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

Adetunji Abigail Bola¹, Aderounmu Ganiyu Adesola², Omidiora Elijah Olusayo³ and Adigun Abimbola Adebisi⁴

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.

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

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

\(\Delta w_{ji}(n)\) 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 \(\Delta w_{ji}\)

\[
\Delta w_{ji} = -\eta \frac{\partial E_d}{\partial w_{ji}}
\]

where, \(E_d\) is the error on training example \(d\), summed overall output units in the network

\(\eta\) is the leaning rate that determines the size of the step that we use for’ moving’ towards the minimum of \(E\). Usually if \(\eta \in \mathbb{R}, 0 < \eta \leq 0.5\). If \(\eta\) 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:

\[
E(\mathbf{w}) \equiv \frac{1}{2} \sum_{d \in D} \sum_{k \in \text{outputs}} (t_{kd} - o_{kd})^2
\]

\(t_{kd}\) and \(o_{kd}\) are the targets and output values associated with the \(k\)th output and the training example \(d\).

The error term for the output units:
\[ \delta_j = (t_j - o_j) o_j (1 - o_j) \]  \hspace{1cm} (4)

The weight update \( \Delta w_{ji} \) for output units is expressed as:

\[ \Delta w_{ji} = -\eta \frac{\partial E_d}{\partial w_{ji}} = \eta (t_j - o_j) o_j (1 - o_j) x \]  \hspace{1cm} (5)

The error term for the hidden units:

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

The weight update \( \Delta w_{ji} \) for hidden units is expressed as:

\[ \Delta w_{ji} = \eta \delta_j x_{ji} \]  \hspace{1cm} (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
   a) Propagate the input forward through the network:
      i) Input the instance \( \hat{x} \) to the network and compute the output \( o_u \) of every unit \( u \) in the network.
      ii) Propagate the errors backwards through the network:
          i) For each network output unit \( k \), calculate its error term \( \delta_k \)
             \[ \delta_k \leftarrow \text{transfer}'(net_k)(t_k - o_k) \]  \hspace{1cm} (7)
          ii) For each hidden unit \( h \), calculate its error term \( \delta_h \)
              \[ \delta_h \leftarrow \text{transfer}'(net_h) \sum_{k \in \text{outputs}} w_{kh} \delta_k \]  \hspace{1cm} (8)
          iii) Update the network weight on the \( nth \) iteration and add momentum \( \alpha \) at the \( (n-1)th \) iteration \( (0 \leq \alpha < 1) \)
              \[ w_{ji} \leftarrow w_{ji} + \Delta w_{ji}(n) \]
              \[ \Delta w_{ji} \leftarrow \eta \delta_j x_{ji} + \alpha \Delta w_{ji}(n-1) \]  \hspace{1cm} (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
             \[ \Delta w_{ji} \leftarrow 1.05\eta \delta_j x_{ji} + \alpha \Delta w_{ji}(n-1) \]  \hspace{1cm} (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 $P(h|D)$ 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, $P(h|D)$ from the prior probability, $P(h)$ together with $P(D)$ and $P(D|h)$.

Mathematically, Bayes’rule states that:

$$p_{\text{osterior}} = \frac{\text{likelihood} \ast \text{prior}}{\text{marginal likelihood}}$$

(11)

This follows from Baye theorem

$$P(h|D) = \frac{P(D|h)P(h)}{P(D)}$$

(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 posterior probability of each candidate hypothesis.

Precisely, $h_{\text{MAP}}$ is a MAP hypothesis provided

$$h_{\text{MAP}} = \arg \max_{h \in H} P(h|D)$$

$$= \arg \max_{h \in H} \frac{P(D|h)P(h)}{P(D)}$$

$$= \arg \max_{h \in H} P(D|h)P(h)$$

(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.

\[
v_{MAP} = \arg \max_{v_j \in V} P(v_j | a_1, a_2, ......, a_n)
\]  

Using Bayes theorem, the expression becomes

\[
v_{MAP} = \arg \max_{v_j \in V} \frac{P(a_1, a_2, ......, a_n | v_j)P(v_j)}{P(a_1, a_2, ......, a_n)}
\]  

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

\[
v_{NB} = \arg \max_{v_j \in V} P(v_j) \prod_i P(a_i | v_j)
\]

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:

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

(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 $G$:

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

(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 \( TR \) like that of NSE data set, while this performance model is concerned with finding model \( f = f_x \) from two training sets, \( \{ TR_1 \text{ and } TR_2 \} \), 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 ‘2-1’ prediction.
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 1 in 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

Appendix

Figure 1: The proposed framework for the Artificial Neural and Bayesian Network
Forecasting Movement of the Nigerian Stock Exchange

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

<table>
<thead>
<tr>
<th>DATE</th>
<th>OPEN</th>
<th>HIGH</th>
<th>LOW</th>
<th>CLOSE</th>
<th>VOLUME</th>
<th>ALL SHARE INDEX</th>
</tr>
</thead>
<tbody>
<tr>
<td>3/1/2005</td>
<td>fall</td>
<td>fall</td>
<td>fall</td>
<td>fall</td>
<td>rise</td>
<td>fall</td>
</tr>
<tr>
<td>3/2/2005</td>
<td>fall</td>
<td>fall</td>
<td>fall</td>
<td>fall</td>
<td>rise</td>
<td>fall</td>
</tr>
<tr>
<td>3/3/2005</td>
<td>fall</td>
<td>fall</td>
<td>fall</td>
<td>fall</td>
<td>rise</td>
<td>fall</td>
</tr>
<tr>
<td>3/7/2005</td>
<td>fall</td>
<td>rise</td>
<td>rise</td>
<td>rise</td>
<td>fall</td>
<td>rise</td>
</tr>
<tr>
<td>3/8/2005</td>
<td>rise</td>
<td>fall</td>
<td>rise</td>
<td>rise</td>
<td>fall</td>
<td>rise</td>
</tr>
<tr>
<td>3/9/2005</td>
<td>rise</td>
<td>fall</td>
<td>rise</td>
<td>fall</td>
<td>rise</td>
<td>fall</td>
</tr>
<tr>
<td>3/10/2005</td>
<td>fall</td>
<td>rise</td>
<td>fall</td>
<td>rise</td>
<td>rise</td>
<td>fall</td>
</tr>
<tr>
<td>3/11/2005</td>
<td>fall</td>
<td>fall</td>
<td>rise</td>
<td>rise</td>
<td>fall</td>
<td>rise</td>
</tr>
<tr>
<td>3/14/2005</td>
<td>rise</td>
<td>rise</td>
<td>rise</td>
<td>rise</td>
<td>fall</td>
<td>fall</td>
</tr>
<tr>
<td>3/15/2005</td>
<td>rise</td>
<td>rise</td>
<td>fall</td>
<td>fall</td>
<td>rise</td>
<td>fall</td>
</tr>
<tr>
<td>3/16/2005</td>
<td>fall</td>
<td>fall</td>
<td>fall</td>
<td>fall</td>
<td>rise</td>
<td>fall</td>
</tr>
<tr>
<td>3/17/2005</td>
<td>fall</td>
<td>fall</td>
<td>fall</td>
<td>fall</td>
<td>fall</td>
<td>fall</td>
</tr>
<tr>
<td>3/18/2005</td>
<td>rise</td>
<td>rise</td>
<td>rise</td>
<td>rise</td>
<td>rise</td>
<td>rise</td>
</tr>
<tr>
<td>3/21/2005</td>
<td>fall</td>
<td>fall</td>
<td>fall</td>
<td>fall</td>
<td>fall</td>
<td>fall</td>
</tr>
<tr>
<td>3/22/2005</td>
<td>fall</td>
<td>rise</td>
<td>fall</td>
<td>rise</td>
<td>rise</td>
<td>fall</td>
</tr>
<tr>
<td>3/23/2005</td>
<td>rise</td>
<td>rise</td>
<td>fall</td>
<td>fall</td>
<td>fall</td>
<td>fall</td>
</tr>
<tr>
<td>3/24/2005</td>
<td>fall</td>
<td>fall</td>
<td>fall</td>
<td>fall</td>
<td>fall</td>
<td>Fall</td>
</tr>
</tbody>
</table>
Table 2: Conditional Probability for the Independent Variables Open rice, Close price, High price, Low price and Volume

<table>
<thead>
<tr>
<th>OPEN PRICE</th>
<th>CLOSE PRICE</th>
<th>HIGH PRICE</th>
<th>LOW PRICE</th>
<th>VOLUME</th>
</tr>
</thead>
<tbody>
<tr>
<td>rise</td>
<td>fall</td>
<td>rise</td>
<td>fall</td>
<td>rise</td>
</tr>
<tr>
<td>0.525</td>
<td>0.475</td>
<td>0.515</td>
<td>0.485</td>
<td>0.495</td>
</tr>
</tbody>
</table>

Table 3: Conditional Probability for the All-Share Index

<table>
<thead>
<tr>
<th>INDEX</th>
<th>OPEN</th>
<th>HIGH</th>
<th>LOW</th>
<th>CLOSE VOLUME</th>
<th>rise</th>
<th>fall</th>
</tr>
</thead>
<tbody>
<tr>
<td>rise</td>
<td>rise</td>
<td>rise</td>
<td>rise</td>
<td>rise</td>
<td>0.163</td>
<td>0.09</td>
</tr>
<tr>
<td>rise</td>
<td>rise</td>
<td>rise</td>
<td>rise</td>
<td>fall</td>
<td>0.106</td>
<td>0.069</td>
</tr>
<tr>
<td>rise</td>
<td>rise</td>
<td>rise</td>
<td>fall</td>
<td>rise</td>
<td>0.016</td>
<td>0.021</td>
</tr>
<tr>
<td>rise</td>
<td>rise</td>
<td>fall</td>
<td>fall</td>
<td>fall</td>
<td>0.016</td>
<td>0.041</td>
</tr>
<tr>
<td>rise</td>
<td>fall</td>
<td>rise</td>
<td>fall</td>
<td>rise</td>
<td>0.033</td>
<td>0.014</td>
</tr>
<tr>
<td>rise</td>
<td>fall</td>
<td>fall</td>
<td>fall</td>
<td>rise</td>
<td>0.16</td>
<td>0.014</td>
</tr>
<tr>
<td>rise</td>
<td>fall</td>
<td>rise</td>
<td>rise</td>
<td>rise</td>
<td>0.024</td>
<td>0.014</td>
</tr>
<tr>
<td>rise</td>
<td>fall</td>
<td>rise</td>
<td>rise</td>
<td>fall</td>
<td>0.033</td>
<td>0.014</td>
</tr>
<tr>
<td>rise</td>
<td>fall</td>
<td>rise</td>
<td>fall</td>
<td>rise</td>
<td>0.016</td>
<td>0.028</td>
</tr>
<tr>
<td>rise</td>
<td>fall</td>
<td>rise</td>
<td>fall</td>
<td>fall</td>
<td>0.041</td>
<td>0.021</td>
</tr>
<tr>
<td>rise</td>
<td>fall</td>
<td>fall</td>
<td>rise</td>
<td>rise</td>
<td>0.008</td>
<td>0.007</td>
</tr>
<tr>
<td>rise</td>
<td>fall</td>
<td>fall</td>
<td>fall</td>
<td>rise</td>
<td>0.008</td>
<td>0.055</td>
</tr>
<tr>
<td>rise</td>
<td>fall</td>
<td>fall</td>
<td>fall</td>
<td>fall</td>
<td>0.016</td>
<td>0.021</td>
</tr>
</tbody>
</table>

Table 4: Instances for Testing

<table>
<thead>
<tr>
<th>INSTANCES</th>
<th>EVIDENCES</th>
<th>INTEGRATION</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>OPEN</td>
<td>CLOSE</td>
</tr>
<tr>
<td>Instance 1</td>
<td>rise</td>
<td>rise</td>
</tr>
<tr>
<td>Instance 2</td>
<td>rise</td>
<td>fall</td>
</tr>
</tbody>
</table>

Table 5: The Testing Result

<table>
<thead>
<tr>
<th>Architecture</th>
<th>Learning rate $\eta$</th>
<th>Momentum rate $\alpha$</th>
<th>Training MSE</th>
<th>Testing MSE</th>
</tr>
</thead>
<tbody>
<tr>
<td>9-5-1</td>
<td>Between 0.01 and 0.05</td>
<td>0.005</td>
<td>9.9296e-7</td>
<td>4.78423-8</td>
</tr>
</tbody>
</table>
Table 6: Simulation Results of Performance of both ANN and BN Systems

<table>
<thead>
<tr>
<th>Date</th>
<th>Actual Movement</th>
<th>Bayesian Network</th>
<th>Neural Network</th>
</tr>
</thead>
<tbody>
<tr>
<td>12-Nov</td>
<td>Rise</td>
<td>Rise</td>
<td>Rise</td>
</tr>
<tr>
<td>13-Nov</td>
<td>Rise</td>
<td>Rise</td>
<td>Fall</td>
</tr>
<tr>
<td>14-Nov</td>
<td>Fall</td>
<td>Rise</td>
<td>Rise</td>
</tr>
<tr>
<td>15-Nov</td>
<td>Rise</td>
<td>Fall</td>
<td>Rise</td>
</tr>
<tr>
<td>16-Nov</td>
<td>Rise</td>
<td>Rise</td>
<td>Rise</td>
</tr>
<tr>
<td>19-Nov</td>
<td>Rise</td>
<td>Rise</td>
<td>Fall</td>
</tr>
<tr>
<td>20-Nov</td>
<td>Rise</td>
<td>Rise</td>
<td>Rise</td>
</tr>
<tr>
<td>21-Nov</td>
<td>Rise</td>
<td>Rise</td>
<td>Rise</td>
</tr>
<tr>
<td>22-Nov</td>
<td>Rise</td>
<td>Fall</td>
<td>Fall</td>
</tr>
<tr>
<td>23-Nov</td>
<td>Rise</td>
<td>Rise</td>
<td>Rise</td>
</tr>
<tr>
<td>26-Nov</td>
<td>Fall</td>
<td>Fall</td>
<td>Fall</td>
</tr>
<tr>
<td>27-Nov</td>
<td>Fall</td>
<td>Rise</td>
<td>Rise</td>
</tr>
<tr>
<td>28-Nov</td>
<td>Fall</td>
<td>Fall</td>
<td>Fall</td>
</tr>
<tr>
<td>29-Nov</td>
<td>Fall</td>
<td>Fall</td>
<td>Fall</td>
</tr>
<tr>
<td>30-Nov</td>
<td>Rise</td>
<td>Rise</td>
<td>Rise</td>
</tr>
<tr>
<td>03-Dec</td>
<td>Rise</td>
<td>Rise</td>
<td>Rise</td>
</tr>
<tr>
<td>04-Dec</td>
<td>Fall</td>
<td>Rise</td>
<td>Rise</td>
</tr>
<tr>
<td>05-Dec</td>
<td>Fall</td>
<td>Fall</td>
<td>Fall</td>
</tr>
<tr>
<td>06-Dec</td>
<td>Fall</td>
<td>Fall</td>
<td>Rise</td>
</tr>
<tr>
<td>07-Dec</td>
<td>Rise</td>
<td>Rise</td>
<td>Fall</td>
</tr>
<tr>
<td>10-Dec</td>
<td>Fall</td>
<td>Fall</td>
<td>Rise</td>
</tr>
<tr>
<td>11-Dec</td>
<td>Rise</td>
<td>Rise</td>
<td>Rise</td>
</tr>
<tr>
<td>12-Dec</td>
<td>Fall</td>
<td>Rise</td>
<td>Rise</td>
</tr>
<tr>
<td>13-Dec</td>
<td>Fall</td>
<td>Fall</td>
<td>Rise</td>
</tr>
<tr>
<td>14-Dec</td>
<td>Fall</td>
<td>Fall</td>
<td>Fall</td>
</tr>
<tr>
<td>17-Dec</td>
<td>Fall</td>
<td>Rise</td>
<td>Fall</td>
</tr>
<tr>
<td>18-Dec</td>
<td>Fall</td>
<td>Rise</td>
<td>Fall</td>
</tr>
<tr>
<td>21-Dec</td>
<td>Rise</td>
<td>Rise</td>
<td>Rise</td>
</tr>
<tr>
<td>24-Dec</td>
<td>Rise</td>
<td>Rise</td>
<td>Fall</td>
</tr>
<tr>
<td>27-Dec</td>
<td>Rise</td>
<td>Rise</td>
<td>Fall</td>
</tr>
<tr>
<td>28-Dec</td>
<td>Rise</td>
<td>Rise</td>
<td>Rise</td>
</tr>
<tr>
<td>31-Dec</td>
<td>Rise</td>
<td>Rise</td>
<td>Fall</td>
</tr>
</tbody>
</table>
Table 6: Performance of the ANN and BN Models

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