Journal of Finance and Investment Analysis, vol. 2, no.1, 2013, 41-59
ISSN: 2241-0998 (print version), 2241-0996(online)
Scienpress Ltd, 2013

Forecasting Movement of the Nigerian Stock Exchange All Share Index using Artificial Neural and Bayesian Networks

Adetunji Abigail Bola[1], Aderounmu Ganiyu Adesola[2], Omidiora Elijah Olusayo[3] and Adigun Abimbola Adebisi[4]

Abstract

This paper presents a study of Artificial Neural Network (ANN) and Bayesian Network (BN) for use in stock index prediction. The data from Nigerian Stock Exchange (NSE) market are applied as a case study. Based on the rescaled range analysis, the neural network was used to capture the relationship in terms of weights between the technical indicators derived from the NSE data and levels of the index. The BayesNet Classifier was based on discretizing the numeric attributes into distinct ranges from where the conditional probability was calculated, stored in the Conditional Probability Table (CPT) and the new instance were classified. The performance evaluation carried out showed results of 59.38% for ANN and 78.13% for BN in terms of predictive power of the networks.

The result also showed that Bayesian Network has better performance than ANN when it comes to predicting short period of time; and that useful prediction can be made for All Share index of NSE stock market without the use of extensive market data.

---

[1]Department of Computer Science & Engineering,
Ladoke Akintola University of Technology, Ogbomoso, Nigeria.
e-mail: abadetunji@yahoo.com
[2]Department of Computer Science & Engineering,
Ladoke Akintola University of Technology, Ogbomoso, Nigeria.
e-mail: gaderoun@oauife.edu.ng
[3]Department of Computer Science & Engineering, Obafemi Awolowo University, Ile-Ife, Nigeria.
e-mail: omidiorasayo@yahoo.co.uk
[4]Department of Computer Science & Engineering,
Ladoke Akintola University of Technology, Ogbomoso, Nigeria.
e-mail: adigunademidun@yahoo.co.uk

Article Info: *Received :* July 12, 2012. *Revised :* September 2, 2012.
Published online : February 10, 2013

JEL classification numbers: E5, C11, C450, D47
Keywords: Artificial Neural Network; Bayesian Network; Financial data; Stock market

1 Introduction

Financial risk has proven in this day and age to be a threat that may cause immeasurable damage and as a result, different measures are taken to prevent or at least reduce the risk. Forecasting is the process of estimation of values in unknown situations for certain specific future times, and it is commonly used with time-series data [21]. It is a process that produces a set of outputs by giving a set of variables. The variables are normally historical data. The basic idea of forecasting is to find an approximation of mapping between the input and output data in order to discover the implicit rules governing the observed movements [15, 21 and 18]. Statistical methods and neural networks are commonly used for time series prediction. A number of techniques have been used in financial forecasting, some of these are: Non-linear modeling, where financial data are regarded as non-linear and therefore require non-linear modeling [8]. Fuzzy rule based system, where the relationship among factors are modeled in a fuzzy relationship [3].
Neural networks, analyze relationships among complex financial data and store relationships in terms of weights as a result from training [10, 14, 20, 21 and 22]. The empirical results in the literature offer mixed support for the neural network models. While some studies reported the superiority of the neural network models over the other models ([3, 6, 7 and 13], no robust superiority could be found in other studies [11, 16 and 17].
The more and accurate the training data, the more accurate the network will perform. Unfortunately abundant data is not always available. The level of accuracy in such cases is not high [9]. Neural networks, being developed primarily for the purpose of pattern recognition from complex sensor data, such as inputs from cameras and microphones, are not well suited for modeling time series because the original application of Neural networks were concerned with detection of patterns in arrays of measurements which do not change in time [5, 12].
In this paper, a modified Artificial Neural Network (ANN) based on the rescaled analysis, to capture the relationship between the Technical Indicators and the levels of the Index in the Nigerian Stock Exchange (NSE) market. A supervised discretization BayesNet classifier; a modified version of the probabilistic BayesNet classifier was also used. An algorithm was developed to evaluate the performance of these models. This developed algorithm is defined as learning from learned Knowledge [2]. Knowledge was extracted from modified ANN and the discretized BayesNet classifier. The algorithm can be viewed as a transformed model that converts the output in one world view to another world view [1].

2 Materials and Method

This study is composed of three phases. The first phase is to generate the data sets, which consisted of daily stock prices; these are the open price, high price, low price, close price, volume and the All-Share index of the Nigeria Stock Exchange market. Technical

Indicators were generated from the data for ANN algorithm, while categorical form of data was generated for Baye Network algorithm. This phase served as the pre-processing stage. The modified ANN and Baye training algorithms rely heavily on the product of this phase.

Learning was carried out in the second phase. This is by applying the algorithms to the pre-processed data to discover knowledge. The knowledge discovered is in the form of classifier or prediction model. The third phase developed an algorithm to show the performance of ANN and Baye prediction models on the NSE data. Figure 1 depicts the research framework.

2.1 Neural Network Learning Algorithm

Besides popular gradient descent of the backpropagation algorithm, the Gradient Descent with Momentum and variable learning rate is adopted in this research work to minimize output error. With standard gradient descent, the learning rate is held constant throughout training. The performance of the algorithm is very sensitive to the proper setting of the learning rate. If the learning rate is set too high, the algorithm may oscillate and become unstable. If the learning rate is set too small, the algorithm will take too long to converge. It is practical to determine the optimal setting for the learning rate before training.

The performance of the standard descent algorithm can be improved as the learning rate is allowed to change during the training process, thus making the optimal learning rate to change during the training process. The momentum is added to alter the weight-update. This is by making the weight-update on the *n*th iteration depend partially on the update that occurred during the $(n-1)$ th iteration, this is given as:

$$\Delta w_{ji}(n) = \eta \delta_j x_{ji} + \Delta w_{ji}(n-1) \tag{1}$$

$\Delta w_{ji}(n)$ is the update performed during the *n*th iteration

$0 \leq \alpha < 1$ is a constant called the momentum

For each training example *d*, descending the gradient of the error E_d, with respect to this single example, every weight w_{ji} is updated by adding to it Δw_{ji}

$$\Delta w_{ji} = -\eta \frac{\partial E_d}{\partial w_{ji}} \tag{2}$$

where, E_d is the error on training example *d,* summed overall output units in the network η is the leaning rate that determines the size of the step that we use for' moving' towards the minimum of *E*. Usually if $\eta \in \Re, 0 < \eta \leq 0.5$ .If η is too large it leads to oscillation around the minimum, while too small can lead to a slow convergence of the ANN.

The objective function is defined by the error function:

$$E(\vec{w}) \equiv \frac{1}{2} \sum_{d \in D} \sum_{k \in outputs} (t_{kd} - o_{kd})^2 \tag{3}$$

t_{kd} and o_{kd} are the targets and output values associated with the kth output and the training example d.

The error term for the output units:

$$\delta_j = (t_j - o_j) o_j (1 - o_j) \tag{4}$$

The weight update Δw_{ji} for output units is expressed as:

$$\Delta w_{ji} = -\eta \frac{\partial E_d}{\partial w_{ji}} = \eta (t_j - o_j) o_j (1 - o_j) x \tag{5}$$

The error term for the hidden units:

$$\delta_j = o_j (1 - o_j) \sum_{k \in Downstream(j)} \delta_k w_{kj}$$

The weight update Δw_{ji} for hidden units is expressed as:

$$\Delta w_{ji} = \eta \delta_j x_{ji} \tag{6}$$

The effect of the momentum is to gradually increase the step size of the search in regions where the gradient is unchanging, thereby speeding convergence.

The summary of the Gradient Descent with Momentum and variable learning rate could be described as:

1. Create a feed-forward network with n_{in} inputs, n_{hidden} hidden units, and n_{out} output units.
2. Initialize all network weights to small random numbers (e.g. between -0.05 and 0.05).
3. Until the termination condition is met, do

For each $(\vec{x}, \vec{t})$ in training examples, do

i) Propagate the input forward through the network:

Input the instance $\vec{x}$ to the network and compute the output o_u of every unit u in the network.

Propagate the errors backwards through the network:

ii) For each network output unit k, calculate its error term δ_k

$$\delta_k \leftarrow transfer'(net_k)(t_k - o_k) \tag{7}$$

iii) For each hidden unit h, calculate its error term δ_h

$$\delta_h \leftarrow transfer'(net_h) \sum_{k \in outputs} w_{kh} \delta_k \tag{8}$$

iv) Update the network weight on the *nth* iteration and add momentum α at the $(n-1)$*th* iteration $(0 \le \alpha < 1)$

$$w_{ji} \leftarrow w_{ji} + \Delta w_{ji}(n)$$

$$\Delta w_{ji} \leftarrow \eta \delta_j x_{ji} + \alpha \Delta w_{ji}(n-1) \tag{9}$$

v) If new error $>$ old error do

$$\Delta w_{ji} \leftarrow 0.7 \eta \delta_j x_{ji} + \alpha \Delta w_{ji}(n-1)$$

else

$$\Delta w_{ji} \leftarrow 1.05 \eta \delta_j x_{ji} + \alpha \Delta w_{ji}(n-1) \tag{10}$$

vi) If the error on the training examples falls below the threshold do

Terminate the process

else, go to step (v)

2.1.1 Bayesian network learning algorithm

In machine learning, the interest is in determining the best hypothesis from space, *H*, given the observed training data, *D*. Bayes theorem is the cornerstone of Bayesian leaning methods because it provides a way to calculate the probability of a hypothesis based on its prior probability, the probabilities of observing various data given the hypothesis, and the observed data itself. (Stutz and Cheeseman, 1994). Bayes theorem uses the following notation:

P(*h*) – Initial probability that hypothesis *h* holds, before we observed the training data or Prior probability of *h*;

P(*D*) – Prior probability that the training data ,*D*, will be observed. It is the probability of *D* given no prior knowledge about which hypothesis holds or the marginal likelihood

P(*D*|*h*) – Probability of observing data, *D*, given some world in which hypothesis *h* holds or the likelihood;

P(*h*|*D*) – Probability that hypothesis, *h*, holds given the observed training data, *D*, (also called the posterior probability). The posterior probability $P(h \mid D)$ reflects the influence of the training data, *D*, in contrast to the prior probability $P(h)$, which is independent of *D*.

Bayes theorem is the cornerstone of Bayesian learning methods because it provides a way to calculate the posterior probability, $P(h \mid D)$ from the prior probability, $P(h)$ together with $P(D)$ and $P(D \mid h)$.

Mathematically, Bayes'rule states that:

$$posterior = \frac{likelihood * proir}{maginallikelihood} \tag{11}$$

This follows from Baye theorem

$$P(h \mid D) = \frac{P(D \mid h)P(h)}{P(D)} \tag{12}$$

In the learning scenarios, the learner considers some set of candidate hypothesis *H* and is interested in finding the most probable hypothesis $h \in H$ given the observed data *D*. Any such maximally probable hypothesis is called a *maximum a posterior* (MAP) hypothesis by using Bayes theorem to calculate the posteriror probability of each candidate hypothesis.

Precisely, h_{MAP} is a MAP hypothesis provided

$$h_{MAP \equiv} \arg\max_{h \in H} P(h \mid D)$$
$$= \arg\max_{h \in H} \frac{P(D \mid h)P(h)}{p(D)}$$
$$= \arg\max_{h \in H} P(D \mid h)P(h) \tag{13}$$

The term $P(D)$ is dropped because it is a constant independent of *h* and it is a normalizing constant.

2.2 Bayesian Network in Finance

The use of Bayesian network in financial predictions appears to be a new research area in the technology world. Bayesian network has been majorly put to work in probability analysis. Since financial forecasting or prediction is a statistical approach, Bayesian network can be used to predict the possibilities of an increase or decrease in the stock prices and market indices of a given financial data over a calculated period of time. In a nutshell, the Bayesian probability of an event *x* is a person's degree of belief in that event. Statistical data from Nigerian Stock Market can be analyzed, taking into consideration the missing data, and transforming the data into the required Bayesian input data form.

2.3 BayesNet Classifier

A Bayesian Learning method often called BayesNet Classifier. It can be compared to that of the Neural Network and the Decision Tree learning in terms of performance. When a learning task is provided, given an instance, *x* ,described by the set of attributes provided in the training data and a target function, *f(x)* which takes on values from a finite set, *V*, a new instance is presented for which a learner is asked to predict a target value on the basis of the target function and the set of attributes provide.
The Bayesian approach to classifying the new instance is to assign the most probable target value given the set of v_{MAP}, given the attributes values $\langle a_1, a_2, ..., a_n \rangle$ that describes the instance.

$$v_{MAP} = \arg\max_{v_j \in V} P(v_j \mid a_1, a_2, ..., a_n) \tag{14}$$

Using Bayes theorem, the expression becomes

$$v_{MAP} = \arg\max_{v_j \in V} \frac{P(a_1, a_2, ..., a_n \mid v_j) P(v_j)}{P(a_1, a_2, ..., .a_n)} \tag{15}$$

The value of $P(v_j)$ can be estimated simply by calculating the frequency of occurrence of the value in the training data provided. Also, the value of $P(v_j \mid a_1, a_2, ..., a_n)$ can be calculated in a similar manner but on the condition that a very large set of training data is provided. Naïve Bayes Classifiers assumes that the set of attributes are conditionally independent given the target value. Therefore, the probability $P(v_j \mid a_1, a_2, ..., a_n)$, can be estimated as the probability of the product of each attribute given the target. Therefore the output of a Naïve Bayes Classifiers is given as:

$$v_{NB} = \arg\, maax_{v_{j \in V}} P(v_j) \prod_i P(a_i \mid v_j) \tag{16}$$

2.4 Experiment

The data used in this study consisted of daily stock prices and volume from the Nigerian All-Share Index and about two hundred stocks traded in this market. The data were obtained from Forte Asset Management Limited and Alangrange Security Limited, in

Lagos Nigeria. The study considered the daily closing data for January 2005 - December 2007, this represented a fairly calmer period in the NSE market.

2.4.1 Neural Network Training

Gradient descent with a variable learning rate and momentum algorithm was used in this research work, implements a gradient descent search through the space of possible network weights and iteratively reducing the error *E*, between the training example target values and the network outputs. Higher percentage of the data set was used for training and the rest for testing and validation.
The network was trained using data from March 16, 2005 to February 23, 2006 as input and index from February 24, 2006 to February 27, 2007 as targets to train against. A three-layer network architecture was used. The required number of hidden nodes is estimated by:

$$\text{No. of hidden nodes} = \frac{(M+N)}{2} \tag{17}$$

where *M* and *N* is the number of input nodes and output nodes respectively.
The sigmoid hyperbolic tangent function is adopted in this research work, with function G:

$$G(z) = \tan(h) = \frac{1-e^{-z}}{1+e^{-z}} \tag{18}$$

The raw data is preprocessed into various technical indicators to gain insight into the direction that the stock market may be going. The parameters for training were as given below. Neural network cannot handle wide range of values. In order to avoid difficulty in getting network outputs very close to the endpoints, the indicators were normalized to the range [-1,1], before being input to the network.

2.4.1.1 Network Parameters

- Network Architecture: 9-5-1.
- Transfer Functions: Hyperbolic tangent sigmoid transfer function and linear transfer function.
- Inputs: Stock moving average convergence/divergence, stock stochastic oscillator, closing momentum, stock relative strength index, stock on-balance volume, and the 5 and 10 days closing moving average.
- Algorithm: Gradient descent with a variable learning rate and momentum

2.4.2 Bayesian Network Training

The design and modeling of the data is realized using a Directed Acyclic Graph (DAG). Figure 3 is the Bayesian network structure for NSE. The graph is built on the basis of the dependency inherent between the variables Open, Close, High, Low, Volume and the All Share Index. From the manner in which the NSE Index is calculated, we can deduce that the Index is dependent on the Open, Close, High, Low and Volume of the NSE Market.
Table 1 shows a discretized form of the summary of the NSE data for 2005 used for training. Discretization involves partitioning the data by placing break-points in it. For NSE data, a break-point is placed where the value changes as compared with the previous.

This can either be a 'rise' or 'fall'. The conditional probability is calculated from this table. The following shows the algorithm with which the table was derived:

- Acquisition of the raw data (2005) in a spread sheet;
- Selection of the first 50 consistent companies between months;
- For each day, the open, close, high, low and volume were summed respectively;
- Comparison of successive days with their previous days to achieve either a 'rise' or a 'fall' in their values.

The Bayesian network was trained using the 2005 NSE data as presented in the Tables 2 and 3 showing the Conditional Probability Distributions (CPD) represented in the Conditional Probability Tables (CPTs).

A new instance, for example, was determined by providing evidences as the opening price of a particular day rose or fell and also same for the rest of the variables. By presenting the test case to the network, the values in the Conditional Probability Table (CPT) was adapted to reflect the data that it received. The system would then forecast whether the target value was 'rise' or 'fall' of the target concept, the All Share Index for the new days data.

3 Performance Evaluation of ANN and Bayesian Networks

In predicting the stock market All Share Index for Nigerian NSE at time t+1, from the methodologies used in this research; the ANN problem for stock price index involved modeling the actual price or value, while the Bayesian problem involved predicting the percentage of rising or falling of the stock index price. To properly evaluate the two networks an algorithm was written. This Bilearning-based is developed to solve the problem of evaluation. The algorithm is defined as learning from learned models or techniques. Learning is concerned with finding model, $f = f_x[i]$ from a single training set $\{TR_i\}$like that of NSE data set, while this performance model is concerned with finding model $f = f_x$ from two training sets , $\{TR_1 and TR_2\}$, each of which has an associated model, that is, the base models. The corresponding outputs or results produced by these base models were used as inputs into this Hybrid Baye-ANN model. The algorithm is as given below:

For Bayesian network model:
Train the network and predict the values for the data set.
Create two vectors for the storage of the network outputs '2-1'
IF the output vector is a 'rise', store the value 2 as the '2-1' vector value for that day
ELSE store 1 for 'fall'
For ANN model:
Train the network and predict the values for the data set.
Create two vectors for the storage of the network outputs '2-1prediction.
Store the previous status to 2.
For each day do
Subtract the previous day from the present day.
IF it is positive store in 2 in the '2-1' vector value.
ELSE store 1in the '2-1' vector value.

otherwise store the value in the previous status in the ‘2-1’ vector value.
Update the previous status value with the last stored value ‘2-1’ vector value.
For the performance model;
Create two new variables for the ANN and Bayesian networks.
IF the ‘2-1’ vector value for a network is equal to the ‘2-1’vector value of original index for the data set, THEN increase that particular network variable by 1.
ELSE store the vector value
Divide each variable by the number of the data set and multiply it by 100 to get the percentage of accurate prediction for each network.
Display the results on a bar chart for the each 2 and 1 value in the ‘2-1’ vectors.

4 Results and Discussion

The ANN was trained using the training data set as provided in Table 1, to find the general pattern of inputs and outputs. To avoid over-fitting of the network, the hold-out validation set was used as cross- testing data set. The data was chosen and segregated in time order. In order words, the earlier-period and later-period data was used for training and validation respectively; newly collected data was also used for testing. The training time for ANN lasted for more than 24 hours. The optimal setting of the learning rate is the trade-off between convergence and generalization. Table 5 shows the ANN basic performance metric used in this research work is the Minimum Square Error (MSE). Figures 4 and 5 show the output result from the training set as against the target All Share index and the forecasting result using the test data.
The Bayes algorithm, a classification algorithm, was applied using the data from the CPT to carry out the testing. Since Bayesian network is probabilistic in nature, the classifier showed whether the All Share index ‘rise’ or ‘fall’. For example, figure 6 gave a ‘rise’ for the index for a test data set from table 6.
To evaluate the performance of ANN and BN models, an algorithm was used, and the result is as displayed in figure 7 and table 6 below. The success of the algorithm for ANN is 59.38% and Bayesian network is 78.13%. Also, Table 7 summarizes the behavioral patterns of both models.

5 Conclusion

In this paper, it has been shown that the index of the NSE market could be forecasted using ANN and BN methods. The historical data set was collected from the NSE market. The past is not fully unrelated with its future since calculating the index values for the predicted and real index showed a slight difference in values. Therefore, it can be said that the spontaneous nature of the time series caused a shift in the value of the predicted index among other minor displacements.
The forecasting models employed are characterized with the following: the ANN used the delayed index levels and some technical indicators calculated from the stock prices as inputs, while the current index level used as output. This research work shows the weakness of ANN, in predicting the NSE financial time series because the data collected is not long-term form. The data may not be suitable to better train the ANN system to

learn properly. From this work, one could easily infer that 'not all financial time series can be efficiently predicted by ANN'.
For BN model, the results are probabilistic values of the All Share Index. This is because BN is a graphical model for probabilistic relationships among a set of variables. The overall assessment of both algorithms showed that BN model performed better than ANN model (Table 6). To improve ANN predictive capabilities in forecasting the NSE stock market, a mixture of technical and fundamental factors as inputs over different time periods should be considered. The characteristics of emerging market like that of NSE stock market should be further researched on to facilitate better market.

References

[1] A.B. Badiru and D. B. Sieger (1998) Neural Network as a Simulation Metamodel in Economic Analysis of Risky Projects, European Journal of Operational Research, **105**, 130-142.
[2] P. Chan and S. Stolfo (1993) Meta-Learning for Multistrategy and Parallel Learning, Proceedings of the Second International Workshop on Multistrategy Learning, 150–165.
[3] P.C.Chang and C. H. Liu (2006). A TSK type fuzzy rule based system for stock price prediction. Elsevier Expert system. Appl. J., **34**(1): 135-144.
[4] A.F. Darrat and M. Zhong (2000). On testing the random -walk hypothesis: A model Comparison Approach. The Financial Review, **35**: 105-124.
[5] G. Dorffner (1996). Neural Network for Time Series Processing. Neural Network world. **6**(4): 447-468.
[6] R. Gencay (1996). Non-linear prediction of security returns with moving average rules. Journal of Forecasting, **15**: 165-174.
[7] R. Gencay and T. Stengos, T. (1998). Moving Average Rules, Volume and the Predictability of StockReturns with Feedforward Networks, Journal of Forecasting, **17**: 401-141.
[8] N. Gradojevic (2006). Non-linear, Hybrid exchange rate modeling and trading profitability in the foreign exchange market. Elsevier J. Econ. Dynamics and Control, **31** (2): pp. 557-574.
[9] L. B. Joko, F. Chastine, and S. Mudji (2009). Hybrid Neural Network- Monte Carlo Simulation for Stock Price Index Prediction. Asian Journal of Information Technology, **8**(1): 1-7.
[10] K. J. Kim (2006). Artificial neural network with evolutionary instance selection for financial forecasting. Elsevier Expert Syst. Appl. J., **30** (3): 519-526.
[11] E. Maasoumi and J. Racine (2002). Entropy and Predicability of Stock Market Returns, Journal of Econometrics, **107**, 291-312.
[12] T. Mitchell (1997), Machine Learning. McGraw Hill Int. Editions. Machine Press, China.
[13] D. Olson and C. Mossman (2002). Neural Network Forecasts of Canadian stock returns using accounting ratios. International Journal of Forecasting, **1**, 1-13.
[14] N. O'Connor and M. G. Madden (2006). A Neural Network approach to predicting stock exchange movements using external factors. Elsevier Knowledge-Based System Journal, **19** (5): 371-378.

[15] L. Ramon (1997) Using Neural Networks to Forecast Stock Market Price, Department of Computer Science, University of Manitoba, Manitoba.
[16] J.V. Rodriguez, S. Torra and J. A. Felix (2005) STAR and ANN models: Forecasting performance on Spanish Ibex-35 stock index. Journal of Empirical Finance, **12**(3): 490-509.
[17] S. Singh (1999). A Long memory pattern modelling and recognition system for financial time series forecasting. Pattern Analysis and Applications, **2**: 264-276.
[18] S.R. Stansell and S.G.Eakins (2003). Forecasting the direction of change in sector indexes: An application of neural networks. Journal of Asset Management, **5**(1): 37-48.
[19] J. Stutz and P. Cheeseman (1994). A short Exposition on Bayesian Inference and Probability. National Aeronautic and Space Administration Ames Research Centre: Computational Sciences Division, Data Learning Group.
[20] P.M.Tsang, P. Kwok, S. Choy, R. Kwan, S. Ng, J. Mak, J. Tsang, K. Koong and T. L. Wong (2007) Design and implementation of NN5 for Hong Kong price forecasting. Elsevier Engineering, Applied Artificial Intelligence Journal, **20** (4): 453-461.
[21] J. Yao and C. L.Tan (2001) Guidelines for Financial Forecasting with Neural Networks., http://www.sfu.ca/~rgencay/jarticles/jef-technical.pdf.
[22] G.P.Zhang (2004). Business Forecasting with Artificial Neural Networks: An overview. 1st Edition Georgia State University, USA, 1-22.

Appendix

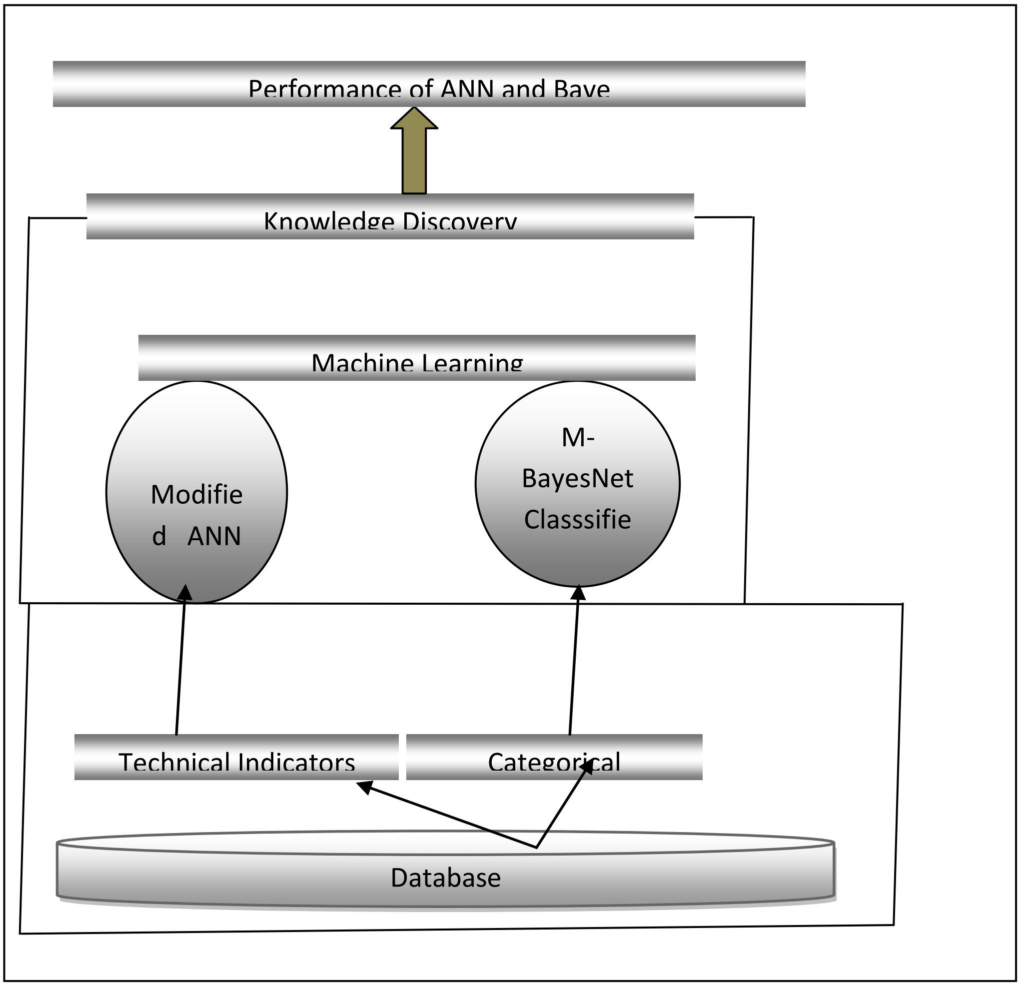
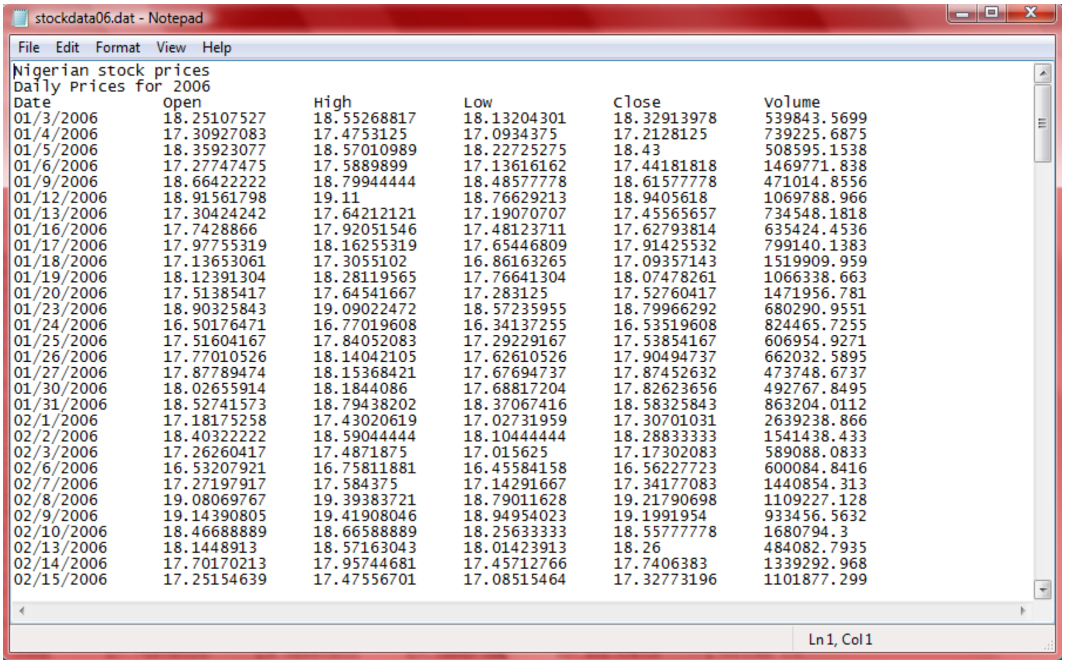


Figure 1: The proposed framework for the Artificial Neural and Bayesian Network



stockdata06.dat - Notepad

File Edit Format View Help

Nigerian stock prices
Daily Prices for 2006

Date	Open	High	Low	Close	Volume
01/3/2006	18.25107527	18.55268817	18.13204301	18.32913978	539843.5699
01/4/2006	17.30927083	17.4753125	17.0934375	17.2128125	739225.6875
01/5/2006	18.35923077	18.57010989	18.22725275	18.43	508595.1538
01/6/2006	17.27747475	17.5889899	17.13616162	17.44181818	1469771.838
01/9/2006	18.66422222	18.79944444	18.48577778	18.61577778	471014.8556
01/12/2006	18.91561798	19.11	18.76629213	18.9405618	1069788.966
01/13/2006	17.30424242	17.64212121	17.19070707	17.45565657	734548.1818
01/16/2006	17.7428866	17.92051546	17.48123711	17.62793814	635424.4536
01/17/2006	17.97755319	18.16255319	17.65446809	17.91425532	799140.1383
01/18/2006	17.13653061	17.3055102	16.86163265	17.09357143	1519909.959
01/19/2006	18.12391304	18.28119565	17.76641304	18.07478261	1066338.663
01/20/2006	17.51385417	17.64541667	17.283125	17.52760417	1471956.781
01/23/2006	18.90325843	19.09022472	18.57235955	18.79966292	680290.9551
01/24/2006	16.50176471	16.77019608	16.34137255	16.53519608	824465.7255
01/25/2006	17.51604167	17.84052083	17.29229167	17.53854167	606954.9271
01/26/2006	17.77010526	18.14042105	17.62610526	17.90494737	662032.5895
01/27/2006	17.87789474	18.15368421	17.67694737	17.87452632	473748.6737
01/30/2006	18.02655914	18.1844086	17.68817204	17.82623656	492767.8495
01/31/2006	18.52741573	18.79438202	18.37067416	18.58325843	863204.0112
02/1/2006	17.18175258	17.43020619	17.02731959	17.30701031	2639238.866
02/2/2006	18.40322222	18.59044444	18.10444444	18.28833333	1541438.433
02/3/2006	17.26260417	17.4871875	17.015625	17.17302083	589088.0833
02/6/2006	16.53207921	16.75811881	16.45584158	16.56227723	600084.8416
02/7/2006	17.27197917	17.584375	17.14291667	17.34177083	1440854.313
02/8/2006	19.08069767	19.39383721	18.79011628	19.21790698	1109227.128
02/9/2006	19.14390805	19.41908046	18.94954023	19.1991954	933456.5632
02/10/2006	18.46688889	18.66588889	18.25633333	18.55777778	1680794.3
02/13/2006	18.1448913	18.57163043	18.01423913	18.26	484082.7935
02/14/2006	17.70170213	17.95744681	17.45712766	17.7406383	1339292.968
02/15/2006	17.25154639	17.47556701	17.08515464	17.32773196	1101877.299

Ln 1, Col 1

Figure 2: Nigerian Stock Market Daily Stock Prices for 2006

Source: Forte Asset Management Limited and Alangrange Security Limited, in Lagos Nigeria

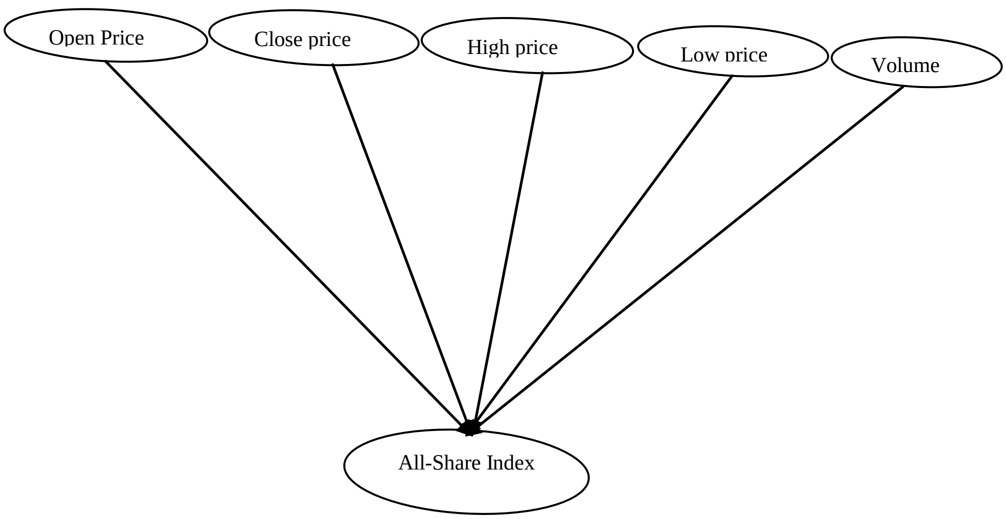


Figure 3: Bayesian Network Structure for NSE

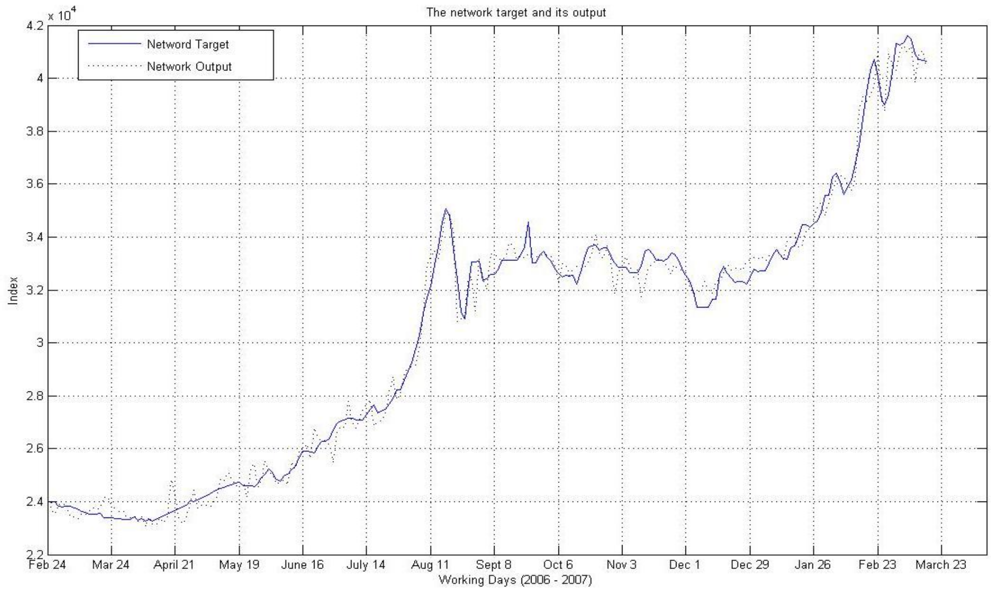


Figure 4: ANN model training set output as against the target, All-Share index.

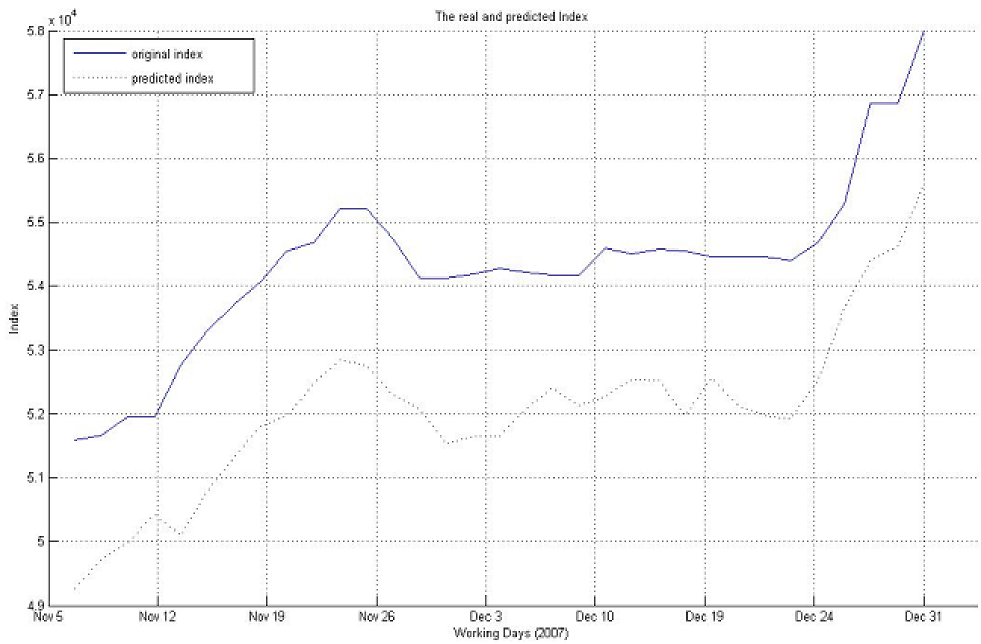


Figure 5: Prediction of All Share Index Price NSE

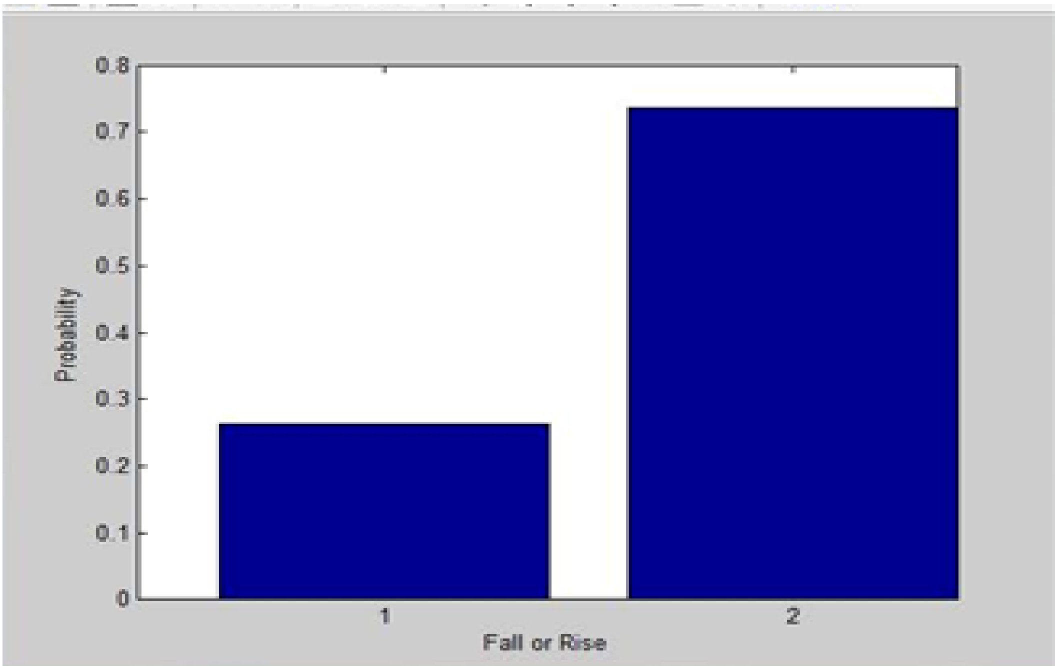


Figure 6: Probability of 'rise' of the All-Share Index

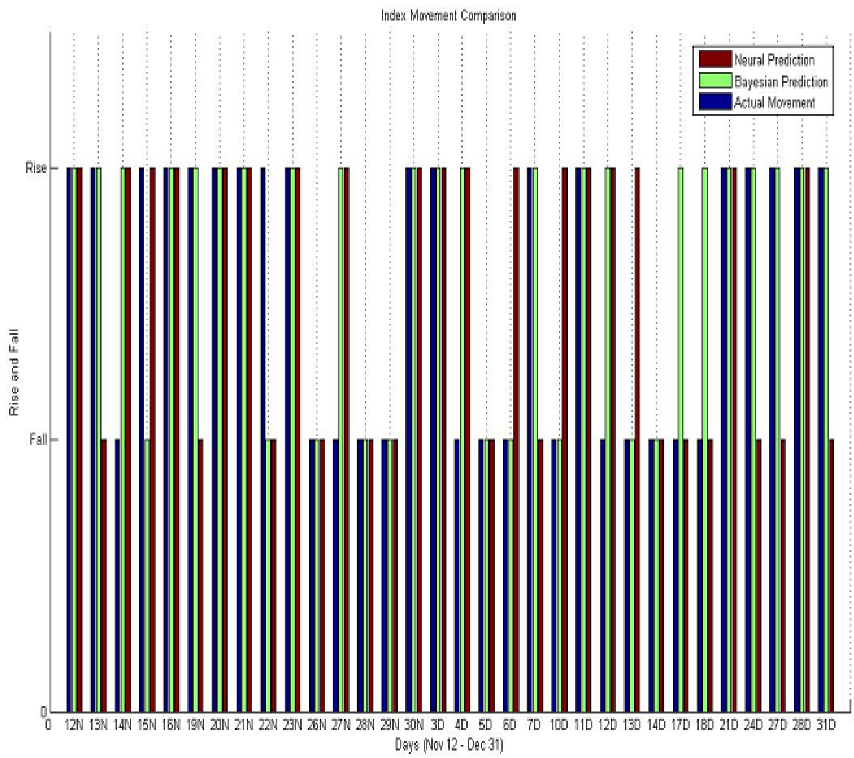


Figure 7: Performance of ANN and BN on All-Share Index

Table 1: Training Instances for the Target Concept All -Share Index: Nominal Attributes

DATE	OPEN	HIGH	LOW	CLOSE	VOLUME	ALL SHARE INDEX
3/1/2005	fall	fall	fall	fall	rise	fall
3/2/2005	fall	fall	fall	fall	rise	fall
3/3/2005	fall	fall	fall	fall	rise	fall
3/7/2005	fall	rise	rise	rise	fall	rise
3/8/2005	rise	fall	rise	rise	fall	rise
3/9/2005	rise	fall	rise	fall	rise	fall
3/10/2005	fall	rise	fall	rise	rise	fall
3/11/2005	fall	fall	rise	rise	fall	rise
3/14/2005	rise	rise	rise	rise	fall	fall
3/15/2005	rise	rise	fall	fall	rise	fall
3/16/2005	fall	fall	fall	fall	rise	fall
3/17/2005	fall	fall	fall	fall	fall	fall
3/18/2005	rise	rise	rise	rise	rise	rise
3/21/2005	fall	fall	fall	fall	fall	fall
3/22/2005	fall	rise	fall	rise	rise	fall
3/23/2005	rise	rise	fall	fall	fall	fall
3/24/2005	fall	fall	fall	fall	fall	Fall

Table 2: Conditional Probability for the Independent Variables Open rice, Close price, High price, Low price and Volume

OPEN PRICE		**CLOSE PRICE**		**HIGH PRICE**		**LOW PRICE**		**VOLUME**	
rise	fall	rise	fall	rise	fall	rise	fall	rise	fall
0.525	0.475	0.51	0.49	0.515	0.485	0.515	0.485	0.495	0.505

Table 3: Conditional Probability for the All-Share Index

					INDEX	
OPEN	**HIGH**	**LOW**	**CLOSE**	**VOLUME**	**rise**	**fall**
rise	rise	rise	rise	rise	0.163	0.09
rise	rise	rise	rise	fall	0.106	0.069
rise	rise	rise	fall	rise	0.016	0.021
rise	rise	rise	fall	fall	0.041	0.034
rise	rise	fall	rise	rise	0.016	0.041
rise	rise	fall	rise	fall	0.033	0.014
rise	rise	fall	fall	rise	0.008	0.034
rise	rise	fall	fall	fall	0.16	0.014
rise	fall	rise	rise	rise	0.024	0.014
rise	fall	rise	rise	fall	0.033	0.014
rise	fall	rise	fall	rise	0.016	0.028
rise	fall	rise	fall	fall	0.041	0.021
rise	fall	fall	rise	rise	0.008	0.007
rise	fall	fall	rise	fall	0.016	0.007
rise	fall	fall	fall	rise	0.008	0.055
rise	fall	fall	fall	fall	0.016	0.021

Table 4: Instances for Testing

INSTANCES	EVIDENCES					INFERENCE
	OPEN	CLOSE	HIGH	LOW	VOLUME	**INDEX**
Instance 1	rise	rise	rise	rise	rise	**?**
Instance 2	rise	fall	fall	rise	rise	**?**

Table 5: The Testing Result

Architecture	**Learning rate** η	**Momentum rate** α	**Training MSE**	**Testing MSE**
9-5-1	Between 0.01 and 0.05	0.005	9.9296e-7	4.78423-8

Table 6: Simulation Results of Performance of both ANN and BN Systems

	Actual Movement	**Bayesian Network**	**Neural Network**
12-Nov	Rise	Rise	Rise
13-Nov	Rise	Rise	Fall
14-Nov	Fall	Rise	Rise
15-Nov	Rise	Fall	Rise
16-Nov	Rise	Rise	Rise
19-Nov	Rise	Rise	Fall
20-Nov	Rise	Rise	Rise
21-Nov	Rise	Rise	Rise
22-Nov	Rise	Fall	Fall
23-Nov	Rise	Rise	Rise
26-Nov	Fall	Fall	Fall
27-Nov	Fall	Rise	Rise
28-Nov	Fall	Fall	Fall
29-Nov	Fall	Fall	Fall
30-Nov	Rise	Rise	Rise
03-Dec	Rise	Rise	Rise
04-Dec	Fall	Rise	Rise
05-Dec	Fall	Fall	Fall
06-Dec	Fall	Fall	Rise
07-Dec	Rise	Rise	Fall
10-Dec	Fall	Fall	Rise
11-Dec	Rise	Rise	Rise
12-Dec	Fall	Rise	Rise
13-Dec	Fall	Fall	Rise
14-Dec	Fall	Fall	Fall
17-Dec	Fall	Rise	Fall
18-Dec	Fall	Rise	Fall
21-Dec	Rise	Rise	Rise
24-Dec	Rise	Rise	Fall
27-Dec	Rise	Rise	Fall
28-Dec	Rise	Rise	Rise
31-Dec	Rise	Rise	Fall

Table 6: Performance of the ANN and BN Models

Performance Indicator	**ANN**	**BN**
Approach for Pre-processing of the Data	Normalization	Discretization
Efficiency	Sorting of parameters and overfitting of the training data	No over-fitting for the nominal attributes
Convergence	Does not converge very fast	There is no need of convergence
Robustness	Cannot work well with limited data	Works very well with very limited data
Speed of operation	Takes longer time to train the data; the validation makes the time longer.	It train very fast and requires Conditional Probability Table (Table 2 and 3)
Memory usage	Large memory required for training	Limited memory required
Overall success	59.38%	78.13%
Predictive Power	Not very good for short-term prediction	Suitable for short period prediction