}]$.

Assume that on the *N*th trading day, the growth rates of gold, bitcoin, and cash are $\left[\overline{\mathrm{g_N}}, \overline{\mathrm{b_N}}, 0.0055\%\right]$ respectively.

In order to avoid the investment risk more accurately and obtain the maximum return, the concept of the Sharp ratio is introduced on this basis. The sharp ratio is one of the three classic indicators that can comprehensively consider the return and risk simultaneously, that is, those portfolios that maximize the expected return at a given risk level or those that minimize the risk at a given expected return level. The Sharp ratio is calculated as follows:

$$\mathrm{SR} = \frac{\mathrm{E(R_P)} - \mathrm{R_f}}{\sigma_\mathrm{p}}$$

In the formula, $\mathrm{E(R_P)}$, $\mathrm{R_f}$ and σ_p represent the expected return rate of the portfolio, risk-free interest rate, and standard deviation of return rate of the portfolio, respectively.

3.3.3 Planning Model Building

On the *N*th trading day, the objective function $F(N)$ is set to the value of the Sharp ratio, which adjusts the position on the trading day and does not change in the next three days. When $\mathrm{F(N)}$ reaches the maximum value, it reaches the state of balance between risk and benefit.

Before the end of each trading day, it is necessary to settle each part's price increase and decrease. After the growth of gold, bitcoin, and cash, the proportion of each part will become $\left[(1+\overline{g_N})G_N, \left(1+\overline{b_N}\right)B_N, (1+0.0055\%)C_N\right]$. As the total assets held have changed, the sum of the proportions of gold, bitcoin, and cash may no longer be 1. After all the settlement is completed, it will be normalized to perform the same operation on the *N*+*1*th trading day.

When we only need to buy gold on the trading day, record the change of the proportion of gold in total assets as $\Delta b_N > 0$, then the change of cash in total assets as $1.02\Delta b_N$. When it only need to sell gold on the trading day, record the change of gold in total assets as $\Delta b_N < 0$, then the change of cash in total assets as $0.98\Delta b_N$.

After the full settlement, the proportion of gold, bitcoin, and cash will become

$\left[(1 \pm \overline{g_N})G_N + \Delta g_N, \left(1 \pm \overline{b_N}\right)B_N + \Delta b_N, (1+0.0055\%)C_N - (1 \pm 0.01)\Delta g_N - (1 \pm 0.02)\Delta b_N\right]$.

Then, it will be normalized, and the proportion of the three parts will be multiplied by the total amount of the previous trading day to get the total amount and return rate of the trading day and the proportion of each part of the trading day.

According to the assumption, for the objective function, the amount of each part needs to remain unchanged in the next three trading days. Therefore, both $\Delta g_{N+1}$and Δb_{N+1}in the next trading day equals 0, and then the Sharp ratio of that day will be calculated until the next trading day.

The constraint condition of the planning model is that when all operations are completed on the trading day, the proportion of gold, bitcoin, and cash in the total assets should be greater than or equal to 0, and the following inequality groups can be obtained:

$(1 + 0.0055\%)\mathrm{C_N} + (1 \pm 0.01)\Delta\mathrm{g_N} + (1 \pm 0.02)\Delta\mathrm{b_N}"0(1 \pm \overline{\mathrm{g_N}})\mathrm{G_N} - \Delta\mathrm{g_N}"0$

$\left(1 \pm \overline{\mathrm{b_N}}\right)\mathrm{B_N} - \Delta\mathrm{b_N}"0$

3.3.4 Model Optimization

It can be found that when the above-mentioned planning model is established, the objective function $F(N)$ will be set to adjust the position of the trading day without any change in the next three days. However, in practice, keeping the position unchanged will have an impact on the trading strategy of each trading day, which will lead to the situation of selling at a low price and buying at a high price, so it is necessary to optimize the model further.

We keep the constraints of the model unchanged, set the $[\Delta g_{N+1}, \Delta b_{N+1}, \Delta g_{N+2}, \Delta b_{N+2}]$ value of the variable, and reuse the model to solve the optimal trading strategy.

However, the range is difficult to determine due to the large number of decision variables, and the convergence speed is too slow in the solution process. Therefore, a particle swarm optimization algorithm speeds up the convergence.

Particle swarm optimization algorithm, or PSO algorithm for short, is a swarm intelligence optimization algorithm in the field of computational intelligence. The PSO algorithm first initializes a group of particles in the feasible solution space. Each particle represents a potential optimal solution of the extremum optimization problem, and each particle has three indexes of position, velocity and fitness value. Particles move in the solution space, and individual positions are updated by tracking individual extremum and population extremum. Individual extremum refers to the optimal position of fitness value calculated from the positions experienced by individuals, and population extremum refers to the optimal position of fitness searched by all particles in the population.

Compared with the traditional genetic algorithm, PSO has the function of memory, and it belongs to one-way information flow. The whole search and update process is the process of following the current optimal solution. Therefore, in general, the convergence speed of PSO is faster. Therefore, here we choose a particle swarm optimization algorithm to optimize the model.

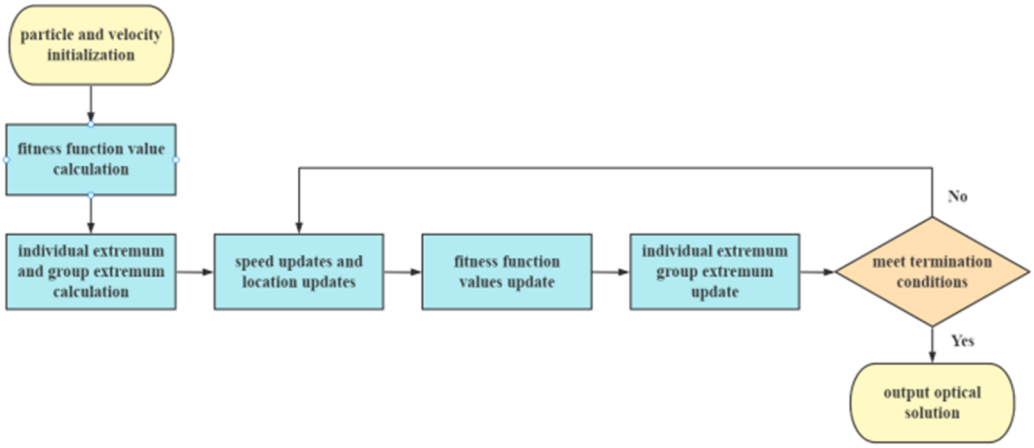

Figure 6 PSO Algorithm Flow Chart

3.3.5 Optimizing Model Results

According to the above-optimized model, and using the Sharpe ratio of the next trading day as the maximum profit of $F(N)$, it can be concluded that when the initial principal is US$1,000, the final capital can be US$1,289.22.

3.4 Sensitivity Analysis

3.4.1 Proof of Optimal Strategy

On the basis of the original model, we interfered with some parameters and re-tested the performance of the new scheme. Here, the local mean value of the objective function is taken as the object of interference to make it deviate from the original value. The interference factors and the final revenue changes are shown in the following table:

Table 2 Earnings After Interference

Parameter	No Change	1%	5%	10%	20%	50%
Returns($)	60569.56	60232.89	53381.56	6358.31	1536.76	0

It can be seen that the final result is not as good as the original model. Therefore, the investment scheme we have chosen is the best one.

3.4.2 Determine How Sensitive the Trading Strategy is to Trading Costs

In order to verify the trading strategy we formulated before, what kind of fluctuations will occur when the fee ratio of trade costs change, adjust the trade cost of gold to $G\%$ of the trade amount, and adjust the trade cost of bitcoin to $B\%$ of the trade amount, use the same verification strategy to reinvest, it can get the data in the following Table 3.

Table 3 Earnings After Adjusting Trade Cost (a%)

Number	No change	1	2	3	4	5
$\boldsymbol{G}\%$	1%	1.2%	0.8%	1%	1%	1.2%
$\boldsymbol{B}\%$	2%	2%	2%	2.4%	1.8%	1.8%
Expect returns($)	60569.56	54686.21	55916.40	35967.59	75237.42	66672.63

From the above table, it is not difficult to find that, based on the investment strategy of the above model, although the trade cost of gold has a specific influence on the final benefit, it is generally not as influential as the trade cost of Bitcoin. Therefore, the theoretical income generated by this model is sensitive to the trade cost of bitcoin.

4. Model Improvements

Due to the constraints of the question conditions, when training the mmm model, we can only use the past transaction prices of gold and bitcoin to train the model, but because the future transaction prices of gold and bitcoin are not only affected by past prices, but also include political, environmental, and other variables. Therefore, some model training results have no high reference value for trading strategies.

In formulating real trading strategies, we can add more characteristic variables. For example, bitcoin holding share, gold holding share, total asset change rate, calculate the importance between each feature and future transaction price, and select more important features to predict future transaction prices more accurately.

5. Summary

The result closest to the expected output value can be automatically obtained by training available data when the input value is given. The number of intermediate layers and the number of neurons in each layer of the neural network can be arbitrarily set according to the specific situation. However,

The learning speed of the BP neural network is very slow. Even a simple problem usually needs hundreds or even thousands of times of learning to converge. In some cases, BP neural network is easy to fall into a local minimum.

References

[1] Takahiro Hattori and Ryo Ishida. The relationship between arbitrage in futures and spot markets and Bitcoin price movements: Evidence from the Bitcoin markets, volume 58, issue 4.

[2] Saketh Aleti and Bruce Mizrach. Bitcoin spot and futures market microstructure, 2017, vol. 4, no. 4, pp. 41–47.

[3] Xiafei Li, Dongxin Li, Xuhui Zhang, Guiwu Wei, Lan Bai, and Yu Wei. Forecasting regular and extreme gold price volatility: The roles of asymmetry, extreme event, and jump, 2021, pp. 126–130.

[4] Manh Cuong Pham, Huu Nhan Duong, and Paul Lajbcygier. A Comparison of the Forecasting Ability of Immediate Price Impact Models, 2016, pp. 31–34.

[5] Angela J. Black, Olga Klinkowska, David G. McMillan, and Fiona J. McMillan. Forecasting Stock Returns: Do Commodity Prices Help?. 2014, vol. 33, no. 8, pp. 9-13