East West University

Stock Market Price Prediction Using Artificial Neural Network

Submitted by:

Ismith Pasha

ID: 2012-1-60-004

Supervised by:

Dr. Mohammad Rezwanul Huq

Assistant Professor

Department of Computer Science and Engineering

East West University

A project submitted in partial fulfillment of the requirements for the degree of Bachelor of Science in Computer Science and Engineering to the Department of Computer Science and Engineering

December 2016

Abstract

This report represents the Artificial Neural Networks approach to predict stock market price. Stock market prices are actually time-series data and Artificial Neural Networks (ANNs) have the ability to find non-linear correlations between time-series data which makes it the best approach to predict stock market prices. Historical data from Dhaka Stock Exchange is used to train and predict the price by using ANN. The Artificial Neural Network (ANN) is implemented using multi-layer Feed-forward Backpropagation algorithm. To predict the specific result the model has been trained in different category of networks based on time period. Three different time period data have been chosen as one year data, six months data and three months data. Each categorized data has been trained by and tested Feed-forward Backpropagation algorithm. After the training and testing process the predicted values are compared with the real data to find the accuracy. The trained network with the highest accuracy rate will able to predict the best possible price of the stock market.

Declaration

I hereby declare that, this project was done under CSE497 and has not been submitted elsewhere for requirement of any degree or for any reason except for publication.

____

Ismith Pasha

ID: **- 2012-1-60-004**

Department of Computer Science and Engineering

East West University

Letter of Acceptance

This thesis is submitted by Ismith Pasha, Id: 2012-1-60-004 to the Department of Computer Science and Engineering, East West University, Dhaka Bangladesh is accepted as satisfactory for the partial fulfillment of the requirement for the degree of Bachelor of Science in Computer Science and Engineering on December 8th, 2016.

1. ____________________

Dr. Mohammad Rezwanul Huq

Assistant Professor (Supervisor)

Department of Computer Science and Engineering

East West University, Dhaka, Bangladesh

2. ________________________

Dr. Md. Mozammel Huq Azad Khan

Professor and Chairperson

Department of Computer Science and Engineering

East West University, Dhaka, Bangladesh

Acknowledgement

First of all, I would like to thank Almighty Allah for giving me the strength and patience to complete this work.

Without the support of supervisor from past few months it will not possible for me to complete this thesis. My sincere thanks to Dr. Mohammad Rezwanul Huq who is the source of inspiration for me throughout the process of the research, writing and implementation. His feedback and insights were always valuable, and never went unused.

My deep gratefulness to all of the teachers, who have been taught and trained us at East West University. Their wonderful teaching methods improved my knowledge of the individual subject and enabled me to complete my studies in time.

In this acknowledgment would not complete without thanking my parents. They always support me on my studies and also throughout every part of my life. I hope this achievement will cheer them up during these demanding times.

Table of Contents

Contents

List of Figures

List of Tables

Chapter 1

1.1 Introduction

A stock market is actually the place where the stock buyers and sellers are met and do their business. In a stock market there are some public listed markets. Those markets are traded their shares to the stock market. Stocks can be both bought or sold when it is listed for an exchange. By trading on stock market a trader can grow his small investment to large one and can build a high paying career for himself. But it is not possible that everyone can get profit at the same time. If someone is making a huge profit than on the other side someone is also losing. So in a stock market every decisions and steps should be taken wisely and carefully. The stock market environment is always changing. Sometimes the price is increasing and sometimes it decreasing. So a trader needs to invest carefully in each and every situation.

Prediction of stock market price is actually helps a trader to take his decision easily. By trading on a stock market one can make his small initials of money into a large amount. But it is only possible if he takes his every decision wisely and carefully. If a trader can predict what is going to be happened with the market price than he can avoid losses. Stock market prices can be predicted with different methods. A time series method can be used for predicting stock market price. Artificial Neural Networks (ANN) [3] is one of the best methods which can be used for predicting stock market price. Using Artificial Neural Networks (ANN) the frequently changing price can be trained as an artificial intelligent agent and using that agent the upcoming or future price can be predicted. As stock market prices are time-series data a time-series algorithm of ANN will be the best choice to do this operation. Feed-forward backpropagation [1] can be used for this purpose. The positive side of feed-forward backpropagation algorithm there is no cycle and it completes its operation with some layers. To train a network using feed-forward backpropagation, some data should be given as input data and some should be set as target data. The algorithm will train the network based on the data. After training data some more data should be given as input for testing purpose. The sample testing data will generate the predicted results. After getting the predicted prices it can be compared with the real prices from the real data. Then if the accuracy rate of prediction is high enough the network can be used to build an application which can help a stock market trader to predict the future price.

In this report the data is collected from Dhaka Stock Exchange (DSE) [2], a stock market in the capital of Bangladesh. For training and testing purpose the data of Grameen Phone (GP) is collected from Dhaka Stock Exchange. The data is categorized in three parts as three months, six months and one year. Each and every category is trained separately. After training process each network has to go through the testing process to find out the best trained network or intelligent agent to predict the price. In testing process the predicted value from sample data is compared with the real value by doing some validation process. Different validation algorithm or process can be used to find the accuracy. In this project K Fold Cross Validation is applied. Using the k-fold cross validation each categorized data is divided into five folds or parts. For each category of data one part is used as testing data and other four parts together used for testing data. Total five time data have been trained and tested for each categorized data. With this process the most accurate network is chosen to predict the stock market price [14].

1.2 Objective

The main objective of the thesis is to predict Stock Market Price by using Artificial Neural Network [10] and to develop a system so that Stock holders can easily take decision whether they are going to buy or sell their stocks. This can bring more profit to the business by helping the traders to take their decisions wisely and appropriately.

1.3 Hypothesis

The goal of the report is to predict the best possible closing price of a stock market. In this report the prediction process is completed with short-term perspective where a short duration of historical data has been used for training and testing purpose using Artificial Neural Networks.

Here, three short time categorized data has been used – (i) One year data, (ii) Six months data and (iii) Three months data.

After training and testing data the best trained network with most accuracy has been chosen by validating the predicted and real value.

Chapter 2

Background

2.1 Stock Market

A stock market is the meeting place of the stock buyers and sellers where they buy sell or exchange their stocks. In a stock market some companies are listed publicly who traded the shares. Stocks can be bought or sold when it is listed on an exchange. By trading on a stock market a trader can grow his small investment to large one and can build a high paying career.

2.2 Dhaka Stock Exchange (DSE)

The Dhaka Stock Exchange was incorporated in 1954, and formal trading began in 1956. Originally, the exchange was called the East Pakistan Stock Exchange Association Ltd, and in 1962 the name was revised to East Pakistan Stock Exchange Ltd. Two years later, on Feb. 22, 1964, the name was changed to current day Dhaka Stock Exchange Ltd. The Dhaka Stock Exchange is registered as a Public Limited Company and is regulated by the Bangladesh Securities and Exchange Ordinance of 1969, the Companies Act of 1994 and the Securities and Exchange Commission Act of 1993.

2.3 Neural Networks

2.3.1 Biological Neuron

Our brains utilize greatly substantial interconnected networks of neurons with methodology majority for processing the data and model, the planet we live in. Electrical sources of information are gone through this system of neurons which result in a yield being delivered. On account of an organic mind this could bring about getting a muscle or flagging your sweat organs to create sweat. A neuron gathers inputs utilizing a structure called dendrites A neuron gathers inputs utilizing a structure called dendrites, the neuron successfully gathers these contributions from the dendrites and if the subsequent esteem is more prominent than it's terminating limit, the neuron fires. At the point when the neuron fires, it sends an electrical impulse through the neuron's axon to it's terminal bulbs. These terminal bulbs can then be networked to thousands of other neurons via connections called synapses. There are about

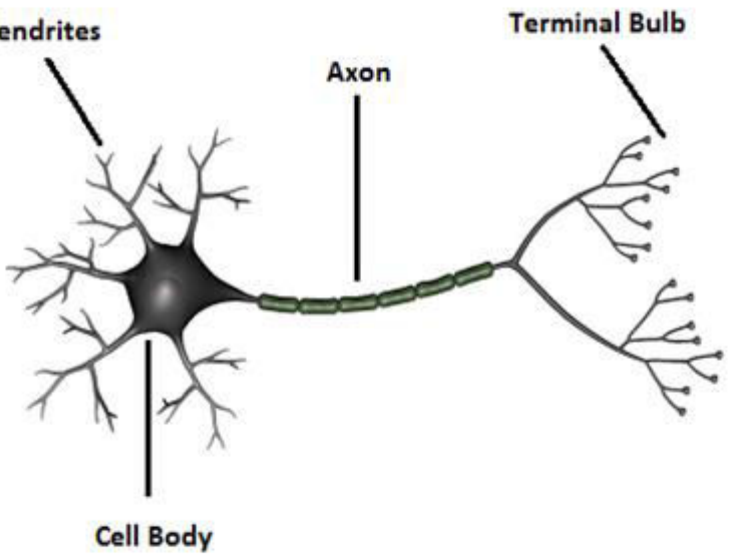


Figure 2.1: Biological Neuron

one hundred billion (100,000,000,000) neurons inside the human brain each with about one thousand synaptic connections. It's effectively the way in which these synapses are wired that give our brains the ability to process information the way they do.

2.3.2 Artificial Neuron model

Artificial neuron models are the core simplified models based on biological neurons [5]. This allows to capture the essence of how a biological neuron functions. These artificial neurons are usually known as 'perceptrons'.

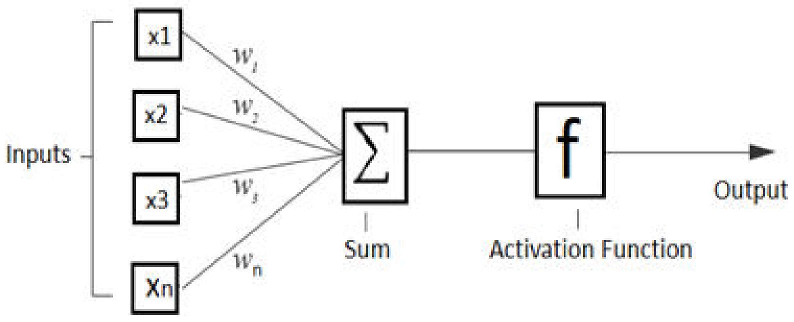


Figure 2.2: Artificial Neuron Model

A perceptron will have many inputs and these inputs are all individually weighted, shown in the figure. The perceptron weights can either amplify or deamplify the original input signal. For example, if the input is 1 and the input's weight is 0.2 the input will be decreased to 0.2. Then these weighted signals will be added together and passed into the activation function. The activation function is used for converting the input into a more useful output. There are many different types of activation function but one of the simplest would be step function. A step function will typically output a 1 if the input is higher than a certain threshold, otherwise it's output will be 0.

For example,

```
Input1 (x1) = 0.8              Weight1 (w1) = 0.6
Input2 (x2) = 0.8              Weight2 (w2) = 0.5

Input3 (x3) =0.6               Weight3 (w3) = 0.4

Threshold = 1.0
```

First multiplying the inputs by their weights and sum them:

$$x1w1 + x2w2 + x3w3 = (0.8 * 0.6) + (0.8 * 0.5) + (0.6 * 0.4) = 1.12$$

Now we compare our input total to the perceptron's activation threshold. In this example the total input (1.12) is higher than the activation threshold (1.0) so the neuron would fire.

2.3.3 Artificial Neural Networks

In an Artificial Neural Network, each input from the input layer is feed up to each node in the

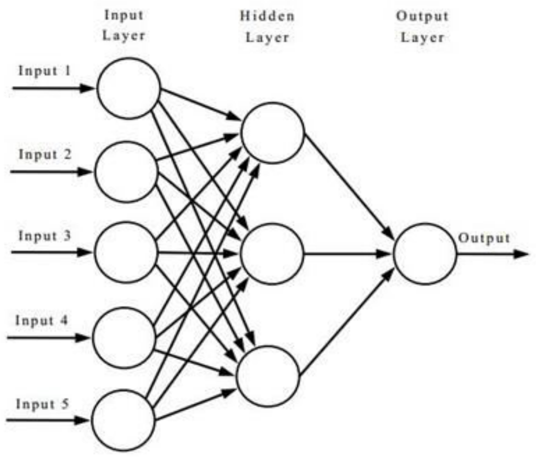


Figure 1.3: Artificial Neural Network

hidden layer, and from there to each node on the output layer. There can be any number of nodes per layer and there are usually multiple hidden layers to pass through before ultimately reaching the output layer. It is important to choose the right number of nodes and layers when optimizing the neural network. As per the diagram, it's called a feed-forward network because of how the signals are passed through the layers of the neural network in a single direction. There are many others neural network present. There are also feedback networks where its architecture allows signals to travel in both directions.

The Input Layer

The input layer can be thought of as the “sensor organ” of the ANN. It is the place where user set the parameters of the environment (i.e. the information ANN to make a decision about). The neurons in this layer have no incoming connections, since their values are set from an external source. The outgoing connections send these values to the neurons of the next layer in the hierarchy.

The Hidden Layer(s)

Normally "hidden" layers are present in between input and output layers. It is called hidden because they are invisible to any external processes that interact with the ANN. The neurons in these layers have both incoming connections from the preceding layer and outgoing connections to the succeeding layer, and work just as described earlier in this section. The hidden layers can be thought of as the "cognitive brain" of the network.

The Output Layer

The output layer is the final layer where the end result of the computations of the ANN. If the input layer holds the parameters of a problem, the information can be interpreted. The neurons in this layer have no outgoing connections, because their φ-values are read directly by whatever external process is using the network.

2.3.4 Learning in Neural Network

There are many different algorithms for learning artificial neural networks, each have their own separate advantages and disadvantages. The learning process within artificial neural networks is a result of altering the network's weights, with some kind of learning algorithm.

The main objective is to find a set of weight matrices, when applied to the network should map any input to a correct output. There are three major learning paradigms, they are-

Supervised Learning: The learning algorithm is used when the desired output for the network is also provided with the input while training the network. By providing the neural network, both an input and output pair, it is possible to calculate an error based on it's target output and actual output. Then it use that error to make corrections to the network by updating it's weights.

Unsupervised Learning: In this paradigm the neural network is only given a set of inputs and it's the neural network's responsibility to find some kind of pattern within the inputs provided without any external aid. This type of learning paradigm is often used in data mining and is also used by many recommendation algorithms due to their ability to predict a user's preferences based on the preferences of other similar users it has grouped together.

Reinforcement Learning: Reinforcement learning is similar to supervised learning, however instead of providing a target output a reward is given based on how well the system performed. The aim of reinforcement learning is to maximize the reward the system receives through trial-and-error. This paradigm relates strongly with how learning works in nature. For example an animal might remember the actions it's previously taken which helped it to find food (the reward).

2.4 MATLAB

MATLAB is a platform, which is developed for solving engineering and scientific problems in an optimized way. The matrix-based language "MATLAB", used worldwide to express computational mathematics in a most natural way. Inherent graphics make it not difficult with visualize also increase insights starting with information. A limitless library from claiming prebuilt toolboxes gives someone to get started right away with algorithms essential to domain. The desktop nature's domain invites experimentation, exploration and also finding. These MATLAB tools and capabilities are all rigorously tested and designed to work together.

2.4.1 Neural Network Toolbox

Neural system toolmaker provides for calculations, capacities, what's more provisions on make, prepare, imagine, and imitate neural frameworks. One can easily perform arrangement, relapse, grouping, dimensionality decrease, time-arrangement estimating, and dynamic framework demonstrating and control.

The Neural Network Toolbox incorporates convolutional neural system and also auto encoder deep learning calculations for picture arrangement and highlight learning undertakings.

2.4.2 Feed-forward network

In a Feed-forward networks system, it's consist with a series of layers. The first layer which is connected with the network input. Each subsequent layer has a connection from the previous layer. The final layer produces the network's output.

It can be used for any sort of input to output mapping. A feed-forward network can fit any finite input-output mapping problem when it has one hidden layer and enough neurons in the hidden layers.

2.4.3 MATLAB GUI

GUI (Graphical User Interfaces) or UI provide point-and-click control of software applications and derives the difficulty of learning a language or type commands in order to run the application.

MATLAB applications are independent MATLAB projects with GUI front ends that automate a task or calculation. The GUI commonly holds control such as menus, toolbars, buttons, and sliders. Many MATLAB products, for example Curve Fitting Toolbox™, Signal Processing Toolbox™, and Control System Toolbox™ include application with custom user interfaces. Users can also create their own custom applications, including their corresponding UIs, for others to use.

2.4.4 MATLAB App Designer

App Designer [6] is a domain for building MATLAB apps which integrates the two primary tasks of app building. One is laying out the visual components and the other one is programming app behavior. It also allows users to quickly move between visual design in the canvas and code development in an integrated version of the MATLAB Editor.

Chapter 3

Architecture of Proposed Solution

3.1 Overview

The architecture to complete the full process is shown in the figure 3.1-

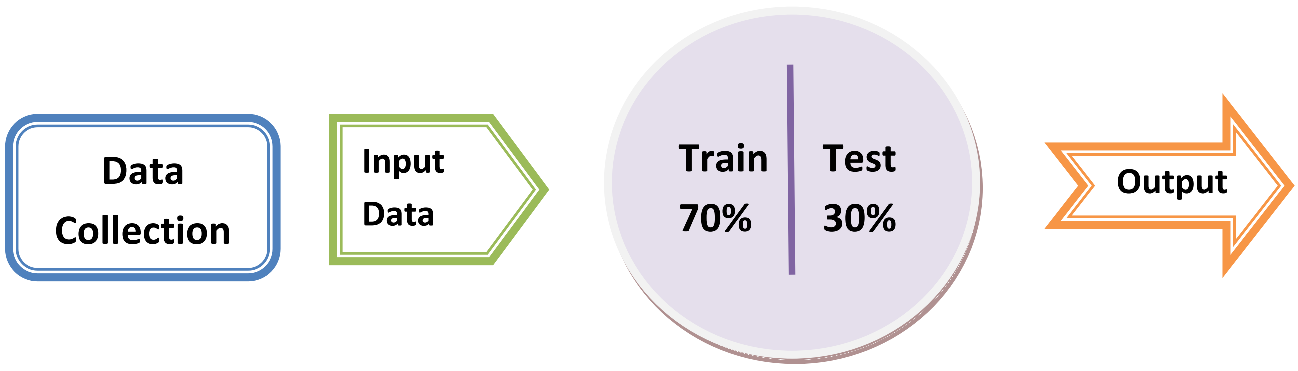


Figure 3.1: Architecture of the Process

The basic steps are –

1) Data Collection
2) Implementation
    a) Input Data
    b) Training
    c) Testing
3) Validation and
4) Result

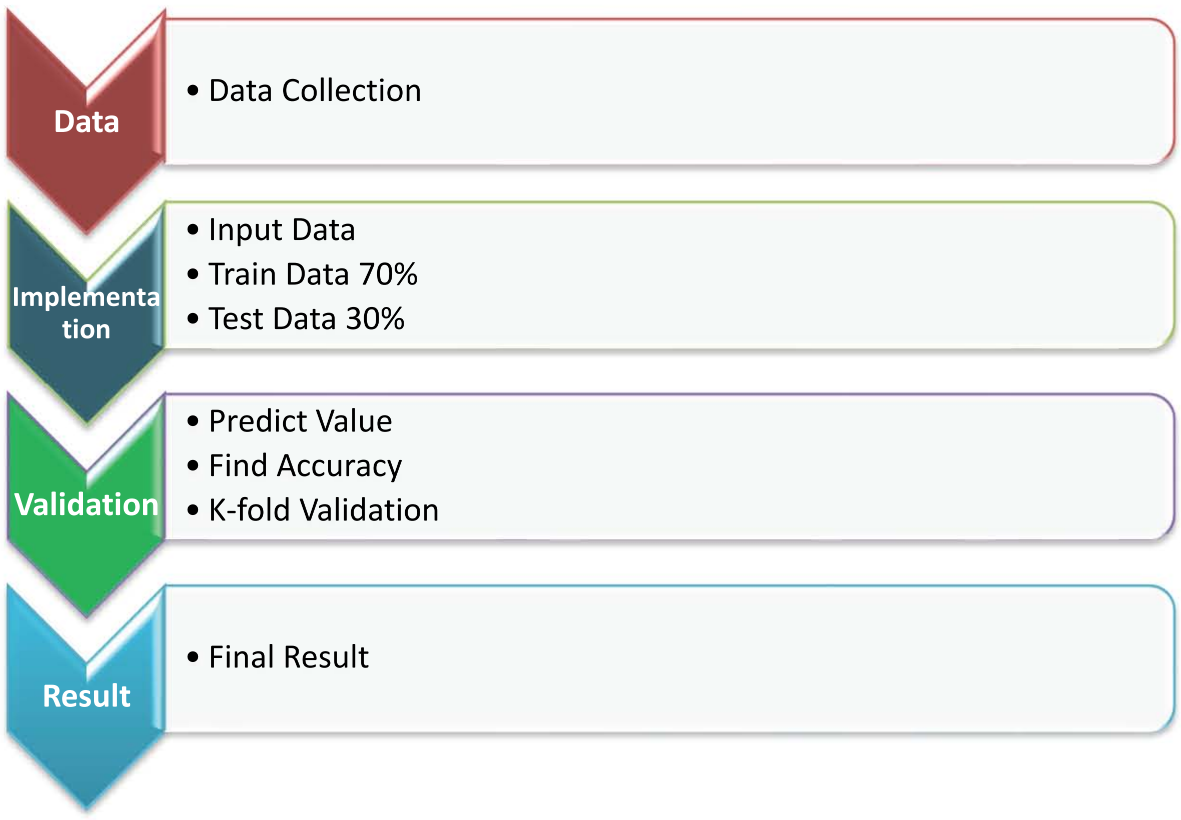


Figure 3.2: Flow Diagram of the Process

3.2 Process

3.2.1 Data Collection

The stock data used in this report is represented by DSE (Dhaka Stock Exchange). In particular, the data represents closing prices and daily starting price for each one of the companies, collected from Grameen Phone (GP). It has been taken into consideration that the stock market is closed on on specific holidays, including weekends.

3.2.2 Implementation

The ANN was implemented by MATLAB 2016a which is a programming language used for mathematical computations. The implementation included construction, training, testing and evaluation of the ANN. The data is trained by Feed-forward [9] backpropagation algorithm using Matlab 2016a.

For training purpose 70% of data from every category have been used. Other 30% data have been used for testing purpose.

3.2.3 Validation

Using the trained network next closing price of share is predicted. To find the accuracy of the network the formula has been used:

$$\text{Accuracy} = \frac{the\ number\ of\ days\ correctly\ classified\ the\ testing\ data}{total\ number\ of\ testing\ days}$$

To calculate the accuracy maximum 5% error is used as tolerance rate.

K-fold validation is used to find the most appropriate trained network.

3.2.4 Result

Finally the result will come out by analyzing the predicted value with the real value and it will help the share holder to do better in Stock Market.

Chapter 4

Development

4.1 Background

4.1.1 Developing Tool Used

MATLAB 2016a, because MATLAB is a highly resource platform with a lot of algorithm. It makes the implement process easier and more accurate.

4.1.2 Algorithm and Programming Language Used

For training, testing and predicting stock market price the algorithm Feed-forward Backpropagation used. Because the price of a stock market is a sequential data and Feed-forward Backpropagation is appropriate for this type of data.

As MATLAB is using for development process so MATLAB's own programming language is used for implementation process.

4.1.3 User Interface (UI)

MATLAB App-designer [6] from MATLAB GUI [13] is used for user interface design.

4.2 Training

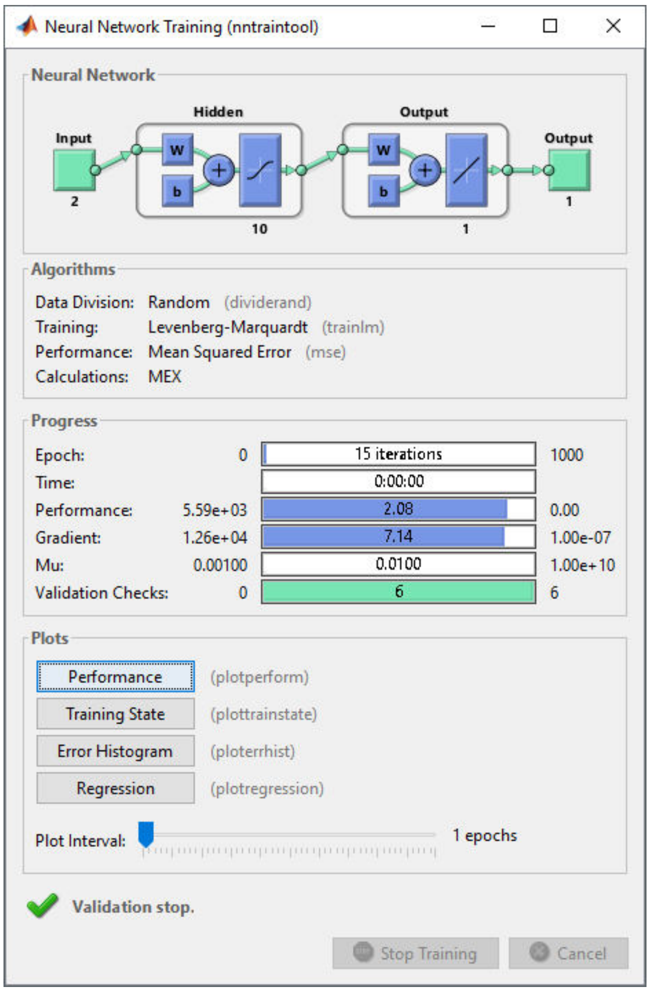
For training purpose 3 different categories of data have been used. The data is collected from Grameen Phone (GP) under Dhaka Stock Exchange (DSE) [2].

- **One Year Data:** The time period of the data is from 1st November, 2015 to 31st October 2016.
- **Six Months Data:** The 6 months time period of the data is from 2nd May, 2016 to 31st October 2016.
- **Three Months Data:** The 3 months time period of the data is from 1st August, 2016 to 31st October 2016.

All of the categories of data are saved in MATLAB Workspace. Each of the categories are also categorized as Input data, Target data and Sample data for testing.

To train the data Feed-forward Backpropagation have been used using MATLAB. The training algorithm is -

```
load ('Gp1Y.mat','-mat');
net1y = feedforwardnet(10,'trainlm');
net1y = train(net1y,input1y',target1y');
output1y = net1y(sample1y')';
```


Neural Network Training (nntraintool)

Neural Network

Input
2
Hidden
W
b
10
Output
W
b
1
Output
1

Algorithms

```
Data Division:  Random  (dividerand)
Training:       Levenberg-Marquardt  (trainlm)
Performance:    Mean Squared Error  (mse)
Calculations:   MEX
```

Progress

```
Epoch:               0     15 iterations   1000
Time:                      0:00:00
Performance:   5.59e+03    2.08            0.00
Gradient:      1.26e+04    7.14            1.00e-07
Mu:             0.00100    0.0100          1.00e+10
Validation Checks:   0     6               6
```

Plots

Performance (plotperform)
Training State (plottrainstate)
Error Histogram (ploterrhist)
Regression (plotregression)
Plot Interval: 1 epochs
Stop Training
Cancel

✓ Validation stop.

Figure 4.1: Training Data using Neural Network

Here,

- 'Gp1Y.mat' file contains one year data of Grameen Phone,
- **"net1y"** is the Network name for training.
- `feedforwardnet(10,'trainlm')` is the function for Feed-forward Backpropagation algorithm and here 10 is the number of node in hidden layer.

- `'trainlm'` [4] is the network training function that updates weight and bias values which is highly recommended for supervised algorithm.
- `train(net1y,input1y',target1y')` [7] is the function to train the network.
- "`input1y`" and "`target1y`" are the input and target data set from `'Gp1Y.mat'` file.
- `sample1y'` is the sample data for testing purpose and `output1y` keeps the predicted values by using "`net1y(sample1y')'`" function.

70% of the data is used for training the network and other 30% is used for testing purpose.

4.3 Testing & Validation

For testing the network 30% of the data have been used. To find the accuracy rate from the predicted data this formula have been used –

$$Accuracy = \frac{(real\ data - predected\ data)}{real\ data} \times 100$$

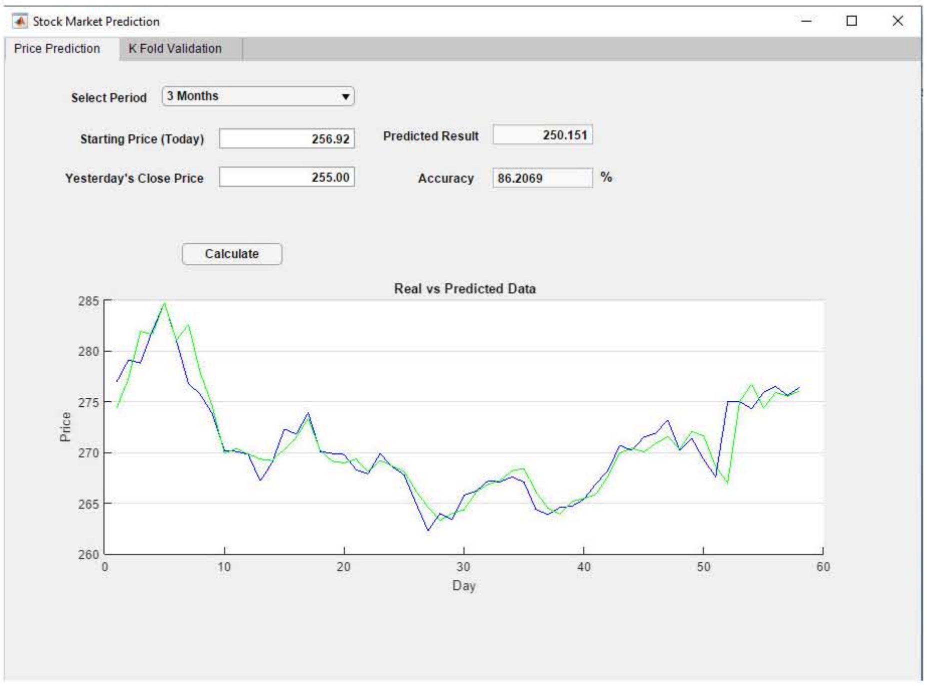
Tolerance rate = 0.8 % is used to find the accuracy of the network.

4.3.1 Find Accuracy

Code to find accuracy:

```
Count=0;
  for row=1 :numel(realData)
    ac = (realData (row)-testingData(row))/( realData(row))*100;
      if(ac>0.8||ac<-0.8)
               Count = Count + 1;
      end
  end

  accuracy = 100 - (Count/numel(realData)*100);
```


Stock Market Prediction

Price Prediction | K Fold Validation

Select Period: 3 Months

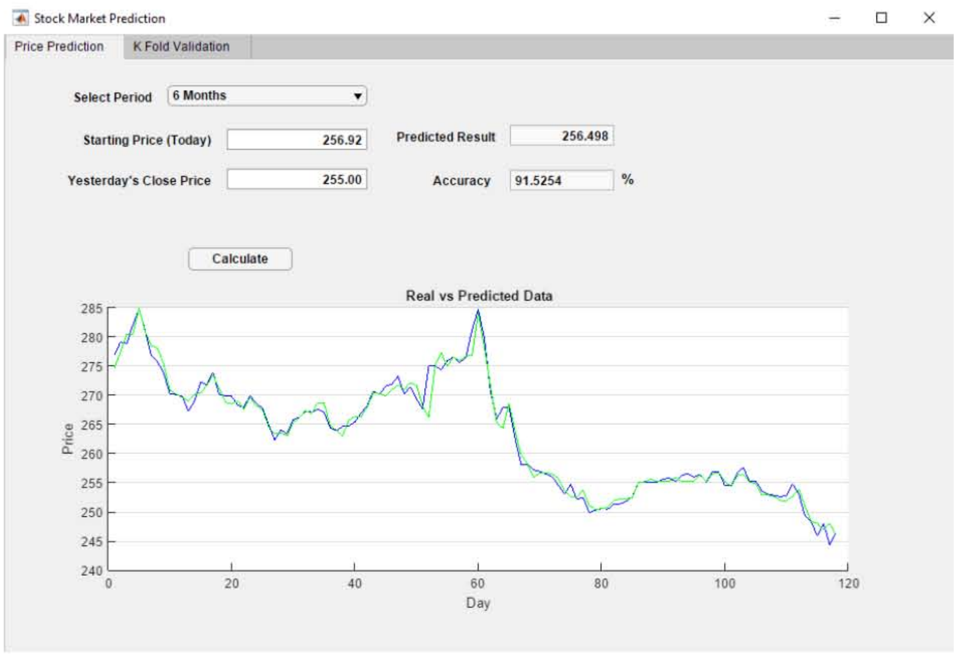
Starting Price (Today): 256.92

Yesterday's Close Price: 255.00

Predicted Result: 250.151

Accuracy: 86.2069 %

Figure 4.2: Testing three months data


Stock Market Prediction

Price Prediction | K Fold Validation

Select Period: 6 Months

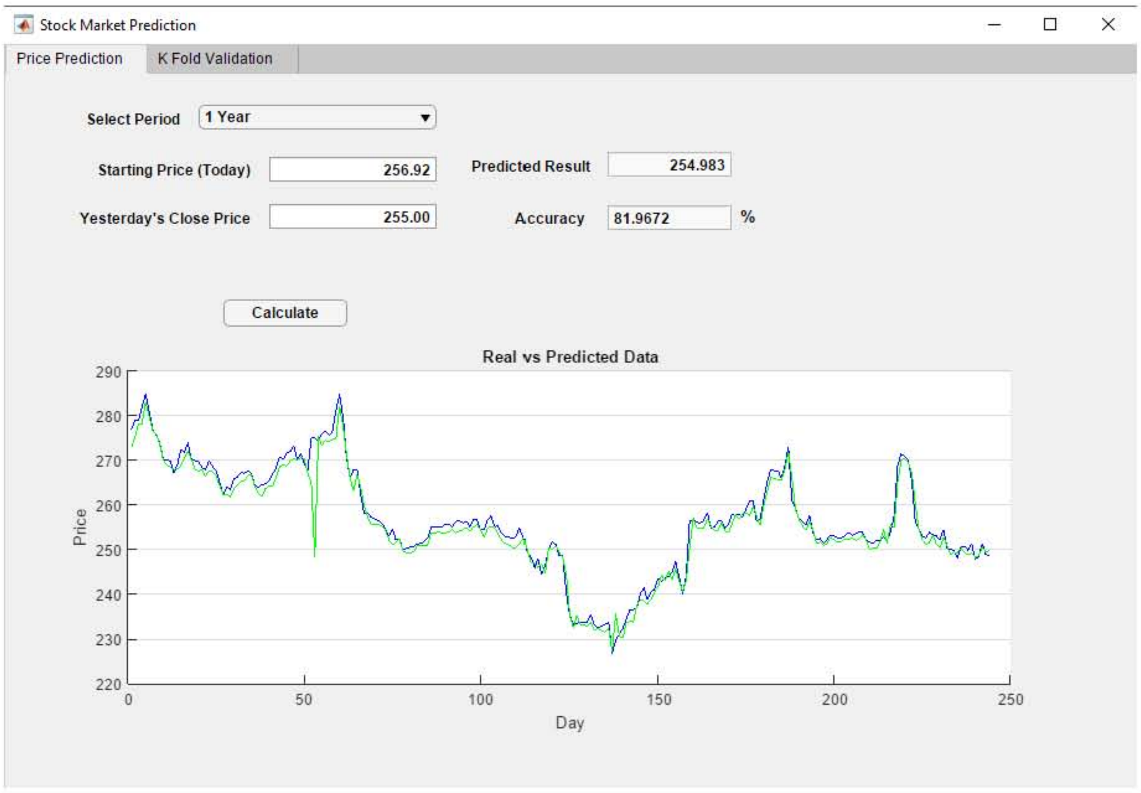
Starting Price (Today): 256.92

Yesterday's Close Price: 255.00

Predicted Result: 256.498

Accuracy: 91.5254 %

Figure 4.3: Testing six months data


Stock Market Prediction

Price Prediction | K Fold Validation

Select Period: 1 Year

Starting Price (Today): 256.92

Yesterday's Close Price: 255.00

Predicted Result: 254.983

Accuracy: 81.9672 %

Calculate

Figure 4.4: Testing one year's data

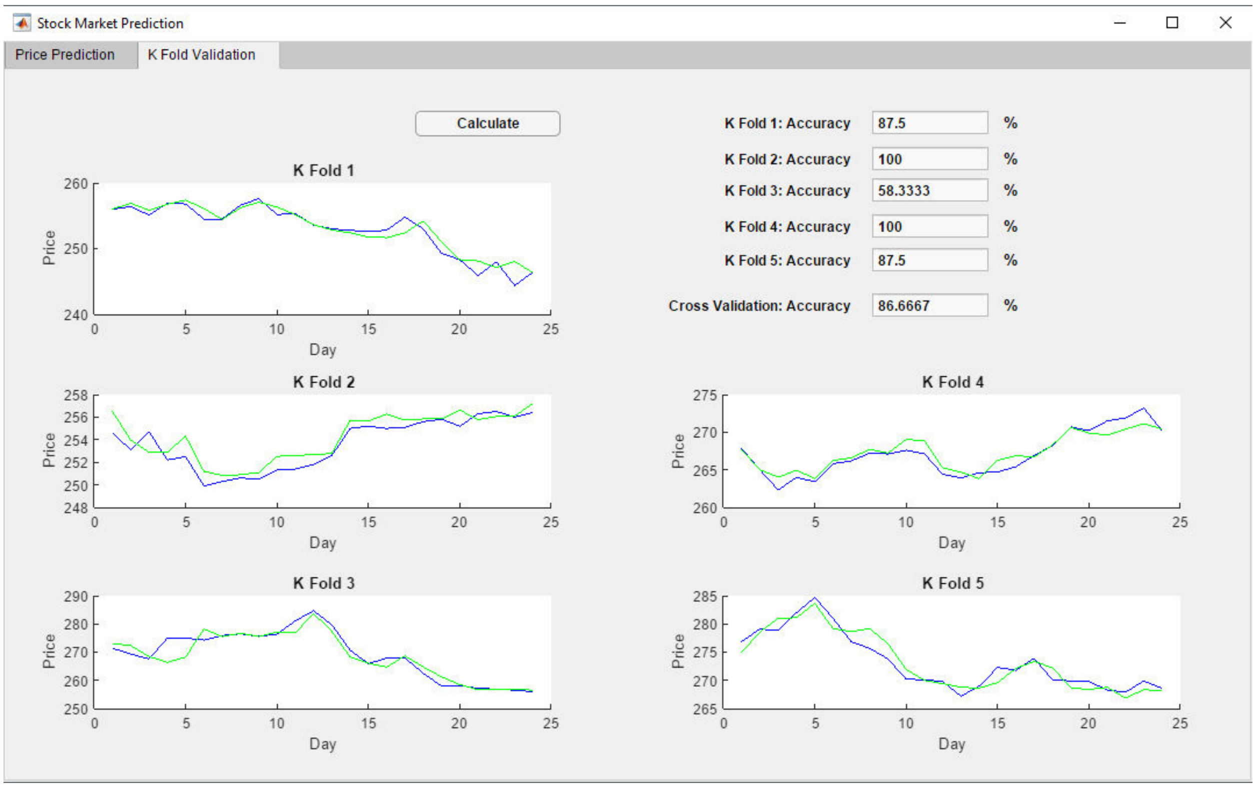
Here, these three figures Figure 4.2, Figure 4.3 and Figure 4.4 show the predicted value and the accuracy of the network. Also it shows the graph of real data vs predicted values.

Each of the network has different accuracy and different predicted values. From Figure 4.2 the accuracy of 3 months trained network **(86.2069%)** is better than 1 year trained network **(81.9672%)** as shown in Figure 4.4. And from Figure 4.3 six months trained network has the highest accuracy rate **(91.5254%)**.

4.3.2 K-Fold Cross Validation

For K-Fold cross validation [11] the data has been divided into five parts or folds. From those five folds every time one fold have been chosen as testing purpose and other four folds are used for training purpose.

After training five different network from the data five accuracies have been found for each network. The average of those five accuracies is called as Cross Validation or K Fold Cross Validation [12].



K Fold 1: Accuracy 87.5 %
K Fold 2: Accuracy 100 %
K Fold 3: Accuracy 58.3333 %
K Fold 4: Accuracy 100 %
K Fold 5: Accuracy 87.5 %
Cross Validation: Accuracy 86.6667 %

Figure 4.5: K Fold Cross Validation

Here, from Figure 4.5 for each network among 118 days of data 24 days data has been used for testing and other 94 days of data used for training.

For each network their accuracy has been found and the average accuracy is **86.6667%** showed in Figure 4.5 which is also called as cross validation accuracy.

4.4 Result and Analysis

After training and testing different categories of data with Artificial Neural Networking (ANN) the predicted value of price is come out with different accuracy.

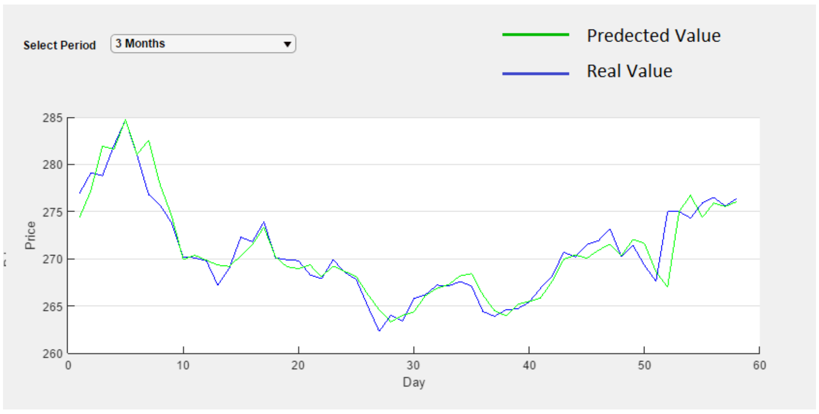


Figure 4.6: Three months data

If we analyze the three months data than from Figure 4.6 we can see this graph plot where green line is the predicted value and blue line is for the real value. If we see deeply between the two lines then we can see, when the real value suddenly increases there occurred underflow with the predicted value and when the real value suddenly decreases there occurred overflow with the predicted value.

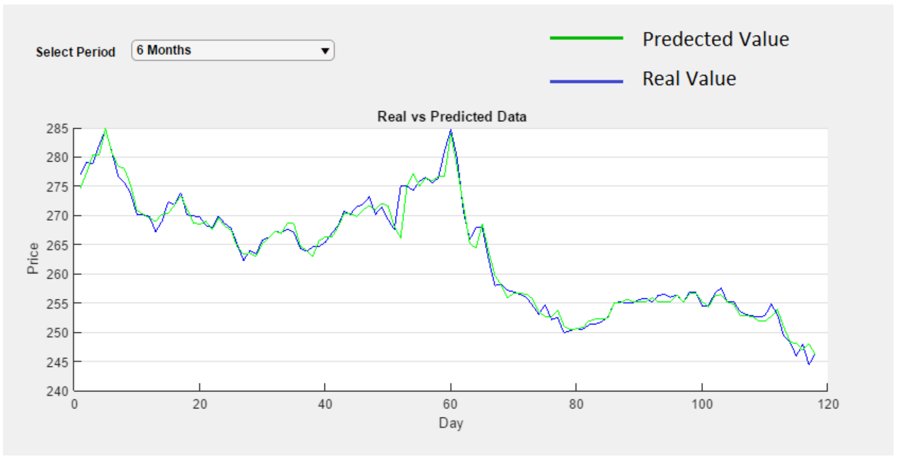


Figure 4.7: Six months data

Similarly for six months (Figure 4.7) and one year's (Figure 4.8) data the same thing occurred. This is happened because during some big changes it too difficult to predict that what is happening. May be the real and the predicted lines have some difference but if we analyze the accuracy rate then we can see this is not so bed prediction at all.

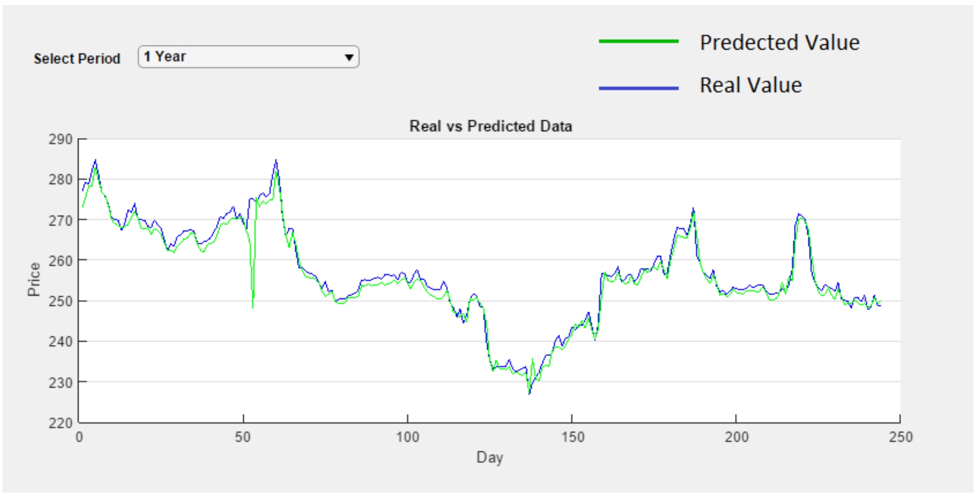
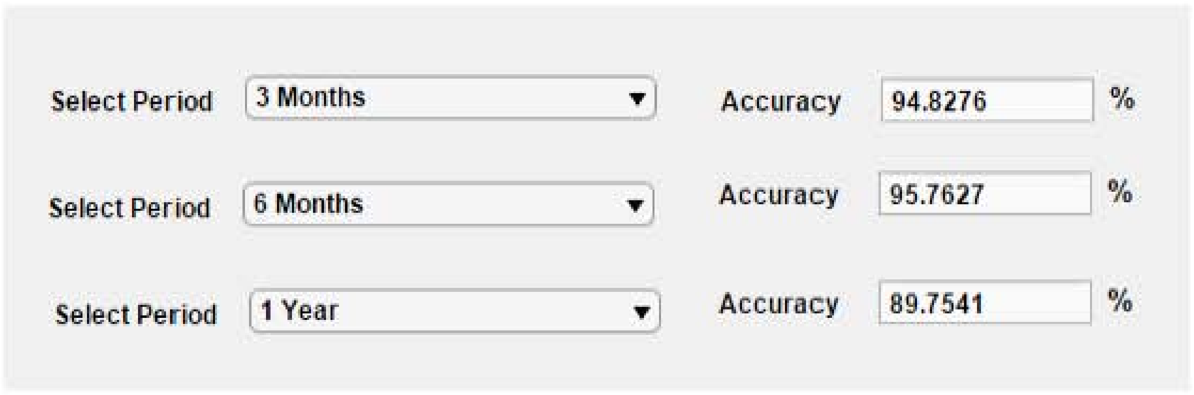


Figure 4.8: One Year's data

Now, if we analyze the accuracy then from Figure 4.9 we can see three categories of data have different accuracy rate. If we set the Tolerance rate as 1% then we can find 3 months accuracy rate is 94.8276%, for six months the accuracy rate is 95.7627% and for 1 year the accuracy rate is 89.7541%.

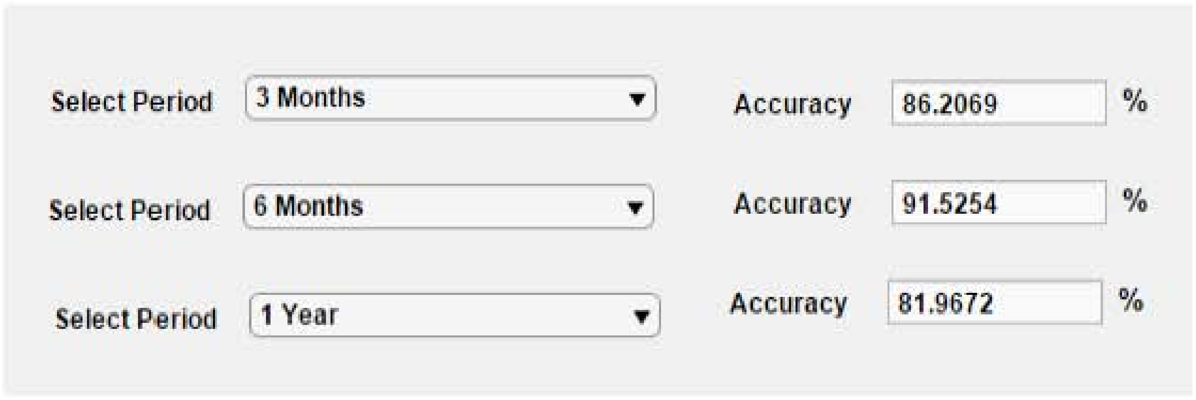


Select Period	3 Months ▾	Accuracy	94.8276	%
Select Period	6 Months ▾	Accuracy	95.7627	%
Select Period	1 Year ▾	Accuracy	89.7541	%

Figure 4.9: Accuracy rate for 1% Tolerance

So, from Figure 4.9 we can see for 1% tolerance rate 6 months training data has the highest accuracy 95.7627%, which is good enough and one year data has the lowest accuracy 89.7541% which is also a good accuracy.

Now, if we set the Tolerance rate a little lower as 0.8% then we can see the accuracy rate is also decreases. From figure 4.10 we can see for three months data the accuracy rate is 86.2069%, for six months the rate is 91.5254% and for one year the accuracy rate is 81.9672%.

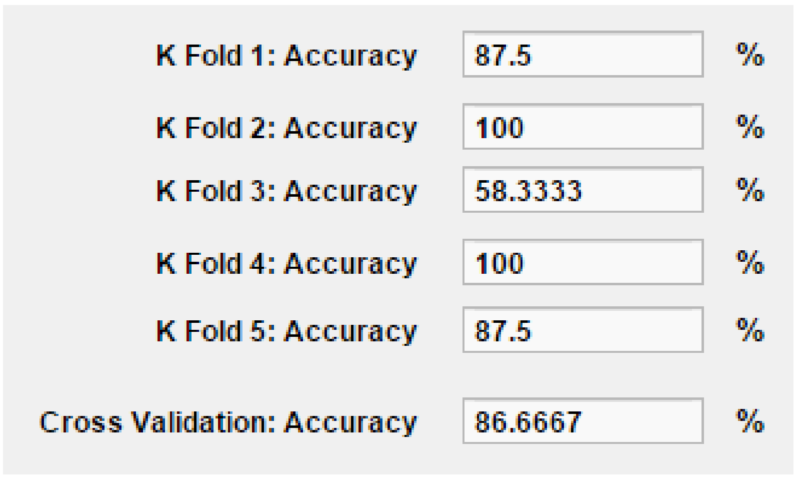


Select Period	3 Months ▾	Accuracy	86.2069	%
Select Period	6 Months ▾	Accuracy	91.5254	%
Select Period	1 Year ▾	Accuracy	81.9672	%

Figure 4.10: Accuracy rate for 0.8% Tolerance

Again six months data has the highest accuracy rate and one year has the lowest.

Now if we run K-Fold Cross Validation over using six months data with Tolerance rate as 0.8% then from Figure 4.11 we can see the cross validation rate is 86.6667% which is not bed at all.



K Fold 1: Accuracy	87.5	%
K Fold 2: Accuracy	100	%
K Fold 3: Accuracy	58.3333	%
K Fold 4: Accuracy	100	%
K Fold 5: Accuracy	87.5	%
Cross Validation: Accuracy	86.6667	%

Figure 4.11: K-Fold Cross Validation for 0.8% Tolerance

So, six months training network is the best network for predicting the stock market price.

Input Value		**Target Value**	**Predicted Value**
Today's Starting Price	**Yesterday Closing Price**	**Real Closing Price**	**Predicted Closing Price**
275	273.2	276.9	274.622
277	276.9	279.1	277.363
279.9	279.1	278.8	280.341
280	278.8	282	280.375
285	282	284.7	284.851
281.2	284.7	281	280.788
277.5	281	276.8	278.476
277.8	276.8	275.7	278.036
275	275.7	273.8	275.28
270.7	273.8	270.2	270.841

Table 4.1: Optimal results for six months trained data using 10 hidden layer nodes

In Table 4.1 Input Values are used as sample input value and it generates some results which is shown as Predicted Values. Target values are the real target data. As an input there is two values. Yesterday Closing Price is the closing price of the company for previous day. Today's Starting price is the stock price started on the current day. Real Closing Price is the real value of closing price on that day and Predicted Closing Price is the predicted value or price generated by trained network on that day. Here 10 hidden layer nodes have been used to traine the network and the accuracy rate is 91.5254% using tolerance rate as 0.8%.

Similarly Figure 4.2 shows the optimal results of 6 months predicted value using 15 hidden layer nodes where the accuracy rate is 90.678% using tolerance rate as 0.8%.

Input Value		Target Value	Predicted Value
Today's Starting Price	**Yesterday Closing Price**	**Real Closing Price**	**Predicted Closing Price**
275	273.2	276.9	274.148
277	276.9	279.1	277.768
279.9	279.1	278.8	280.462
280	278.8	282	280.716
285	282	284.7	284.173
281.2	284.7	281	281.874
277.5	281	276.8	276.979
277.8	276.8	275.7	278.508
275	275.7	273.8	275.164
270.7	273.8	270.2	270.407

Table 4.2: Optimal results for six months trained data using 15 hidden layer nodes

Table 4.3 shows the optimal results of 6 months predicted value using 20 hidden layer nodes where the accuracy rate is 92.3729% using tolerance rate as 0.8%. So, if we change the number of hidden layer nodes the accuracy rate is also change.

Input Value		Target Value	Predicted Value
Today's Starting Price	**Yesterday Closing Price**	**Real Closing Price**	**Predicted Closing Price**
275	273.2	276.9	275.039
277	276.9	279.1	277.082
279.9	279.1	278.8	280.882
280	278.8	282	280.969
285	282	284.7	284.884
281.2	284.7	281	281.625
277.5	281	276.8	277.641
277.8	276.8	275.7	277.768
275	275.7	273.8	274.368
270.7	273.8	270.2	269.972

Table 4.3: Optimal results for six months trained data using 20 hidden layer nodes

Table 4.4 and Table 4.5 is constructed with the accuracy rate in different condition. In Table 4.4 the general accuracy rate of different categories of data with different tolerance rate is given. In Table 4.5 the accuracy rate of six months trained data using K-Fold Cross Validation is shown with different tolerance rate.

Tolerance Rate	**Data Period**	**Accuracy**
1 %	Three months	94.8276 %
1 %	Six months	95.7627 %
1 %	One year	89.7541 %
0.8 %	Three months	86.2069 %
0.8 %	Six months	91.5254 %
0.8 %	One year	81.9672 %
0.5 %	Three months	72.4138 %
0.5 %	Six months	77.1186 %
0.5 %	One year	52.8689 %

Table 4.4: Accuracy rates for predicted data

K-Fold Validation	**0.5 % Tolerance**	**0.8 % Tolerance**	**1.0 % Tolerance**
Fold 1	79.1667 %	87.5 %	95.8333 %
Fold 2	79.1667 %	100 %	100 %
Fold 3	54.1667 %	58.3333 %	70.8333 %
Fold 4	66.6667 %	100 %	100 %
Fold 5	50 %	87.5 %	95.8333 %
Cross Validation	65.8333 %	86.6667 %	92.5 %

Table 4.5: Accuracy rates for K-Fold Cross Validation using six months data

From table 4.4 the highest accuracy rate is 95.7627% and the lowest is 52.8689% and maximum rates are over 80%. From Table 4.5 the highest cross validation rate is 92.5% with 1% tolerance and the lowest is 65.8333% with 0.5% tolerance rate. All the reports showing the result is higher than average accuracy rate and so the network is well trained and providing a good solution. Among all of the trained network six months trained network is the best network for predicting the stock market price.

Chapter 5

Conclusion & Future Work

5.1 Conclusion

It is clear that stock market price can be predicted and from the predicted values and accuracy rate it is also clear that it can be a very useful tool for the people who are related to the stock market. By using Artificial Neural Network any frequent data from stock market [8] can be trained, tested and then the trained network can be used for predicting the stock market price. Though it is not possible to predict the actual price from future but the predicted price which is almost close to the real value can be very useful for a trader who want to make a good profit from stock market. During the prediction sometimes there may occurred some big difference between the real value and predicted value. Because sometimes the real value suddenly make a big change. If the change is go to very high than the predicted value stay a little lower from the real value and if the real value become very low then the predicted value stay on a little higher position. But in regular days the predicted value is almost close with the real value. So, using Backpropagation [15] algorithm in Artificial Neural Networking it is possible to make good artificial intelligence agent which can help to predict the future price of stock market by analyzing the historical data.

5.2 Future Work

This project is mainly focusing on short-time perspective analysis. In future, it can be extended for predicting the long time perspective analysis. Further testing could be included to the project by increasing or decreasing the number of nodes in hidden layers. Also this project can be extended by adding some new features -

i. Predict the daily maximum and minimum possible price from stock market.
ii. Predict upcoming few days possible price.
iii. Suggest a possible date to the user for buying or selling to earn a better profit.
iv. Suggest a strategy of investment by analyzing the market status.

References

[1] A Step by Step Backpropagation Example, by Matt Mazur. https://mattmazur.com/2015/03/17/a-step-by-step-backpropagation-example/ [Accessed on August 15, 2016].

[2] Dhaka Stock Exchange Data Archives Available at: http://www.dsebd.org/data_archive.php [Accessed on August 20, 2016].

[3] Hal Daumé III, A Course in Machine Learning, 2012, Chapter 8, Page 113-124.

[4] trainlm - Levenberg-Marquardt backpropagation Available at: http://www-rohan.sdsu.edu/doc/matlab/toolbox/nnet/trainlm.html , [Accessed on November 07, 2016].

[5] How neural networks work - A simple introduction Available at: http://www.explainthatstuff.com/introduction-to-neural-networks.html , [Accessed on November 07, 2016].

[6] UI Axes (App Designer) Properties Available at: https://www.mathworks.com/help/matlab/ref/uiaxesappdesigner-properties.html#property_Color [Accessed on November 12, 2016].

[7] sim - Simulate neural network, MathWorks Available at: https://www.mathworks.com/help/nnet/ref/sim.html , [Accessed on November 14, 2016].

[8] Stock Prediction using Artificial Neural Networks, https://people.eecs.berkeley.edu/~akar/IITK_website/EE671/report_stock.pdf , [Accessed on November 14, 2016].

[9] Multilayer Neural Network Architecture, MathWorks Available at: https://www.mathworks.com/help/nnet/ug/multilayer-neural-network-architecture.html , [Accessed on November 14, 2016].

[10] Mayankkumar B Patel , Sunil R Yalamalle : Stock Price Prediction Using Artificial Neural Network. In: International Journal of Innovative Research in Science, Engineering and Technology (ISSN: 2319-8753) Vol. 3, Issue 6, June 2014.

[11] Machine Learning :: Model Selection & Cross Validation Available at: https://www.youtube.com/watch?v=hihuMBCuSlU , [Accessed on November 20, 2016].

[12] Lecture 77, K-fold cross validation, Available at: https://www.coursera.org/learn/ml-regression/lecture/FJcUw/k-fold-cross-validation , [Accessed on November 20, 2016].

[13] Making GUI Apps in Matlab using App Designer , Available at: https://www.youtube.com/watch?v=3YLYMtBmqEI , [Accessed on November 24, 2016].

[14] Zabir Haider Khan, Tasnim Sharmin Alin, Md. Akter Hussain: Price Prediction of Share Market using Artificial Neural Network (ANN). In: International Journal of Computer Applications (0975 – 8887) Volume 22– No.2, May 2011.

[15] Chapter 2, How the backpropagation algorithm works , Available at: http://neuralnetworksanddeeplearning.com/chap2.html , [Accessed on November 27, 2016].

Appendix

Implementation in MATLAB GUI

```
  classdef DseGpGUI < matlab.apps.AppBase

     methods (Access = private)
```

Function for Prediction

```
   function results = predict(app,x,y,v)

              if v==1

              S = load ('Gp1yTrained.mat','-mat');

           net1y = S.net1y;

           results = net1y([x,y]');

               compare(app,1);

          elseif v==3

               S = load ('Gp3mTrained.mat','-mat');

            net3m = S.net3m;

           results = net3m([x,y]');

              compare(app,3);

          elseif v==6

               S = load ('Gp6mTrained.mat','-mat');

           net6m = S.net6m;

           results = net6m([x,y]');

            compare(app,6);

          end

         end
```

Function to Compare and Find Accuracy

```
 function results = compare(app,v)

       if v==1

           S = load ('Gp1yTrained.mat','-mat');

          targetData = S.target1y;

          inputData = S.input1y;

          net1y = S.net1y;

          results = net1y(inputData');

         elseif v==3

          S = load ('Gp3mTrained.mat','-mat');

          targetData = S.target3m;

          inputData = S.input3m;

          net3m = S.net3m;

          results = net3m(inputData');

          elseif v==6

           S = load ('Gp6mTrained.mat','-mat');

          targetData = S.target6m;

          inputData = S.input6m;

          net6m = S.net6m;

          results = net6m(inputData');

       end

            hold(app.dataAxes,'off');

            plot(app.dataAxes,targetData,'-b');

             hold(app.dataAxes,'on');

             plot(app.dataAxes,results,'-g');
```

```
     c5=0;
   for row=1 :numel(targetData)
    ac = (targetData(row)-results(row))/(targetData(row))*100;
            if(ac>0.8||ac<-0.8)
            c5=c5+1;
            end
    end
 accuracy6m = 100- (c5/numel(targetData)*100);
 app.acField.Value = num2str(accuracy6m);
  end
```

Function for K-Fold Validation

```
        function results = kFold(app)
        S = load ('kFold6mTrained.mat','-mat');
        rD1 = S.r6m1;
        sam1 = S.sam6m1;
        net1 = S.net1;
        t1 = net1(sam1');
            hold(app.k1,'off');
          plot(app.k1,rD1,'-b');
           hold(app.k1,'on');
          plot(app.k1,t1,'-g');
    c1=0; c2=0; c3=0; c4=0; c5=0;
 for row=1 :numel(rD1)
    ac = (rD1(row)-t1(row))/(rD1(row))*100;
     if(ac>0.8||ac<-0.8)
    c1=c1+1;
     end
end
accuracy1 = 100-( c1/numel(rD1)*100);
app.k1Field.Value = num2str(accuracy1);
          rD2 = S.r6m2;
          sam2 = S.sam6m2;
          net2 = S.net2;
          t2 = net2(sam2');
            hold(app.k2,'off');
          plot(app.k2,rD2,'-b');
           hold(app.k2,'on');
          plot(app.k2,t2,'-g');
for row=1 :numel(rD2)
   ac = (rD2(row)-t2(row))/(rD2(row))*100;
    if(ac>0.8||ac<-0.8)
            c2=c2+1;
    end
end
accuracy2 = 100- (c2/numel(rD2)*100);
app.k2Field.Value = num2str(accuracy2);
         rD3 = S.r6m3;
          sam3 = S.sam6m3;
          net3 = S.net3;
          t3 = net3(sam3');
```

```
            hold(app.k3,'off');
          plot(app.k3,rD3,'-b');
           hold(app.k3,'on');
          plot(app.k3,t3,'-g');
  for row=1 :numel(rD3)
   ac = (rD3(row)-t3(row))/(rD3(row))*100;
    if(ac>0.8||ac<-0.8)
            c3=c3+1;
    end
end
accuracy3 = 100- (c3/numel(rD3)*100);
app.k3Field.Value = num2str(accuracy3);
          rD4 = S.r6m4;
          sam4 = S.sam6m4;
          net4 = S.net4;
          t4 = net4(sam4');
            hold(app.k4,'off');
          plot(app.k4,rD4,'-b');
           hold(app.k4,'on');
          plot(app.k4,t4,'-g');
for row=1 :numel(rD4)
   ac = (rD4(row)-t4(row))/(rD4(row))*100;
   % out1(row)=ac;
    if(ac>0.8||ac<-0.8)
          c4=c4+1;
```

```
        end

    end

    accuracy4 = 100- (c4/numel(rD4)*100);

    app.k4Field.Value = num2str(accuracy4);

                rD5 = S.r6m5;

              sam5 = S.sam6m5;

              net5 = S.net5;

              t5 = net5(sam5');

                hold(app.k5,'off');

              plot(app.k5,rD5,'-b');

               hold(app.k5,'on');

              plot(app.k5,t5,'-g');

    for row=1 :numel(rD5)

       ac = (rD5(row)-t5(row))/(rD5(row))*100;

        if(ac>0.8||ac<-0.8)

                c5=c5+1;

        end

    end

    accuracy5 = 100- (c5/numel(rD5)*100);

    app.k5Field.Value = num2str(accuracy5);

            end

        end
```

Functions for Buttons

```
    methods (Access = private)

            % Code that executes after component creation
    function startupFcn(app)

    end

    % calculate button pushed function

    function calculateButtonPushed(app)

        option = app.selectTime.Value;

        x = app.num1.Value;

        y = app.num2.Value;

      if strcmp(option ,'3 Months')

          z = predict(app,x,y,3);

        app.result.Value = z;

      elseif strcmp(option ,'6 Months')

        z = predict(app,x,y,6);

        app.result.Value = z;

      elseif strcmp(option ,'1 Year')

      z = predict(app,x,y,1);

        app.result.Value = z;

       else

        fig = app.DSEPrediction;

        uialert(fig,option,'Warning',...

        'Icon','warning');

      end

    end

end
```