PRIMER ON USING NEURAL NETWORKS FOR FORECASTING MARKET VARIABLES

Shaikh A. Hamid
Associate Professor of Finance
Southern New Hampshire University

Working Paper No. 2004-03

School of Business, Southern New Hampshire University, 2500 N. River Road, Manchester, NH 03106

PRIMER ON
USING NEURAL NETWORKS FOR FORECASTING MARKET VARIABLES

Abstract

Ability to forecast market variables is critical to analysts, economists and investors. Among other uses, neural networks are gaining in popularity in forecasting market variables. They are used in various disciplines and issues to map complex relationships.

We present a primer for using neural networks for forecasting market variables in general, and in particular, forecasting volatility of the S&P 500 Index futures prices. We compare volatility forecasts from neural networks with implied volatility from S&P 500 Index futures options using the Barone-Adesi and Whaley (BAW) model for pricing American options on futures. Forecasts from neural networks outperform implied volatility forecasts. Volatility forecasts from neural networks are not found to be significantly different from realized volatility. Implied volatility forecasts are found to be significantly different from realized volatility in two of three cases.

Keywords: Neural networks, Volatility forecasting, Implied standard deviation, Realized standard deviation.

PRIMER ON
USING NEURAL NETWORKS FOR FORECASTING MARKET VARIABLES

INTRODUCTION

Accurate forecasting of market variables is critical to economists, analysts, and investors. This task gets complex as world financial markets get increasingly interconnected and interdependent. This complexity has created opportunities for neural networks which have the ability to explore interrelationships among a large number of market variables. Hence they are gaining popularity.

Though neural networks have been around for almost half a century, only since the late 1980s they have gained significant use in scientific and technical use. They have found applications in wide ranging fields (Exhibit 1).

Neural networks have gained use in economics and finance more recently. The networks have been used in issues like economic prediction, stock picking, portfolio construction, identifying insider trading, analyzing corporate financial health, bond risk assessment, recognizing financial distress, detecting credit card fraud, improving real estate appraisal, identifying good credit or insurance candidates, exchange rate prediction, valuing options, commodity trading.

Back in 1990, Hawley, Johnson and Raina identified various potential uses of neural networks in corporate finance, financial institutions and investments. In the last fourteen years, the extent of use of the networks in finance has indeed increased. Exhibit 2 presents a list of research works on the application of neural networks in economics and finance. If the exhibit is an indicator, the research works involving the technology have been devoted predominantly to two areas: financial distress prediction (about 20 percent of the listed studies) and prediction of

stock price/stock index (about 12 percent). Neural networks have yet to see other potential applications in financial economics. Hawley, Johnson and Raina (1990) mention a number of potential uses of neural networks on which academic research is yet to be seen. Exhibit 3 lists some of the areas in which the efficacy of using neural networks could be researched. For example, the technology can be applied to corporate finance (financial simulation, prediction, evaluation, etc.), IPO pricing, identifying arbitrage opportunities, security risk profiling, locating tax evaders, etc. Evaluation of uses in these areas is yet to be seen.

NEURAL NETWORKS IN FINANCIAL APPLICATIONS

The general finding from the studies in Exhibit 2 by and large is that neural networks have promising applications in many fields of economics and finance.

Many of the studies on early warning failure prediction studies compared the predictive powers of neural networks and conventional statistical models like multiple discriminant analysis and logistic regression. A number of studies found neural networks to be superior to these models (e.g., Coats and Fant, 1993; Lenard, Alam and Madey, 1995; Fletcher and Goss, 1993; Salchenberger, Cinar and Lash, 1992). Yet other studies found both approaches yield balanced degree of accuracy (Altman, Marco and Varetto, 1994; Boritz, Kennedy, de Mirande e Albuquerque, 1995; Yang, Platt and Platt, 1999). Boritz and Kennedy (1995) show that the performance of neural networks is sensitive to the choice of variables selected and that the networks cannot be relied upon to evaluate and focus on the most important variables. Zurada, Foster, Ward, and Barker (1998) find neural networks are not superior to logistic regression models for traditional dichotomous response variables, but are superior for more complex financial distress response variables.

Neural networks have been employed with success to make stock market predictions and stock selection (e.g., White, 1988, Yoon and Swales, 1991; Kryzanowski, Galler and Wright, 1993; Rhee, 1994, Gencay, 1998; Qi, 1999; Qi and Maddala, 1999). The networks have been used to determine optimal buy and sell timing for an equity index (Kimoto, Asakawa, Yoda, and Takeoka, 1990) and recognize a specific price pattern, such as the Japanese "candlestick" triangle (Kamijo and Tanigawa, 1990).

Neural networks have been found to generate improved risk ratings of bonds (Dutta and Shekhar, 1988; Moody and Utans, 1991; Surkan and Singleton, 1991; Kim, Weistroffer and Redmond, 1993; Maher and Sen, 1997) and useful in mortgage risk assessment (Collins, Ghosh and Scofield, 1988; Reilly, Collins, Scofield and Ghosh, 1991; Grudnitski, Quang, and Shilling, 1995).

Vishwakarma, (1994) and Qi (2001) have found neural networks to be useful in predicting business cycle turning points. Three studies by Swanson and White (1995, 1997a,b) find that nonlinear neural networks are useful in economic time series forecasting of interest rates, unemployment, GNP, etc.

In prediction of corporate takeover targets, Sen and Gibbs (1994) found several neural network models map the data very well, but did not predict merger targets significantly better than logistic regression.

Furthermore, the technology has been found useful in other diverse applications like commodity trading (Kaastra and Boyd, 1995); exchange rate forecasting (Zhang, 1994; Kuan and Liu, 1995; Gencay, 1999); real estate valuation (Worzala, Lenk and Silva, 1995); option pricing (Hutchinson, Lo and Poggio, 1995; Garcia and Gencay, 2000); detection of management fraud (Fanning, Cogger and Srivastava, 1995); earnings forecast (Charitou and Charalambous, 1996); Kim, 1996).

In theory, neural networks are suitable for nonlinear problems. Zhang (2001) found that neural networks are quite effective in linear time-series modeling and forecasting. This implies that the technology can compete with linear models for linear problems.

Since the first draft of this article in 1995, neural networks have been used in volatility forecasting by some other authors. Ormoneit and Neuneier (1996) predict volatility of the German stock market using two types of networks. Donaldson and Kamstra (1997) have proposed the use of neural network-GARCH model to capture volatility effects in stock returns. González Miranda, and Burgess (1997) have used the networks to predict intraday volatilities for the Spanish stock market. Schittenkopf, Dorffner and Dockner (1998) predict the volatility of the Austrian stock market and find neural networks outperform ARCH models. Schittenkopf, Dorffner and Dockner (2000) use daily DAX data and find that volatility predictions from neural network are superior to GARCH models in that they have higher correlations with implied volatilities. Meissner and Kawano (2001) use a combined GARCH-neural network approach to capture the volatility smile of options on high-tech stocks.

In sum, neural networks have been found useful in many different types of applications in economics, finance and business. They have been found to outperform linear models in a variety of circumstances. However, the performance of the networks has not been consistent in all cases. They have been particularly effective in capturing complex relationships in which linear models fail to perform well (see White, 1989b, and Kuan and White, 1994).

The vast majority of the about 100 studies on which Exhibit 2 is constructed, were published in nonfinance journals and nonfinance researchers authored many of them. This can lead one to wonder if finance/economics academics may still find neural networks to be esoteric "black-box" enveloped in mystique. This article seeks to remove some of the mist. It explains in nontechnical terms what a neural network is and how a version of it works.

A better understanding of the technology will hopefully generate needed research in the many unresolved issues relating to its applications in economics and finance. Practitioners in greater numbers can reap the benefits of present and potential uses of the technology. (For some finance academics, the lack of formal theoretical foundation behind neural networks may be the reason to avoid it.)

This paper provides a primer on using neural networks for forecasting market variables in general, and in particular, forecasting volatility of the S&P 500 Index futures prices using over ten years of daily data on a number of input variables. Volatility forecasts from neural networks are compared with implied volatility forecasts using Barone-Adesi and Whaley (1987) American futures options pricing model. The forecasts from neural networks and the options pricing model are for similar horizons and time periods. Volatility forecasts from the two methods are then compared with realized volatility.

The next section briefly explains neural network. Then we describe neural network architecture and operation of possibly the most commonly used type of network for forecasting -- the back-propagation network. We then show how the networks can be used for forecasting volatility of the S&P 500 Index futures prices using data on a number of market variables. Finally, we present analysis of the results we obtain and compare against implied volatility forecasts and realized volatility.

WHAT IS A NEURAL NETWORK?

A neural network is a computational technique that benefits from techniques similar to ones employed in the human brain. It is designed to mimic the ability of the human brain to process data and information and comprehend patterns. It imitates the structure and operations of the three dimensional lattice of network among brain cells (nodes or neurons, and hence the term

"neural"). The technology is inspired by the architecture of the human brain, which uses many simple processing elements operating in parallel to obtain high computation rates. Similarly, the neural network is composed of many simple processing elements or neurons operating in parallel whose function is determined by network structure, connection strengths, and the processing performed at computing elements or nodes.

The learning process of the neural network can be likened to the way a child learns to recognize patterns, shapes and sounds, and discerns among them. For example, the child has to be exposed to a number of examples of a particular type of tree for her to be able to recognize that type of tree latter on. In addition, the child has to be exposed to different types of trees for her to be able to differentiate among trees.

The human brain has the uncanny ability to recognize and comprehend various patterns. The neural network is extremely primitive in this aspect. The network's strength, however, is in its ability to comprehend and discern subtle patterns in a large number of variables at a time without being stifled by detail. It can also carry out multiple operations simultaneously. Not only can it identify patterns in a few variables, it also can detect correlations in hundreds of variables. It is this feature of the network that is particularly suitable in analyzing relationships between a large number of market variables. The networks can learn from experience. They can cope with "fuzzy" patterns – patterns that are difficult to reduce to precise rules. They can also be retrained and thus can adapt to changing market behavior.

The network holds particular promise for econometric applications. Multilayer feedforward networks with appropriate parameters are capable of approximating a large number of diverse functions arbitrarily well (see White, 1989a). Even when a data set is noisy or has irrelevant inputs, the networks can learn important features of the data. Inputs that may appear

irrelevant may in fact contain useful information. The promise of neural networks lies in their ability to learn patterns in a complex signal.

Time series models are one of the more widely used approaches for prediction purpose. Although useful, time series models pose a problem in the very beginning. Identification of the model (autoregressive versus moving average, or a combination of the two) that will fit a particular time series of data, and the order specification that will be appropriate is the difficult first step in using time series models. Neural networks do not depend on assumptions regarding the data but adapt to the data (see Davies, 1995). Also, statistical models encounter difficulty when a data series is noisy. This happens to be the case with most financial market data -- they are hard to model or hide obvious pattern. Neural networks are adept at handling such data. They have performed well in a number of applications in which linear models fail to perform well (see White, 1989b, Kuan and White, 1994). Specially, when it comes to forecasting financial market variables characterized by nonstationarity, neural networks incorporating nonlinear regression models have distinct edge.

A neural network can be described as a type of multiple regression in that it accepts inputs and processes them to predict some output. Like a multiple regression, it is a data modeling technique.

Neural networks have been found particularly suitable in complex pattern recognition compared to statistical multiple discriminant analysis (MDA) since the networks are not subject to restrictive assumptions of MDA models (see Coats and Fant, 1993).

Network architecture

Network architecture deals with arrangement of neurons into layers and the connection patterns within and between layers. The type of problem to be solved has a great deal to do with

network architecture. The networks have been applied to solve problems in a wide variety of areas. Problem solving approaches in which the networks have been used include classification, filtering, pattern association, optimization, conceptualization and prediction. (Our concern in this paper is the last one). For each problem solving approach more than one architecture may be used. Each architecture goes with numerous variations depending on parameters selected. Network parameters vary in factors as the following:

- The **number of layers** in the network through which input variables are processed and the **number of neurons** or nodes per layer (neurons are the elements that process the inputs and learn about relationships between input and the output variables);
- **Connections between neurons** in each layer and the strength (weight) of each connection;
- **Transfer function**, through which the network seeks to relate the input data with the output data.

These factors are explained below.

Number of layers and neurons in each layer

It is helpful to visualize neurons as being grouped in layers. The number of layers will depend on the type and complexity of problem we explore. For prediction purpose, the most common form of architecture is probably the feedforward multi-layered network commonly termed back-propagation, or simply back-prop.

A typical architecture incorporating back-propagation is shown in Figure 1. Networks of this type have at least three layers (see Katz, 1995). The first is the input layer that presents data into the network. This layer has as many neurons as there are input categories.

The next layer is called the hidden layer because it is essentially hidden from the access of inputs and outputs. A network may have more than one hidden layer depending on the

complexity of the problem. The hidden layer(s), in conjunction with the input layer, creates an internal mapping of the input data. This process explores hidden patterns, correlations and causal relationships in the data set in order to make generalizations. For financial forecasting, typically one hidden layer may be sufficient to map input data to output data.

No analytical formula can tell us how many hidden layers to use. That will need experimentation. How many neurons to use in the hidden layer? Too few neurons prevent the network from correctly mapping inputs into outputs. Too many neurons may cause the network to "memorize" trivial patterns that can impair its ability to "assimilate" important features or trends and make proper generalizations. For example, suppose we want to predict the price of a stock using fifteen explanatory variables and so we use fifteen neurons in the input layer. If we use only five neurons in the hidden layer, we will not enable the network to exhaustively map the different ways in which the stock price might evolve. On the other hand, if we use fifty or forty neurons, we risk overloading the network so that it overlooks the nontrivial and captures trivial relationships. Mendelsohn (1993) suggests hidden neurons between half the number of input variables and two times that number.

The final layer is the output layer. Each neuron in the output layer receives input from each neuron in the hidden layer immediately proceeding. The number of neurons in the output layer will equal the number of output categories we want.

In nearly all cases, a three-or-four-layer network will do well; layers beyond four rarely perform better and increase the training time. Qi (1996) outlines a few optimal network guidelines that have been proposed.

Connections between neurons and their strengths

Connectivity of the neurons is an important part of network architecture. The neurons can be fully connected or partially connected. In the case of a fully connected network, all first layer neurons are connected to all hidden neurons in the next layer, and all second layer neurons are connected to all third layer neurons. Finally, all neurons in the last hidden layer are connected to all neurons of the output layer. Figure 2 shows a fully connected network with three layers. The connections carry initial weights (connection strengths) which are altered during training. The network has to be fed with some random "seed values". The seed values provide initial weights (‘hints’) which the network learns to adjust in subsequent runs. The algorithm of the back-propagation network (called "generalized delta rule") provides the "learning rule" or the method through which the network changes the connection weights during training, and thereby finds optimal values for the weights.

Apart from the architecture, neural networks set themselves apart by the way values of the weights are set for training purpose. In supervised training method, a network is trained by presenting it with a series of training cases (vectors) each with associated target output values. The weights are then adjusted based on the learning rule specified. In unsupervised training, a self-organizing network groups together similar input vectors without using training cases. It specifies characteristics of typical member of each group or the group to which each vector belongs. Weights are modified such that the most similar input values are assigned to the same output category.

Transfer function

Another very important functional issue involves the transfer function. Through this function a network seeks to make sense of the input data. The position and role of a transfer

function are illustrated in Figure 3. The figure also illustrates the basic functioning of an individual neuron. Each of the interconnected processing elements -- or neurons -- in a neural network sends and receives data to and from other processing elements. Input data (I_1, I_2,..., I_n) are multiplied by the weights (W_1, W_2,.., W_n) associated with the connection to the neuron. These products are then passed through a transfer function, which converts the sum into a value in a specified interval between 1 and -1 or 1 and 0. The output from this neuron is then multiplied by another, separate weight and fed into the next neuron. If a neuron is in the output layer, then the output from this neuron is not multiplied by a new weight. It becomes the output itself.

Figure 4 shows a typical transformation in a transfer function using sigmoid function (which produces an S-shaped curve). The purpose is to scale the input values to reasonable levels (e.g., between 0 and 1). It is done before the output is passed on to the next level. Without this transformation the output value may be very large. A sigmoid transfer function dulls the effect of outliers. When financial market data is used, such function is preferred because of the presence of outliers. In addition to standard sigmoid, variations of sigmoid, Gaussian, hyperbolic tangent and sine transfer functions are appropriately used in various problem-solving approaches.

Operation of the back-propagation network

We go back to Figure 1 to understand the forecasting operation of a simple, fully connected, feedforward back-propagation network. Feedforward refers to network architecture, whereas back-propagation is a training method. To forecast, the network has to be trained using historical data. Data inputs have to be in numbers -- prices, volume, ratios, etc.

A single input category is fed into a single neuron in the input layer. Thus, there are as many neurons in the input layer as there are input categories. Each neuron multiplies the input data by some initial weight and passes on the value to every neuron in the hidden layer. Thus,

each hidden layer neuron receives input from every input layer neuron in a fully-connected network.

Each neuron in the hidden layer sums the values it receives and runs the summed value through a transfer function contained in it. The transfer function determines a weight with which the summed value is multiplied and then passed on to the single neuron in the output layer.

The neuron in the output layer receives values from all the neurons in the hidden layer. It sums the values, runs the result through its transfer function and multiplies the value by some weight to produce an output. The network compares this output with the desired output and determines the difference between the two. This forms the error signal after the first run.

The error signal is fed back through the output layer and the hidden layer(s) to the first layer. As the error signal goes backward, each hidden neuron can determine how strongly it is connected to the output unit and the error at that neuron. A hidden neuron modifies its weight in proportion to the error times the input signal which reduces the error in the direction of most rapid reduction in error. The transfer function specified for each hidden neuron provides the rule for adjusting the weights based on the magnitude of the error in the output neuron. The extent of change in a given weight, as per the generalized delta rule, is the derivative of the transfer function with respect to its total input (see Rumelhart, Hinton and Williams, 1986). The process of feedforward and back propagation of values continues so that the error between the output generated and the output desired, is gradually minimized over a series of data runs. In this way, the network trains itself to generate output closer and closer to desired output. This process of trial and error enables the network to recognize patterns, relationships and correlations between the input variables and the output category. The errors will not be reduced to zero specially with real financial data. When the network can hardly minimize errors as more input data is fed, it

reaches a steady state and the output can then be used for testing. Lippmann (1987) provides back-propagation training algorithm.

LIMITATIONS OF NEURAL NETWORKS

A major shortcoming is that the steps or process through which a network produces an output cannot be debugged or decomposed. As Hawley, Johnson and Raina (1990) note, that process which involves the lattice of connection weights cannot at present be translated into an algorithm that would be intelligible outside neural networks. Another major shortcoming is that a network may "overfit" the data. Kean (1993) points to this generic tendency in neural networks: difficult problems like financial market prediction can "actuate memorization" of idiosyncratic patterns in the training data that will not be of help in out-of-sample data. If a network cannot minimize error by learning significant relationships between input variables and the output variable, it tends to do so by memorizing trivial relationships. It may find any pattern however spurious and coincidental, think it to be significant and predict based on it. To prevent overfitting one solution is to use "fuzzy logic" which instructs the network not to be emphatic when its conclusion is tentative. Another approach is to use "genetic algorithm" which also uses trial and error and its mechanism is similar to how evolution works by mutation and natural selection (see Ridley, 1993). Yet another solution is not to use too many data columns. Kean (1993) contends that too much of extraneous information not only lengthens the learning period, the output will probably suffer.

The technology also suffers from lack of optimal algorithm to ensure the global minimum because of multi-minima error surface (see Qi, 1996).

The lack of formal theory behind neural networks to help in model building implies users need a certain degree of sophistication in terms of selection of input variables and specifying

appropriate network architecture. One approach may be to use in-sample model selection criteria like Akaike's information criteria (AIC) and Bayesian information criteria (BIC). However, Qi and Zhang (2001) find that the commonly used in-sample model selection criteria are not able to identify the best neural network model for out-of-sample prediction. That implies that trial and error will continue to be an essential aspect of use of the networks. However, some networks, as the one we use, provide coefficients to reflect the relative contribution of input variables to the prediction of desired output. Finding correlation between input variables and the output variable(s) is also helpful. The combined use of these two tools can at least ease the input selection issue.

Like factor analysis, neural networks cannot be used to confirm ex-post the identity of causal factors. Also, ex-ante identification of factors does not provide strong grounds to assert causality even when we have a good empirical fit.

Few statistical concepts have been applied in the development of neural networks. Nevertheless, the technology bears relationship with statistical models (see Qi, 1996).

EXPERIMENTAL DESIGN

Data

We want to predict the volatility of the S&P 500 Index futures prices. Our raw data series consists of closing settlement prices of sixteen nearest futures contracts and three spot indexes. We take the futures contract class that will mature in the nearest maturity month. The maturity months are March, June, September and December. The nineteen variables are listed in Table 1. Seven of the sixteen futures contracts are on commodities, three on Treasury obligations, and six on foreign currencies. The three spot indexes are DJIA, NYSE Composite Index and S&P 500 Index. We also use one-day lagged S&P 500 futures prices as an additional explanatory variable

for a total of twenty such variables. We select these variables because of availability of ten years of daily data on them -- from February 1, 1984 to January 31, 1994 – 2,531 observations per variable. The data set was obtained from Knight-Ridder Financial Publishing. Since neural networks need to be trained with a large data set, it fits well with our needs. From the raw data series we calculate 20-day rolling historical standard deviations (HSD). We calculate HSDs from daily percentage price changes of the twenty variables calculated as natural log relatives of the price or index series. The percentage change for day 2 based on prices P_1 and P_2 will be given by: $Ln(P_2/ P_1)$. We use about five hundred HSD observations to train the network, and the rest for forecasting.

We obtain ninety forecasts as of the days shown in Table 2. Thirty of the forecasts are over 55 days following the forecast dates (55-day forecasts), and equal numbers of 35-day and 15 day forecasts. These forecast dates and horizons correspond to implied volatility forecasts we obtain using the Barone-Adesi and Whaley (1987) American futures options pricing model.

ISDs from Barone-Adesi and Whaley (BAW) futures options pricing model

The BAW model is for pricing American options on futures which has no analytic solution. We devise an algorithm for extracting ISDs from the model. We execute the algorithm on a connection machine to extract the implied volatilities from the BAW model. (Connection machines are parallel processors that have speed beyond the reach of present-day PCs.) The implied volatilities are obtained from call options on S&P 500 Index futures contracts. We use just-out-of-the-money options on dates shown in Table 2 to extract the implied volatilities. These are options for which futures price minus exercise price is nearest to 1 but negative.

The futures and options maturity classes run from January 1986 to June 1993 -- four maturity classes per year. For each futures maturity class, we extract forecasts at three different

points in the life of the series. These three forecasts serve as forecasts for the three horizons: 55, 35, and 15 trading days to maturity of a futures contract. As shown in Table 2, the forecast dates are 55, 35 and 15 days prior to the maturity of nearest futures contracts and corresponding options contracts. This will enable us to compare forecasts from neural networks with implied volatility -- a type of forecast that is highly regarded. We compare forecasts from neural network and from BAW model with volatility realized over each of the three forecast horizons. Since the realized volatilities are obtained on the basis of trading days as opposed to calendar days, to be consistent we modify the BAW model to get ISDs based on trading days.

Volatility realizations

We derive 55-day realized standard deviation (RSD) from daily log relatives of the present value of S&P 500 Index futures settlement prices from 55 days before maturity until the day of maturity. For the 35-day and 15-day forecast horizons, the daily returns are based on daily log relatives of the present value of the index futures settlement prices respectively from 35 and 15 days before maturity until the day of maturity. RSD on day t is calculated as follows:

$$RSD_t = [\sum_{j=t}^{t+n} (R_j - \bar{R})^2 / n]^{1/2} \quad \text{.....(1)}$$

where:

$R_j = \ln [F_j / F_{(j-1)}]$

$\bar{R} = R_j / n$

F_j = Present value of futures price on date j

n = 55, 35, 15 respectively for RSDs over the three horizons.

As Figure 5 shows, each 55-day forecast from a futures maturity class will be non-overlapping with the 55-day forecast from the previous or subsequent maturity classes. This will ensure that the forecast error of each period will be uncorrelated with the errors of other periods. The error of the 35-day forecast obtained from a series will be uncorrelated with the errors of other 35-day forecasts. Similar is the case of errors of 15-day forecasts.

Then we compare the two sets of 55-day forecasts with 55-day volatility realizations. Similarly, we separately evaluate the accuracy of the 35-day forecasts and the 15-day forecasts by comparing with corresponding volatility realizations.

Forecast accuracy

To measure forecast accuracy, we calculate mean of absolute errors (MAE) and root mean of squared errors (RMSE) of volatility forecasts compared to realized volatility for the three forecast horizons, as follows:

$$MAE = 1/30 * \sum_{i=1}^{30} [Abs(Y_{it} - \hat{Y}_{it})] \quad \text{.....(2)}$$

$$RMSE = [1/30 * \sum_{i=1}^{30} (Y_{it} - \hat{Y}_{it})^2]^{1/2} \quad \text{.....(3)}$$

where:

$\hat{Y}_{it}$ = forecasted standard deviation.

t = forecast horizon (15, 35, 55 days);

Y_{it} = realized standard deviation.

We also separately test for the differences in the means of each type of forecast with respect to the means of realized volatility for each of the three forecast horizons using standard test statistics. Since the standard t-tests assume normal distribution of the data and our data may

not be normally distributed, we also perform a nonparametric test -- the Mann-Whitney test -- sometimes also known as the Wilcoxon rank sum test. This test performs a two-sample rank test for the difference between two population medians.

VOLATILITY FORECASTS USING NEURAL NETWORKS

Steps in forecasting volatility

We use the following steps to forecast volatility from a backpropagation neural network using the 20-day rolling historical standard deviation series of the input variables.

(a) Selection of input variables

(b) Preprocessing the input data

(c) Specifying a neural network

(d) Training the network and forecasting

These four steps are explained below.

(a) Selection of input variables

This step identifies the variables that contribute the most to forecasting the target variable. Too many variables can unnecessarily overload the system. If we omit important variables, then its effect on the performance of the neural network can be significant. Our perspective on the markets will affect the choice of input data.

Mendelsohn (1993) proposes a synergistic market analysis -- combining technical analysis and fundamental analysis approaches with intermarket analysis -- implemented using a neural network to predict, for example, the next day's high and low for the Treasury bond market. Technical price data on Treasury bonds would be fed into the network, allowing it to learn the general price patterns and characteristics of the target market. In addition, fundamental data that

affect the market can also be input into the network. Few examples are federal funds rate, Gross Domestic Product, money supply, inflation rate and consumer price index. Mendelsohn argues that using fundamental data as well as technical data can improve the overall performance of the network. He further claims that incorporating intermarket input data on related markets enables the network to utilize this information to find intermarket relationships and patterns that affect the target market. A few examples of intermarket data are US dollar index, S&P 500 Index and currency exchange rates.

To predict the next days high and low for the Treasury bond market one can select any number of variables. But the larger the number of variables, the longer will be the training period required and greater the possibility that the data will overfit the model determined by the network. As an upper limit to the number of input variables, Kean (1993) suggests around ten percent of the number of data observations. Thus, if there are three hundred days of observations to train a network, Kean recommends thirty variables.

The selection of input variables will depend on the knowledge of what affects the target variable and the use of statistical tools to find correlation between the target and the other variables. It can be a lengthy process of trial and error. Multiple regression analysis can help to identify statistically significant variables that can be used as input variables; principal component analysis and stepwise regression analysis would also be helpful.

To simplify this step, we initially included twenty explanatory variables mentioned earlier and shown in Table 1. We find the correlations of the daily price changes of eighteen of the twenty variables with the daily price changes of S&P 500 Index futures contracts. (We do not calculate the correlation of the S&P 500 Index futures prices with its lag or with S&P 500 Index since the correlations will be very high.) Table 1 reports the correlation coefficients. From the

twenty variables, we select thirteen explanatory variables as indicated in the last column of Table 1. In general, a variable is selected if it meets one of the two conditions:

- correlation with futures prices is greater than 5% and less than -5%; or,
- high relative contribution to forecasting (relative contribution coefficient greater than 0.07 is the criteria used; this coefficient is provided by the network we use and is explained in McCormick, 1992).

This leaves us with sixteen explanatory variables. Under the apprehension that this number may be on the high side -- in which case the network may "overfit" the data -- we dropped three more variables. We dropped the Canadian dollar (relative contribution coefficient of 0.33) because of its very low correlation (0.005) with S&P 500 Index futures prices. We dropped Eurodollar in spite of its modest correlation with S&P 500 Index futures prices (0.15) and modest relative contribution coefficient (0.23). We also dropped DJIA (relative contribution coefficient of 0.078 and correlation of 0.94) under the assumption that its effect will be captured by the included variable -- S&P 500 Index. This implies there is scope for obtaining better forecasts from the networks by using a different number of input variables than we used.

(b) Preprocessing the input data

Neural networks need properly transformed data to be able to process them and generate sound forecasts. Transformation, normalization and data smooothing are three common ways of preprocessing data. Through transformation we can coalesce a few input variables to form a single input category. Methods include taking differences between inputs or ratios of inputs. Reducing the inputs may help the network learn better. However, it is not necessary to transform data before feeding into a network.

Normalization makes the statistical distribution of each input and output data roughly uniform. The values are scaled to match the range that the input neurons use. Data normalization methods, which include simple linear scaling and statistical measures of central tendency and variance, remove outliers and spread out the distribution of the data.

Data smoothing filters out noise in the data. Smoothing techniques suggested are simple and exponential moving averages, and polynomial regression. Data smoothing serves two purposes. First, the network has been given useful information at a reasonable level of detail. Second, the noise entering the data is reduced.

Some of the networks available in the market have limited built-in preprocessing capabilities like scaling and randomly rearranging data to remove serial dependence. However, these networks cannot transform or smooth data. If we need to transform or smooth data, we have to do that before feeding the data into the system.

Preprocessing has two parts: (i) arranging the data in the form that the neural network can read, and, (ii) scaling the data so that the maximum and minimum of each variable falls in a range of 1 and –1 (or 0 and 1 depending on the type of transfer function specified) respectively, and the other values are scaled accordingly.

(c) Specifying a neural network

Since a typical backpropagation network should have at least three-layers, we specify a three-layered network. Appropriate specification of number of layers is an art. It needs experimentation. The countless combinations of layers and neurons that we can make and the time it takes to train a network after each specification is an arduous exercise. A single or two-layer network would be rather inadequate in capturing the complex interrelationships between market variables. The number of layers specified -- three -- is such that it is not too few and not

too many. We could also use four layers. But that would make the training time prohibitive. The resulting improvement in forecast accuracy may not be worth the extra time. However, a backpropagation network should have at least three layers.

The number of neurons in the first layer -- thirteen -- is equal to the number of explanatory variables. We specified two times that many neurons in the second layer. Lesser number of neurons could also have been specified. But since we use three layers rather than four, we wanted to be rigorous in terms of number of neurons in the hidden layer. More layers and/or neurons would have increased the training time for each of the desired ninety forecasts. The results would also change.

In the network specification stage we can adjust a number of default parameters or values that influence the behavior of the training process. These deal with the learning, forgetting and error tolerance rates of the network, the overlearning threshold, the maximum number of runs, stop value for terminating training and randomizing weights with some specified dispersion. In the absence of objective guidelines, we set most of these parameters to default values of the network.

(d) Training the network and forecasting

A neural network learns from past experience and so has to be trained with a sufficient number of training cases. (This contrasts with expert systems, which have to be fed with rules.) The use of too few training cases will cause the network to map an inadequate model for the output. One rule of thumb suggests four times as many test cases as weights. That means a network with 100 weights needs to be fed with at least 400 training cases. The time taken to train the network will depend on the number of layers, the neurons per layer, the number of iterations

on each day's data set and the speed of the computer. Much of the art of using neural networks comes into play in the training phase. Among the decisions involved at this stage are:

- the number of layers and neurons per layer; these can be varied depending on the performance of the network during the training phase;
- type of transfer function; the standard sigmoid is the most commonly used for market forecasting;
- number of times each day's data set will run through the system;
- the learning rate: extent of correction applied to the connection weights at each step of training; this is not a crucial decision, it can be set to default value.

We train the network using the thirteen explanatory variables mentioned earlier. We use 500 days of data for training purpose. Each set of observations is run 500 times through the network. After each run, the network compares the forecasted volatility of futures prices with the desired volatility. It calculates and feeds the error backward. The neurons reset their weights each time the errors are fed back.

The desired volatility on a particular day for the 55-day forecast horizon is the volatility realized in the subsequent 55 days. The desired volatility on a particular day for the 35-day forecast horizon is the volatility realized in the subsequent 35 days. For the 15-day horizon, the desired volatility on a particular day is the 15-day volatility realization.

After each volatility forecast, we change the training observations. In each case, we use 500 20-day rolling HSD observations for training. These observations are for the 500 trading days prior to a forecast date. In all we get thirty forecasts for each of 55, 35 and 15 trading days -- a total of ninety forecasts.

ANALYSIS OF RESULTS

Table 3 shows realized standard deviations (RSD), volatility forecasts from neural networks (Neural) and implied standard deviations (ISD) using Barone-Adesi and Whaley American futures options pricing model. The realized and forecasted volatilities are shown for the three forecast horizons of this study: 55, 35 and 15 days to the maturity of nearest futures and corresponding options contracts. For each forecast class we have thirty forecasts. The bottom of the table shows the means of RSD, Neural and ISD. T-tests for differences in means of RSD versus Neural show no significant differences in the means in the case of all three forecast classes. T-tests for differences in means of RSDs versus ISDs show significant difference in the means of 15-day and 35-day forecasts (p-value = 0.00 and 0.01 respectively in one-tailed test), and no significant difference in the case of 55-day forecasts (p-value =0.44 in one-tailed test).. That means, whereas ISDs have provided good forecast over the 55-day horizon, neural forecasts have been good in the case of all three horizons. Results of Mann-Whitney nonparametric tests yield similar conclusions: in 2-tailed tests neural forecasts do not show significant differences from realized volatilities in the case of all three forecast classes; ISDs are significantly different from volatility realizations in the case of 15-day and 35-day forecast classes (p-values = 0.01 in both cases) and not so significantly different in the case of 55-day forecast class (p-value = 0.13).

It clearly implies that for the data in our sample, volatility forecasts from neural networks yielded superior results compared to implied volatilities.

Table 3 also shows the mean absolute errors and root mean squared errors of the two types of forecasts. On both measures, and for all three forecast classes, neural forecasts outperform ISDs.

CONCLUSION

The use of neural networks in finance is a promising field of research specially given the ready availability of large mass of data sets and the reported ability of neural networks to detect and assimilate relationships between a large number of variables. However, the realization of the potentials of neural networks in forecasting the market involves more of an art than science. There are many ways and combinations in which a neural network can be specified and many types of data can be simultaneously used. Much of that is yet unexplored. The principles that can guide us in effectively utilizing the networks in financial applications remain a fertile area of research.

DIRECTION FOR FUTURE RESEARCH

In our forecasting, we used thirteen input variables. We also used eleven input variables; the results did not improve. Could it have improved with more than thirteen variables? Could the results have improved with a different number of neurons in the hidden layer, or with two rather than one hidden layer? Guidelines available at present are rather sketchy at best. A great deal of trial and error experimentation is called for. Programming skills are not essential with many commercial software tools for neural networks that are available. A useful site listing many such softwares with links to description on them is: www.emsl.pnl.gov:2080/proj/neuron/neural/systems/software.html.

This article is intended as a primer on how to use neural networks for market forecasting. It makes a rather straightforward exposition of forecasting volatility of S&P 500 Index futures prices. However, the field is advancing. Extension of this research may proceed in the following directions: use simulated and real data to validate the robustness of results following Zhang

(2001); specify both linear and nonlinear neural networks to compare the in-sample fitting and out-of-sample forecasts; use AIC and BIC criteria to add to o the results following Qi and Zhang (2001); use a recursive model similar to Qi (2001) to capture possible structural changes in the economy that may have an impact on forecasts.

.

ACKNOWLEDGMENTS

We gratefully acknowledge insightful suggestions and ideas provided by two anonymous reviewers.

Exhibit 1

Applications of Neural Networks

Among the areas in which neural networks have been used are:

- sensor signal processing and data fusion;
- filtering out noise in electronic communications systems;
- pattern classification, image processing, and machine vision; for example, in designing an airport security system;
- automated inspection to diagnose malfunctions in automobiles;
- robotics and sensor-motor control;
- speech recognition and synthesis, and natural language; for example, converting written into spoken English;
- knowledge processing;
- database retrieval;
- computer-based handwriting and character recognition;
- medical diagnosis, healthcare, and biomedical applications, such as hybrid scheme for diagnosing skin diseases;
- manufacturing and process control;
- defense applications;
- assessing credit/insurance risk;
- financial forecasting applications;
- stock picking/portfolio management/automated trading.

Exhibit 2
Studies on application of neural networks in economics/finance*

- **Prediction of stock price/stock index/stock selection:** White (1988), Sharda and Patil (1990), Kimoto, Asakawa, Yoda, and Takeoka (1990), Kamijo and Tanigawa (1990), Yoon and Swales (1991), Wong, Wang, Goh and Quek (1992), Kryzanowski, Galler and Wright (1993), Rhee (1994), Kohara, Ishikawa, Fukuhara and Nakamura (1997), Brown, Goetzmann and Kumar (1998), Gencay (1998), Qi and Maddala (1999).
- **Business cycle forecasting:** Hoptroff, Bramson and Hall (1991), Vishwakarma (1994), Qi (2001).
- **Economic time-series forecasting:** Swanson and White (1997a,b); Model selection criteria: Qi and Zhang (2001); Data requirement: Walczak (2001); Linear time-series forecasting: Zhang (2001).
- **Interest rate prediction:** Swanson and White (1995), Kim and Noh (1997).
- **Volatility prediction:** Ormoneit and Neuneier (1996), Donaldson and Kamstra (1997), González Miranda and Burgess (1997), Schittenkopf, Dorffner and Dockner (1998), Schittenkopf, Dorffner and Dockner (1999); Capturing volatility smile of options: Meissner and Kawano (2001).
- **Earnings prediction:** Charitou and Charalambous (1996), Kim (1996).
- **Investment analysis:** Valuing manufacturing flexibility: Feurstein and Natter (1998).
- **Predicting merger target:** Sen and Gibbs (1994).
- **Financial distress prediction:** Bell, Ribar and Verchio (1990), Salchenberger, Cinar and Lash (1992), Tam and Kiang (1992), Coats and Fant (1993), Fletcher and Goss (1993), Udo (1993), Altman, Marco and Varetto (1994), Fanning and Cogger (1994), Boritz and Kennedy (1995), Boritz, Kennedy and de Mirande e Albuquerque (1995), Lenard, Alam and Madey (1995), Back, Laitinen and Sere (1996), Greenstein and Welsh (1996), Barniv, Anurag and Leach (1997), Bell (1997), Etheridge and Sriram (1997), Hongkyu, Han and Lee (1997), O'Leary (1998), Zurada, Foster, Ward and Barker (1998), Yang, Platt and Platt (1999).
- **Fraud detection:** Fanning, Cogger and Srivastava (1995), Fanning and Cogger (1998).
- **Financial statement analysis:** Kryzanowski and Galler (1995).

- **Bond risk analysis:** Dutta and Shekhar (1988), Moody and Utans (1991), Surkan and Singleton (1991), Kim, Weistroffer and Redmond (1993), Maher and Sen (1997).
- **Mortgage risk assessment:** Collins, Ghosh and Scofield (1988), Reilly, Collins, Scofield and Ghosh. (1991), Grudnitski, Quang and Shilling (1995).
- **Real estate valuation:** Worzala, Lenk and Silva (1995).
- **Commodity trading:** Collard (1991), Bergerson and Wunsch (1991), Trippi and DeSieno (1992), Kaastra and Boyd (1995).
- **Exchange rate forecasting:** Kuan and Liu (1995), Dropsy (1992), Trippi and DeSieno (1992), Refenes (1993), Zhang (1994), Gencay (1999).
- **Pricing derivatives:** Malliaris and Salchenberger (1992), Hutchinson, Lo and Poggio (1995), Lajbcygier and Connor (1997), Geigle and Aronson (1999), Hanke (1997), Keber (1999), Carelli, Silani and Stella (2000), Garcia and Gencay (2000), Zapart (2003).
- **Real option valuation:** Taudes, Natter and Trcka (1998).
- **Model testing:** Testing APT: Ahmadi (1990); testing multilayer networks: White (1989a); testing model selection criteria: Qi and Zhang (2001); data requirements: Walczak (2001).

* For about eight more studies, see Trippi and Turban (1993). Four of the studies are on financial distress prediction, three on financial forecasting approaches, and one on commodity trading. Also, see Refnes (1995) for more articles using neural networks in finance; the articles deal with equity applications, foreign exchange applications, bond applications, and macroeconomic and corporate performance.

Exhibit 3
Potential research areas in finance using neural networks*

A. CORPORATE FINANCE

1. Financial simulation

- Simulate the behavior of firm's credit customers as economic conditions change: to plan for:
  - Planning for bad-debt expenses and accounts receivable cyclicity
  - Evaluating credit terms and limits
- Cash management
- Capital budgeting
- Exchange rate risk management
- Prediction of credit costs and availability
- Sales prospect selection
- Analyze corporate financial health

2. Prediction

- Train network to mimic the behavior of investors in response to changes in economic conditions or company policies (dividend policy, capital structure, accounting standards, etc.)
- Predicting changes in market trends
- Forecast personnel requirement

3. Evaluation

- Value an acquisition target based on financials.
- Identify desirable acquisition targets based on qualitative criteria or learn personal preferences of human expert.

B. FINANCIAL INSTITUTIONS

- Pricing IPOs
- Simulation of market behavior

C. INVESTING

1. Arbitrage pricing/identifying arbitrage opportunities
2. Technical analysis
3. Fundamental analysis
4. Security risk profiling
5. Index construction

D. OTHERS

- Locating tax evaders
- Property tax analysis
- Mining of financial and economic data bases
- Identification of explanatory economic factors

*Largely based on Hawley, Johnson, and Raina. (1990). Also see Qi (1996).

Figure 1: A neural network system with feedforward backpropagation configuration.

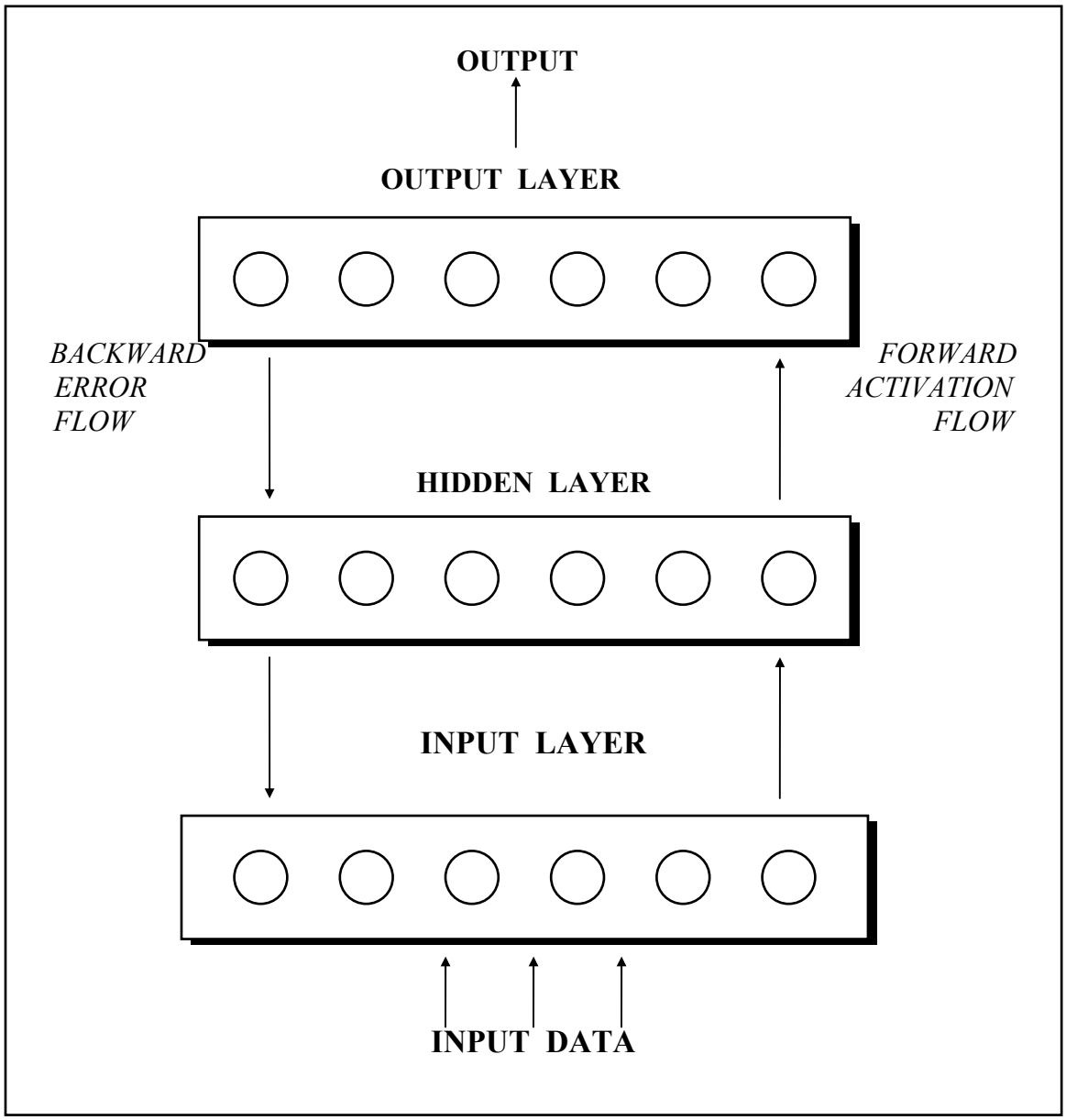


Figure 2: A fully connected, three-layered network. All three neurons in the input layer are connected to all three neurons in the hidden layer. All hidden layer neurons are connected to the neuron(s) in the output layer. Neurons in a given layer do not interconnect.

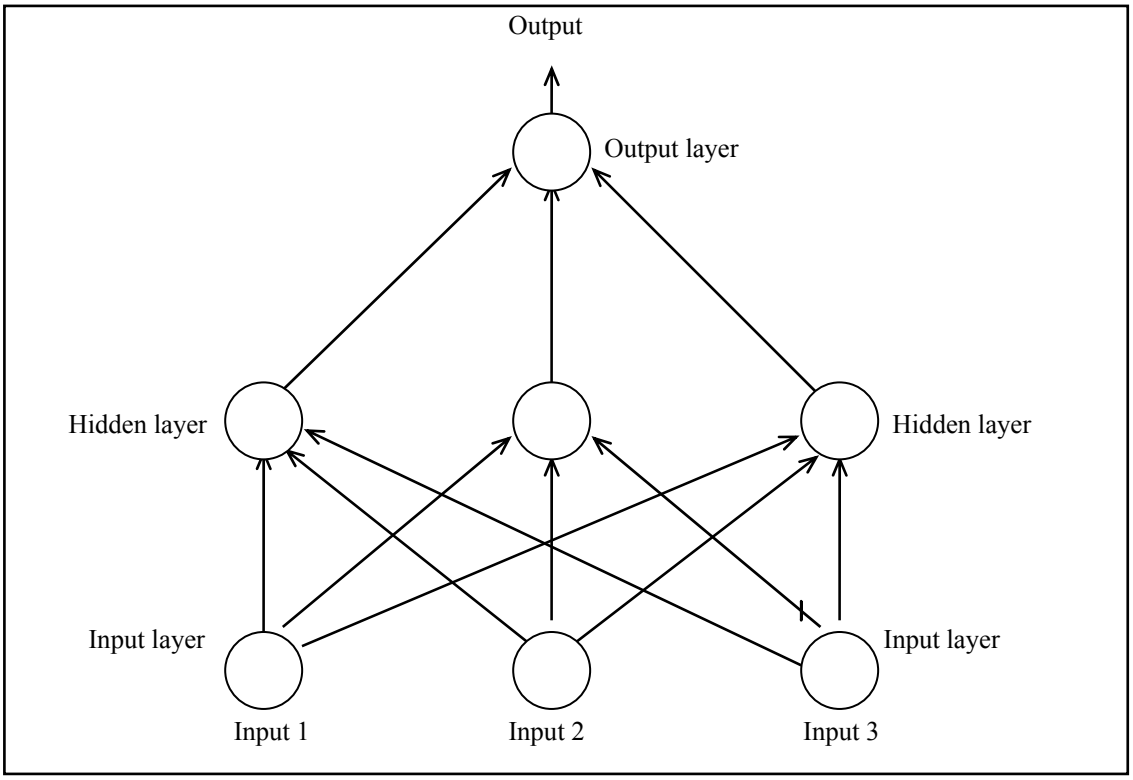
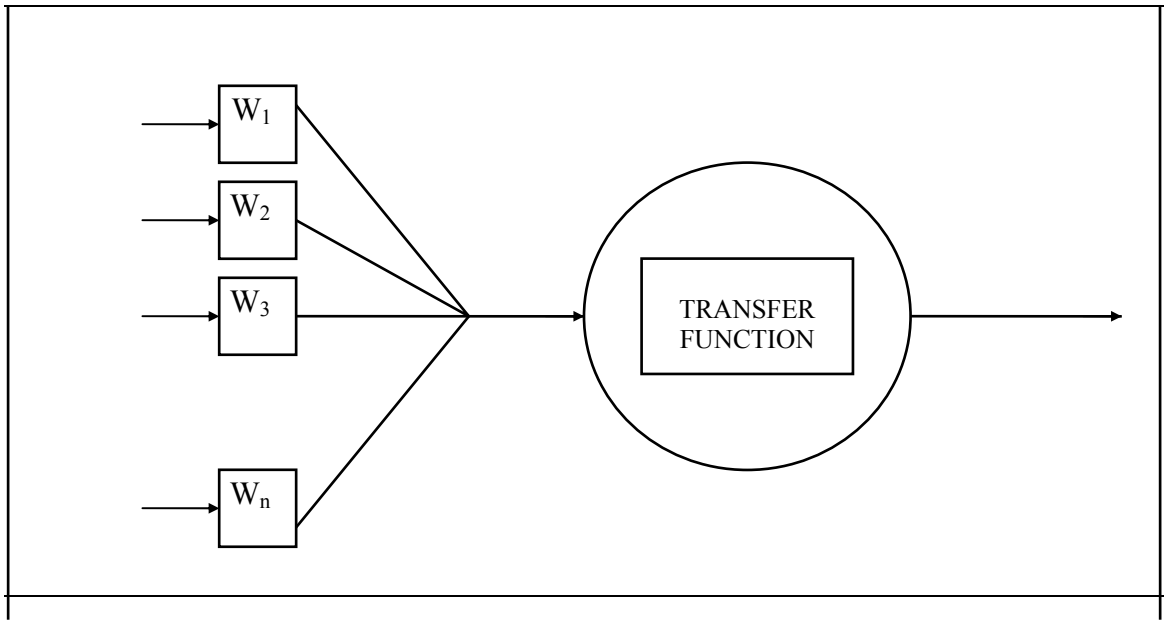


Figure 3: Typical processing element or neuron of a neural network. Such individual interconnected processing elements are the brain cells of neural networks.



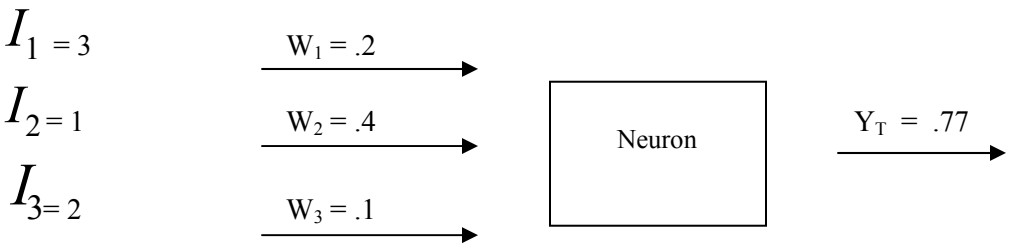
A transfer function is specified in the neuron. The transfer function sums the weighted inputs $I_i*W_i,...,I_n*W_n$ and multiplies the summed value with a new weight determined by the type of the function. For financial forecasting, a nonlinear, continuously differentiable transfer function is needed. The weight by which the neuron multiplies the sum of the weighted inputs is proportional to the derivative of the transfer function with respect to its total input. Typical transfer functions used are sigmoid and hyperbolic tangent.

Adapted from Mendelsohn (September 1993) and Rumelhart et al. (1986).

Figure 4

Transfer Function

An example of a neuron using a sigmoid transfer function:



The summation function results in:

$$Y = 3(.2) + 1\ (.4) + 2\ (.1) = 1.2$$

And the sigmoid transformation results in:

$$Y_T = \frac{1}{1 + e^{-1.2}} = .77$$

0.77 is the transformed or normalized value of 1.2

Figure 5: Nonoverlapping forecast horizons. The errors of 55-day forecast made on January 2, 1986 will be uncorrelated with the errors of the next 55-day forecast (made on April 3, 1986). Similarly, the errors of 35-day forecast made on January 30, 1986 will be uncorrelated with the errors of the next 35-day forecast (made on May 1, 1986). Similar is the case with 15-day forecasts. Thus, each forecast of a particular horizon will be uncorrelated with previous or subsequent forecasts.

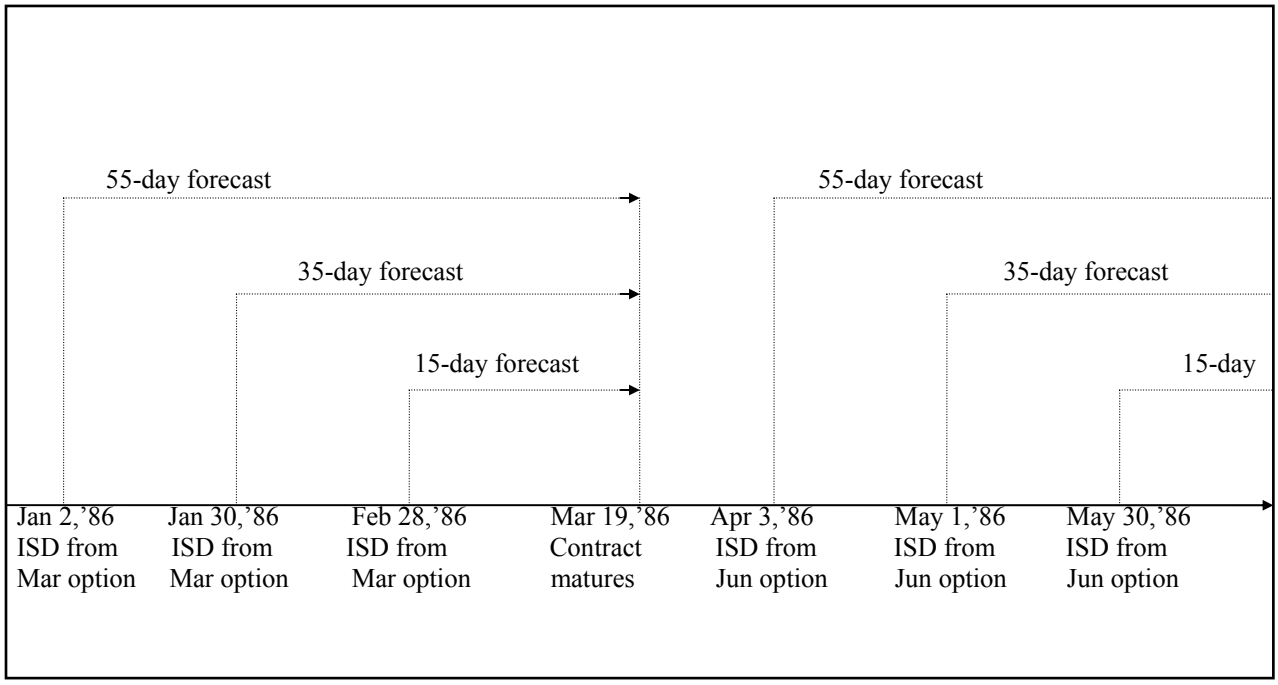


Table 1

Correlation between S&P 500 Index futures daily settlement price changes and the daily price changes of 16 futures contracts and 2 spot indexes using data from February 1984 to January 1994. Also shown are the relative contributions of 20 variables in forecasting volatility of S&P 500 Index futures prices; last column shows contribution of 13 selected variables in forecasting index futures price volatility. The relative contribution coefficients are taken after training the network with 15-day forecast file. The training was with 500 days of data on the 13 variables.

	Contract	Correlation coefficient	Relative contribution coefficient	
			All 20 variables	13 select variables
1	Swiss frank	-0.0572	0.0439	***
2	Japnese yen	-0.0293	0.3293	0.6539
3	NYSE#	0.9000	0.1243	0.1455
4	Treasury bonds	0.3164	0.0736	0.6335
5	Treasury notes	0.2779	0.0676	***
6	Treasury bills	0.0595	0.3239	0.4947
7	Silver	-0.1026	0.1026	0.1609
8	Platinum	-0.0465	0.2686	0.3944
9	Palladium	-0.0488	0.7043	0.8373
10	Heating oil	-0.0682	0.3735	0.1655
11	Copper	0.0590	0.4742	0.6173
12	Gold	-0.1220	0.0675	0.1326
13	Euro-dollar	0.1466	0.2311	***
14	German mark	-0.0393	0.0663	***
15	DJIA#	0.9384	0.0776	***
16	Crude oil	-0.0742	0.4462	0.1513
17	Canadian dollar	0.0054	0.3316	***
18	British pound	-0.0368	0.0684	***
19	S&P 500 Index#	NC	0.0655	0.2042
20	S&P 500 Futures-L	NC	0.1102	0.1617

Notes:

1. #: Represents spot indexes.
2. Relative cotribution measures extent of contribution of rolling historical standard deviation (HSD) series of 17 futures contracts and 3 spot indexes in forecasting the realized volatility of S&P 500 Index futures prices usingdata from 1984-1989. Higher the relative contribution coefficient, higher the contribution of a particular variable in forecasting. Appendix 3 explains the concept.
3. ***: Represents variables not included in network training and forecasting.
4. L: Rolling HSDs computed from log relatives of 1-day lagged index futures prices.
5. NC: Not calculated.

Table 2

Experimental design: Dates for which forecasts are generated for the three forecast horizons: 55, 35 and 15 days before maturity of S&P 500 Index futures contracts:

1986-1993 (30 contract classes)

Obs.	Futures maturity class	55 days to maturity	35 days to maturity	15 days to maturity
1	Mar.-86	02-Jan-86	30-Jan-86	28-Feb-86
2	June-86	03-Apr-86	01-May-86	30-May-86
3	Sept.-86	02-Jul-86	31-Jul-86	28-Aug-86
4	Dec.-86	02-Oct-86	30-Oct-86	28-Nov-86
5	Mar.-87	31-Dec-86	29-Jan-87	27-Feb-87
6	June-87	01-Apr-87	30-Apr-87	29-May-87
7	Sept.-87	01-Jul-87	30-Jul-87	27-Aug-87
8	Dec.-87	01-Oct-87	29-Oct-87	27-Nov-87
9	Mar.-88	30-Dec-87	28-Jan-88	26-Feb-88
10	June-88	30-Mar-88	28-Apr-88	26-May-88
11	Sept.-88	29-Jun-88	28-Jul-88	25-Aug-88
12	Dec.-88	29-Sep-88	27-Oct-88	25-Nov-88
13	Mar.-89	28-Dec-88	26-Jan-89	24-Feb-89
14	June-89	30-Mar-89	27-Apr-89	25-May-89
15	Sept.-89	28-Jun-89	27-Jul-89	24-Aug-89
16	Dec.-89	28-Sep-89	26-Oct-89	24-Nov-89
17	Mar.-90	27-Dec-89	25-Jan-90	23-Feb-90
18	June-90	28-Mar-90	26-Apr-90	24-May-90
19	Sept.-90	05-Jul-90	02-Aug-90	30-Aug-90
20	Dec.-90	04-Oct-90	01-Nov-90	30-Nov-90
21	Mar.-91	26-Dec-90	24-Jan-91	22-Feb-91
22	June-91	04-Apr-91	02-May-91	31-May-91
23	Sept.-91	03-Jul-91	01-Aug-91	29-Aug-91
24	Dec.-91	03-Oct-91	31-Oct-91	29-Nov-91
25	Mar.-92	02-Jan-92	30-Jan-92	28-Feb-92
26	June-92	01-Apr-92	30-Apr-92	29-May-92
27	Sept.-92	01-Jul-92	30-Jul-92	27-Aug-92
28	Dec.-92	01-Oct-92	29-Oct-92	27-Nov-92
29	Mar.-93	30-Dec-92	28-Jan-93	26-Feb-93
30	June-93	31-Mar-93	29-Apr-93	27-May-93

Table 3

Realized standard deviations (RSD) and forecasts from neural networks and implied standard deviation (ISD) from Barone-Adesi and Whaley (BAW) model for 15-, 35- and 55-day forecast horizons. t-statistics for tests of differences in the means of forecasts with respect to RSDs is shown below with p-values. Neural forecasts are not significantly different from RSDs. ISDs are not significantly different from RSDs only in case of 55-day forecasts. Mean absolute errors (MAE),root mean squared errors (RMSE) of forecasts with respect to RSDs show that neural forecasts on average have lower errors compared to errors of ISDs.

	15-day horizon			35-day horizon			55-day horizon		
Obs.	RSD	Neural	ISD	RSD	Neural	ISD	RSD	Neural	ISD
1	0.009198	0.011000	0.010089	0.008276	0.009000	0.009783	0.009923	0.012000	0.009783
2	0.011406	0.012000	0.009771	0.010022	0.009000	0.011388	0.011015	0.010000	0.011388
3	0.017427	0.016000	0.009656	0.013535	0.008000	0.010745	0.012890	0.011000	0.010745
4	0.007874	0.010000	0.009718	0.009552	0.009000	0.010770	0.009380	0.009000	0.010770
5	0.008228	0.009000	0.012194	0.008512	0.009000	0.012882	0.010166	0.009000	0.012882
6	0.006964	0.011000	0.011128	0.010561	0.015000	0.014738	0.013913	0.013000	0.014738
7	0.011139	0.010000	0.012426	0.010021	0.007000	0.009125	0.008790	0.009000	0.009125
8	0.020287	0.012000	0.020818	0.021492	0.038000	0.032794	0.058768	0.045000	0.032794
9	0.009187	0.016000	0.015446	0.009319	0.014000	0.019159	0.017147	0.010000	0.019159
10	0.012025	0.013000	0.012297	0.011030	0.014000	0.014072	0.012645	0.014000	0.014072
11	0.008707	0.013000	0.012115	0.009885	0.013000	0.012264	0.010301	0.010000	0.012264
12	0.005930	0.009000	0.010000	0.006907	0.010000	0.011705	0.007654	0.009000	0.011705
13	0.007648	0.009000	0.009600	0.008261	0.010000	0.009255	0.007665	0.010000	0.009255
14	0.007428	0.004000	0.008444	0.006983	0.004000	0.008398	0.006746	0.005000	0.008398
15	0.005255	0.016000	0.008666	0.007722	0.008000	0.009068	0.007788	0.006000	0.009068
16	0.005393	0.013000	0.008051	0.007121	0.009000	0.014414	0.015035	0.023000	0.014414
17	0.007268	0.009000	0.011115	0.007841	0.009000	0.013629	0.010295	0.007000	0.013629
18	0.009851	0.009000	0.009508	0.009231	0.008000	0.010463	0.008744	0.006000	0.010463
19	0.010864	0.011000	0.015917	0.015323	0.008000	0.011123	0.013175	0.013000	0.011123
20	0.006910	0.011000	0.010422	0.009459	0.012000	0.016777	0.011586	0.012000	0.016777
21	0.009506	0.008000	0.011807	0.010080	0.009000	0.012102	0.011263	0.012000	0.012102
22	0.007055	0.008000	0.008817	0.008364	0.009000	0.009417	0.008695	0.008000	0.009417
23	0.005553	0.006000	0.007723	0.008149	0.010000	0.008492	0.007196	0.008000	0.008492
24	0.006697	0.008000	0.010214	0.009282	0.009000	0.008768	0.008601	0.009000	0.008768
25	0.010568	0.006000	0.008176	0.008483	0.008000	0.009562	0.007620	0.007000	0.009562
26	0.006152	0.005000	0.007265	0.005870	0.006000	0.008303	0.007714	0.007000	0.008303
27	0.006132	0.006000	0.007439	0.005359	0.007000	0.007490	0.006046	0.007000	0.007490
28	0.005089	0.006000	0.006457	0.005283	0.006000	0.008454	0.005931	0.006000	0.008454
29	0.008717	0.006000	0.006224	0.008120	0.006000	0.006608	0.007080	0.007000	0.006608
30	0.004788	0.003000	0.006406	0.006164	0.008000	0.007002	0.006418	0.007000	0.007002
Mean:	0.008642	0.009533	0.010264	0.009207	0.010033	0.011625	0.011340	0.010700	0.011625
Std:	0.003453	0.003461	0.003098	0.003162	0.005840	0.004942	0.009382	0.007349	0.004942
t-stat.		-1.00	**-1.92**		-0.68	**-2.26**		0.29	-0.15
p (2-tail)		(0..20)	**(0.00)**		(0.49)	**(0.03)**		(0.77)	(0.88)
p (1-tail)		(0.10)	**(0.00)**		(0.25)	**(0.01)**		(0.38)	(0.44)
MAE		0.002447	0.002598		0.002535	0.003079		0.001923	0.002379
RMSE		0.003237	0.003114		0.003996	0.004114		0.003432	0.005113
Mann-Whitney nonparametric test results:									
Median	0.007760	0.009000	0.009740	0.008500	0.009000	0.010600	0.009090	0.009000	0.010600
W-stat.		823	738		912	727		949	812
p(2-tailed)		0.18	0.01		0.97	0.01		0.62	0.13

References

Ahmadi, H. (1990) Testability of the Arbitrage Pricing Theory by Neural Networks. *Proceedings of the IEEE International Conference on Neural Networks*: 1385-1393.

Altman, E. I., Marco, G., and Varetto, F. (1994) Corporate Distress Diagnosis: Comparisons Using Linear Discriminant Analysis and Neural Networks (the Italian Experience). *Journal of Banking and Finance* 18 (May): 505-529.

Back, B., Laitinen T., and Sere K. (1996) Neural Networks and Genetic Algorithms for Bankruptcy Predictions. *Expert Systems with Applications* 11(4): 407-413.

Barniv, R., Agarwal A., and Leach, R. (1997) Predicting the Outcome Following Bankruptcy Filing: A Three-state Classification Using Neural Networks. *International Journal of Intelligent Systems in Accounting, Finance & Management* 6(3).

Barone-Adesi, G., and Whaley, R. E. (1987) Efficient Analytic Approximation of American Option Values. *Journal of Finance* 42: 301-320.

Bell, T. B., Ribar, G. S., and Verchio, J. (1990) Neural Nets Versus Logistic Regression: A Comparison of Each Model's Ability to Predict Commercial Bank Failures, in *Proceedings of the 1990 Deloitte and Touche/University of Kansas Symposium on Audit Problems*: 29-52.

Bell, T. B. (1997) Neural Nets or the Logit Models? A Comparison of Each Models Ability to Predict Commercial Bank Failures. *International Journal of Intelligent Systems in Accounting, Finance & Management* 6(3).

Bergerson, K, and Wunsch, D. (1991) A Commodity Trading Model Based on a Neural Network-Expert System Hybrid, in *Proceedings of the IEEE International Conference on Neural Networks*: pp. 1289-1293.

Boritz, J. E., Kennedy, D. B., and de Mirande e Albuquerque, A. (1995) Predicting Corporate Failure Using a Neural Network Approach. *International Journal of Intelligent Systems in Accounting, Finance and Management* 4(2) (June): 95 –111.

Boritz, J. E., and Kennedy, D.B. (1995) Effectiveness of Neural Network Types for Prediction of Business Failure. *Expert Systems with Applications* 9(4): 503-512.

Brown, S., Goetzmann, W., and Kumar, A. (1998) The Dow theory: William Peter Hamilton's track Record Reconsidered. *Journal of Finance* 53 (August): 1311-1333.

Carelli, A., S. Silani and F. Stella. (2000) Profling neural networks for option pricing, *International Journal of Theoretical and Applied Finance* 3:2, 183-204.

Charitou, A., and Charalambous, C. (1996) The Prediction of Earnings Using Financial Statement Information: Empirical Evidence using Logit Models & Artificial Neural Networks. *International Journal of Intelligent Systems in Accounting, Finance & Management* 5(4).

Coats, P. K., and Fant, L. F. (1993) Recognizing Financial Distress Patterns Using a Neural Network Tool. *Financial Management*: 142-155.

Collard, J. E. (1991) A B-P ANN Commodity Trader, in *Advances in Neural Information Processing System,* vol. 3, R. P. Lippmann, J. E. Moody, and D. S. Touretzky, ed., Morgan Kaufmann, San Mateo, CA.

Collins, E., Ghosh, S., and Scofield, C. (1988) An Application of a Multiple Neural Network Learning System to Emulation of Mortgage Underwriting Judgments. *Proceedings of the IEEE International Conference on Neural Networks* II (July): 459-466.

Davies, P. C. (1995) Neural Network Techniques for Financial Time Series Analysis in Virtual Trading, J. Lederman and R. A. Klein, ed., Probus Publishing, 81-87.

Dropsy, V. (1992) Exchange Rates and Neural Networks. *Working Paper 1-92*, California State University, Dept. of Economics, Fullerton.

Donaldson, R. G., and Kamstra, M. (1997) An Artificial Neural Network- GARCH Model for International Stock Return Volatility. *Journal of Empirical Evidence* 4: 17-46.

Dutta, S., and Shekhar, S. (1988) Bond Rating: A Non-conservative Application of Neural Networks, in *Proceedings of the IEEE International Conference on Neural Networks,* vol. II, IEEE Press, San Diego, New York, (July), 443-450.

Etheridge, H. L., and Sriram, R.S. (1997) A Comparison of the Relative Costs of Financial Distress Models: Artificial Neural Networks, Logit and Multivariate Discriminant Analysis. *International Journal of Intelligent Systems in Accounting, Finance & Management* 6(3).

Fanning, K. M., and Kenneth, O. C. (1994) A Comparative Analysis of Artificial Neural Networks Using Financial Distress Prediction. *International Journal of Intelligent Systems in Accounting, Finance and Management* 3(4) (December): 241-252.

Fanning, K. M., Cogger, Kenneth O., and Srivastava, R. (1995) Detection of Management Fraud: A Neural Network approach. *International Journal of Intelligent Systems in Accounting, Finance & Management* 4(2) (June): 113-126.

Fanning, K. M and Kenneth O. C. (1998) Neural Network Detection of Management Fraud Using Published Financial Data. *International Journal of Intelligent Systems in Accounting, Finance & Management* 7(1).

Feurstein, M., and Natter, M. (2000) Fast High Precision Decision Rules for Valuing Manufacturing Flexibility. *European Journal of Operational Research* 120(1) (January): 108-117.

Fletcher, D., and Goss, E. (1993) Forecasting With Neural Networks: An Application Using Bankruptcy Data. *Information and Management* 24(3): 159-167.

Garcia, R., and Gencay, R.: Pricing and Hedging Derivative Securities with Neural Networks and a Homogeneity Hint. *Journal of Econometrics* 94 (2000): 93-115.

Gencay, R. (1998) The Predictability of Security Returns with Simple Technical Trading Rules. *Journal of Empirical Finance* 5: 347-359.

Geigle, D.S. and J.E. Aronson (1999) An artificial neural network approach to the valuation of options and forecasting of volatility, *Journal of Computational Intelligence in Finance* 7:6 (November/December).

Gencay, R. (1999) Linear, Nonlinear and Essential Foreign Exchange Rate Prediction with Simple Technical Trading Rules. *Journal of International Economics* 47: 91-107.

Gonzales, Miranda F., and Burgess, N. (1997) Modeling Market Volatilities: The Neural Network Perspective. *European Journal of Finance* 3: 137-157.

Greenstein, M. M., and Welsh, M. J. (1996) Bankruptcy Prediction Using Ex Ante Neural Networks and Realistically Proportioned Testing Sets. Working Paper, Lehigh University.

Grudnitski, G., Quang, A., and Shilling, J. D. (1995) A Neural Network Analysis of Mortgage Choice. *International Journal of Intelligent Systems in Accounting, Finance and Management* 4(2) (June): 127-135.

Hanke, M. (1999) Option pricing using neural networks vs. Black/Scholes: An empirical comparison of two fundamentally different option pricing methods, *Journal of Computational Intelligence in Finance* 5:1 (January), 26-34.

Hawley, D. D., Johnson, J. D., and Raina, D. (1990) Artificial Neural Systems: A New Tool For Financial Decision-Making. *Financial Analysts Journal* (November/December): 63-72.

Hongkyu, J., Han, I., and Lee, H. (1997) Bankruptcy Prediction Using Case-based Reasoning, Neural Networks and Discriminant Analysis. *Expert Systems with Applications* 13(2) (August): 97-108.

Hoptroff, R., Bramson, M., and Hall, T. (1991) Forecasting Economic Turning Points with Neural Nets, in *IEEE INNS International Joint Conference of Neural Networks*, 1347-1350.

Hutchinson, J. M., Lo, A., and Poggio, T. (1995) A Nonparametric Approach to Pricing and Hedging Derivative Securities via Learning Networks. *Journal of Finance* 49, 851-889.

Kaastra, I., and Boyd, M. S. (1995) Forecasting Futures Trading Volumes Using Neural Networks. *Journal of Futures Markets* 15, 953-970.

Kamijo, K., and Tanigawa, T. (1990) Stock Price Pattern Recognition: A Recurrent Neural Network Approach, in *Proceedings of the International Joint Conference on Neural Networks,* vol. I, 215-21, San Diego: IEEE Network Council.

Katz, Jeffrey, O. (1995) Neural Networks in Trading, in V*irtual Trading*, J. Lederman, and Robert A. Klein, ed., Probus, Danvers, MA.

Kean, J. (1993) Neural Nets and Stocks. *Training a Predictive System* PC A1 (September-October): 45-47.

Keber, C. (1999) Option pricing with the genetic programming approach, *Journal of Computational Intelligence in Finance* 7:6, 26-36.

Kim, S. H, and Noh, H. J. (1997) Predictability of Interest Rates Using Data Mining Tools: A Comparative Analysis of Korea and the US. *Expert Systems with Applications* 13(2) (August): 85-95.

Kim, J. W., Weistroffer, H. R., and Redmond, R. T. (1993) Expert System for Bond Rating: A Comparative Analysis of Statistical, Rule-based, and Neural Network Systems. *Expert Systems* 10(3), 167-171.

Kim, Moon K. (1996) Accuracy of Quarterly Earnings Forecasts by Analysts and an Artificial Neural Network Model, in *Financial Management Association Meeting*.

Kimoto, T., Asakawa, M., Yoda, and Takeoka, M. (1990) Stock Market Prediction System with Modular Neural Networks, in *Proceedings of the International Joint Conference on Neural Networks,* vol. I, 1-6.

Kohara, K., Ishikawa, T., Fukuhara, Y., and Nakamura, Y. (1997) Stock Price Prediction Using Prior Knowledge and Neural Networks. *International Journal of Intelligent Systems in Accounting, Finance & Management* 6(1).

Kryzanowski, L. and Galler, M. (1995) Analysis of small-business financial statements using neural nets, *Journal of Accountng, Auditing and Finance* 10:1, 147-170.

Kryzanowski, L., Galler, M., and Wright, D. W. (1993) Using Artificial Neural Networks to Pick Stocks. *Financial Analysts Journal* 49 (July/August): 21-27.

Kuan, C. M., and Liu, T. (1995) Forecasting Exchange Rates Using Feedforward and Recurrent Neural Networks*,* in *Bureau of Economic and Business Research*, University of Illinois at Urbana-Champaign, Faculty Working Paper 92-0128.

Kuan, C. M., and White, H. (1994) Artificial Neural Networks: An Econometric Perspective. *Econometric Reviews* 13, 1-91.

Lajbcygier, P.R. and Connor, J.T. (1997) Improved option pricing using artificial neural networks and bootstrap methods, *International Journal of Neural Systems* 8:4, 457-471.

Lenard, M. J., Alam, P., and Madey, G. R. (1995) The Application of Neural Networks and A Qualitative Response Model to the Auditor's Going Concern Uncertainty Decision. *Decision Sciences* 26(2), 209 –227.

Lippmann, R. P. (1987) An Introduction to Computing with Neural Nets. *IEEE ASSP Magazine* (April): 4-22.

Maher, J. J., and Sen, T. K. (1997) Predicting Bond Ratings Using Neural Networks: A Comparison with Logistic Regression. *International Journal of Intelligent Systems in Accounting, Finance & Management* 6(1).

Malliaris, M.E. and Salchenberger, L. (1992) A neural network model for estimating option prices, *Applied Intelligence* 3:3, 193-206.

Meissner, G., and Kawano, N. (2001) Capturing the volatility smile of options on high-tech stocks-a combined GARCH-Neural Network approach. *Journal of Economics and Finance* 25(3) (Fall): 276-293.

McCormick, D. L. (1992) *N-TRAIN: Neural Network Development System* Users Manual 1(2), Scientific Consultant Services, Inc., Selden, NY.

Mendelsohn, L. (1993) Preprocessing Data for Neural Networks. *Technical Analysis of Stocks and Commodities* (September): 52-58.

Moody, J., and Utans, J. (1991) Principled Architecture Selection for Neural Networks: Applications to Corporate Bond Rating Predictions, in *Advances in Neural Information Processing Systems,* vol 4, J. E. Moody, S. J. Hanson, and R. P. Lippmann, ed., Morgan Kauffman, San Mateo, 683-690.

O'Leary, D.E. (1998) Using Neural Networks to Predict Corporate Failure. *International Journal of Intelligent Systems in Accounting, Finance & Management* 7(3), 187-197.

Ormoneit, D., and Neuneier, R. (1996) Experiments in Predicting the German Stock Index DAX with Density Estimating Neural Networks, in *Proceedings of the 1996 Conference on Computational Intelligence in Financial Engineering (CIFEr),* 66-71.

Qi, M. (1996) Financial application of artificial neural networks, in *Handbook of Statistic: Statistical Methods in Finance,* vol 14, G. S. Maddala, and C. R. Rao, ed., Elsevier, Amsterdam, 529-552.

Qi, Min. (1999) Nonlinear predictability of stock returns using financial and economic variables. *Journal of Business and Economic Statistics* 17(4), 419-429.

Qi, M., and Maddala, G. S. (1999) Economic factors and the stock market: A new perspective. *Journal of Forecasting* 18, 151-166.

Qi, M. (2001) Predicting US recessions with leading indicators via neural network models. *International Journal of Forecasting* 17(3) (Jul-Sep), 383-401.

Qi, M., and Zhang, Guoquiang P. (2001) An investigation of model selection criteria for neural network time series forecasting. *European Journal of Operational Research* 132(3) (August), 666-680.

Refenes, A. P., ed. (1995) *Neural Networks in the Capital Markets*, John Wiley & Sons, Chichester, New York.

Rhee, M. J. (1994) Forecasting Stock Market Indices with Neural Networks, in *International Workshop on Neural Networks in the Capital Markets* II, Pasadena, CA., November.

Ridley, M. (1993) Frontiers of Finance. *The Economist* (October), 9-15.

Reilly, D., Collins, E., Scofield, C., and Ghosh, S. (1991) Risk Assessment of Mortgage Applications With A Neural Network System: An Update As the Test Portfolio Age, in *Proceedings of the IEEE International Conference on Neural Networks,* vol. II, July, 479-482.

Rumelhart, D. E., Hinton, G. E., and Williams, R. J. (1986) Learning Internal Representations by Error Propagation, in *Parallel Distributed Processing: Explorations in the Microstructures of Cognition*, vol. 1, D. E. Rumelhart and J. L. McClelland, ed., MIT Press, Cambridge, MA, 18-362.

Salchenberger, L. M., Cinar, E. M, and Lash, N. A. (1992) Neural networks: A New Tool for Predicting Thrift Failures. *Decision Sciences*. 23(4), 899-916.

Schittenkopf, C., Dorffner, G., and Dockner, E. J. (1998) Volatility Predictions With Mixture Density Networks, in *ICANN 98 – Proceedings of the 8th International Conference on artificial Neural Networks,* L. Niklasson, M. Bodén and T. Ziemka, ed., Berlin, 929-934.

Schittenkopf, C., Dorffner, G., and Dockner, E. J. (2000) Forecasting time-dependent conditional densities: A semi-nonparametric neural network approach. *Journal of Forecasting* 19(4) (July), 355-374.

Sen, T. K., and Gibbs, A. M. (1994) An Evaluation of the Corporate Takeover Model Using Neural Networks. *International Journal of Intelligent Systems in Accounting, Finance and Management* 3(4) (December): 279-292.

Sharda, S., and Patil, B. P. (1990) Neural Networks as Forecasting Experts: An Empirical Test. *Proceedings of the International Joint Conference on Neural Networks* 2, 491-494.

Surkan, A., and Singleton, J. (1991) Neural Networks for Bond Rating Improved by Multiple Hidden Layers, in *Proceedings of the IEEE International Conference on Neural Networks,* vol. II, July, pp. 157-162.

Swanson, N. R., and White, H. (1995) A model selection approach to assessing the information in the term structure using linear models and artificial neural networks. *Journal of Business and Economic Statistic* 13, 265-275.

Swanson, N. R., and White, H. (1997a) A model selection approach to real-time macroeconomic forecasting using linear models and artificial neural networks. *The Review of Economic and Statistic* 79, 540-550.

Swanson, N. R., and White, H. (1997b) Forecasting economic time series using flexible vs. fixed specification and linear vs. nonlinear economic models. *International Journal of Forecasting* 13, 493-461.

Tam, K.Y., and Kiang, M.Y. (1992) Managerial Applications of Neural Networks: The Case of Bank Failure Prediction. *Management Science* 38(7), 926-947.

Taudes, A., M. Netter and Trcka, M. (1998) Real option valuation with neural networks, *International Journal of Intelligent Systems in Accounting, Finance and Management* 7:1 (March), 43-52.

Trippi, R. R., and DeSieno, D. (1992) Trading Equity Index Futures with a Neural Network. *The Journal of Portfolio Management* (Fall), 27-33.

Trippi, R.R. and Turban, E., ed. (1993) *Neural Networks in Finance and Investing*. Probus Publishing Co., Chicago.

Udo, G. (1993) Neural Network Performance on the Bankruptcy Classification Problem. *Computers and Industrial Engineering* 25(1-4), 377-380.

Vishwakarma, K. P. (1994) Recognizing business cycle turning points by means of a neural network. *Computational Economics* 7, 175-185

Walczak, S. (2001) An empirical analysis of data requirements for financial forecasting with neural networks. *Journal of Management Information Systems* 17(4) (Spring): 203-222.

White, H. (1988) Economic Prediction Using Neural Networks: The Case of IBM Daily Stock Returns, in *Proceedings of the IEEE International Conference on Neural Networks*, vol. II, 451-458.

White, H. (1989a) Connectionist nonparametric regression: Multilayer feedforward networks can learn arbitrary mappings. *Neural Networks* 3, 535-549.

White, H. (1989b) Learning in Artificial Neural Networks: A Statistical Perspective. *Neural Computation* 1, 425-464.

Wong, F. S., P.Z. Wang, T. H. Goh, and Quek, B.K.. (1992) Fuzzy Neural Systems for Stock Selection. *Financial Analysts Journal*, (January-February): 47-52.

Worzala, E., Lenk, M., and Silva, A. (1995) An Exploration of Neural Networks and Its Application to Real Estate Valuation. *Journal of Real Estate Research* 10, 185-201.

Yang, Z. R, Platt, M. B., and Platt, Harlan D. (1999) Probabilistic Neural Networks in Bankruptcy *Prediction.* Journal of Business Research *44(2) (Feb): 67-74.*

Yoon, Y., and Swales, G. (1991) Predicting Stock Price Performance. *Proceedings of the IEEE 24th Annual Hawaii International Conference on Systems Sciences,* vol. IV, IEEE Computer Society Press, Hawaii, 156-162.

Zapart, C.A. (2003) Beyond Black-Scholes: A neural networks –based approach to options pricing, *International Journal of Theoretical and Applied Finance* 6:5, 469-489.

Zhang, X. (1994) Non-linear Predictive Models for Intra-day Foreign Exchange Trading. *International Journal of Intelligent Systems in Accounting, Finance and Management* 3(4) (December), 293-302.

Zhang, G. P. (2001) An investigation of neural network for linear time-series forecasting. *European Computers & Operational Research* 28(12) (Oct), 1183-1202.

Zurada, J. M., Foster, B. P., Ward, T. J., and Barker, R. M. (1998-1999) Neural Networks Versus Logit Regression Models for Predicting Financial Distress Response Variables. *The Journal of Applied Business Research* 15(1) (Winter), 21-29.