Predicting Trading Signals of Stock Market Indices Using Neural Networks

Chandima D. Tilakaratne[1,2], Musa A. Mammadov[2], and Sidney A. Morris[2]

[1]Department of Statistics, University of Colombo,
Colombo 3, Sri Lanka
[2]Center for Informatics and Applied Optimization
School of Information Technology and Mathematical Sciences
University of Ballarat, PO Box 663
Ballarat, Victoria 3353, Australia
cdt@stat.cmb.ac.lk, m.mammadov@ballarat.edu.au, s.morris@ballarat.edu.au

Abstract. The aim of this paper is to develop new neural network algorithms to predict trading signals: buy, hold and sell, of stock market indices. Most commonly used classification techniques are not suitable to predict trading signals when the distribution of the actual trading signals, among theses three classes, is imbalanced. In this paper, new algorithms were developed based on the structure of feedforward neural networks and a modified Ordinary Least Squares (OLS) error function. An adjustment relating to the contribution from the historical data used for training the networks, and the penalization of incorrectly classified trading signals were accounted for when modifying the OLS function. A global optimization algorithm was employed to train these networks. The algorithms developed in this study were employed to predict the trading signals of day (t+1) of the Australian All Ordinary Index. The algorithms with the modified error functions introduced by this study produced better predictions.

Keywords: Neural networks, Classification, Stock market predictions, Global optimization.

1 Introduction

Past studies have suggested that trading strategies guided by forecasts on the direction of price change may be more effective than on value of price indices and may lead to higher profits [16]. Furthermore, it was found that the classification models based on the direction of stock returns outperformed the models based on the level of stock return in terms of both predictability and profitability [4].

One of the most commonly used techniques to predict the trading signals of stock market indices is Feedforward neural networks (FNN) [8], [9], [17]. The FNN outputs the value of the stock market index (or a derivative) and subsequently this value is classified into classes (trading signals).

Almost all of the past studies, aimed at forecasting trading signals, considered only two classes: the upward and the downward trend of the stock market movement, which are corresponding to buy and sell signals [9], [17]. It was noticed

W. Wobcke and M. Zhang (Eds.): AI 2008, LNAI 5360, pp. 522–531, 2008.

that the time series data used for these studies are approximately symmetrically distributed among these two classes.

In practice, the traders do not participate in trading (either buy or sell shares) if there is no substantial change in the price level. Instead of buying/selling, they will hold the money/shares in the hand. In such a case it is important to consider the additional class which represents a hold signal. For instance, the following criterion can be applied to identify three trading signals: buy, hold and sell [14]:
Criterion A

$$\begin{array}{ll} \text{buy if} & Y(t+1) \geq 0.005 \\ \text{hold if} & -0.005 < Y(t+1) < 0.005 \\ \text{sell if} & Y(t+1) \leq -0.005 \end{array}$$

where $Y(t+1)$ is the relative return of the day $(t+1)$ of the Close price of the stock market index of interest. However, in this case, one cannot expect a symmetric distribution of data among the three classes, because more data falls into the hold class while less data falls into other two classes.

Due to the imbalance of data, the most classification techniques produce inaccurate results [1], [14]. FNN can be identified as a suitable alternative technique for classification when the data to be studied has an imbalanced distribution. However, a standard FNN itself shows some disadvantages: (1) usage of local optimization methods which do not guarantee a 'deeper' local optimal solution; (2) because (1), FNN needs to be trained many times with different initial weights and biases (multiple training results in more than one solution and having many solutions for network parameters prevent getting a clear picture about the influence of input variables); and (3) use of the ordinary least squares (OLS) as an error function to be minimised may not be suitable for classification.

To overcome the problem of being stuck in a local minima, finding a global solution to the error minimisation function is required. Several past studies attempted to find global solutions for the parameters of the FNNs by developing new algorithms [3], [7], [19].

This study aims at developing new neural network algorithms to predict the trading signals: buy, hold and sell, of a given stock market index. When developing new algorithms two matters were concerned: (1) using a global optimization algorithm for network training and (2) modifying the ordinary least squares error function. By using a global optimization algorithm for network training, this study expected to find better solutions to the error function. Also this study attempted to modify the OLS error function in a way suitable for the classification problem of interest.

The organisation of the paper is as follows: the next section explains the development of new algorithms. The third section describes the network training and the measures of evaluating the performance of the algorithms. Section four presents the results obtained from training the proposed algorithms together with their interpretations. The last section presents the conclusions of the study.

2 Development of New Algorithms

We designed new neural network algorithms for forecasting the trading signals of stock market indices. These new algorithms are based on the FNN.

FNN adopts backpropagation learning for weight modification. Backpropagation learning is an error minimising procedure and the network weights are changed according to an error function which compares the network output with the training targets ([18]). The most commonly used error function is the Ordinary Least Squares function (OLS):

$$E_{OLS} = \frac{1}{2N} \sum_{i=1}^{N} (a_i - o_i)^2 \tag{1}$$

where N is the total number of observations in the training set while a_i and o_i are the target and the output corresponding to the ith observation in the training set.

2.1 Alternative Error Functions

Minimisation of the absolute errors between the target and the output may not produce the desired accuracy of directional predictions [18]. Having this idea in mind, some past studies aimed to modify the error function associated with the FNNs [2], [10], [18]. These studies incorporated factors which represent the direction of the prediction [2], [18] and recency of the data that used as inputs [10], [18].

Functions proposed in [2] and [18] penalised the incorrectly predicted directions more heavily, than the correct predictions. In other words, higher penalty was applied if the predicted value (o_i) is negative when the target (a_i) is positive or vice-versa.

Yao & Tan [18] argued that a higher penalty should be imposed if a wrong direction is predicted for a larger change while it should be less if a wrong direction is predicted for a smaller change and so on. Based on this argument, they proposed the Directional Profit adjustment factor:

$$f_{DP}(i) = \begin{cases} c_1 \text{ if } (\Delta a_i \times \Delta o_i) > 0 \text{ and } \Delta a_i \leq \sigma, \\ c_2 \text{ if } (\Delta a_i \times \Delta o_i) > 0 \text{ and } \Delta a_i > \sigma, \\ c_3 \text{ if } (\Delta a_i \times \Delta o_i) < 0 \text{ and } \Delta a_i \leq \sigma, \\ c_4 \text{ if } (\Delta a_i \times \Delta o_i) < 0 \text{ and } \Delta a_i > \sigma. \end{cases} \tag{2}$$

where $\Delta a_i = a_i - a_{i-1}$, $\Delta o_i = o_i - o_{i-1}$ and σ is the standard deviation of the training data (including validation set). For the experiments authors used $c_1 = 0.5$, $c_2 = 0.8$, $c_3 = 1.2$ and $c_4 = 1.5$ [18].

Based on this Directional Profit adjustment factor (2), they proposed Directional Profit (DP) model [18]:

$$E_{DP} = \frac{1}{2N} \sum_{i=1}^{N} f_{DP}(i)(a_i - o_i)^2. \tag{3}$$

Refenes et al. [10] proposed Discounted Least Squares (LDS) function by taking the recency of the observations into account.

$$E_{DLS} = \frac{1}{2N} \sum_{i=1}^{N} w_b(i)(a_i - o_i)^2 \tag{4}$$

where $w_b(i)$ is an adjustment relating to the contribution of the ith observation and is described by the following equation:

$$w_b(i) = \frac{1}{1 + exp(b - \frac{2bi}{N})}. \tag{5}$$

Discount rate b, decides the recency of the observation. Authors suggested $b = 6$ ([10]).

Yao & Tan [18] proposed another error function, Time Dependent Directional Profit (TDP) model, by incorporating the approach suggested by [10] to their Directional Profit Model (3):

$$E_{TDP} = \frac{1}{2N} \sum_{i=1}^{N} f_{TDP}(i)(a_i - o_i)^2 \tag{6}$$

where $f_{TDP}(i)$=$f_{DP}(i) \times w_b(i)$. $f_{DP}(i)$ and $w_b(i)$ are described by (2) and (5), respectively.

2.2 Modified Error Functions

We are interested in classifying trading signals into three classes: buy, hold and sell. The hold class includes both positive and negative values (refer Criterion A in Section 1). Therefore, the least squares functions in which the cases with incorrectly predicted directions (positive or negative) are penalised (for example (3) and (6)), will not give the desired prediction accuracy. Instead of the weighing schemes suggested by previous studies, we proposed a different scheme of weighing.

This scheme is based on the correctness of the classification of trading signals. Let the codes for buy, hold, and sell signals be 1, 2, and 3 respectively. Also let $d = |a_i - o_i|$ where a_i and o_i are the targeted value and the output of the ith observation of the training set. Therefore:

d=0 implies that the predicted trading signal is correct

d=1 implies that either a buy or sell signal is misclassified as a hold signal or a hold signal is misclassified as a buy or sell signal. In the former case a trader loses only the opportunity of making profits, but not money while in the later case a monetary loss is incurred; however, this loss is not a 'big loss'. Therefore, d=1 indicates a less serious mistake.

d=2 implies that either a buy signal is misclassified as a sell signal or vice-versa. In this case, the impact of the misclassification is very high as there is a high tendency for a major monetary loss.

Considering these issues, this study proposes the following weighing scheme:

$$w_d(i) = \begin{cases} \delta \text{ if } d = 0, \\ 1 \text{ otherwise.} \end{cases} \tag{7}$$

where δ is a very small value. The value of δ needs to be decided according to the distribution of data.

Proposed Error Function 1: The weighing scheme, $f_{DP}(i)$, incorporated in the Directional Profit (DP) error function (3) considers only two classes, upward and downward trend (direction) which are corresponding to buy and sell signals. In order to deal with three classes, buy, hold and sell, we modified this error function by replacing $f_{DP}(i)$ with the new weighing scheme, $w_d(i)$ (see (7)). Hence, the new error function (E_{CC}) is defined as:

$$E_{CC} = \frac{1}{2N} \sum_{i=1}^{N} w_d(i)(a_i - o_i)^2 \tag{8}$$

When training backpropagation neural networks using (8) as the error minimisation function, the error is forced to take a smaller value, if the predicted trading signal is correct. On the other hand, the actual size of the error is considered in the cases of misclassifications.

Proposed Error Function 2: Recency of the data also plays an important role in the prediction accuracy of financial time series. Therefore, Yao & Tan [18] went further, by combining DP error function (3) with LDS error function (4) and proposed Time Dependent Directional Profit (TDP) error function (6).

Following Yao & Tan [18], this study also proposed a similar error function, E_{TCC}, by combining first new error function (E_{CC}) described by (8) with the DLS error function (E_{DLS}). Hence the second proposed error function is:

$$E_{TCC} = \frac{1}{2N} \sum_{i=1}^{N} w_b(i) \times w_d(i)(a_i - o_i)^2 \tag{9}$$

where $w_b(i)$ is defined by (5) while (7) defines $w_d(i)$.

The difference between the TDP error function (6) and this second new error function (9) is that $f_{DP}(i)$ is replaced by $w_d(i)$, in order to deal with three classes, buy, hold and sell.

2.3 New Neural Network Algorithms

New neural network algorithms were developed by: (1) using OLS error function as well as modified least squares error functions; and, (2) employing a global optimization algorithm to train the networks.

The importance of using global optimization algorithms for FNN training was discussed in the Introduction (Section 1). In this paper, we applied the global optimization algorithm, AGOP (introduced in [5], [6]), for training the proposed network algorithms.

As the error function to be minimised, we considered E_{OLS} (see (1)) and E_{DLS} (see (4)), together with the two modified error functions E_{CC} (see (8)) and E_{TCC} (see (9)). Based on these four error functions, we proposed the following algorithms:

NN_{OLS} - Neural network algorithm based on Ordinary Least Squares error function, E_{OLS} (see (1))
NN_{DLS} - Neural network algorithm based on Discounted Least Squares error function, E_{DLS} (see (4))
NN_{CC} - Neural network algorithm based on the newly proposed error function 1, E_{CC} (see (8))
NN_{TCC} Neural network algorithm based on the newly proposed error function 2, E_{TCC} (see (9))

These networks consist of three layers and out of these three one is a hidden layer. The layers are connected in the same structure as the FNN (Section 2). A tan-sigmoid function was used as the transfer function between the input layer and the hidden layer while the linear transformation function was employed between the hidden and the output layers.

3 Network Training and Evaluation

The Australian All Ordinary Index (AORD) was selected as the stock market index whose trading signals are to be predicted. Previously Tilakaratne et al. [12], [15] showed that Close prices of day t of the US S&P 500 Index (GSPC), the UK FTSE 100 Index (FTSE), French CAC 40 Index (FCHI), German DAX Index (GADXI) have an impact on the Close price of day $(t+1)$ of the AORD.

If there is a set of influential markets to a given dependent market, it is not straightforward to separate the influence from individual influential markets. The influence from one market on a dependent market may include the influence from one or more stock markets on the former. Therefore, in order to estimate the direct influence from one market to another, intermarket influence[1] needs to be quantified. Furthermore, Tilakaratne et al. [13] revealed the effectiveness of using quantified intermarket influences from the above mentioned markets for forecasting. Therefore, this study considered the quantified relative returns of day t of the Close prices of those markets as the input features for neural network.

Quantification of intermarket influences on the AORD was carried out by finding the coefficients, ξ_i, i=1, 2, ..., which maximise the median rank correlation between the relative return of day $(t+1)$ the Close of the AORD market and the sum of ξ_i multiplied by the relative returns of day t of the Close prices of a

[1] Relationship between the current price (or a derivative of price) of a dependent market with lagged price (or a derivative thereof) of one or more influential markets [12].

combination of influential markets over a number of small non-overlapping windows of a fixed size. The combination of markets, which is mentioned above, was considered. ξ_i measures the contribution from the ith influential market to the combined influence which equals to the optimal correlation.

Daily relative returns of the Close prices of the selected stock market indices from 2nd July 1997 to 30th December 2005 were used for this study. If no trading took place on a particular day, the rate of change of price should be zero. Therefore, before calculating the relative returns, the missing values of the Close price were replaced by the corresponding Close price of the last trading day.

The minimum and the maximum values of the data (relative returns) used for network training are -0.137 and 0.057, respectively. Therefore we selected the value of δ (Section 2.2) as 0.01. This value is small enough to set the value of the proposed error functions (8 and 9) approximately zero, if the trading signals are correctly predicted.

Since, influential patterns between markets are likely to vary with time [11], the whole study period was divided into a number of moving windows of a fixed length. Overlapping windows of length three trading years were considered. A period of three trading years consists of enough data (768 daily relative returns) for neural network experiments. Also the chance that outdated data (which is not relevant for studying current behaviour of the market) being included in the training set is very low.

The most recent 10% of data (the last 76 trading days) in each window was accounted for out of sample predictions while the remaining 90% of data was allocated for network training. Different number of neurons for the hidden layer was tested when training the networks with each input set.

As described in Section 2.1, the error function, E_{DLS} (see (4)), consists of a parameter b (discount rate) which decides the recency of the observations in the time series. Refenes et al. [10] fixed b=6 for their experiments. However, the discount rate may vary from one stock market index to another. Therefore, this study tested different values for b when training network NN_{DLS}. Observing the results, the best value for b was selected and this best value was used as b when training network NN_{TCC}.

3.1 Evaluation Measures

The networks proposed in Section 2.3 output the relative returns day (t+1) of the Close price of the AORD. Subsequently, the output was classified into trading signals according to Criterion A (Section 1).

The performance of the networks was evaluated by the overall classification rate (r_{CA}) as well as by the overall misclassification rates (r_{E1} and r_{E2}), which are defined as follows:

$$r_{CA} = (N_0/N_T) \times 100 \tag{10}$$

where N_0 and N_T are the number of test cases with d=0 (Section 2.2) and the total number of cases in the test sample, respectively;

$$r_{E1} = (N_1/N_T) \times 100 \tag{11}$$

$$r_{E2} = (N_2/N_T) \times 100 \quad (12)$$

where N_1 and N_2 are the number of test cases with d=1 and d=2 (Section 2.2), respectively. Because of the seriousness of the mistake (Section 2.2), r_{E2} plays a more important role in performance evaluation.

4 Evaluating the Performance of the New Algorithms

As mentioned in Section 3, different values for the discount rate, b, was tested. b=1, 2, ..., 12 was considered when training NN_{DLS}. The prediction results improved with the value of b up to b=5. For $b \geq 5$ the prediction results remained unchanged. Therefore, the value of b was fixed at 5. As previously mentioned (Section 3), b=5 was used as the discount rate also in NN_{TCC} algorithm.

The best four prediction results corresponding to the four networks were obtained when the number of hidden neurons equals two. Therefore, only the results relevant to networks with two hidden neurons are presented. Table 1 presents the results obtained from training four neural network algorithms.

Table 1. Results obtained from training four neural network algorithms

NN Algorithm	Average r_{CA}	Average r_{E2}	Average r_{E1}
NN_{OLS}	64.6930	0.0	35.3070
NN_{DLS}	64.4737	0.2193	35.3070
NN_{CC}	63.8158	0.0	36.1842
NN_{TCC}	66.2281	0.0	33.7719

Table 1 shows that the performance of NN_{DLS} and NN_{CC} are poorer than that of the algorithm based on the OLS error function (NN_{OLS}). However, NN_{TCC} produced better classification results than NN_{OLS}. It prevented from producing series misclassifications and gave the highest overall classification accuracy.

The classification results obtained by four algorithms broke down into classification and misclassification rates. These rates indicate the patterns of classification/misclassification of data belonging to a class. The classification rate indicates the proportion of correctly classified signals to a particular class out of the total number of actual signals in that class whereas, misclassification rate indicates the proportion of incorrectly classified signals from a particular class to another class, out of the total number of actual signals in the former class.

Table 2 shows the average (over the six windows) classification rate /misclassification rate corresponding to the results obtained from the four algorithms. These results also confirm that NN_{TCC} produces the best results among the four algorithms considered.

Table 2. Average (over the six windows) classification rate /misclassification rate corresponding to the results obtained from the four algorithms (The best prediction results are shown in bold colour)

Actual class	Average classification/misclassification rates					
	NN_{OLS}			NN_{DLS}		
	Predicted class			Predicted class		
	Buy	Hold	Sell	Buy	Hold	Sell
Buy	23.46%	76.54%	0.00%	23.54%	76.46%	0.00%
Hold	5.00%	88.74%	6.27%	4.97%	89.26%	5.77%
Sell	0.00%	79.79%	20.21%	1.39%	80.62%	17.99%
Actual class	NN_{CC}			NN_{TCC}		
	Predicted class			Predicted class		
	Buy	Hold	Sell	Buy	Hold	Sell
Buy	21.68%	78.32%	0.00%	**27.00%**	**73.00%**	**0.00%**
Hold	4.58%	87.90%	7.52%	**4.56%**	**89.22%**	**6.22%**
Sell	0.00%	79.72%	20.28%	**0.00%**	**75.49%**	**24.51%**

5 Conclusions

The algorithm which is based on the error function which takes the recency of data and the correctness of the predicted class into account (see (9)) showed the best performance of classifying trading signals. This algorithm produced the best predictions when the network consisted of one hidden layer with two neurons. The quantified relative returns of the Close prices of the US S&P 500 Index, the UK FTSE 100 Index, French CAC 40 Index, German DAX Index were used as the input features. This network prevented serious misclassifications such as misclassification of buy signals to sell signals and vice-versa and also predicted trading signals with a higher degree of accuracy.

The algorithms proposed in this paper can be used to predict the trading signals, buy, hold and sell, of any given stock market index or a sector of a stock market index.

References

1. Akbani, R., Kwek, S., Japkowwicz, N.: Appying Support Vector Machines to Imbalanced Datasets. In: Boulicaut, J.-F., Esposito, F., Giannotti, F., Pedreschi, D. (eds.) ECML 2004. LNCS (LNAI), vol. 3201, pp. 39–50. Springer, Heidelberg (2004)
2. Caldwell, R.B.: Performances Matrics for Neural Network-based Trdaing System Development. NeuroVe$t J. 3, 22–26 (1995)
3. Jordanov, I.: Neural Network Training and Stochastic Global Optimization. In: 9th International Conference on Neural Information Processing, vol. 1, pp. 488–492 (2002)
4. Leung, M.T., Daouk, H., Chen, A.S.: Forecasting stock indices: a comparison of classification and level estimation models. In. J. Fore. 16, 173–190 (2000)

5. Mammadov, M.A.: A new global optimization algorithm based on dynamical systems approach. In: Rubinov, A., Sniedovich, M. (eds.) 6th International Conference on Optimization: Techniques and Applications, Ballarat, Australia (2004)
6. Mammadov, M.A., Rubinov, A.M., Yearwood, J.: Dynamical systems described by relational elasticities with applications to global optimization. In: Jeyakumar, V., Rubinov, A. (eds.) Continuous Optimisation: Current Trends and Applications, pp. 365–387. Springer, Heidelberg (2005)
7. Minghu, J., Xiaoyan, Z., Baozong, Y., Xiaofang, T., Biqin, L., Qiuqi, R., Mingyan, J.: A fast hybrid algorithm of global optimization for feedforward neural networks. In: 5th International Conference on Signal Processing, Beijing, China (2000)
8. Pan, H., Tilakaratne, C., Yearwood, J.: Predicting the Australian Stock Market Index Using Neural networks Exploiting Dynamical Swings and Intermarket Influences. J. Resear. and Pract. in IT 37, 43–55 (2005)
9. Qi, M., Maddala, G.S.: Economic factors and the stock market: A new perspective. J. Fore 18, 151–166 (1999)
10. Refenes, A.N., Bentz, Y., Bunn, D.W., Burgess, A.N., Zapranis, A.D.: Financial Time Series Modelling with Discounted Least Squares Backpropagation. Neurocomp. 14, 123–138 (1995)
11. Tilakaratne, C.D.: A Study of Intermarket Influence on the Australian All Ordinary Index at Different Time Periods. In: 2nd Australian Business and Behavioural Seiences Association International Conference, Adelaide, Australia (2006)
12. Tilakaratne, C.D., Mammadov, M.A., Hurst, C.P.: Quantification of Intermarket Influence Based on the Global Optimization and Its Application for Stock Market Prediction. In: International Workshop on Integrating AI and Data Mining (AIDM 2006), Hobart, Tasmania, Australia (2006), http://doi.ieeecomputersociety.org/10.1109/AIDM.2006.14
13. Tilakaratne, C.D., Mammadov, M.A., Morris, S.A.: Effectiveness of Using Quantified Intermarket Influence for Predicting Trading signals of Stock Markets. In: 6th Australian Data Mining Conference (AusDM 2007), Gold Coast, Australia (2007), http://crpit.com/confpapers/CRPITV70Tilakaratne.pdf
14. Tilakaratne, C.D., Morris, S.A., Mammadov, M.A., Hurst, C.P.: Predicting Stock Market Index Trading Signals Using Neural Networks. In: 14th Annual Global Finance Conference (GFC 2007), Melbourne, Australia (2007)
15. Tilakaratne, C.D., Morris, S.A., Mammadov, M.A., Hurst, C.P.: Quantification of Intermarket Influence on the Australian All Ordinary Index Based on Optimization Techniques. ANZIAM J. 48, C104–C118 (2007)
16. Wu, Y., Zhang, H.: Forward Premiums as Unbiased Predictors of Future Currency Depreciation: A Non-parametric Analysis. J. Int. Mon. and Fin. 16, 609–623 (1997)
17. Yao, J., Tan, C.L., Poh, H.L.: Neural Networks for Technical Analysis: A Study on KLCI. Int. J. Theo. and App. Fin. 2, 221–241 (1999)
18. Yao, J., Tan, C.L.: Time Dependent Directional Profit Model for Financial Time Series Forecasting. In: IEEE-INNS-ENNS Internationsl Joint Conference on Neural Networks (IJCNN 2000), Como, Italy (2000)
19. Ye, H., Lin, Z.: Global optimization of neural network weights using subenergy tunneling function and ripple search. In: 2003 International Symposium on Circuits and Systems: ISCAS 2003, vol. 5, pp. 725–728 (2003), http://ieeexplore.ieee.org/xpl