STOCK MARKET PRICE PREDICTION USING ARTIFICIAL NEURAL NETWORK: AN APPLICATION TO THE KENYAN EQUITY BANK SHARE PRICES

J. M. Kihoro and E. L. Okango
Department of Statistics and Actuarial Sciences, Jomo Kenyatta University of Agriculture and Technology, Nairobi, Kenya
Email: kangphas@gmail.com

Abstract
This paper looks at the application of the artificial neural networks (ANN) in predicting stock market prices in Kenya. In particular the paper looks at the application of ANN in predicting future Equity Bank share prices using historical data. We have assumed that only previous prices affect future prices, then fitted ARIMA models to the stock prices data in order to identify the best input lags into the ANN model. The best combination of lags was taken for input lags and led to optimal result in terms of the least mean squared error between the predicted values and the test data. The 3-3-1 network architecture gave the best results in terms of the Mean Squared error. The paper demonstrates that artificial neural networks can effectively model local stock market prices for reliable forecasts.

Key words: stock market, price prediction, artificial neural networks, Equity Bank

1.0 Introduction

Stock market price prediction is not an easy task and this is because of the random walk behavior of the stock prices. Several classes of prediction techniques exist and they include: technical analysis, fundamental analysis, traditional time series methods and the machine learning techniques. The artificial neural networks fall under the machine learning techniques. Traditional approaches such as moving averages, exponential smoothing and linear regression methods have been used in predicting stock market prices. Patel and Marwala (2006) used the multi-layer perceptron (MLP) and the Radial Basis Function (RBF) neural architectures in forecasting the closing price indices in the Johannesburg stock exchange. Other advanced methods that have been applied in solving problems of predicting future stock prices include: Markov models, genetic algorithm and fuzzy logic.

The most important factor to be considered in construction of an artificial neural network is what the network will learn. We would like to determine whether to buy or sell securities based on the past market indicators. The uphill task is determining which indicators and input data will be used so that the system is trained appropriately. The input data may be raw data ranging from value, price, technical indicators such as moving averages, and trend line indicators.

Yoon and Swales (1993) used phrases from the president's speech to shareholders as input. The input data should allow the network to generalize market behavior while containing limited redundant data. The Johannesburg Stock Exchange-system by Robert (1996) modeled the performance of the Johannesberg stock exchnage and had 63 indicators but after training, the analysis showed that many of the inputs were unnecessary. They employed cross-validation techniques and sensitivity analysis to discard 20 input values thereby reducing the number of hidden nodes from 21 to 14 with negligible effects on the system.

This paper seeks to address the prediction of stock prices using ANN with historical data (past prices) and their lags as the input data. We also seek to explain the choice of lags as the input data and the network architecture. A network with more weights models a complex function and is therefore prone to over-fitting while a network with less weights may not be sufficiently powerful to model the underlying function. In particular, a network with no hidden layers actually models a simple linear function.

The statistical approach to forecasting involves the construction of stochastic models to predict the value of an observation X_{t+d} using previous observation. This is often accomplished using linear stochastic difference equation models with random input. Before we present our results, we give a brief review of the class of such models and also a review of the artificial neural network.

2.0 Standard Time series Models

2.1 Autoregressive Model

Also known an infinite impulse response filter (IIR), it is based on the idea that a present value can be forecasted with respect to the previous p values where p is the number of steps one has to go back to the past in order to predict the present value. The autoregressive order of order p, AR(p) is of the form:

$$X_t=\mu+\beta_1 X_{t-1}+\beta_2 X_{t-2}+...+\beta_p X_{t-p}+\varepsilon_t \quad (1)$$

Where $\beta_1,\beta_2,...,\beta_p$ are constants, μ is the mean of the series and ε_t is a white noise with mean 0 and variance σ^2. Therefore an Autoregressive model of order one AR(1) can be written as:

$$X_t=\mu+\beta_1 X_{t-1}+\varepsilon_t \quad (2)$$

2.2 Moving Average Model

Moving averages are very useful in identifying the trend of the time series. A moving average of order q, MA(q) written:

$$X_t=\mu+\sum_{i=1}^{q}\Theta_i\varepsilon_{t-i}+\varepsilon_t \quad (3)$$

where $\Theta_1,\Theta_2,...,\Theta_q$ are parameters of the model, ε_t is a white noise with mean 0 and variance σ^2 calculates each term of the time series from the moving average of the last i terms of the error sequence. Therefore the moving average is also a linear function and stationary process too.

2.3 Autoregressive Moving Average Model

When an Autoregressive model of order p is combined with a moving average model of order *q*, the result is an autoregressive moving Average of order (*p*,*q*). This model was introduced by Box and Jenkins (1976) for forecasting time series. It is written as:

$$X_t=\mu+\varepsilon_t+\sum_{i=1}^{p}\beta_i X_{t-1}+\sum_{j=1}^{q}\Theta_i\varepsilon_{t-i} \quad (4)$$

Clearly the current value in an ARMA model is a combination of the past values and the past errors. All the models discussed previously need the data to be stationary. If the time series is not stationary, one can make it stationary by differencing it. Box and Jenkins developed a generalization of ARMA model, the autoregressive integrated moving average which uses differenced time series.

2.4 Autoregressive Integrated Moving Average Model

The Autoregressive Integrated Moving Average (ARIMA) model is very effective and flexible in non-stationary and continuous data. An Autoregressive Integrated Moving Average of order(p,d,q) is defined as:

$$\Delta^d Y_t = \beta_1 \Delta^d Y_{t-1} + \beta_2 \Delta^d Y_{t-2} + \ldots + \beta_p \Delta^d Y_{t-p} + \sum_{j=1}^{q} \Theta_j \varepsilon_j \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots(5)$$

Where:

p is the order of the Autoregressive part
d is the order of the differencing needed to stationarize the data.
q is the order of the moving average part.

The ARIMA model is used for prediction of non-stationary time series when linearity between variables is supposed. In many practical situations however, linearity does not hold and therefore ARIMA models do not produce accurate results when used for capturing and explaining nonlinear relations of many real world problems. Real world time series and especially stock market prices fluctuate with seasonal patterns. When seasonality presents a dynamic pattern, ARIMA models become ineffective. Therefore ARIMA (p,d,q) can be extended forming the seasonal ARIMA(p,d,q)(P,D,Q) denoted SARIMA(p,d,q)(P,D,Q) where p,d,q are model parameters and P,D,Q are seasonal parameters, and is given by:

$$\varphi_p(B)\Phi_P(B)x_t = \Theta_q(B)\tau_Q(B^s)\varepsilon_t \quad (6)$$

where $\phi_p, \Phi_P, \Theta_q, \tau_Q$ are polynomials of order p,P,q,Q respectively and , $x_t = \Delta^d \Delta_s^D Y_t$ is the the original series differenced d times and seasonaly differenced D times. Identification of ARIMA model is done by analyzing the auto-correlation function (ACF) and partial auto-correlation function (PACF) or alternatively by AIC (Akaike's information criterion), Bayesian or Schwartz information criterion or by FPE (Final prediction error criterion).

3.0 Artificial Neural Network

3.1 Network architecture

ANN is a parallel connection of a set of nodes called neurons (weights) whose functionality is loosely based on the animal neuron. The ability of neural networks to discover nonlinear relationships in input data makes them ideal for modeling nonlinear dynamic systems such as the stock market. An ANN has an input, one or more hidden layers, and an output layer. The network architecture is as shown below.

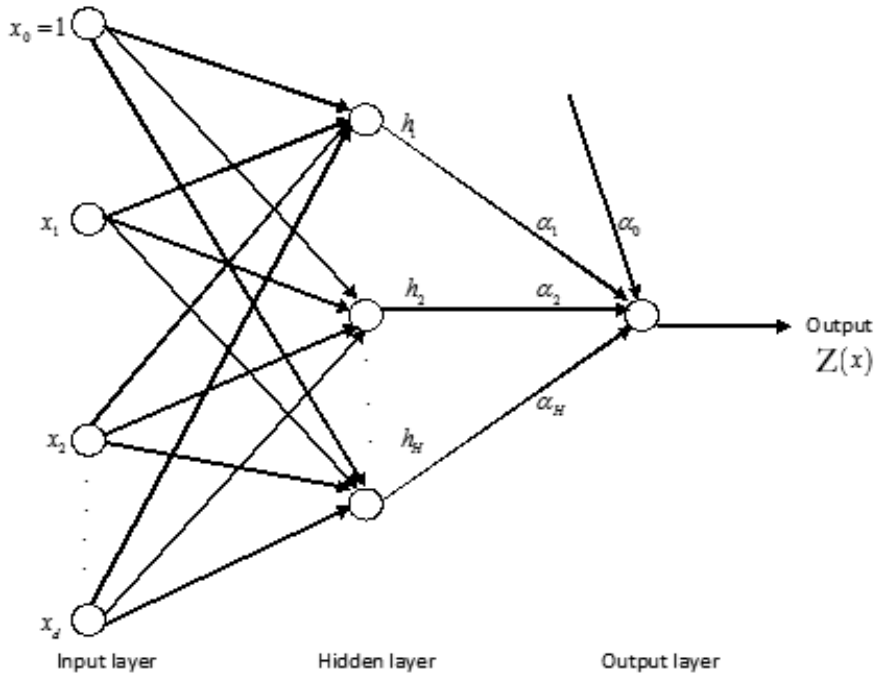

Figure 1: Neural network architecture

In the above input-output architecture, we have d input nodes, one layer of H hidden nodes and an activation function $\varphi(x)$. The input at the hidden layer nodes are connected by weights $W_{h,j}$, for $h\in(1,2,...,H)$ and $j\in(0,1,2,...,d)$ where $W_{h,0}$ is the bias for the i^{th} hidden node.The hidden and the output layers are connected by weights α_h for $h\in(0,1,...,H)$. Considering an input vector $X=(x_1,x_2,...,x_d)\in\Re^d$ the input $V_h(x)$ to the h^{th} hidden node is the value;

$$V_h(x)=W_{h,0}+\sum_{j=1}^{d} W_{h,j}x_j \quad \text{.....(7)}$$

The output $\phi_h(x)$ of the h^{th} hidden node is;

$$\phi_h(x)=\varphi(V_h(x)) \quad \text{.....(8)}$$

The net input to the output node is the value;

$$P(x)=\alpha_0+\sum_{h=1}^{H} \alpha_h\phi_h(x) \quad \text{.....(9)}$$

Finally the output $Z(x)$ of the net is the value;

$$Z(x)=\psi(P(x))$$

The connection weights are adjusted through training. There exist two training paradigms: non supervised and supervised training. In this study we discuss and later apply supervised training. The supervised training of a neural network requires the following.

1. A sample of d input vectors $X=(x_1,x_2,...,x_d)$ and an associated output $Y=(y_1,y_2,...,y_d)$

2. The selection of an initial weight set.

3. A repetitive method to update the current weights to optimize the input output map.

3.2 Choice of Inputs into the Network

The choice of inputs is a vital step in model development especially in the data driven technique such as the artificial neural networks. Some researchers have used past values or their transformation as input variables into the neural networks. Nerraj (2005) studied the efficacy of neural networks in modeling the Bombay Stock Exchange SENSEX weekly closing values. They developed two neural networks, which are denoted as NN1 and NN2. NN1 takes as its inputs the weekly closing value, 52-week Moving Average of the weekly closing SENSEX values, 5-week Moving Average of the same, and the 10-week Oscillator for the past 200 weeks. NN2 takes as its inputs the weekly closing value, 52-week Moving Average of the weekly closing SENSEX values, 5-week Moving Average of the same, and the 5-week volatility for the past 200 weeks. Yim (2002) mapped lagged returns to current returns by using three neural networks with a back propagation algorithm in trying to predict the Brazilian daily index returns. The first neural network had nine lagged returns in the input layer (lags 1 to 9), the second one had only four neurons in the input layer (lags 1, 2, 5 and 9) and the third one having only two neurons (lags 1 and 9) produced the best results. Since our data is univariate, we use lags as the input to the network. To determine which lags to include in an input set *X* of variables, autocorrelations and partial autocorrelations analysis together with AIC and its variation have been used, but they have not been very helpful Kihoro (2004) et al. An input selection method referred to as automatic relevance determination (ARD) by Mackay (1992), and Neal (1996), is based on a regression problem with many input variables. Input variables of a neural network should not be much correlated because the correlated input variables may degrade the prediction performance by interacting with each other as well as other elements and producing a biased effect. The autocorrelation coefficient of a series y_t at lag k is given by:

$$r_k = \sum_{t=k+1}(y_t-\bar{y})(y_{t-k}-\bar{y}) / \sum_{t=1}(y_t-\bar{y})^2 \quad \text{.....(10)}$$

Therefore lags with least autocorrelations can be conveniently used as inputs into the network. We fitted ARIMA models to the data and performed correlations on the fitted values. The pairs (lags) with least correlations were used as inputs into the network and the results were compared with those with the highest correlations and also with the results when all the lags were used as input into the model.

4 .0 Empirical Results

4.1 The Data

The data was obtained from the Nairobi's securities exchange financial and investment segment, comprising 487 daily share prices for Equity Bank. The volume weighted average price(VWAP) is obtained by dividing the turnover per counter by the total share traded. However there were some missing values as trading did not occur in some days. The missing values were imputed. Imputation was done by taking the average of the previous week's and following week's share prices. This brought the total number of daily prices to 498.

4.2 Properties of the data

We looked at some properties of the data before fitting. The properties are illustrated in the figures below.

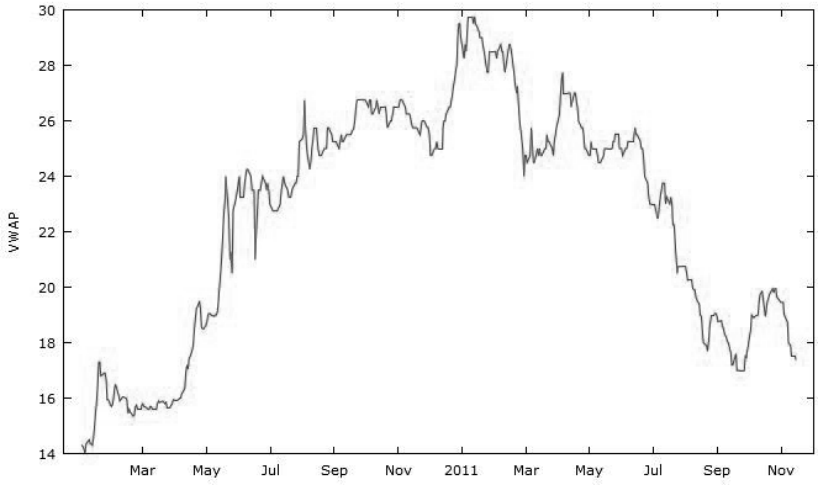

Figure 2: Plot of Equity Bank share prices

Figure 2 above shows that the Equity Bank share price has been on the upward trend until the setting in of the year 2011. The prices are not mean reverting as it should be. Afterwards the prices plummet in 2011. This is reflected in the overall dwindling of the NSE 20 share index and the all share index, caused by tough economic times. The NSE was the worst performing bourse in Africa in 2011. The highest share price recorded during this period is 29.75 shillings and the lowest was 14 shillings.

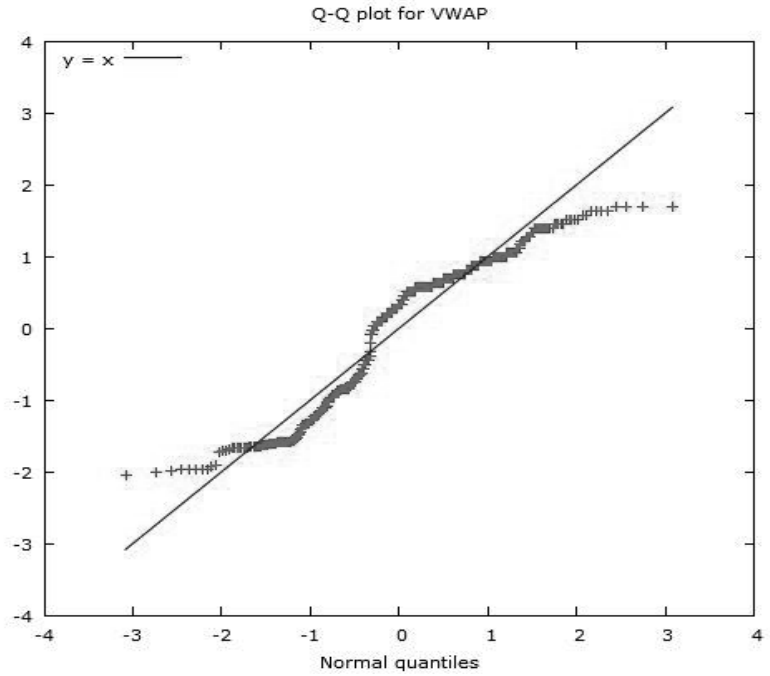

Figure 3: Q-Q plot of the Equity bank share prices

From the normal Q-Q plot we establish that the data is not normally distributed.

4.2 Forecasting

First the 498 values are split into two, the first 448 values (90 percent) were used to train the network and the remaining 50 (10 percent) were used for testing. We employed a feed forward network comprising an input layer, a hidden layer and an output layer. The input layer had a varying number of inputs i.e. the first model had 10 inputs(the lags), the second network had three lags with three largest correlations as input and the third network had also three lags with three smallest correlations as input. The best network was the one that gave the least mean squared error. The output layer contains a neuron, with a single output corresponding to a category of dependent variable. Connecting the input and the output layer is the hidden layer. In our study, we considered a single hidden layer with a varying number of neurons. The number of hidden neurons affects how well the network is able to separate the data. The larger the number of hidden neurons, the better the learning and the network is able to correctly predict the data it has been trained on. This comes with cost as its performance on new data and ability to generalize is compromised.

4.3 Fitting ANN to Equity Bank Data

Case 1:10 inputs (10-3-1 architecture)

Here all the 10 lags were used as inputs into the network and the following results were obtained.

Table 1: Table of mean squared error of 10-3-1 network

Training examples	438	Reserved data 10%	48
No of inputs	10	Total train MSE	0.004321
No of layers	1	Total test MSE	0.006342
No of neurons	3		
No of outputs	1		
No of predictions	48		
Noise %	0		

We observe that with 10 inputs, one hidden layer with three neurons, the mean squared error both for training and testing is acceptable. The train MSE is obtained by getting the mean squared error between the actual values and the fitted values. The test MSE is obtained by getting the mean squared error between the forecast and the test data. A plot of the fitted values superimposed on the predicted values depicts an almost perfect fit as shown below.

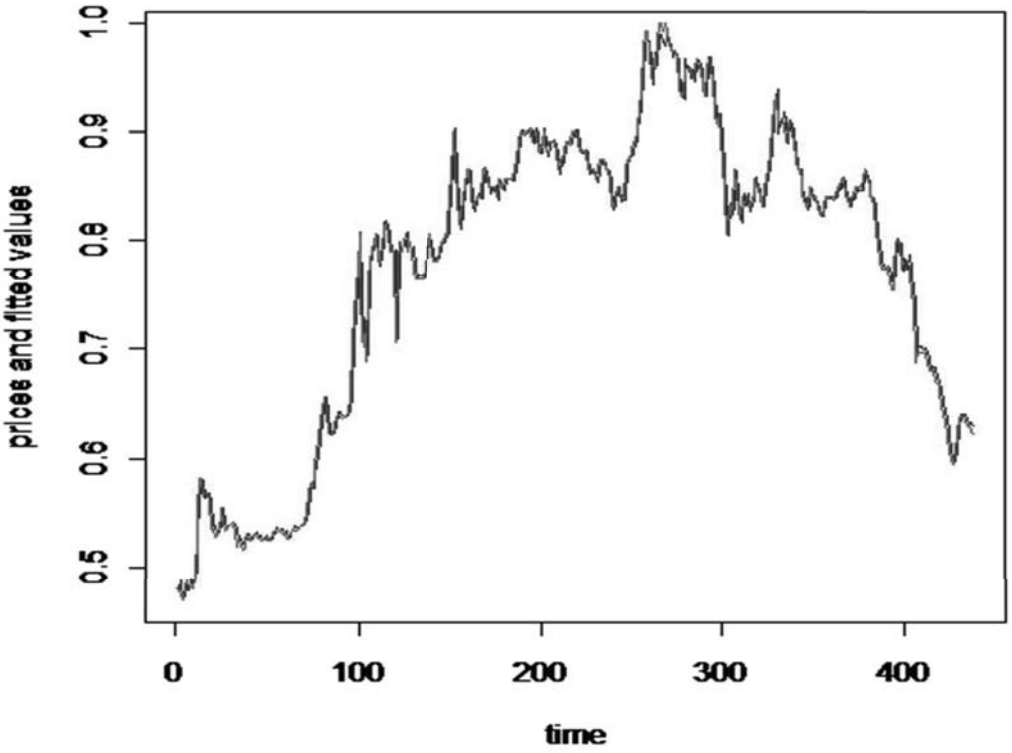

Figure 4: Plot of superimpose of fitted values on prices

4.3.1 Case 2:3 Inputs (3-3-1 architecture)

Here 3 lags with three highest correlation are used as inputs into the network.

Table 2: Mean squared error of 3-3-1 network (three highest correlated input lags)

Training examples	438	Reserved data 10%	48
No of inputs	10	Total train MSE	0.005321

No of layers	1	Total test MSE	0.008342
No of neurons	3		
No of outputs	1		
No of predictions	48		
Noise %	0		

4.3.2 Case 3:3 Inputs (3-3-1 architecture)

Here 3 lags with three least correlations are used as inputs into the network.

Table 3: Mean squared error of 3-3-1 network (least highest correlated input lags)

Training examples	438	Reserved data 10%	48
No of inputs	10	Total train MSE	0.003321
No of layers	1	Total test MSE	0.003842
No of neurons	3		
No of outputs	1		
No of predictions	48		
Noise %	0		

It is observed that the MSE is slightly for the 3-3-1 architecture with least highest correlated input lags. We used this architecture (3-3-1) to obtain a 15 day ahead forecast and the values are shown in the table below.

Table 4: Table of 15 day ahead price forecast

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
17.4	17.3	17.1	16.9	16.85	16.85	16.7	16.6	16.75	16.9	17.1	17.4	17.6	18	18.1

The plot of the forecast with the past prices is as shown below.

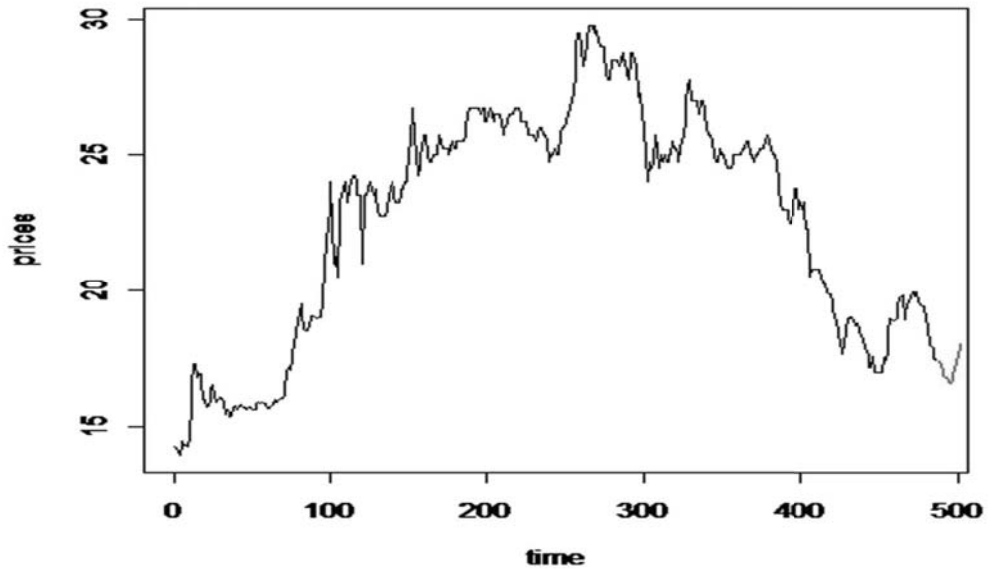

Figure 5: Plot of 15 days-ahead price forecast

5 .0 Conclusion

We have observed that artificial neural networks can effectively model the stock market prices. Their ability to discover non linear relationship in data is evidenced in the fact that the mean squared error between the predicted and the desired share prices is very minimal. The selection of input variables also plays a vital role as they would determine what the network will learn and hence predict. The idea of garbage in garbage out could never be more manifest in any other field than in an artificial neural network. In our study we have observed that the MSE reduces when lags with low correlations are used. The 3-3-1 network architecture gave the best results in terms of the mean squared error.

In this study, we only used the past prices, and in particular lags as the input into the network. We also dwelt on one company: Equity Bank. It would be interesting if study could be extended to other companies while incorporating other inputs to the network.

References

Box, G. and Jenkins G. (1976), Time analysis forecasting and control., San Francisco Holden Day.

Kihoro, J., Otieno R. and Wafula C. (2004), 'Seasonal time series forecasting: A comparative study of ARIMA and ANN models.', African Journal of Science and Technology (AJST) **5**, pp. 41–49.

MacKay, D. J. C. (1992). Bayesian interpolation, Neural Computation, **4(3)**, pp. 415-447.

Neal, R. M. (1996). Bayesian Learning for Neural Networks. New York: Springer.

Neeraj, M. (2005), 'Artificial neural network models for forecasting stock price index in Bombay Stock Exchange', Indian Institute of Management Ahmedabad, Research and Publication Department. .

Patel, P. and Marwala. T (2006), 'Neural networks, fuzzy inference systems and. adaptive neuro fuzzy inference systems for financial decision making', Proceeding ICONIP'06 Proceedings of the 13th international conference on neural information processing **3**, pp. 430–439.

Robert, J. (1996), The application of neural networks in the forecasting of share prices., Finance and Technology Publishing.

Yim, J. (2002), Comparison of neural networks with time series models for forecasting returns on a stock market index, Technical report, Springer-Verlag Berlin Heidelberg.

Yoon, Y. and G. Swales. (1993), Neural Networks in Finance and Investing., Probus Publishing Company.