International Journal of Soft Computing and Engineering (IJSCE)
ISSN: 2231-2307, Volume-7 Issue-2, May 2017

Systematic Investment Plan Date Prediction

Abhishek Bhatt, Romil Gopani, Lukash Chaddwa, Gajanan Bherde

Abstract*: Neural networks have been used on variety of prediction problems in field of finance. Mutual funds in particular SIP (Systematic Investment Plan) have been very lucrative form of high gain investment in recent years [A3]. In the paper, we have proposed a way to maximize investors return by providing an insight on possible values of NAV thought the month in the beginning of the month so they can buy units at Low rates. We have used artificial neural network (ANN) along with resilient propagation algorithm for prediction. We want to create a system which will help an investor to gain more profit compared to another investor investing in the same SIP. The proposed system will notify the user the date on which investment to be made to maximize profit. Results of our experiment have been attached which shows good performance on HDFC TOP 200 fund (G).

***Index Terms*: Systematic Investment Plan, Mutual Fund, Artificial Neural Net.**

I. INTRODUCTION

Predicting the correct date to buy a SIP is one of the major issues in finance. Predicting the correct date gives enables the user to get more number of SIP units as compared to user who buys SIP on a fixed date. The method of predicting the Net Asset Value is similar to the method of the predicting the stock market.

Nowadays, ANNs have been applied in order to predict exchange index prediction. ANN is one of data mining techniques that are learning capability of the human brain. Data patterns may perform dynamics and unpredictable because of complex financial data used. Several researches efforts have been made to improve efficiency of share values [1].

A. Artificial Neural Network

Artificial Neural Network (ANN) is a system that processes information similar to that of human neuron (Fig – 1). Each neuron receives some signals from other neurons or outside. Above figure has three layers of neurons, where one input layer is present. Every neuron employs activation function that fires when total input is more than a given threshold.

In this paper, we focus that Multi-Layer Perception (MLP) networks are layered feed-forward networks typically trained with back propagation.

MLP neural networks select one of the examples of training; make a forward and a backward pass. The first advantage of MLP networks is that eases an approximation of any input or output map. The first disadvantage is that they train very slowly and require lots of training data [2].

Revised Version Manuscript Received on April 07, 2017.

Abhishek Bhatt, Student, Department of Computer Engineering, K. J. Somaiya College of Engineering, Mumbai (Maharashtra), India.

Romil Gopani, Student, Department of Computer Engineering, K. J. Somaiya College of Engineering, Mumbai (Maharashtra), India.

Lukash Chhadwa, Student, Department of Computer Engineering, K.J. Somaiya College of Engineering, Mumbai (Maharashtra), India.

Gajanan Bherde, Professor, Department of Computer Engineering, K. J. Somaiya College of Engineering, Mumbai (Maharashtra), India.

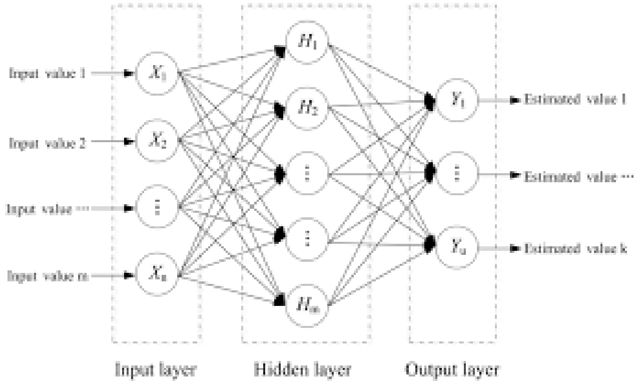


Fig-1: Architecture of Artificial Neural Network.

II. METHODOLOGY

1. Preparing data (normalizing)
2. Constructing network
3. Training the network
4. Comparing the output

2.1. Preparing Data (Normalizing)

The data is partitioned in 3 sets of which 91% (20 months of NAV data) of it is used to train the network, 4.5%(1 month of NAV data) is used to validate the result, used as target while training the network. The rest 4.5% (1 month of NAV data) is used to see how the network performs on untrained data by sliding the window of input data by 1 month.

The NAV of each SIP cannot be directly used to train the neural network; it has to be normalized first (Table-1).

Following formulae has been used to normalize the data (The formulae is feature scaling add citation)

$f(x) =((x-dL)(nH-nL)/(dH-dL)) + nL$

Here

X = value to be normalized

dL = data Low

dH = data High

nH = normalized High

nL = normalized Low

for de-normalizing the formulae used is

$f(x) = ((dL-dH)x-(nH\cdot dL)+dH\cdot nL)/ (nL-nH)$

The variables holding the same meaning as in normalization for our purpose, we have scaled all NAV to a range between -1 to 1.

Table1: 2 Months of Normalized Data

Date	**Data**	**Normalized Data**
1/2/2012	171.301	-0.124375744
1/3/2012	176.269	-0.098981256
1/4/2012	176.2	-0.099333957

1/5/2012	176.119	-0.099747998
1/6/2012	175.708	-0.10184887
1/9/2012	175.624	-0.102278245
1/10/2012	179.326	-0.083355058
1/11/2012	180.191	-0.078933513
1/12/2012	180.013	-0.07984338
1/13/2012	181.569	-0.071889712
1/16/2012	182.003	-0.069671272
1/17/2012	185.183	-0.053416346
1/18/2012	184.006	-0.059432713
1/19/2012	186.294	-0.047737345
1/20/2012	188.293	-0.037519232
1/23/2012	187.897	-0.039543431
1/24/2012	191.475	-0.021254083
1/25/2012	193.406	-0.01138356
1/27/2012	194.899	-0.003751923
1/30/2012	190.045	-0.028563688
1/31/2012	194.47	-0.005944805
2/1/2012	195.731	5.01E-04
2/2/2012	197.314	0.00859262
2/3/2012	199.22	0.018335352
2/6/2012	201.285	0.028890831
2/7/2012	200.121	0.022940915
2/8/2012	201.539	0.030189181
2/9/2012	203.156	0.038454657
2/10/2012	202.826	0.036767825
2/13/2012	202.698	0.036113539
2/14/2012	203.888	0.042196357
2/15/2012	208.664	0.066609417
2/16/2012	208.644	0.066507184
2/17/2012	210.199	0.074455741
2/21/2012	211.59	0.081565993
2/22/2012	205.312	0.049475293
2/23/2012	204.811	0.046914375
2/24/2012	203.278	0.039078274
2/27/2012	197.694	0.010535032
2/28/2012	203.017	0.037744143

2.2. Constructing Network

We have created Multi-Layer Perceptron (MLP) [A1] which is then trained using resilient algorithm.
The network takes 20 months of data as input as input and provides one month data as output. The architecture of the network is _ layers in input layer _ many hidden layers, the initial _ hidden layers have _ number of hidden neurons the latter _ layers have _ neurons and the output layer has _ neurons.

All the weights are initially assigned random value.

2.3. Training The Network

We have constructed a network which takes 1 set each consisting of 10 days of NAV data in input layer, 40 such sets are given input to network to give a precise result of target output for upcoming month data.

We have selected resilient propagation (RPROP) algorithm to train the network.

The weights are updated in the following way

$$\Delta w_{ij}^{(t)} = \begin{cases} -\Delta_{ij}^{(t)} & , \text{ if } \frac{\partial E}{\partial w_{ij}}^{(t)} > 0 \\ +\Delta_{ij}^{(t)} & , \text{ if } \frac{\partial E}{\partial w_{ij}}^{(t)} < 0 \\ 0 & , \text{ else} \end{cases}$$

$$w_{ij}^{(t+1)} = w_{ij}^{(t)} + \Delta w_{ij}^{(t)}$$

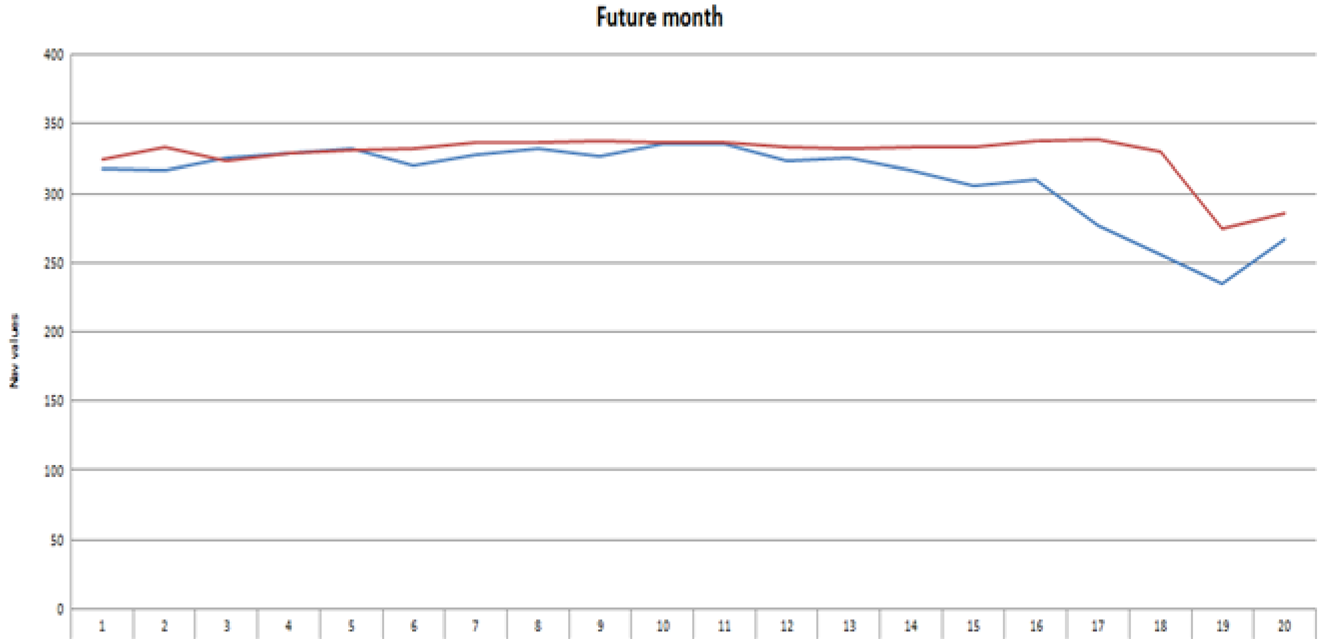
Every time the partial derivative of the corresponding weight wij changes its sign, which indicates that the last update was too big and the algorithm has jumped over a local minimum, the update-value delta ij is decreased by the factor q-. If the derivative retains its sign, the update-value is slightly increased in order to accelerate convergence in shallow regions.

Once the update-value for each weight is adapted, the weight-update itself follows a very simple rule: if the derivative is positive (increasing error), the weight is decreased by its update-value, if the derivative is negative, the update-value is added [A1]

The error is calculated after every iteration, the error used is the degree to which the neural network output matches the desired output. Finally, the network is saved when the output is close to the target output, i.e. the error is minimum the network is saved.

2.4. Comparing The Output

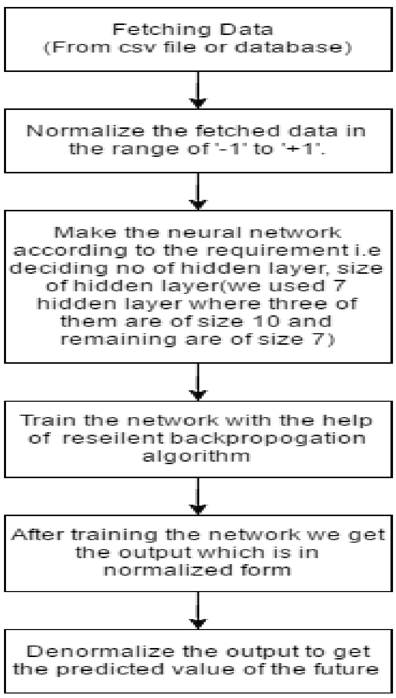
The output computed is compared to the desired output and the error remains within _%.
The network provides little deviating results when it works with untrained data _ % of deviation in output is encountered.



	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
actual	317.7869	317.3845	326.4652	329.2125	332.7700	320.8851	328.4315	332.6915	327.6738	336.0567	335.6548	323.5393	326.5010	316.6440	306.3276	310.2764	277.6495	255.6522	234.8788	267.3136
ideal	324.7517	333.2307	323.1624	329.0429	330.7623	332.3656	336.4728	336.3318	338.1105	337.112	336.8645	333.5331	332.6131	332.9144	332.8498	337.7898	338.8056	329.8241	275.1896	285.8503

III. CONCLUSION

Prediction model used proper data sources which are clean and authentic; using the official websites of the mutual fund is a suggested. Different steps like collecting, cleaning, normalizing, creating and training network and moving the prediction window forward are integral steps to implement successful model. The experiment in the paper uses simple, efficient and well known resilient propagation algorithm for training purpose. The experiment has shown that for above setup of network the error of day of month can vary for maximum of 22% to minimum 0.5% on untrained data. The model is very useful for small investors as they can maximize the units they can buy using insight of the model, it is also beneficial to financial analysts and corporate investors.



REFERENCES

1. X. Wu, M. Fund and A. Flitman, "Forecasting Stock Performance using Intelligent Hybrid Systems", Springerlink, 2001, pp. 4 4 7-4 56.
2. Yunus YETISl, Halid KAPLAN2, and Mo JAMSHIDI3, Fellow IEEE Department of Electrical and Computer Engineering, University of Texas at San Antonio San Antonio, Texas, USA
3. D. E. Rumelhart, G. E. Hinton, R. 1. Wiliams, "Learning Internal Representation by Error Propagation Parallel Distributed Processing Explorations in the Microstructures of Cognition ", McClelland J. L. (eds.), 1:318-362, MIT Press, Cambridge, 1986.
4. Riedmiller, Martin, and Heinrich Braun. "A direct adaptive method for faster backpropagation learning: The RPROP algorithm." Neural Networks, 1993, IEEE International Conference on. IEEE, 1993. APA
5. http://www.livemint.com/Money/VBRowqYA6XWPQo55asrVDJ/Mutual-fund-folios-rise-a-record-14-in-fiscal-year-201516.html
6. Data source http://www.hdfcfund.com/products/equity-growth-fund

Published By:
Blue Eyes Intelligence Engineering
& Sciences Publication