A Stock Decision Support System Based on ELM

Chengzhang Zhu, Jianping Yin and Qian Li

Abstract People often tend to use a reliable way to predict the stock market in order to get a substantial return on investment. However, with plenty of uncertainty and noise, prediction is full of challenging and risk when it comes to stock markets. This chapter combines extreme learning machine (ELM) and the Oscillation box theory together to construct a stock decision support system, which can help people make decisions on stock trading through suggestion buy or sell stock. In experiments, 4 typical stock movements have been tested trading and 400 stocks in S&P500 are used to detect the performance of the system. Results show that our method is much better than buy-and-hold strategy.

Keywords Stock predict · ELM · Oscillation box theory

1 Introduction

The study of stock market, which helps people make lucrative investment decisions, is a focus of attention. However, owing to the fact that stock market indices are essentially dynamic, nonlinear, complicated, nonparametric, and chaotic, the stock time-series forecasting is regarded as one of the most challenging applications of time-series forecasting [1]. In recent years, a lot of work had been done and trying to analyse and predict stock prices or trends in the future [2]. Although nobody

C. Zhu (✉)
College of Computer, National University of Defense Technology, Changsha 410073, China
e-mail: kevin.zhu.china@gmail.com

J. Yin
State Key Laboratory of High Performance Computing, National University of Defense Technology, Changsha 410073, China

Q. Li
School of Economics, Minzu University of China, Beijing 100081, China

F. Sun et al. (eds.), *Extreme Learning Machines 2013: Algorithms and Applications*,
Adaptation, Learning, and Optimization 16, DOI: 10.1007/978-3-319-04741-6_6,

can predict a stock market with high enough accuracy, one could be able to predict the overall trend in this market based on historical data. Therefore, people may obtain high profit by using some well trading strategies with the prediction results. A successful stock market prediction is characterized by achieving best results using minimum required input data and the least complex stock market model [3]. Since it is affected by many macro economic factors, the stock market cannot be well comprehensively described by traditional model [4]. As a comparatively accurate solution can be found in the complex, noisy environment due to artificial neural networks [5], lots of attentions have been devoted to applying different neural networks into stock prediction [6–9]. In addition, with the application of SVM in regression, some work introduced SVM method in stock market prediction and got outstanding result [10, 11]. All the methods mentioned above have made breakthrough achievements, which made the stock market prediction significantly accurate and robust however, there still exists an unresolved problem that is the speed will become very slow when a large number of historical data in the stock market need to be learned. It limits computer dynamic learning new data.

Recently, a new type of learning machine called extreme learning machine (ELM), which is a methodology for learning single-hidden layer feedforward neural networks (SLFN) and is proposed by Huang et al. [12–15], has been proved to be extremely fast and it can also provide excellent generalization performance. Different with the traditional neural network training algorithms such as back-propagation algorithm (BP), ELM does not need any other extra time to adjust the hidden weights and biases since it chooses them at random and then obtains the output layer weights and biases analytically. For this reason, we can introduce ELM to forecast the stock price trends in the future in order to get a better performance in a short time.

A powerful trading strategy is necessary for stock transactions. Nicolas proposed a box theory, which indicates the price of stock would generally oscillate in a certain range in a period of time named price box. The price will fall when it is close to the upper boundary of the price box and rise on the contrary. If the price breaks the upper boundary or the lower boundary of the oscillation box, it will enter another oscillation box in which the price will start a new upward or downward trend. So it will be the best time to buy or sell the stock [16]. It is fairly clear that the most important and difficult work is to accurately identifying the boundary of the box and confirm the price breakout it, since one can only predict it based on experience in daily life.

The box theory and extreme learning machine algorithm are combined in this chapter. We train extreme learning machine by history price data and utilize it to predict the highest and lowest stock price in the next period as the upper and lower boundary of the oscillation box. Meanwhile, we have developed an inspection rule to confirm whether the stock price breakout the boundary or not. Then we can formulate our trading strategy based on the box theory to make decisions. Experiments show that our approach has obvious performance advantages compared to hold-and-buy strategy in which an investor buys stocks and holds them for a long period of time, regardless of fluctuations in the market. The advantages of the systems are mainly reflected in two aspects. On one hand, the system has a great learning speed to learn

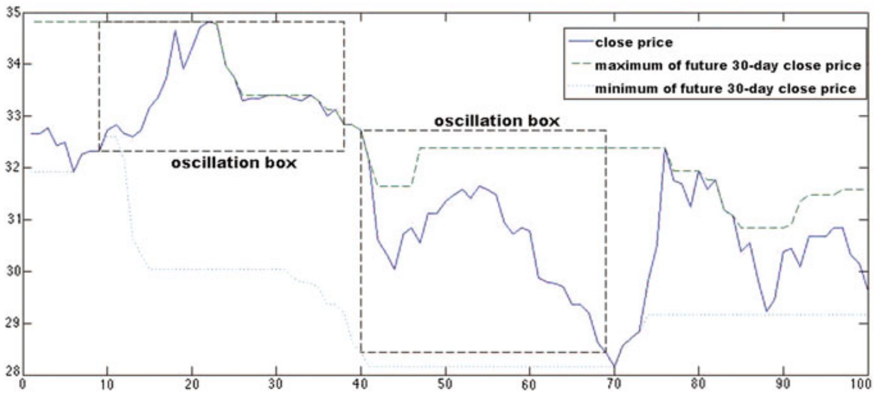

Fig. 1 The oscillation box samples

from a large number of historical stock price data and make decisions. On the other hand, the system can always get a better performance than it compared with other methods.

The remainder of this chapter is organized into four sections. Section 2 briefly reviews the oscillation box theory and extreme learning machine theory. Section 3 details our system trading strategy. Section 4 shows experiments and analysis. Finally, Sect. 5 contains the concluding remarks.

2 Related Work

2.1 Oscillation Box Theory

Nicolas proposed the Oscillation box theory. The basic idea of this theory is that the stock price is always has a certain shock range in a period of time, thus it has a maximum and a minimum price during this time. Imaging there are two ends of a box—the upper boundary and the lower boundary, thus Nicolas had it called oscillation box in his theory. When the stock price close to the lower boundary it has the rising trend and on the contrary close to the upper boundary. Furthermore, the price will go into another box to start a new shock in a range after it breaks through the boundary. The Oscillation box is showed in Fig. 1. Obviously, we can get a fruitful profit if we buy the stock when the price breaks the upper boundary and sell it as soon as it breaks the lower boundary. However, effective to detect the price when it breaks through the boundary, which is always based on experience, is quite challenging. In our system, we proposed a method to detect it automatically based on the ELM prediction.

2.2 Extreme Learning Machine

Extreme learning machine is a novel algorithm proposed by Huang et al. in [14]. The theory provides a new approach to training the single hidden layer feedforward networks, which makes the training completed within a very short time to achieve the effect of extreme learning. A SLFN consists of three layers, namely input layer, hidden layer and output layer. We can train the network by adjusting the connection weights and biases of layers.

Denote the numbers of nodes in input, hidden and output layers as n_1, n_2 and n_3, we can represented a SLFN by

$$\boldsymbol{t}_r = f_r(\boldsymbol{x}_j) = \sum_{i=1}^{n_2} \beta_{ir} G_i(\boldsymbol{a}_i, b_i, \boldsymbol{x}_j) \qquad (j = 1, 2, \ldots, n_1; r = 1, 2, \ldots, n_3). \quad (1)$$

where $\boldsymbol{t}_r = [t_{r1}, t_{r2}, \ldots, t_{rn}]^T$ is the output vector; $\boldsymbol{x}_j = [x_{j1}, x_{j2}, \ldots, x_{jn_1}]^T$ is the input vector; $\boldsymbol{a}_i = [a_{1i}, a_{2i}, \ldots, a_{n_1 i}]$ represents the connection weights between the input layer and ith node in the hidden layer; $\boldsymbol{\beta}_i = [\beta_{i1}, \beta_{i2}, \ldots, \beta_{in_3}]^T$ represents the connection weights between the ith node in the hidden layer and the output layer; b_i means the ith hidden node bias; $G_i(\boldsymbol{a}_i, b_i, \boldsymbol{x}) = g(\boldsymbol{a}_i \cdot \boldsymbol{x} + b_i)$ is the value provided by the network for $\boldsymbol{x}_i$ in the hidden layer, where $g(\cdot)$ represents the activation function of the hidden layer. The $g(\cdot)$ can have a variety of options such as Sigmoid function, Sine function, Hard Limit function, Triangular basis function and Radial basis function.

The above Eq. (1) can be written compactly as

$$\boldsymbol{T} = \boldsymbol{G} \cdot \boldsymbol{\beta}. \quad (2)$$

where

$$\boldsymbol{G} = \begin{bmatrix} g(\boldsymbol{a}_1 \cdot \boldsymbol{x}_1 + b_1) & \ldots & g(\boldsymbol{a}_{n_2} \cdot \boldsymbol{x}_1 + b_{n_1}) \\ \vdots & \ddots & \vdots \\ g(\boldsymbol{a}_1 \cdot \boldsymbol{x}_n + b_1) & \ldots & g(\boldsymbol{a}_{n_2} \cdot \boldsymbol{x}_n + b_{n_1}) \end{bmatrix}_{n \times n_2}. \quad (3)$$

$$\boldsymbol{\beta} = \begin{bmatrix} \boldsymbol{\beta}_1 \\ \boldsymbol{\beta}_2 \\ \vdots \\ \boldsymbol{\beta}_{n_2} \end{bmatrix}_{n_2 \times n_3}. \quad (4)$$

$$\boldsymbol{T} = \begin{bmatrix} \boldsymbol{t}_1 \\ \boldsymbol{t}_2 \\ \vdots \\ \boldsymbol{t}_n \end{bmatrix}_{n \times n_3}. \quad (5)$$

The extreme learning machine is trying to minimize the empirical and structural error by adjusting the weights and biases. The objective can be written as

$$minE(\boldsymbol{a}_i, \boldsymbol{\beta}_i) = \sum_{r=1}^{n} ||\boldsymbol{t}_r - \boldsymbol{T}_r||. \tag{6}$$

where $\boldsymbol{T}_r$ represents the real target values. In ELM theory, $\boldsymbol{a}_i$ and b_i, which are the weights and biases of hidden layer, can be randomly assigned. We only need to focus on $\boldsymbol{\beta}_i$, which are the weights of output layer. The theory provides that solving the optimization problem Eq. (6) is equivalent to Eq. (2) for its least square solution $\boldsymbol{\beta}$. It will be easy to get the weights $\boldsymbol{\beta} = \boldsymbol{G}^{\dagger} \cdot \boldsymbol{T}$ based on the Moore Penrose generalized inverse matrix theory, where $\boldsymbol{G}^{\dagger}$ is the generalized inverse matrix of $\boldsymbol{G}$.

2.3 Gray Correlation Degree (GCD)

In [17], Deng proposed the gray correlation degree, which has been applied in many fields [18]. The method is using the geometric shape of sequence curves to present the relational degree between two data sequences. The closer the two curves are, the higher degree is it. If we have a feature $\boldsymbol{X} = [x_1, x_2, \ldots, x_n]$ and target $\boldsymbol{T} = [t_1, t_2, \ldots, t_n]$ we can calculate the feature GCD as follow:

$$r(t_i, x_i) = \frac{min|t_i - x_i| + \xi max|t_i - x_i|}{|t_i - x_i| + \xi max|t_i - x_i|}. \tag{7}$$

$$r(\boldsymbol{T}, \boldsymbol{X}) = \frac{1}{n} \sum_{i=1}^{n} r(t_i, x_i). \tag{8}$$

where $\xi \in (0, 1)$ is the discernibly coefficient which often set to 0.5. The $r(t_i, x_i)$ is the gray correlation degree of $\boldsymbol{T}$ and $\boldsymbol{X}$ at ith point. The $r(\boldsymbol{T}, \boldsymbol{X})$ is the gray correlation degree of $\boldsymbol{T}$ and $\boldsymbol{X}$.

3 Detail of the Decision Support System

The decision support system is in accordance with the following steps. First, the system calculates the related indicators from the historical data of the stock market and scales it. Then it obtains relationships of the scaled indicators and stock prices time-series, which could set as the input value weight of the ELM. Next step comes to training the ELM, using the weighted indicators sequence as input values and stock history prices time-series as target values. The third step is using the trained ELM to predict the stock price sequence for the next period of time in order to

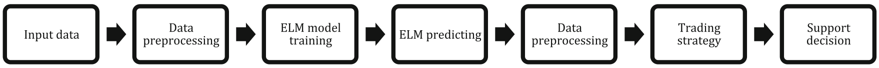

Fig. 2 Support system architecture

get the minimum and maximum values of stock prices, which can set as the lower and upper bounds of the oscillation box. Finally, the transaction is carried out in accordance with the trading strategy based on box theory. The architecture of the system is showed in Fig. 2.

3.1 Pre-processing History Indictors

The system uses closing price to present stock market price, which is the target value. We have selected some indicators as the ELM input feature values, which are OPEN, HIGH, LOW, CLOSE, VOL, AMOUNT, MA, ROC, RSI, FASTK, SLOWK, SLOWD. The computations can be found in [19]. In order to get the boundary in next few days through prediction the stock market time-series, the boundary in period few days can be used as features. In this system, we use the highest price and lowest price of the stock in the future n_1 days as the upper boundary Up_k and lower boundary Low_k.

$$Up_k = max(C_{i+1}, C_{i+2}, \ldots, C_{i+n_1})$$

$$Low_k = min(C_{i+1}, C_{i+2}, \ldots, C_{i+n_1})$$

were C_k represent the closing price in the kth day.

There are totally 14 indictors and 1 target in our system. If a indictor data sequences are $X = (x(1), x(2), ..., x(n))$, all data of the indictor will be normalized to $[-1, 1]$ by

$$x(i)_{normalize} = -1 + 2\frac{x(i) - min(X)}{max(X) - min(X)} \tag{9}$$

Otherwise, the prediction result will be denormalized by

$$p_{denormalize} = \frac{p(max(X) - min(X)) + max(X) + min(X)}{2}. \tag{10}$$

where p is the prediction result.

However, the influence degrees of these indicators on the prediction results are not equal. Obviously, enlarge the indicator, which has considerable impact on the result, can help get a more accurate prediction result. Therefore we use the gray relation analysis method to get relationships of the scaled indicators and stock market prices time-series and use it as the input weight w_i.

3.2 Stock Prices Prediction Based on ELM

This system needs to predict prices in the next n_1 days based on history data in n_2 previous days. We define the target vector as $\boldsymbol{T}_i = [C_{i+1}, C_{i+2}, \ldots, C_{i+n_1}]$, the feature vector as

$$\boldsymbol{F}_i = [O_k, H_k, L_k, C_k, VOL_k, MA_k, ROC_k, RSI_k, FastK_k, SlowK_k, SlowD_k, Up_k, Low_k]$$

where $k = (i-1, i-2, \ldots, i-n_2)$, all features are indictors mentioned in Sect. 3.1.

Since the input weights w_i has been acquired based on gray relation analysis method, we can descript the weight vector as $\boldsymbol{W}_i = [w_{O_k}, w_{H_k}, \ldots, w_{Low_k}]$, where $k = (1, 2, \ldots, n_2)$. The input vector $\boldsymbol{I}_i$ can calculate as follow:

$$\boldsymbol{I}_i = \boldsymbol{F}_i \circ \boldsymbol{W}_i. \tag{11}$$

where $\circ$ is Hadamard product, i presents ith day.

Due to the recent data do more contribution to learning stock market, we need to set up a window that contains recent data for ELM training. If we want to predict stock prices after ith-day, and the window size is set to n days. The input vectors can form a matrix as $\boldsymbol{I} = [\boldsymbol{I}_{i-n}, \boldsymbol{I}_{i-n+1}, \ldots, \boldsymbol{I}_{i-n_1}]$ while the target vectors can form a matrix as $\boldsymbol{T} = [\boldsymbol{T}_{i-n}, \boldsymbol{T}_{i-n+1}, \ldots, \boldsymbol{T}_{i-n_1}]$. After training ELM, which used $\boldsymbol{I}$ as input matrix and $\boldsymbol{T}$ as target matrix, we can predict stock price $\boldsymbol{T}_i$ using input vector $\boldsymbol{I}_i$. The upper boundary and lower boundary can set to maximize and minimize price of $\boldsymbol{T}_i$.

3.3 Trading Strategy Based on Box Theory

Our trading strategy is based on the oscillation box theory. After predicting the upper and lower boundary, which are described as Up_i and Low_i, in next n_1 days after ith-day, we need to set a standard to detect whether the price series crossing the border. Obviously, two conditions need to be met when the price series up through the box. The first thing is that the price is very close to the lower boundary of the new box, and the second thing is that the lower boundary of the new box is moved upward. Similarly, the price will close to the upper boundary of the new box and the upper boundary of the new box will move downward when it crosses the lower boundary. Thus our strategy can be defined as Algorithm 1.

Algorithm 1 Trading strategy.

if next trade $==$ buy **then**
  if $\frac{|C_i - Low_i|}{C_i} \leqslant \sigma$ and Low_i is in uptrend **then**
    if $sellprice - C_i \geqslant \varphi$ **then**
      Buy,$buyprice = C_i$
      next trade=sell
    end if
  end if
else if $\frac{|C_i - Up_i|}{C_i} \leqslant \sigma$ and Up_i is in downtrend **then**
  if $C_i - buyprice \geqslant \phi$ **then**
    Sell,$sellprice = C_i$
    next trade $=$ buy
  end if
  if $\frac{buyprice - C_i}{buyprice} \geqslant \theta$ **then**
    Sell,$sellprice = C_i$
    next trade=buy
  end if
end if

4 Experiments

We conducted some experiments to verify the system's feasibility and efficiency. In these experiments, several typical stock movements, such as bull market, bear market, fluctuant market and so on, are selected to carry out a comparative analysis. After that we tested 400 stocks in the S&P500, in order to detect the average performance of our system. Finally, the optimal sets of parameters are discussed and tested. All of the experiments are run in MATLAB environment.

4.1 Performance Evaluation

There are two performance indicators in our experiments. One of them is MSE (mean squared error), which is used to illustrate the accuracy of ELM regression. The MSE is defined as follow:

$$MSE = \frac{1}{N}\sum_{i=1}^{N}(y_i - y_i^*)^2. \tag{12}$$

where y_i is the actual output and y_i^* is the estimate.

The other is rate of profit which can defined as

$$\text{rate of profit} = (Y - Y_0)/Y_0 \times 100\,\%. \tag{13}$$

where Y is the money after trade and Y_0 is the initial money.

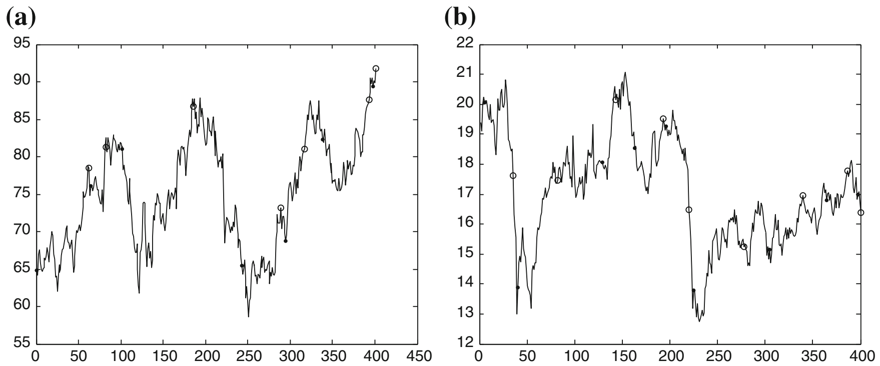

Fig. 3 **a** The fluctuant and bull market movement, where • means buy point and ∘ means sell point. **b** The fluctuant and bear market movement

In our experiments, we suppose \$10000 initial money and use all money or stock to trade at each operation. As the real trading, we set the transaction cost of each trading 0.5 %. Then we let the system trade on a stock for a period of time and get the average MSE and rate of profit in the final. In particular, we short-selling of all stocks hold at the last day of test trading.

4.2 Typical Stock Movement Trading

A movement of fluctuant and bull market is showed as Fig. 3a. The transaction rate σ is set to 0.01 and stop-loss rate θ is set 0.1 and φ, ϕ set to 0, 0.05, respectively. The window size is 120. In the experiment, our system profits to 93.20 % while the market gains about 41.69 %. The MSE of the ELM is $7.2957e^{-30}$. The data set is samples from March 13, 2002 to August 29, 2003.

A movement of fluctuant and bear market is showed as Fig. 3b. The transaction rate σ is set to 0.01 and stop-loss rate θ is set 0.1 and φ, ϕ set to 0, 0.05, respectively. The window size is 120. In the experiment, our system can profit to 38.61 % while the market losses about 15.48 %. The MSE of the ELM is $5.6965e^{-30}$. The data set is samples from October 26, 2000 to June 6, 2002.

Obviously, our system is significantly better than the buy-and-hold strategy in a fluctuant market.

A movement of overall bull market is showed as Fig. 4a. The transaction rate σ is set to 0.05 and stop-loss rate θ is set 0.1 and φ, ϕ set to 0, 0.05, respectively. The window size is 120. In the experiment, our system can profit to 137.17 % while the market gains about 94.38 %. The MSE of the ELM is $1.0645e^{-29}$. The data set is samples from March 17, 2004 to October 17, 2005.

A movement of overall bear market is showed as Fig. 4b. The transaction rate σ is set to 0.05 and stop-loss rate θ is set 0.1 and φ, ϕ set to 0, 0.05, respectively.

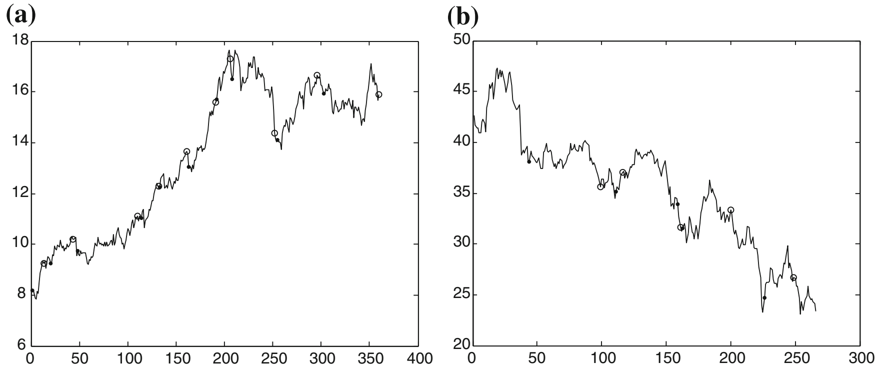

Fig. 4 **a** The overall bull market movement. **b** The overall bear market movement

The window size is 90. In the experiment, our system only losses 2.74 % while the market losses about 45.13 %. The MSE of the ELM is $2.3462e^{-30}$. The data set is samples from August 8, 2001 to September 9, 2002.

4.3 Trade on S&P500

We simulate trading of 400 stocks in 400 trading days, which are selected in S&P500 from Mar 18, 2004 to Oct 17, 2005, to examine the average performance of the system. There are 212 of these stocks movement are bull, the other are bear. In these trading test, the same parameters are used. The transaction rate σ is set to 0.01 and stop-loss rate θ is set to 0.05 and φ, ϕ set to 0, 0.05, respectively. While the window size is set to 120. The test results are showed in Table 1. Clearly, our system is in most cases superior to the buy-and-hold strategy, and can gains a much higher average profit.

4.4 The Optimal Sets of Parameters

Firstly, we consider the impact on the training window size. Obviously, the learning of history data in a period of time before can generate important guiding significance for the prediction of future case. However, if we take a long time to learn the prediction accuracy, it might be decreased because of much noise data. At the same time, if we take a short time to learn, it might lack of experience. On experience, a stock with bull movement always has less noise data over a longer period of time. Conversely, a stock with bear movement is often accompanied by more noise data. Therefore, if we set a large window size to a bull movement stock and set a small size to a bear movement stock, the system may give better results. We have already conducted experiment

Table 1 Average performance of trading 400 stocks in S&P500 for 400 days

Market pattern	Stock number	Less than buy-and-hold	Less (%)	Loss number	Loss (%)	Average profit (%)	Average profit of buy-and-hold (%)
Bull	212	47	21.17	0	0	40.16	27.32
Bear	188	1	0.53	25	13.30	14.18	−20.68
Total	400	48	12.00	25	6.25	**27.95**	**4.76**

Table 2 Average performance of trading 400 stock in S&P500 for 400 days with window size control

Market pattern	Stock number	Less than buy-and-hold	Less (%)	Loss number	Loss (%)	Average profit (%)	Average profit of buy-and-hold (%)
Bull	212	32	15.09	0	0	57.28	27.32
Bear	188	0	0	8	4.26	14.73	−20.68
Total	400	32	8.00	8	2.00	**37.28**	**4.76**

Table 3 Average performance of trading 50 stock by varying n_2

	15/15	15/30	15/45	15/60	15/75
Average profit (%)	30.25	35.32	36.64	**37.38**	34.32
Number of transaction	15.2	14.6	13.4	12.8	12.2

Table 4 Average performance of trading 50 stock by varying n_1

	5/20	8/32	10/40	15/60	20/80
Average profit (%)	28.21	32.48	35.64	**37.38**	36.32
Number of transaction	14.2	13.4	12.2	12.4	11.2

while the window size is set to 120 trading days (nearly 3 months). Table 1 shows the results. Now we have tested same stocks again with window size control while other parameter settings are not change. The results are shown in Table 2. Revenue in this experiment has been significantly increased.

Secondly, since we have used the previous n_2 days for feature value extraction and prediction, the extreme value stock may reach in next n_1 days, the n_1 and n_2 may be considered. We first fix n_1 as 15 and chance n_2 from 1 to 5 times of n_1 to find the optimal value of n_2. The result illustrates in Table 3, which shows that the most profit will be gained while n_2 is 4 times of n_1. Then we adjust n_1 from 5 to 20, and set n_2 as 4 times of n_1. At this time we can see the average yield arrive maximum when n_1 is set to 15 in Table 4.

The last but not the least, the σ also need to be considered. On one hand this parameter controls the speed of transactions, on the other hand it also eliminates some of the impact caused by the prediction error. For fluctuations or bull movement, we can set it to a larger value to get more transactions. In contrast, for the bear movement,

Table 5 Performance of trading MSFT by varying σ

σ	0.005	0.010	0.015	0.020	0.025	0.030	0.035	0.040	0.045	0.050
Profit (%)	2.41	7.20	**8.15**	4.53	5.75	4.84	4.97	4.73	5.18	6.07
Number of transaction	8	9	9	9	9	9	9	9	9	8

it can be set a small value to reduce transactions. However, the parameter must be set within a range, otherwise it will reduce the accuracy. We selected a stock to show the impact for system profit of σ. The value of σ adjust from 0.005 to 0.04 while window size is set to 90, stop-loss rate θ is set to 0.05 and φ, ϕ set to 0, 0.05, respectively. The data set is samples from September 7, 2004 to March 24, 2006. In Table 5 we can see when σ is 0.015 the profit will be highest.

5 Conclusion

We have proposed a stock decision support system in this chapter. In the experiment, we have shown that this system is capable of superior performance to give investors considerable returns, especially when it is in a fluctuant movement the system can bring more lucrative benefits. It is mainly based on two reasons. The first one is the fast learning ability and high precision of ELM. The second one is that trading with box theory is based on the highest and lowest values the stock could reach in a period of time, which reduces the impact of noise and uncertainty in the stock market on the prediction accuracy. Similarly, using gray relation degree method to obtain each factor weight, to a certain extent, helps the ELM get more precise results.

The whole system is based on the ELM's prediction value to produce the result. However, features, which are used to train ELM, cannot fully represent all the influencing factors in the stocks. For this reason, ELM can only reach a certain degree of prediction accuracy. Therefore, how to reasonably model on the stock market and select more representative and comprehensive features become the future work.

Acknowledgments This work was supported by the National Natural Science Foundation of China (Project No. 60970034, 61170287 and 61232016).

References

1. S. Chen, P.M. Grant, A clustering technique for digital communications channel equalization using radial basis function networks. IEEE Trans. Neural Netw. **4**, 570–578 (1993)
2. X. Zhang, H. Fuehers, P.A. Gloor, in *Predicting Asset Value Through Twitter Buzz*. Advances in Collective Intelligence 2011 (Springer, New York, 2012), pp. 23–34
3. S.A. George, P.V. Kimon, Forecasting stock market short-term trends using a neuro-fuzzy based methodology. Expert Syst. Appl. **36**(7), 10696–10707 (2009)

4. T.Z. Tan, C. Quek, N.G. See, Biological brain-inspired genetic complementary learning for stock market and bank failure prediction. Comput. Intell. **23**(2), 236–261 (2007)
5. G. Grudnitski, L. Osburn, Forecasting S&P and gold futures prices: an application of neural networks. J. Futur. Market **13**, 631–643 (1993)
6. A.S. Chen, M.T. Leung, Regression neural network for error correction in foreign exchange forecasting and trading. Comput. Oper. Res. **31**, 1049–1068 (2004)
7. Y.K. Kwon, B.R. Moon, A hybrid neurogenetic approach for stock forecasting. IEEE Trans. Neural Netw. **18**(3), 851–864 (2007)
8. H.J. Liu, H.M. Chen, Y.R. Hu, Financial characteristics and prediction on targets of M&A based on SOM-Hopfield neural network, in *IEEE International Conference on Industrial Engineering and Engineering Management 2007*, pp. 80–84 (2007)
9. X. Lin, Z. Yang, Y. Song, T. Washio, The application of echo state network in stock data mining. PAKDD **5012**, 932–937 (2008)
10. L.J. Cao, E.H. Francis Tay, support vector machine with adaptive parameters in financial time series forecasting. IEEE Trans. Neural Netw. **14**(6), 1506 –1518 (2003)
11. Y.K. Bao, Forecasting stock composite index by fuzzy support vector machines regression, in *Proceedings of the Fourth International Conference on Machine Learning and Cybernetics*, Guangzhou, 18–21 Aug 2005
12. G.B. Huang, Q.Y. Zhu, C.K. Siew, Extreme learning machine: theory and applications. Neurocomputing **70**, 489–501 (2006)
13. G.B. Huang, D.H. Wang, Advances in extreme learning machines. Neurocomputing **74**(16), 2411–2412 (2011)
14. G.B. Huang, Q.Y. Zhu, C.K. Siew, Real-time learning capability of neural networks. IEEE Trans. Neural Netw. **17**(4), 863–878 (2006)
15. G.B. Huang, H. Zhou, X. Ding, R. Zhang, Extreme learning machine for regression and multiclass classification. IEEE Trans. Syst. Man Cybern. **42**, 513–529 Part B (2011)
16. D. Nicolas, How I made two million dollars in the stock market. BN Publishing, Illinois, America (2007)
17. J.L. Deng, Control problems of gray systems. Syst. Control Lett. **1**(4), 288–294 (1982)
18. L.J. Zhang, Z.J. Li, Gene selection for classifying microarray data using grey relation analysis. Discov. Sci. **4265**, pp. 378–382 (2006)
19. P. Martin, *Technical Analysis Explained*, 4th edn. Paperback. McGraw-Hill Company. ISBN0071226699