Applied Science & Engineering Journal for Advanced Research
ISSN (Online): 2583-2468
Volume-2 Issue-2 || March 2023 || PP. 19-25 DOI: 10.54741/asejar.2.2.4

A Perceiving and Recognizing Automaton Prediction for Stock Market

Om Prakash Tere[1] and Domesh Kshatriya[2]
[1]*Student, Department of Civil Engineering, Dr. D.Y. Patil College of Engineering, Pmpri, Pune, India*
[2]*Guide, Department of Civil Engineering, Dr. D.Y. Patil College of Engineering, Pmpri, Pune, India*

[1]*Corresponding Author: omprakash8787@gmail.com*

Received: 15-02-2023 Revised: 01-03-2023 Accepted: 24-03-2023

ABSTRACT
The skill of forecasting the value of a company's equity on the stock market In order to forecast stock market prices, this research suggests a machine-learning (ML) artificial neural network model. The back-propogation algorithm is integrated into the suggested algorithm. Here, we use the back-propogation algorithm to train our ANN network model. Additionally, we conducted research using the TESLA dataset for this publication.

Keywords: *back-propogation, artificial neural network, stock-market prediction*

I. INTRODUCTION

In the past few decades, prediction of stock price is gaining more attention as the profitability of the investors in the stock market is mainly depends on the predictability. If the direction of market is successfully predicted then investor can yield enough profit. For solving the such kind of financial problem the relationship between the input and output is very complex so that's why we have used ANN for solving or predicting the stock price.

An artificial neural network model is computer model whose architecture essentially mimics the learning capability of human brain. The processing element of artificial neural network resembles the biological structure of neuron and the internal operation of human brain.

In this paper , Multilayer feed forward back-propogation neural network is used for the prediction purpose. Feed forward neural network is unidirectional connection between the neurons that means the information can flow only in forward direction. Here there is no connection between the neurons present in the same layer. Input has been fed into first layer and with the help of hidden layers connected to the last layer that produces the output. And since all of the information is constantly feeding forward from one layer to the next hence it is called feed forward network.

One of the learning methods in multilayer Perceptron Neural Networks is the error back-propogation in which the network learns the pattern in the dataset and justifies the weight of the connections in the inverse direction respect to the gradient vector of error function which is usually regularized sum of

Square error. The back-propogation method picks a training vector from training data set and moves it from the input layer toward the output layer. In the output layer the error is calculated and propogated backward so the weight of the connection will be corrected. This will usually go on until the error reaches a pre-defined value. Its proved that we can approximate any continuous function with a three layer feedback network with any precision. It should be said that the learning speed will dramatically decreases according to the increase of the number of neurons and layer of the network.

1.1. Multilayer Feed Forward Perceptron

In this paper we have used multilayer feed forward perceptron below figure illustrate the how the multilayer feed forward perceptron looks like.

Multilayer: In multilayer neural network what happened there are more number of hidden layer are available in between the input layer and output layer called as hidden layer so in multilayer perceptron neural network more than one hidden layer are available.

Feed forward : in feed forward neural network what happened there is no edges are available in between the neurons present in the same layer here the synaptics are available in between only the neuron present in the different layer of neural network.

Applied Science & Engineering Journal for Advanced Research
ISSN (Online): 2583-2468
Volume-2 Issue-2 || March 2023 || PP. 19-25 DOI: 10.54741/asejar.2.2.4

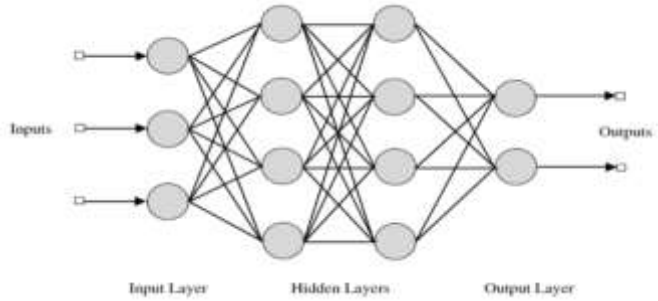

Figure 1: Feed forward Multilayer Perceptron

II. ALGORITHMS

2.1 Back-propogation Algorithm

The back-propogation algorithm is falls into the general categ-ory of gradient descent algorithm. Purpose of gradient descent algorithm is to find the minima and maxima of a function by iteratively moving in the direction of negative slope of the function that we want to minimize or maximize.

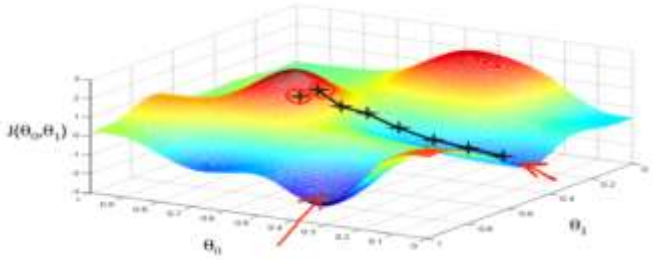

Figure 2: Gradient Descent

In back-propogation algorithm the network is trained by repete-dly processing the training data set and comparing the network output with the actual output if anything differs between the what output we are getting from our network and the desire output then this is called as error so after that what we are doing is we are again back propogate the whatever the output we got from our neural network to the back layer of the neural network till first hidden layer so that again our network will assign different weight randomly to the synaptic and try to generate the output with lesser error from the previous time and at every iteration our network output should be less from the output of previous time so that after some iteration we should get desire output and here we are keep back propogate the output from our neural network till our network trained completely. This below flowchart shows how back-propogation algorithm will work.

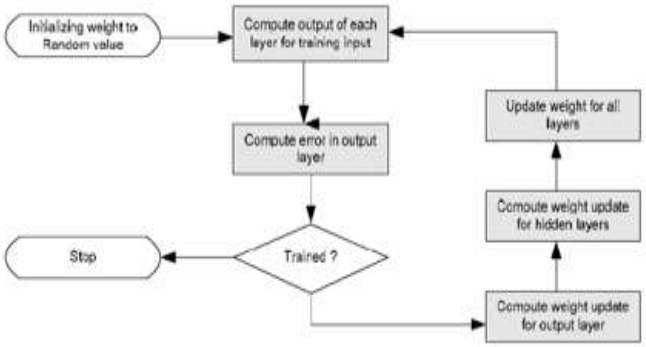

Figure 3: Flowchart of Back-propagation Algorithm

2.2 Mathematical Derivation for Back-propogation Algorithm

In back-propogation our main intension is to find out the what changes is supposed to be happened on the weight assigned to the synaptic when the output we are getting from our network model is again backpropogated to the again first hidden layer so in this computation our main intension is to calculate Δwij.

First equation is about finding out the error generated from the neural network.

$$Ej\ (n) = dj\ (n) – yj\ (n) \quad ..equ(1)$$

The above equation shows the error generated from our network and it can be calculated by just taking the difference of desired output and output we are getting from out neural network. Here the term dj(n) will represent the desired output we are getting from out neural network and the term yj(n) is the output we got from our neural network . after that we found out the square error by just doing this.

$$E\ (n) = ½ \sum j€C \ \ ej2\ (n) \quad equ(2)$$

The above equation shows the error energy here we are actually intended to calculate the square energy by just doing square of all of the error and summing up the squared error we are getting from all of the neurons present in the output layer. Now we will find the average error energy.

$$Eavg\ (n) = 1/N \ \sum Nn=1\ E(n) \quad equ(3)$$

In above equation N is the number of iteration and we are summing up all the error energy from 1st iteration to N iteration for calculating the average error energy. Now we will calculate the value for induced field.

$$Vj(n) = \sum mi=1\ Wij(n) * Yij(n) \quad equ(4)$$

Local induced field value can be calculated by just doing summation of multiplication of Wij(n) and Yij(n) from number neurons in previous layer where is represent the number of neuron in previous layer and the value of Yij can be calculated by applying the activation function over induced field of Jth layer neuron below equation will illustrate this.

$$Yj(n) = ϕ(Vj(n)) \quad equ(5)$$

Now we are intended to find out the what is change is happened in error with respect to change in the weight by just applying the chain rule of differentiation.

Here From equ(2) ej(n)

From equ(1) = -1

From equ(5) = ϕ,(Vj(n))

From equ(4) = Yj(n)

By putting this all of the value in equation 6 we got the following result:

$$-ej(n).\ ϕ,(Vj(n)).\ Yj(n) \quad ..equ(7)$$

Δwij is applied to the Wij and which is proportional to the so according the definition of proportionality we can write as .

$$\Delta Wij = -η \quad equ(8)$$

In above equation η is constant of proportionality and the value of η is 9.25.from the equation (7) we can write equation (8) as .

$$\Delta Wij = η\ ej(n).\ ϕ,(Vj(n)).\ Yj(n)$$

$$δj(n) = ej(n).\ ϕ,(Vj(n)). \quad equ(9)$$

$$\Delta Wij = \eta\ \delta j(n)\ Yj(n) \quad \text{equ(10)}$$

The below figure shows the signal flow of back-propogation algorithm .

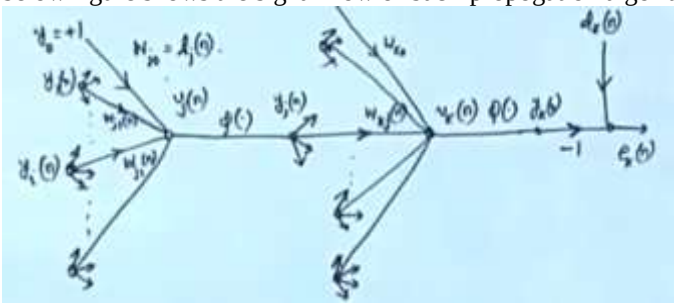

Figure 4: Signal Flow of Back-propagation Algorithm

Now we can find out the $\delta_{j,}$

$$\delta_j = -\frac{dE(n)}{dvj(n)}$$

$$= -\frac{dE(n)}{dyj(n)}.\frac{dyj(n)}{dvj(n)}$$

$$= -\frac{dE(n)}{dyj(n)}.\phi,(Vj(n))$$

Now as above figure if we consider k as output layer and j as preceding layer then calculation might looks likes:

$$E(n) = \tfrac{1}{2} \sum_{k\epsilon C} e_k^{\,2}(n) \quad \text{equ(11)}$$

$$\frac{dE(n)}{dyj(n)} = \sum_k e_k(n).\frac{dek(n)}{dyj(n)} \quad \text{equ(12)}$$

$$\frac{dE(n)}{dyj(n)} = \sum_k e_k(n).\frac{dek(n)}{dvk(n)}\ .\frac{dvk(n)}{dyj(n)} \quad \text{equ(12)}$$

$$e_k(n) = d_k(n) - y_k(n)$$

$$= d_k(n) - \phi(V_k(n)) \quad \text{equ(13)}$$

$$\frac{dek(n)}{dvk(n)} = -\phi(Vk(n)) \quad \text{equ(14)}$$

For neuron k,

$$V_k(n) = \sum_{j=0}^{m} W_{kj}(n) * Y_j(n) \quad \text{equ(15)}$$

$$\frac{dvk(n)}{dyj(n)} = W_{kj}(n) \quad \text{equ(16)}$$

So now equation 12 can be written as

$$\frac{dE(n)}{dyj(n)} = -\sum_k e_k(n).\ \phi\big(Vk(n)\big).W_{kj}(n)$$

$$= -\sum_k \delta_k(n)\ .W_{kj}(n)$$

$$\delta_k(n) = e_k(n)\ \phi'\big(Vk(n)\big)$$

So if we talk about back-propogation algorithm in a nutshell then It can be represented like this:

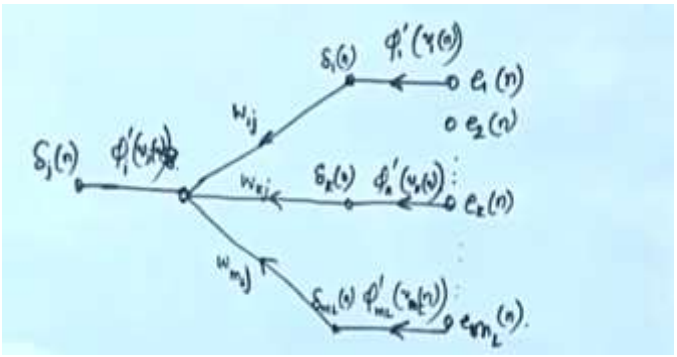

Figure 5: Back-propagation Algorithm in Nutshell

III. METHODOLOGY

We will be using stock market data to predict closing price the workflow for the general neural network design has five primary steps:
a) Data collection and preparation
b) Network creation
c) Training the network
d) Validating the network
e) Using the network

a) Data Collection and Preparation

Data collection is the primary step and it is necessary in order to train, validate and test the neural network.

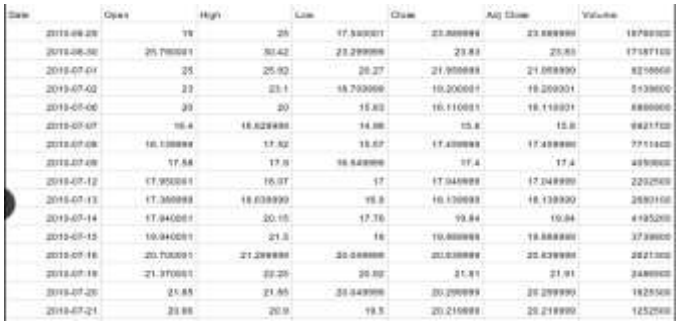

Figure 6: Tesla Data Set

For collection the data google finance has been used for collection the historical stock price details of any one of the company. This all of the data set are then feded into the network.

b) Network Creation

After collecting the all of the data set the next hectic task is to create your neural network model here the selection of what type of neural network is going to be difficult task. Neural network model like supervised or unsupervised and single layer or multilayer you should choose any one of those which is appropriate to solve your problem in our case it is supervise and multilayer perceptron.

c) Training the Network

As we are going to solve the problem with the help of artificial neural network then actually what we are trying to do is we are actually mimicking the functionality of biological neural to some extent since there is requirement of training your brain here also there is requirement of training your network for doing the task by itself once the your network trained this will

perform every task correctly so for training the network there is lots of way but in our case we have used back-propogation algorithm because the efficiency of back-propogation algorithm is high very high as we are doing back propogate error.

d) Validating the Network

Once the training has done the network are validated using the validated data to enhance the performance of the network.

e) Using the Network

Once the network are optimized . It has been tested using the test data. In our case collected data of TESLA has been used to predict the adjusted closing price of stock.

Some of the screenshot our results are shown below:

Date	Open	High	Low	Close	Volume	Ex-Dividend
2010-06-29	19.00	25.0000	17.54	23.89	18766300.0	0.0
2010-06-30	25.79	30.4192	23.30	23.83	17187100.0	0.0
2010-07-01	25.00	25.9200	20.27	21.96	8218800.0	0.0
2010-07-02	23.00	23.1000	18.71	19.20	5139800.0	0.0
2010-07-06	20.00	20.0000	15.83	16.11	6866900.0	0.0

Date	Split Ratio	Adj. Open	Adj. High	Adj. Low	Adj. Close
2010-06-29	1.0	19.00	25.0000	17.54	23.89
2010-06-30	1.0	25.79	30.4192	23.30	23.83
2010-07-01	1.0	25.00	25.9200	20.27	21.96
2010-07-02	1.0	23.00	23.1000	18.71	19.20
2010-07-06	1.0	20.00	20.0000	15.83	16.11

Date	Adj. Volume	100ma
2010-06-29	18766300.0	23.890000
2010-06-30	17187100.0	23.860000
2010-07-01	8218800.0	23.226667
2010-07-02	5139800.0	22.220000
2010-07-06	6866900.0	20.998000

Accuracy: 0.9395375335435104

Figure7: Stock Data Set

Figure 8: Closing and Moving Average

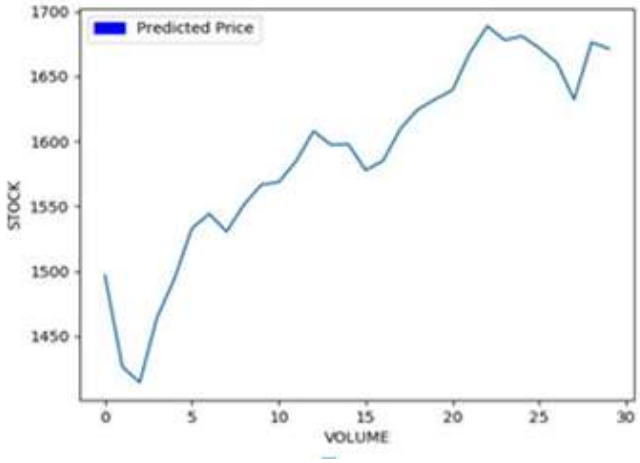

Figure 9: Predicted Price

IV. CONCLUSION

By using historical stock market value information, we applied neural network models to forecast stock share values in these papers. Multilayer feed-forward networks are used to achieve this goal and address the issue. The outcome demonstrates that, with a 94% accuracy rate, there is no approach that is superior to the back-propogation algorithm for predicting the direction of changes in stock value.

REFERENCES

1. Palavi Ranka. (2019). Stock market prediction using artificial neural networks. *IME611 - Financial Engineering Indian Institute of Technology, Kanpur (208016), India*.
2. Rosenblatt, F. (1997). The perceptron: A perceiving and recognizing automaton. *Cornell Univ. Ithaca. NY. Project PARA Cornell Aeronaut Lab*, pp. *85-460*.
3. Mandic, D. P., & Chambers, J. A. (2001). *Recurrent neural networks for prediction*. John Wiley & Sons, LTD.
4. Goh, T. H., Wang, P. Z., & Lui, H. C. (1992). Learning algorithm for enhanced fuzzy perceptron. *Proceedings of IJCNN, 2*, pp. 435.
5. Lippmann, R.P. (1987). An introduction to computing with neural nets. *IEEE Accost. Speech Signal Process. Mag.*, pp. 4-22.