Stock Market Prediction Using Machine Learning

Sankar Gireesan Nair, Veera Venkata Sasanka Uppu
College of Computer and Information Science
Northeastern University
gireesannair.s@husky.neu.edu, uppu.v@husky.neu.edu

Abstract

Stock market is one of the most complicated systems in the world, and it has connection with almost every part in our life. Predicting its trend by interpreting the seemly chaotic market data has always been attractive topic to both investors and researchers. In this project, we use different machine learning techniques like Neural Network, Logistical Regression, Decision Tree and Support Vector Machine to predict whether the stock price of next day would increase or decrease before trade ends by analyzing historic stock data. The performance of our implementation was analyzed under different algorithms by comparing the average accuracy among the test set.

Introduction

Stock market analysis has always been a hot area for researchers and investors. But predicting its trend accurately is a challenging task because of the high data intensity, noise, hidden structures and the correlation with the whole world. A clear understanding of the timing of lead-lag relations among many factors, understanding the statistical significance of these lead-lag relations and learning which variables are more important ones to watch as signals for predicting the market moves is required to forecast any trends. People have come up with a lot of theoretical foundation in mathematics, and developed a variety of methods to analyze the stock market with the help of modern computer technology. Among those popular methods that have been employed, Machine Learning techniques are very popular due to the capacity of identifying stock trend from massive amount of data that capture the underlying stock price dynamics.

Objective of this study is to compare performance of different machine learning techniques in predicting the trend of next-day stock price (whether the stock would increase or decrease) for selected NASDAQ stocks.

Our model uses robust machine learning techniques like Neural Network, Logistical Regression, Decision Tree and Support Vector Machine algorithms to predict the next-day price. In order to train the model, we had collected relevant data of stable companies like Google, Apple etc. for last six years. We used 75% of the obtained data to train our model and 25% for testing. Obtained accuracies were compared among each algorithm under different factors.

In the upcoming sections of this paper the methodology and machine learning algorithms used to solve this problem are described, how features are extracted from the data set and the results which are analyzed to obtain some conclusions and discuss any further work that could be done to improve the accuracy of the predictions.

Dataset

As stated, the data set used consists of historical stock data of different companies (Google, Apple, Microsoft and Amazon) for past six years (12/09/2010 to 12/09/2016). We also collected historical data of Nikkei 225, a stock market index for Tokyo Stock Exchange (Japan) for the same time period. Specifically, each entry in the historical stock data includes the following items:

- Date (YY-MM-DD)
- Open (float)
- High (float)
- Low(float)
- Close(float)
- Volume(float)
- Adjclose(float)
- J-index (Integer)

Where Date represents the trading date, Open/Close is the opening/closing price of that stock in that trading date, High/Low is the highest/lowest price the stock reached on that trading date. Volume is the amount of stocks moved in that day (buy + sell). Adjclose is the closing price after adjustment at end of the trading. The historical data set was collected using Yahoo Finance [1], NASDAQ [2] and we also used Yahoo Finance API to collect up to date values.

J-index was added manually which represents the trend of Tokyo Stock Exchange(TSE) Index, Nikkei 225. It has values: +1 (indicating an increase in Nikkei 225), 0 (no change) and -1 (indicating a decrease in Nikkei 225). The values above were calculated by matching the trading dates of TSE with trading dates of NASDAQ and assigning zeros for dates when the market was closed for Japan, but open for USA.

Methods Used

Multilayer Perceptron

An artificial neuron network (ANN) is a computational model based on the structure and functions of biological neural networks. Information that flows through the network affects the structure of the ANN because a neural network changes - or learns, in a sense - based on that input and output. Each individual neural unit have a summation function which combines the values of all its inputs together. [7]

A multilayer perceptron (MLP) is a feedforward artificial neural network model that maps sets of input data onto a set of appropriate outputs. An MLP consists of multiple layers of nodes in a directed graph, with each layer fully connected to the next one. Except for the input nodes, each node is a neuron (or processing element) with a nonlinear activation function. After a lowermost input layer there are usually any number of intermediate, or hidden, layers followed by an output layer at the top. There exist no interconnections within a layer while all neurons in a layer are fully connected to neurons in adjacent layers. Weights measure the degree of correlation between the activity levels of neurons that they connect. [6]

An external input vector is supplied to the network by clamping it at the nodes in the input layer. For conventional classification problems, during training, the appropriate output node is clamped to state 1 while the others are clamped to state 0. This is the desired output supplied by the teacher.

Consider the network given in Fig. 1. The total input, x_j^{h+1}, received by neuron j in layer h + 1 is defined as:

$$x_j^{h+1} = \sum_i y_i^h w_{ji}^h - \theta_j^{h+1} \qquad (1)$$

where y_i^h is the state of the ith neuron in the preceding hth layer, w_{ji}^h; is the weight of the connection from the ith neuron in layer h to the jth neuron in layer h + 1, and θ_j^{h+1} is the threshold of the jth neuron in layer h + l. Threshold θ_j^{h+1} may be eliminated by giving the unit j in layer h + 1 an extra input line with a fixed activity level of 1 and a weight of $-\theta_j^{h+1}$.

The output of a neuron in any layer other than the input layer (h > 0) is a monotonic nonlinear function of its total input and is given as

$$y_j^h = \frac{1}{1 + e^{-x_j^h}}. \qquad (2)$$

For nodes in the input layer,

$$y_j^0 = x_j^0, \qquad (3)$$

where x_j^0 is the jth component of the input vector clamped at the input layer. All neurons within a layer, other than the input layer, have their states set by (1) and (2) in parallel while different layers have their states set sequentially in a bottom up manner until the states of the neurons in the output layer H are determined. The learning procedure has to determine the internal parameters of the hidden units based on its knowledge of the inputs and desired outputs. Hence learning consists of searching a very large parameter space and therefore is usually little slow.

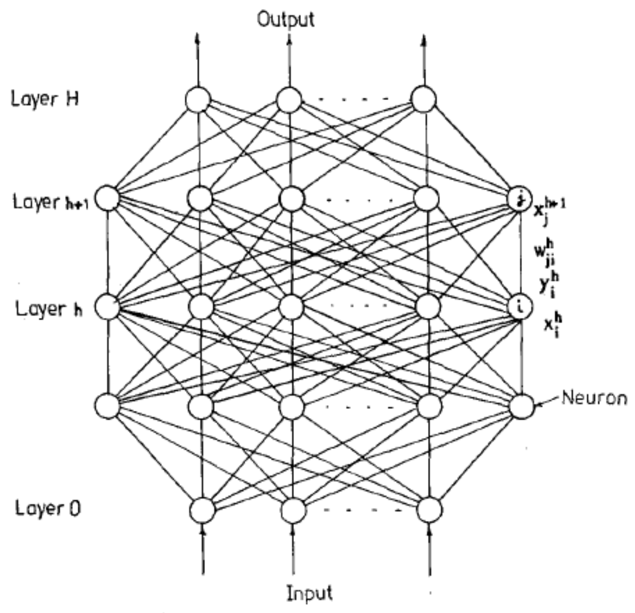


Fig. 1. A neural network with three hidden layers. Neurons connected to each other via variable connection weights.

This type of network is trained with the backpropagation learning algorithm. The backward propagation of errors or backpropagation, is a common method of training artificial neural networks which repeats a two phase cycle, propagation and weight update. When an input vector is presented to the network, it is propagated forward through the network, layer by layer, until it reaches the output layer. The output of the network is then compared to the desired output, using a loss function, and an error value is calculated for each of the neurons in the output layer. The error values are then propagated backwards, starting from the output, until each neuron has an associated error value which roughly represents its contribution to the original output.

MLPs are widely used for pattern classification, recognition, prediction and approximation. Multi-Layer Perceptron can solve problems which are not linearly separable which can be very useful for classifying whether the stock trend of a company would go up or down on a certain trading day.

Logistic Regression

Logistic Regression is used to determine the magnitude of relationships between variables as well as to model relationships between variables and for predictions based on the models. Simple linear regression or multiple linear is applicable when this relationship is assumed to be linear [16]. However, many non-linear techniques could be used to obtain a more accurate regression if the relationship between variables is not linear in parameters.

Logistic regression is preferred in case the response variable can take only binary values (yes or no). In stock price prediction, the trend of next day's price can be categorized into two classes, which is whether the price of the stock is increasing or decreasing hence logistic regression can be used. [5]

The significance of logistic regression can be evaluated by the log likelihood test, given as the model chi-square test, evaluated at the $p < 0.05$ level, or the Wald statistic.

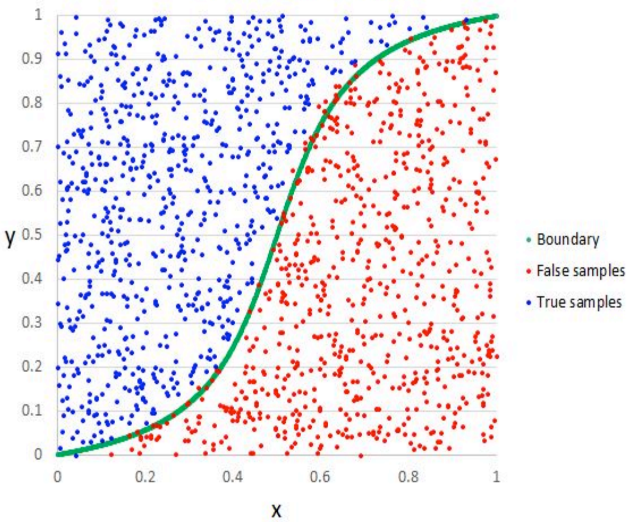


Fig. 2 An example of Logical Regression

Logistic Regression Model is represented by:

$$p = \frac{\exp(c_0 + c_1 x_1 + c_2 x_2 + \cdots + c_k x_k)}{1 + \exp(c_0 + c_1 x_1 + c_2 x_2 + \cdots + c_k x_k)} \quad (4)$$

To predict stock price trend of average price of the next-day, stock price is expected to have downtrend if the Logistic Regression value p_i (4)is close to 0 (or is equal to 0). Otherwise, such stock price is considered to have uptrend if the Logistic Regression value pi is close to 1(or is equal to 1). And as we could from Fig 2. how logistic regression could classify the data, so taken a selected set of features which from historical stock data allows data points to be tightly packed for one class from another which allows separability.

Support Vector Machine with RBF Kernel

A Support Vector Machine (SVM) is a classifier formally defined by a separating hyperplane. In other words, given labeled training data (supervised learning), the algorithm outputs an optimal hyperplane which is used to categorize the testing data set. Any hyperplane can be written such that as the set of points x satisfying:

$$\vec{w} \cdot \vec{x} - b = 0,$$

SVM is a non-probabilistic binary linear classifier, but in addition to performing linear classification, SVMs can efficiently perform a non-linear classification using what is called the kernel trick, implicitly mapping their inputs into high-dimensional feature spaces. This type of implementation is called Radial Basis Function (RBF) SVM. The RBF kernel on two samples x and x', represented as feature vectors in some input space, is defined as:

$$K(\mathbf{x}, \mathbf{x'}) = \exp\left(-\frac{||\mathbf{x} - \mathbf{x'}||^2}{2\sigma^2}\right)$$

Based on the value of σ the feature space of the kernel has infinite number of dimensions.

As we can see in the fig 3 the performance of SVM with the radial basis function versus the linear SVM. Since the stock prediction involves vast amount of non-linear data, SVM with RBF Kernel should be a good choice in classifying.

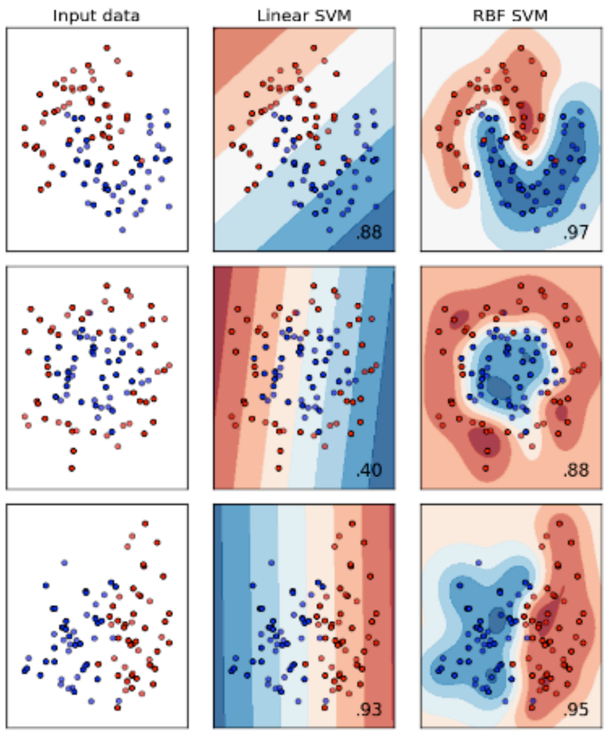


Fig. 3 Simple SVM and SVM using RBF Kernel to classify data

Decision Trees

Decision trees are tree-like graph in which each node represents a rule and each outgoing edge represents some possible value of that rule. The algorithm repeatedly splits the data set based on a criterion that maximizes the separation of the data, resulting in a tree-like structure. Leaf nodes represent on of multiple possible candidate classes. Decision trees are mainly used for classification. The cost of using decision trees in predictions is log of the number of data points used to train the tree. [8]

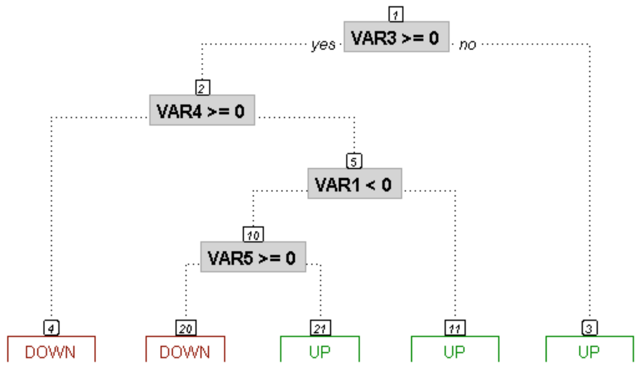
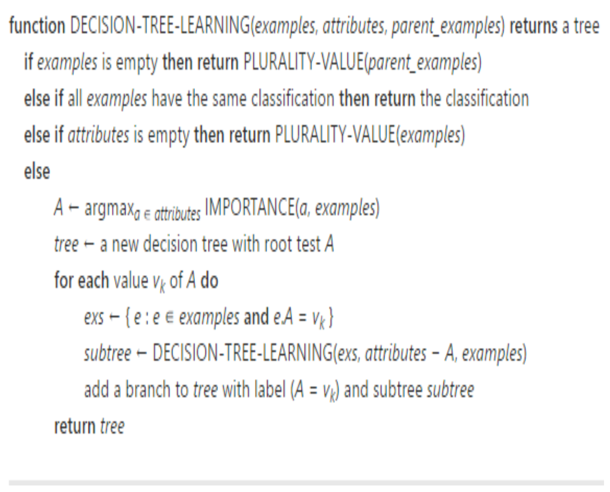


Fig. 4 Example of Decision Tree Learning for predicting stock market

In the above figure (Fig4) we can see how ideally a decision tree would look like while predicting the stock market trend. It would establish few rules on each node based on different parameters to classify the data and assign a label in the leaf node. A parameter called min sample split is used in the predictor which tells upon how many minimum samples should the decision tree split a node. A major disadvantage of decision trees is they can create overcomplex trees that do not generalize the data well which may result in overfitting or underfitting the data. Another drawback lies in the fact that continuous variables are implicitly split every time losing necessary information in the process. Below in Fig5. is the general algorithm for decision tree.



```
function DECISION-TREE-LEARNING(examples, attributes, parent_examples) returns a tree
  if examples is empty then return PLURALITY-VALUE(parent_examples)
  else if all examples have the same classification then return the classification
  else if attributes is empty then return PLURALITY-VALUE(examples)
  else
      $A \leftarrow \mathrm{argmax}_{a \in attributes}$ IMPORTANCE(a, examples)
      tree ← a new decision tree with root test A
      for each value $v_k$ of A do
          $exs \leftarrow \{ e : e \in examples \text{ and } e.A = v_k \}$
          subtree ← DECISION-TREE-LEARNING(exs, attributes − A, examples)
          add a branch to tree with label $(A = v_k)$ and subtree subtree
      return tree
```

Fig 5. Pseudocode for Decision Tree Learning

Feature Selection and Processing

The first group of feature sets was comprised of finding difference between each attribute of today's stock to yesterdays, we gradually added more features until we kept getting better accuracy. These were the following features we tried –

- f1: Open(d_i)-Open(d_{i-1})
- f2: Close(d_i)-Close(d_{i-1})
- f3: Opening/(diff of high,low)
- f4: Closing/(diff of high,low)
- f5: Moving average data
- f6: Average directional movement index
- f7: Money Flow Index
- f8: Relative Strength Index
- f9: j-index(d_{i-1})
- f10: High(d_i) - High(d_{i-1})
- f11: Low(d_i) - Low(d_{i-1})

Our Feature Vector was tried for multiple combinations and net average accuracy was calculated for different combinations out of which the Feature Vector F-

F=[f1,f3,f4,f5,f6,f7,f8,f9]

turned out giving a better overall net accuracy over the tested set of stocks (GOOGL, AAPL, AMZN, MSFT).
The set of features in detail are-

- f1, f2 are the difference in opening/closing price vectors of today's stock value from yesterday's.
- f3, f4 are the ratios of opening/closing price of stock by the difference in stock's high low over the entire time frame.
- f5 is the moving average data of the stock which is given by

  $$\frac{\sum_{i=0}^{j} data_i}{j \times data_j}$$

  where data is the entire row of the stock on day j.
- f6 is average directional movement index(ADX) which is obtained from Ta-Lib[3] library. ADX is a momentum indicator developed by J. Welles Wilder which is constructed from smoothing out the Directional Movement Index (DMI) which includes two other Wilders' indicators: the Positive Directional indicator (+DI) and the Negative Directional Indicator (-DI). ADX factor is useful in measuring the strength of a trend. [15]
- f7 is the money flow index(MFI) of a stock which is obtained from Ta-Lib [3] library. MFI is a momentum indicator that measures the inflow and outflow of money into a security over a specific period of time. The MFI is generally calculated by using stock's price and volume to measure trading pressure. [14]
- f8 is the relative strength index(RSI) which is obtained from Ta-Lib [3] library. RSI is a technical indicator used in the analysis of financial markets. It is intended to chart the current and historical strength or weakness of a stock or market based on the closing prices of a recent trading period. [13]
- f9 is the j-index which gets a value based on the trend shown by Nikkei 225 Index. It is a price-weighted index comprised of Japan's top 225 blue-chip companies traded on the Tokyo Stock Exchange. A +1 in f9 means that Nikkei 225 index increased and a -1 means vice versa. A Zero is given if the index hasn't changed from previous day. The historical data of index for six years were obtained from Yahoo Finance and Japan Exchange Group Japan is ranked third based on their GDP and can influence the world economics significantly. So any trend in Tokyo Stock Exchange is likely to be followed by NASDAQ Stock Exchange. But There is a time difference of 14 hours between Japan Standard Time (JST) and Eastern Standard Time (EST). Therefore while using the dataset, f9 for particular date is mapped to the previous day of stock's data. And unwanted data was filtered and unavailable data for f9 is assigned zero assuming index was closed on that particular day.
- f10, f11 are the difference in high/low price vectors of today's stock value from yesterday's.

Only the features (f1,f5,f6,f7,f8) are normalized before using for predictions since the remaining set of features(f3,f4,f9) are already normalized while calculating.

Features are normalized by the given formula.
If X is the vector of length n to be normalized, then we calculate

$$Z = ||x||_2 = \sqrt{\sum_{i=1}^{n} x_i^2}$$

And the new Normalized Vector

$$Y=X/Z$$

Result Discussion

Stock market prediction was carried out using the above-mentioned features. All algorithms mentioned above are implemented with the help of sci-kit learn library [4]. We divided the dataset into 75% for training and 25% data for testing the implementations. Accuracy obtained from different algorithms is plotted in Fig 6. Highest accuracy among the implementation was shown by Multi-Layer Perceptron (85.7%).

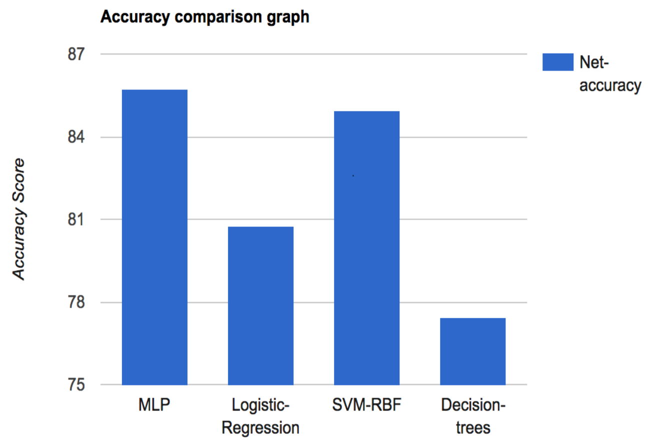


Fig 6. Accuracy Graph

As we can see, the performance of decision tree is relatively low. The minimum number of samples to be split parameter in decision tree is the one which decides upon how many samples should the algorithm split a node.

From the below fig 7, we can see that the decision tree performs poorly when the minimum number of samples is too less or too high which is clearly the case of overfitting or underfitting. And the maximum accuracy attained by decision tree is 0.774 with minimum samples to be split as 27.

Min samples to be split	Net Accuracy
1	0.632
27	0.774
35	0.763
60	0.69

Fig7. Decision tree performance on min-sample-split factor

The accuracy obtained using logistic regression is 80.75%. Since the classifiable data is nonlinear, it is expected that SVM-RBF performs better than logistic regression, which obtained an accuracy of 84.8%. SVM-RBF performance depends on C value and gamma value. C value is penalty cost for misclassification, a large C gives you a low bias, high variance and vice versa. The gamma parameter defines how far the influence of a single training example reaches, with low values meaning 'far' and high values meaning 'close'. The gamma parameter is the inverse of the radius of influence of samples selected by the model as support vectors. It can be observed from Fig.8 and Fig.9 that with higher C value and lower gamma value the net accuracy of SVM using RBF kernel improves.

C Value	Net Accuracy
10e-3	0.75
10e0	0.841
10e1	0.847
10e2	0.848

Fig 8. Performance of RBF under different C values for a default gamma value of 1/number of features

Gamma Value	Net Accuracy
10e-2	0.841
10e-1	0.837
10e1	0.667
10e2	0.539

Fig 9. Performance of RBF under different gamma values for a default C value of 1.0

Feature (f9), which depends on Nikkei 225 Index, was taken because Tokyo Stock Exchange's trading time will be over by the time NASDAQ's trading starts. Using this feature the overall accuracy was increased by 2.7% which clearly conveys the dependency of stock price on other world markets.

Conclusion

In this paper, we described the application of different machine learning algorithms to the task of stock market prediction. We described the theory behind multilayer perceptron, logistic regression, SVM with RBF kernel and decision tree algorithm. The results obtained in all these cases were fairly accurate as evident from Fig. 6, with MLP giving the highest accuracy score. Even though stock market depends on a lot of factors and it is impossible to collect all those factors and supply to a machine learning model, training with important features that involve around the stock will help us getting a high accuracy score. Thus we can use machine learning algorithms to predict the trend of stock market accurately which is otherwise considered unpredictable. This project can be further extended to improve the performance by using more data mining techniques.

References

[1]. https://finance.yahoo.com/
[2]. http://www.nasdaq.com/
[3]. https://pypi.python.org/pypi/TA-Lib
[4]. http://scikit-learn.org/stable/
[5]. Forecasting Stock Market Trends By Logistic Regression And Neural Networks Evidence From Ksa Stock Market Makram Zai.
[6]. Stock Prediction using Artificial Neural Networks Abhishek Kar (Y8021), Dept. of Computer Science and Engineering, IIT Kanpur
[7]. https://en.wikipedia.org/wiki/Artificial_neural_network
[8]. https://en.wikipedia.org/wiki/Decision_tree
[9]. Short-term prediction of exchange traded funds (ETFs) using logistic regression generated client risk profiles
[10] Zahid Iqbal, R. Ilyas, W. Shahzad, Z. Mahmood and J. Anjum, " Efficient Machine Learning Techniques for Stock Market Prediction" in Int. Journal of Engineering Research and Applications, ISSN : 2248-9622, Vol. 3, Issue 6, Nov-Dec 2013, pp.855-867.
[13] Marc-André Mittermaye, "Forecasting Intraday Stock Price Trends with Text Mining Techniques" in the 37th Hawaii International Conference on System Sciences – 2004. [11] Ruchi Desai, Prof. Snehal Gandhi, "Stock Market Prediction Using Data Mining" in International Journal of Engineering Development and Research, 2014 IJEDR | Volume 2, Issue 2 | ISSN: 2321-9939.
[12] Prakash Ramani, Dr. P. D. Murarka, "Stock Market Prediction Using Artificial Neural Network" in International Journal of Advanced Research in Computer Science and Software Engineering. ISSN: 2277-128x, Volume 3, Issue 4, April 2013
[13]. https://en.wikipedia.org/wiki/Relative_strength_index
[14].http://stockcharts.com/school/doku.php?id=chart_school:technical_indicators:money_flow_index_mfi
[15].http://stockcharts.com/school/doku.php?id=chart_school:technical_indicators:average_directional_index_adx
[16].https://repositories.lib.utexas.edu/bitstream/handle/2152/1532/davisv09008.pdf