International Journal of Engineering & Technology, 7 (4.10) (2018) 169-176

International Journal of Engineering & Technology

Website: www.sciencepubco.com/index.php/IJET

Research paper

Artificial Neural Network Algorithms based Nonlinear Data Analysis for Forecasting in the Finance Sector

Jitendra Kumar Jaiswal[1], Raja Das[2*]

[1,2]VIT Institute of Technology, Vellore – 632014, Tamilnadu, India
**Corresponding author E-mail: rajadas@vit.ac.in*

Abstract

The involvement of big populace in the quantitative trading has been increased remarkably since the wired and wireless systems have become quite ubiquitous in the fields of finance and economics. Statistical, mathematical and technical analysis in parallel with machine learning and artificial intelligence are frequently being applied to perceive prices moving pattern and forecasting. However stock price do not follow any deterministic regulatory function, factor or circumstances rather than many considerations such as economy and finance, political environments, demand and supply, buying and selling tendency, trading and investment, etc. Historical data assist remarkably for prices forecasting as an important option for mathematicians and researchers. In this paper, we have followed backpropagation and radial basis function neural network for predicting future prices by modifying these techniques as per requirements. We have also performed a comparative analysis of the two ANN techniques for existing and our modified models.

Keywords: *Moving Average; Forecasting; Neurons; Backpropagation; RBFN; and Sliding Window*

1. Introduction

Financial sectors are the keen premises for the investors to procure high returns with of course an edifice or anticipations for the movements of securities' prices. The concerned activities in this sector are also termed as quantitative investments. This generation is also very much intended toward financial investment as of about 30-40% share has occupied globally in this sector only. Financial investments refer to put some amount of money and anticipating some returns in a certain span of period. A financial market is considered as a place where individuals perform financial transactions. It consists of two bodies called buyers and sellers, who become involved in sales and purchase of financial products such as stocks, bonds, mutual funds, and so on. It is also called as capital market. Capital market is divided in two phase: primary market and secondary market. In the primary market, different companies release new stocks, bonds and shares at first time for investors as IPOs (Initial Public Offering). In the secondary market, already issued securities are bought and sold. Types of capital market refer to stock, bonds, commodities, money, derivatives, insurance, future, foreign exchange, and private market, etc. There are numerous determinants which influence the maneuver of market prices like economy, supply-demand, advertising, sentiments, company news, naturals disasters, expectations and speculations, political environments, GDP growth, etc ([7], [15]).
In the financial market, the forecasting of movements of stock values on daily or a certain duration basis is the primary concern and challenging as well for both investors and researchers. As we mentioned many factors those regulates the stock values ripples, the characteristics of stock market reflect a dynamic, non-stationary, nonparametric, nonlinear, noisy, and chaotic nature ([5], [19]). The identification of interaction among these factors is very complex ([18]). However many researchers and investors find some patterns of prices movements, and perform forecasting and investment recommendations on those basis. There has been developed many models form mathematics, statistics, computer science, machine learning, and artificial intelligence, etc., for financial time series forecasting. Some reviews on forecasting models have been presented by [8], [13], [1], [12], and [14]. Statistical techniques have been traditionally applied to perform time series analysis using ARMA and ARIMA models ([3]) along with some more sophisticated ARCH techniques ([6]). On the other hand machine learning techniques are also being applied for finding future movements of security prices ([20]). We have performed some more literature reviews in the respective sections.
We have downloaded 5-years automobile NSE data for the stocks Ashok Leyland Limited, TATA Motors, Mahindra & Mahindra Limited, Maruti, and Suzuki India Limited from 17/07/2012 to 17/07/2017 on daily basis from the website [17]. We have performed all our programming and result analysis on the MATLAB-software from MathWorks®.
In this paper we have considered moving averages and multiple regressions as conventional analysis and after that we have applied backpropagation technique from artificial neural networks independently and in conjugation with conventional methods. In Section-2, we have explained about moving averages to perform time series analysis as the trend of data. Artificial neural network structure has been elaborated in the section-3 and the study of backpropagation learning model is performed in Section - 4. Next in Section-5, radial basis function neural network has been explored.
Further in the Section-6, we have performed result comparison and analysis from the various independent and conjugated techniques. Finally we have summarized the paper along with its future scope in the section-7.

2. Moving Average

Moving average is characterized by calculating n-days averages with different techniques in a sliding window manner. The values of n are followed in different perspectives. It may opt two ways such as short and long time period. Different researchers consider different spans of times for short and long. Short time period may consider 5-days, 10-days, 14-days, 15-days, or one-month while long time periods may follow 50-days, 100-days, 150-days, or 200-days. Some best combinations of periods for moving averages and threshold return prices have been explored by Horne et al. ([9]). Mitra ([4]) has also explained the advantages of trading rules in India.

Moving average is process of considering one most recent value and dropping one most old value, and proceeding in this sliding widow manner. There are different procedures for the calculation of moving average. There are six prime techniques: Simple moving average (SMA), Exponential moving average (EMA), Weighted moving average (WMA), Adoptive moving average (AMA), Triangular moving average (TMA), and Typical Price moving average (TPMA). Wang et al. ([16]) have presented these averages with the respective formulas. In this paper we shall be followed only the Simple moving Average and Exponential moving average because will shall be using the data generated by these moving averages in further techniques.

2.1 . Simple Moving Average

Simple moving average (SMA) is the simple average of n-most recent period of stock values. It is calculated for n-time period by the following formula:

$$ma_t = \frac{1}{n}\sum_{i=0}^{n-1} v_{t-i} \qquad (1)$$

where, v_{t-i} is the stock value for $i = 1, 2, ..., n-1$ days.

Some researchers also follow the volume factor that is total number of shares that being traded collectively. So the formula for SMA can be considered as follows:

$$va_t = \frac{1}{k}\sum_{i=0}^{k-1} vol_{t-1} \qquad (2)$$

Where, volume average va_t is the average for first k shares in t time period.

2.2 Exponential Moving Average

Exponential moving average (EMA) is also called Weight exponential moving average since it considers weight factors in exponentially decreasing manner but never zero. The EMA of stock prices X for n-time period can be calculated recursively as follows:

$$ema_t = \begin{cases} X_1, & t = 1 \\ \alpha X_t + (1-\alpha) ema_{t-1}, & t > 1 \end{cases}$$

Where, $\alpha \in (0,1)$ is the degree of weights in decreasing order, also known as smoothing factor. X_t is the stock price at time period t and ema_t is EMA at t time period

3. Artificial Neural Network Structure

Zahra and Seyedmohsen ([21]) performed the comparison analysis and efficiency evaluation of ANN along with data normalization analysis for predictions of profitability in corporate. Anyaeche and Ighracwe ([2]) have applied linear regression, ANN, and back-propagation ANN to compare forecasting efficiency.

The concept of artificial neural networks is very much relevant or analogous to natural nervous systems in human brain. It can be considered as substantially parallel adaptive networks of neurons. Neurons are here cognitive to simple nonlinear computing components. The mere intention of neural networks is to perform both analysis and establishment of such substantive parallel computing systems. Neurons are categorized into three types: input, hidden, and output. Input neurons receive inputs from external sources as a stimulant to the network. Output of neurons produces output signals of the network. The intermediate functions are calculated by hidden neurons, and these neurons are not visible from the external sites.

A neural network model can be created as a weighted directed graph containing neurons as nodes and directed weighted edges as links between neurons. Two possible types of connections may ne there:

- a feed forward architecture, which is without loops, and,
- a feedback (recurrent) architecture, having loops in the network, due to the feedback links

Artificial neural network models have the capability of adopting the change of environments that is also learning. In this process it generates an internal model with sampled data, which represent structured weight vectors. Learning algorithms develop an architecture-based approach to assign patterns into weights to produce internal models. Learning process continues with update of connections weights. According to the learning nature, it is categorized into two types: supervised and unsupervised.

3.1. Supervised Learning

Let us consider a discrete data sample for learning process as $\zeta = \{(X_k, D_k)\}_{k=1}^{Q}$ from population space where input vector $X_k \in R^n$ to an output vector $D_k \in R^P$. The unknown function $f : R^n \to R^p$ is characterized by the sample data. Note that the sample data may be noisy.

The structure of the supervised learning may be presented as follows. X_k is the input to the system, and it produces an output Y_k. Supervised learning corresponds to the desired output D_k to reduce the error ($D_k - Y_k$) in the response to the system. The network is being trained with the pairs of input-output samples, in the form of learning of error correction that is also called as steepest descent or gradient descent weight adaptation. The error correction is performed here in the global environment that is estimated from difference between desired value (D_k) and output of the network (Y_k). It is implemented usually with the help of difference equation that is established to perform with such global information. Here we want to generate the output from system which is close to the desired output D_k, and we say that the system has learned the underlying phenomena if the stimulus $X_k^{'}$ close to X_k invoke a response $Y_k^{'}$ which is adequately lose to D_k .

3.2. Unsupervised Learning

Supervised learning deals with the development of well defined "clusters" with a provided set of data samples, $\{X_i\}, X_i \in R^n$, in such a way that each cluster comprises a class of vectors which are have some similar properties.

3.3. $\alpha -$ Least Mean Square Learning

We have considered a training set of the form $\zeta = \{X_k, d_k\}, X_k \in R^{n+1}, d_k \in R$. The resultant activation of the neuron:

$$s_k = y_k = X_k^T W_k \tag{3}$$

We can follow the following definition.

Definition3.1

The linear error e_k due to the dedicated training pair (X_k, d_k) is calculated as the difference between the desired output d_k and the neuronal signal $s_k : e_k = d_k - s_k$.

Therefore the linear error becomes,

$$e_k = d_k - s_k = d_k - X_k^T W_k \tag{4}$$

With the implementation of linear error measure into the weight update process, $\alpha -$least mean squared $(\alpha - LMS)$ learning algorithm is established. In the case of single adaptive linear neuron, $\alpha - LMS$ are of maintaining minimal disturbance based on past learning when new data into weight vector is added.

The recursive update equation from the $\alpha - LMS$ can be constructed as follows:

$$W_{k+1} = W_k + \eta e_k \frac{X_k}{\|X\|^2} \tag{5}$$

Here the weights are revised by multiplication of the scaled error and the normalized input vector as follows:

$$\Delta W_k = \frac{\eta e_k}{\|X_k\|} \frac{X_k}{\|X_k\|} = \frac{\eta}{\|X_k\|} e_k \hat{X}_k = \overset{\wedge}{\eta}_k e_k \hat{X}_k \tag{6}$$

Where η the rate of learning is, $\overset{\wedge}{\eta}_k$ is the pattern-normalized learning rate $\hat{X}_k$ is the unit vector in the direction of X_k.

The error changes for X_k follows the scope of the change in the weight vector W_k.

$$\Delta e_k = (d_k - X_k^T W_{k+1}) - (d_k - X_k^T W_k) \tag{7}$$

$$= -X_k^T \Delta W_k \tag{8}$$

$$= -X_k^T \left(\frac{\eta e_k X_k}{\|X\|^2} \right) \tag{9}$$

$$= \eta e_k \tag{10}$$

The error correction term Δe_k is proportional to the error e_k itself, and it is reduced by the factor η iteratively, which can be called as regulatory factor as it regulates the stability and the convergence speed. Generally, stability is achieved if $0 < \eta < 2$.

4. Back Propagation Neural Network (BPNN)

4.1. Notations

The notation summary is given in the following Table 1

Table 1: Table of Notations

	Input	Hidden	Output
Number of neurons	n+1	q+1	p
Signal Function	Linear	sigmoidal	sigmoidal
Index range of neurons	i=0,....,n	h=0,....,q	j=1,....,p
Activation	x_i	z_h	y_j
Signal	$S(x_i)$	$S(z_h)$	$S(y_j)$

The neuronal weights from input-to-hidden and hidden-to-output have been represented by w_{ih} and w_{hj} respectively. Iteration index is k. At the input layer, neurons follow linear function as follows:

$$S(x) = x \tag{11}$$

and at the hidden and output layers, sigmoidal function

$$S(x) = \frac{1}{1 + e^{-\lambda x}} \tag{12}$$

λ typically considers the value 1. Let us assume a set of Q training vector pairs $\zeta = \{(X_k, D_k)\}_{k=1}^{Q}$, where input vector $X_k \in R^n$ to an output vector $D_k \in R^P$. Training pairs are therefore $(X_1, D_1), (X_2, D_2), ..., (X_k, D_k), ...$ are selected from the training set. The generated output signal vector Y_k becomes the activation vector of output layer neurons.

4.1 Squared Error Function

The instantaneous error defined by $k_t h$ training pair X_k, D_k is given by:

$$E_k = D_k - S(Y_k) \tag{13}$$

Where

$$E_k = (e_1^k, e_2^k, ..., e_p^k)^T = (d_1^k - S(y_1^k), d_2^k - S(y_2^k), ... d_p^k - S(y_p^k))^T \tag{14}$$

The summation of squared error ξ_k is the sum of squares of each individual output error e_j^k, scaled by one-half of convenience:

$$\xi_k = \frac{1}{2}\sum_{j=1}^{p}\left(d_j^k - S(y_j^k)\right)^2 = \frac{1}{2}E_k^T E_k \tag{15}$$

The mean square error ξ can be calculated over the entire training set:

$$\xi = \frac{1}{Q}\sum_{k=1}^{Q}\xi_k \tag{16}$$

4.2. The Learning Procedure Skeleton

The basic process of steepest descent based learning is structured as:

Step 1. Give the input pattern X_k form the training set ζ to the network. 1.

Step 2. Calculate activations, input signals, hidden and output neurons sequentially.

Step 3. Calculate the error vector with the help of output neurons and desired output.

Step 4. With help of the calculated error in Step 3, estimate the changes in hidden-to-output and input-to-hidden layer weights (along with all biased weights), such that global error gets reduced.

Step 5. According to the all changes in Step 4, update all weights as follows:

Hidden-to-output layer weights

$$w_{hj}^{k+1} = w_{hj}^k + \Delta w_{hj}^k \tag{17}$$

Input-to-hidden layer weights

$$w_{ij}^{k+1} = w_{ij}^k + \Delta w_{ij}^k$$

Where Δw_{hj}^k and Δw_{ij}^k are changes in weights calculated in Step 4.

Step 6. Repeat Steps 1-5 until the global error gets down below considered threshold.

4.3. Calculations in the Backpropagation Algorithm

We have considered the sigmoid gain scale factor $\lambda = 1$ throughout the discussion.

Calculations of Neuronal Signals

1. At the input layer:

$$S(x_i^k) = x_i^k, \quad i = 1,2,...,n \tag{19}$$

$$S(x_0^k) = x_0^k = 1 \tag{20}$$

2.
3.

Where, x_i^k is i^{th} value of X_k and x_0^k is bias neuron signal at input.

2. At hidden layer:

$$z_h^k = \sum_{i=0}^{n} w_{ih}^k S(x_i^k) = \sum_{i=0}^{n} w_{ih}^k x_i^k,\ h = 1,2,...,q \tag{21}$$

$$S(z_h^k) = \frac{1}{1+e^{-z_h^k}}, \quad h = 1,2,...,q \tag{22}$$

$$S(z_0^k) = 1 \tag{23}$$

Where w_{ih}^k are the neuronal weights of output signal $S(z_i^k)$ from hidden layer, and w_{0h}^k is biased weight at this layer.

3. At output layer:

$$y_i^k = i = h\sum^{q} w_{hj}^k S(z_h^k),\ h = 1,2,...,p \tag{24}$$

$$S(y_i^k) = \frac{1}{1+e^{-y_j^k}}, \quad h = 1,2,...,p \tag{25}$$

where w_{hj}^k are the neuronal weights from outputs.

Calculations of error gradients

1. Weight gradients at hidden-to-output layer:

Using chain rule of calculus:

$$\frac{\partial \xi_k}{\partial w_{hj}^k} = \frac{\partial \xi_k}{\partial S(y_j^k)}\frac{\partial S(y_j^k)}{\partial y_j^k}\frac{\partial y_j^k}{\partial w_{hj}^k} \tag{26}$$

The intermediate partial derivatives can be calculated as follows:

$$\frac{\partial \xi_k}{\partial S(y_j^k)} = (d_j^k - S(y_j^k)) = e_j^k \tag{27}$$

$$\frac{\partial \xi_k}{\partial S(y_j^k)} = S^{'}(y_j^k) = S(y_j^k)(1 - S(y_j^k)) \tag{28}$$

$$\frac{\partial y_j^k}{\partial w_{hj}^k} = S(z_j^k) \tag{29}$$

which collectively yields:

$$\frac{\partial \xi_k}{\partial w_{hj}^k} = -e_j^k S^{'}(y_j^k) S(z_h^k) \tag{30}$$

$$= -\delta_j^k S(z_h^k) \tag{31}$$

Where, $\delta_j^k = e_j^k S^{'}(y_j^k)$ is a signal slope as an error, δ_j^k is slope scaled error.

2. Weight gradients at input-to-hidden layer:

$$\frac{\partial \xi_k}{\partial w_{ih}^k} = \frac{\partial \xi_k}{\partial S(z_h^k)}\frac{\partial S(z_h^k)}{\partial z_h^k}\frac{\partial z_h^k}{\partial w_{ih}^k} \tag{32}$$

which can be expressed as:

$$\frac{\partial \xi_k}{\partial w_{ih}^k} = \sum_{j=0}^{p}\left\{\frac{\partial \xi_k}{\partial y_h^k}\frac{\partial (y_j^k)}{\partial s(z_h^k)}\right\} S^{'}(z_h^k) S(x_i^k) \tag{33}$$

$$= \sum_{j=0}^{p} \left\{ \frac{\partial \xi_k}{\partial S(y_j^k)} \frac{\partial S(y_j^k)}{\partial y_j^k} \frac{\partial y_j^k}{\partial S(z_h^k)} \right\} S'(z_h^k) S(x_i^k) \quad (34)$$

$$= \sum_{j=0}^{p} \left\{ -e_j^k S'(y_j^k) w_{hj}^k \right\} S'(z_h^k) x_i^k \quad (35)$$

$$= \underbrace{-\sum_{j=0}^{p} (\delta_j^k w_{hj}^k)}_{\text{error backpropagation}} \; S'(z_h^k) x_i^k \quad (36)$$

The error factor of the h^{th} hidden node is:

$$e_h^k = \sum_{j=1}^{p} (\delta_j^k w_{hj}^k) \quad (37)$$

$$\delta_h^k = e_h^k S'(z_h^k) \quad (38)$$

and therefore, we have

$$\frac{\partial \xi_k}{\partial w_{ih}^k} = -\delta_h^k x_i^k \quad (39)$$

Weight updates

1. For hidden-to-output layer weights:

$$w_{hj}^{k+1} = w_{hj}^k + \Delta w_{hj}^k \quad (40)$$

$$= w_{hj}^k + \eta \left(-\frac{\partial \xi_k}{\partial w_{hj}^k} \right) \quad (41)$$

$$= w_{hj}^k + \eta \delta_j^k S(z_h^k) \quad (42)$$

2. For input-to-hidden layer weights:

$$w_{ih}^{k+1} = w_{ih}^k + \Delta w_{ih}^k \quad (43)$$

$$= w_{ih}^k + \eta \left(-\frac{\partial \xi_k}{\partial w_{ih}^k} \right) \quad (44)$$

$$= w_{ih}^k + \eta \delta_h^k x_i^k \quad (45)$$

4.4. Delta Rule Generalization: Momentum Introduction

The momentum term is used in weight update calculation to increase the learning rate along with taking care of stability

$$\Delta w_{hj}^k = \eta \delta_j^k S(z_h^k) + \alpha \Delta w_{hj}^{k-1} \quad (46)$$

$$\Delta w_{ih}^k = \eta \delta_h^k x_i^k + \alpha \Delta w_{ih}^{k-1} \quad (47)$$

Where, $\alpha > 0$ is the momentum. The algorithm follows the *delta rule generalization* when weights are updated according to Equations (46) and (47).

5. Radial Basis Function Network

RBFN is also a network of input, hidden, and output layers while it can comprise only one neuron in the hidden layer and it follows only a feed-forward technique, however it may contain any number of nodes in the three layers. The basis stricture of RBFN is contains *m*-nodes in input layer, *h*-nodes in hidden layer, and 1-node in the output layer. The synaptic weights are not assigned from input to hidden layer whereas the weights are charged with applied mathematical functions for hidden to output layer, and the output nodes trade on the heels of linear simulation.
The input-output pairs $\boldsymbol{T} = \{X_i, d_i\}$ perform calculations with interpolation to acquire function *f* which takes input X_i and produce output close to desired output d_i for *n* data sample.

$$f(x_i) = d_i; i = 1, 2, ..., n \quad (48)$$

RBFN produces *n* function as basis $\varphi(||X - X_i||)$, $i = 1,2,...,n$ as non-linear with Euclidean distance $(||X - X_i||)$, where *X* as applied input and *Xi* as points of training data. Then the mapping *f* is

$$f(x) = \sum_{i=0}^{n} w_i \varphi(||\, x - x_i \,||) \quad (49)$$

From equation, (48) and (49)

$$\sum_{i=0}^{n} w_i \phi(||x - x_i||) = d_i; i = 1,2,\ldots,n \quad (50)$$

For basis determination at the hidden layer, many functions can be applied. Some of them are given as follows.

1. Gaussian function:

$$\varphi(x) = e^{\left(-\frac{(x-t)^2}{2\sigma^2}\right)}; \alpha > 0; x, t \in R \quad (51)$$

2. Multiquadrics:

$$\varphi(x) = (x^2 + t^2)^{1/2} \quad (52)$$

3. Inverse multiquadrics:

$$\varphi(x) = 1/(x^2 + t^2)^{1/2} \quad (53)$$

The basis function φ is symmetric and weights *W* can be estimated with correct selection of φ as:

$$W = \varphi^{-1} D \quad (54)$$

For, $W = (w_1, ..., w_n)'$ and $D = (d_1, ...d_n)'$.

There are mainly three parameters those regulates the functioning of RBFN, namely *activation function*, *spread factor*, and *basis*. Empirical results show that combination of all these factors with different set of values may reflect a distinct set of outcomes. Spread factor σ may be determined by regulating the outputs based on observations.
In this paper we have focused on regulating the spread factor σ, which has been given in the forthcoming algorithm, and determi-

nation of basis, for which we have proposed four methods explained in the following subsection.

5.1. Basis Determination in RBFN

It is obvious that the maximum length of basis may not be more than the total length of training data and weights are estimated with the assistance of basis function. The following approaches have been followed to determine the centers of the basis functions.

1. *Random Subset basis:* A random row may be as the basis from the considered set of training data, and once a satisfactory result is obtained, the model can be further employed for testing and prediction.
2. *Random mean Subset basis:* A random mean value for small data subset can be provided for every iteration, until the network is satisfactorily trained.
3. *k-Mean subset basis:* The center for basis is determined as the cluster mean values for training the model.
4. *Hybrid basis:* The hybrid approach can be followed as considering combination of any two of above mentioned three approaches.

We have presented a consolidated modified RBFN algorithm here to predict the stock prices as follows.

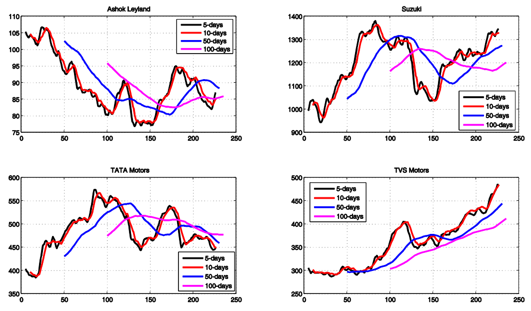
Algorithm: ModifiedRBFNAlgorithm

```
1. $\alpha = 0.1; error = threshold;$
2. $[m,n] = size(traindata);[p,q] = size(testdata)$
3. **while** $error > threshold$
$\sigma = \sigma + 0.1;$                # Regulation of spread factor $\sigma$
a)  **for** $i = 1:m$
      $k=1;$
      **for** $j = 1:m/2$
        $\varphi(i,j) = \exp(-(traindata(i) - c(k))^2 / 2\sigma^2)$;
      $k = k + 1;$
b)  $pseudoinv = inv(\varphi' * \varphi) * \varphi$;
c)  $W = pseudoinv * traindata(:,n)$
d)  **for** $i = 1:1$
      $k=1;$
      **for** $j = 1:m/2$
        $test\varphi(i,j) = \exp(-(testdata(i) - c(k))^2 / 2\sigma^2)$;
      $k = k + 1;$
e)  $f = test\phi * W$ ;
f)  Estimate biased weight $W_0$
g)  Predict Value, $Y = W_0 + f$
h)  *Calculate error*
4. Estimate center($c$) on random basis; go to step (2);
5. Estimate center($c$) on random mean basis. ; go to step (2);
6. Estimate center($c$) on k-mean subset basis; go to step (2);
7. Estimate center($c$) on hybrid manner subset basis; go to step (2);
8. Find minimum error from steps (4), (5), (6), and (7);
9. Perform prediction with the basis function providing minimum error.
```

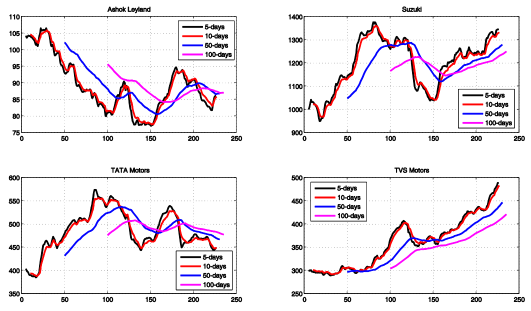
6. Result Analysis and Comparison

6.1. Results Discussion with Moving Averages

Moving averages are generally advantageous for observing the trend of data as they are important for long term investing. They directly indicate whether after a long time the value of a stock or security will increase or decrease. Simple and exponential moving averages can be observed in the following Figure:1.



(a) Simple Moving Average graphs for the stocks Ashok Leyland, Suzuki, TATA motors, and TVS motors



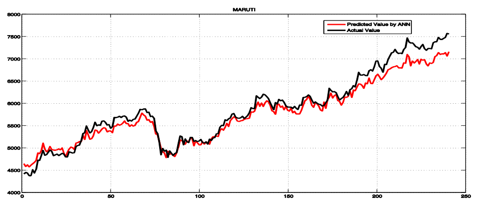
(b) Exponential Moving Average graphs for the stocks Ashok Leyland, Suzuki, TATA motors, and TVS motors

Figure 1: Moving Average graphs

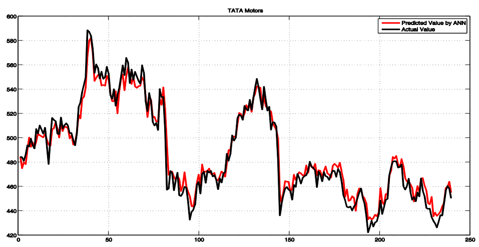
6.2. Results Discussion with BPNN

The BPNN has been trained to predict the stock value for the next day adding data for the last day. Data has been provided to the network in sliding window manner for 15-days, i.e., initially 70% of the data is allotted for training to the network and for in the testing phase, 15-days data was given. For the next testing, new 15-days data was given again for training and the most former data for 15-days was discarded. This process was carried out for whole testing data. We have plotted the prediction graphs in the Figure:2 for stocks MARUTI, TATA Motors, MAHINDRA & MAHINDRA and ASHOK LEYLAND and observed the closeness of predicted values as compared to original value.

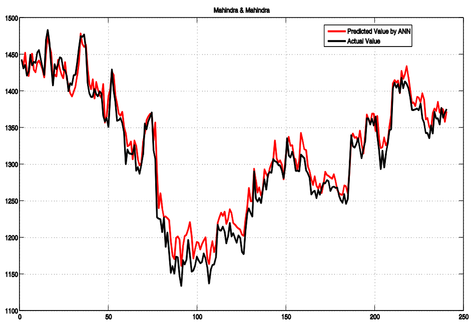
Further we have also sketched plot regression in Figure:3 based on predicted and actual values of stock prices where the R values in the respective sections of the figure demonstrate efficacies of the ANN technique.



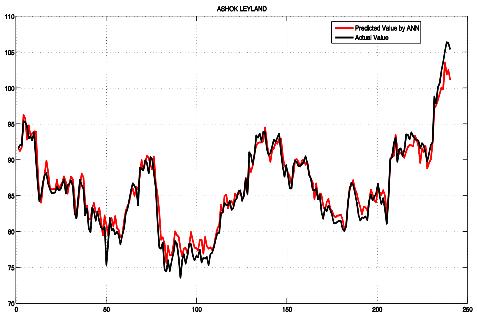
(A) MARUTI



(B) Tata Motors



(C) Mahindra & Mahindra



(D) Ashok Leyland

Figure 2: Prediction with BPNN in Sliding Window Manner

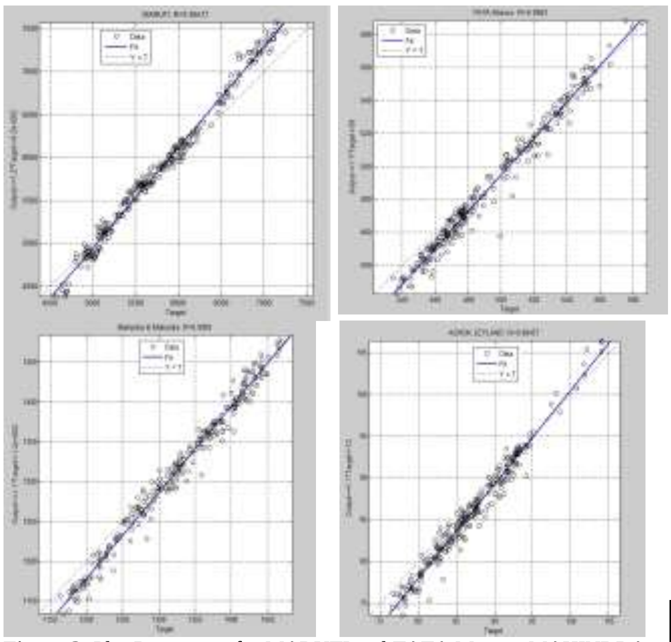
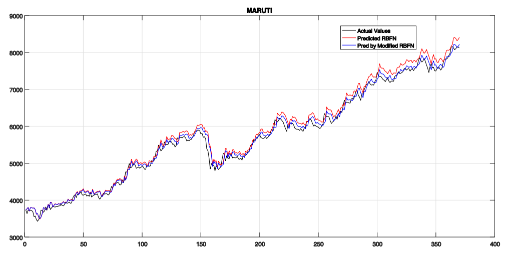


Figure 3: Plot Regression for MARUTI and TATA Motors, MAHINDRA & MAHINDRA and ASHOK LEYLAND with BPNN Technique

6.3. Results Discussion with Modified Rbfn

The next day closing value is predicted on the input of past data for the above considered stocks of companies. The most recent data will be added for the next day prediction. The effectiveness of models can be observed with the correlation coefficient values produced by corresponding plot-regression graphs given in the forthcoming figures.

Following graphs in Figure 8 represent the prediction for MARUTI and TATA Motors with existing RBFN and our modified RBFN techniques. The plot-regression graphs in Figure 9 have been produced for observing effectiveness of both RBFN and modified-RBFN.



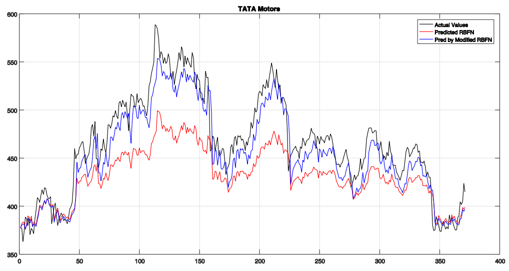


Figure 4: Prediction for MARUTI and TATA Motors with RBFN and modifiedRBFN Methods

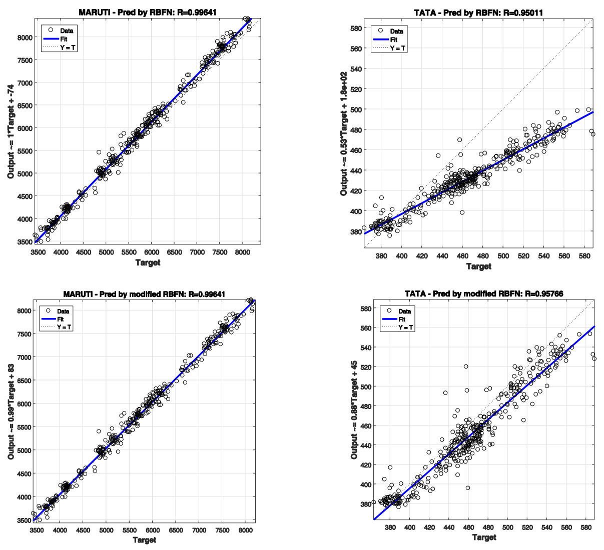


Figure 5: Plot Regression for MARUTI and TATA Motors with RBFN and modifiedRBFN Methods

Table 2 has been given as follows for comparative effectiveness analysis for forecasting, based on correlation-coefficient values for RBFN and modified RBFN models studied in this paper.

Table 2: Correlation-Coefficient values

	RBFN	Modified RBFN
Maruti	0.99641	0.99641
TATA	0.95011	0.95766

7. Conclusion

In ANN techniques are in trend for fitting the models in numerous sectors these days. Forecasting is one of the premises where any technique may not be appeared as a de facto for a particular model. The fact with ANN techniques for forecasting is that any model may not perform equally for all the considered stocks, however we require to examine the efficiency of any algorithm that is fit to a specified data with error values or correlation factor. So the model and data combination is important. In this paper, we observed the trend of data with moving averages. Further BPNN technique was examined with sliding window manner data and effectiveness of this model was noticed with correlation factor in plot regression

graphs. Radial basis function neural network was developed with regulation of basis as preprocessing of input data to the model.
These days data preprocessing are being focused so frequently as input data to artificial neural network model and we also followed the same in both of our models. There are numerous opportunities to regulate ANN techniques with optimization and data preprocessing methods from statistical and machine learning approaches such as principal component analysis, support vector machines, regression analysis, particle sworm optimization, etc. as the future works.

References

[1] Andersen T.G., Bollerslev T., Christoffersen P.F. and Diebold F.X, *Volatility and correlation forecasting*, Handb. Econ. Forecast., 1 (2006), 777–878.

[2] Anyaeche C. O. and Ighravwe D. E. , *Predicting performance measures using linear regression and neural network: A comparison*, African Journal of Engineering Research Vol. 1(3) (2013), 84-89.

[3] Box, G. E. P. , Jenkins, G. M. , and Reinsel, G. C. (1994), *Time series analysis: Forecasting and control*, Wiley.

[4] Mitra S. K, Usefulness of Moving Average Based Trading Rules in India, International Journal of Business and Management, 6(7) (2011).

[5] Deboeck G. J. (1994), *Trading on the edge: Neural, genetic, and fuzzy systems for chaotic financial markets*, New York, Wiley.

[6] Engle R. F, *Autoregressive conditional heteroscedasticity with estimates of the variance of United Kingdom inflation*, Econometrica, 50 (4) (1982), 987–1007.

[7] Enke D. and Thawornwong S, *The use of data mining and neural networks for forecasting stock market returns*, Expert Systems with Applications, 29 (4) (2005), 927–940.

[8] Franses P.H. and McAleer M, *Financial volatility: an introduction*, J. Appl. Economet. 17 (2002), 419–424.

[9] Horne V., James C. and Parker George G. C, *The Random-Walk Theory: An Empirical Test*, Financial Analysts Journal, (1967), 87-92.

[10] Ismail Z., Yahya A. and Shabri A. (2009), *Forecasting Gold Prices Using Multiple Linear Regression Method*, American Journal of Applied Sciences, 6 (8) (2009), 1509-1514.

[11] Ismail Z., Jamaluddin F. and Jamaludin F, *Time series regression model for forecasting, Malaysian electricity load demand*. Asian J. Math., 1(3) (2008), 139-149.

[12] Knight J.L. and Satchell S.S. (2007), *Forecasting Volatility in the Financial Markets*, 3*rd* Ed., Butterworth-Heinemann.

[13] Poon S. H. and Granger C, *Practical issues in forecasting volatility types ofvolatility models*, Financ. Anal. J. 61 (2005) 45–56.

[14] Pradeepkumar D and Ravi V, *Forecasting financial time series volatility using Particle Swarm Optimization trained Quantile Regression Neural Network,* Applied Soft Computing, 58 (2017), 35–52.

[15] Wang J. Z. , Wang J. J. , Zhang Z. G. and Guo S. P, *Forecasting stock indices with back propagation neural network*, Expert Systems with Applications, 38 (11) (2011), 14346–14355.

[16] Wang L, Haizhong A, Xiaohua X, Xiaojia L, Xiaoqi S and Xuan H (2014), *Generating Moving Average Trading Rules on the Oil Futures Market with Genetic Algorithms*, Hindawi Publishing Corporation , Mathematical Problems in Engineering, Article ID 101808, 10 pages, http://dx.doi.org/10.1155/2014/101808.

[17] Yahoo Finance, https://in.finance.yahoo.com/lookup, 17/07/2017, 12:15 PM.

[18] Yao J. , Tan L. C. and Poh H, *Neural networks for technical analysis: A study on KLCI*, International Journal of Theoretical and Applied Finance, 2 (2) (1999), 221–241 .

[19] Yaser S. and Atiya A. F, *Introduction to financial forecasting*, Applied Intelligence, 6 (3) (1996), 205–213.

[20] Yoo P. D., Kim M. H. and Jan T, *Machine learning techniques and use of event information for stock market prediction: A survey and evaluation*, International Conference on Computational Intelligence for Modelling, Control and Automation and International Conference on Intelligent Agents, Web Tech- nologies and Internet Commerce (CIMCA-IAWTIC’06): 2 (2007), 835–841, IEEE. doi: 10.1109/CIMCA.2005.1631572

[21] Zahra P. and Seyedmohsen R. (2014), *Comparing the Capabilities of Neural Networks and Data Envelopment Analysis in Predicting Corporate Profitability*, Technical Paper, 40 (2014), 1-111.