Deep Learning in Financial Analytics : Exchange Rate Modelling

Sonali Agarwal [1]

Abstract

In finance, a major enthralling research question has been the accurate determination of future market and economic movements. A lot of researchers have tried to develop different models with different accuracies of prediction over the years. It appears that the full potential of deep learning has not been explored to study FX rates. The current study, therefore, explored the proficiency of deep neural networks in predictive modeling. I tested different models of artificial neural networks (using hyperparameters' tuning like training algorithms, number of hidden layers, and hidden nodes) using neural network input-output fitting and tried to find the best fit model. The model was also validated by layered digital dynamic time series modeling using autoregression with two delays. The appraisal metrics used were regression *R* - value, MSE, time-series response plot, and error autocorrelation plot. It was concluded that the artificial neural network with a single hidden layer having 17 nodes and trained using the Levenberg– Marquardt algorithm gave the best performance in a minimum number of iterations. This study marks an extensive examination of ANN modeling. This model can be used by traders, investors, financiers, economists, bankers, speculators, hedgers, and governments to get insights into future forex rates and thus make profitable decisions. Various trading policies, import-export policies, and pricing of commodities in indigenous markets can be managed precisely. Future studies can use these models in simulated trading and help establish an alliance between statistical significance and economic significance.

Keywords : **Forex forecasting, predictive modeling, deep neural network, input-output fitting, training algorithm, hidden layer, error autocorrelation, back-propagation, hyperparameter, incremental training**

JEL Classification Codes : C880, F470, G170

Paper Submission Date : June 25, 2021 ; Paper sent back for Revision : April 26, 2022 ; Paper Acceptance Date : May 20, 2022 ; Paper Published Online : September 15, 2022

An important and virtually invincible area in finance is the "prediction of market trends" for the future. The real-time data of many variables that govern the national and international markets need to be utilized efficiently so that maximum information can be deduced for future forecasting. The markets are important for professional traders, fund managers, borrowers, economists (Sager & Taylor, 2006), etc. The forex markets behave differently as compared to financial stock markets, making the exchange rate a crucial factor. There is high short-term trading in the FX market as compared to the securities markets (Lyons, 2001). Forex rates prediction has been at the heart of investigation since the disintegration of the Bretton-Woods monetary system, and for a country like India, where the floating rate exchange system is now followed, the "exchange rate" becomes one of the most dominant variables for the functioning of the economy. The valuation of export, import, FII & FDI investments, the value of foreign reserves, and the price of commodities like gold, oil, etc., all hinge on the forex rate. Whenever there is a fall in the exchange rate, there ensues a feeling of fear and distrust among investors, financers, economists, bankers, and citizens.

[1] *Assistant Professor,* University School of Humanities and Social Sciences, Guru Gobind Singh Indraprastha University, Sector-16-C, Dwarka, Delhi - 110 078. (Email : sonali1600@yahoo.in) ; ORCID iD : https://orcid.org/0000-0002-3753-428X

DOI : https://doi.org/10.17010/ijf/2022/v16i9/172157

It has been believed that exchange rates show independent and identically distributed distribution, are stochastic, and are highly non-stationary (Kayacan et al., 2010). Thus, to handle such strong non-linearities (Hong & Lee, 2003; Medeiros et al., 2001) and establish patterns, ANN modeling gives a possibly superior alternative to long-established forecasting techniques. Studies conducted by Lenard et al. (1995), Salchenberger et al. (1992), Fletcher and Goss (1993), Tyree and Long (1995), and Hu et al. (1999) established the better performance of ANN over conventional modeling using statistics. Empirical studies (Zurada et al. 1999) also demonstrated that ANN was superior to many time series models. Agarwal and Khan (2019) tested the effect of hyperparameter modification in their research. Agarwal (2022) went on to use deep learning in sentiment analysis. The papers highlighted the growing use of ANN in analytics.

Some researchers, Kiani (2005), Swanson and White (1995, 1997 a,b), Kuan and White (1994), and Zhang and Hu (1998) also found ANN to be very useful in modeling economic time series and argued that the results were better than both linear and other nonlinear models. Machine learning algorithms have been changing quickly, and improved models can be built and tested due to improved software. I, therefore, attempt a research by successively testing different models of ANN created by adjusting and tuning hyperparameters. The study aims to find the best fit model that can make near-perfect exchange rate predictions and give generalizable results. This is the need of the hour as a lot of money rides on forex transactions, trading, exports, imports, etc. Not much has been done to find a perfect model for exchange rate prediction using ANN. This model, once developed, can be used by traders, investors, financiers, economists, bankers, speculators, hedgers, and governments to get an insight into future forex rates and thus make profitable decisions. Various trading policies, import-export policies, and pricing of commodities in indigenous markets can be managed precisely. We begin with an explanation of a few basic concepts and methods used in our study.

Deep Neural Networks

Significant analytical advancements have been achieved with the penetration of deep learning in allied fields. The introduction of ANN, optimization algorithms, and regularization procedures have rejuvenated prediction modeling. These are marvelous paradigms of artificial intelligence which facilitate the modeling of a complex nonlinear relationship between sets of inputs and outputs. The iterative revision of internal parameters expedites the process until the inputs fit the target outputs both in the train set and validation set. Deep learning involves automatic learning of multiple levels of data representations using multiple layered models.

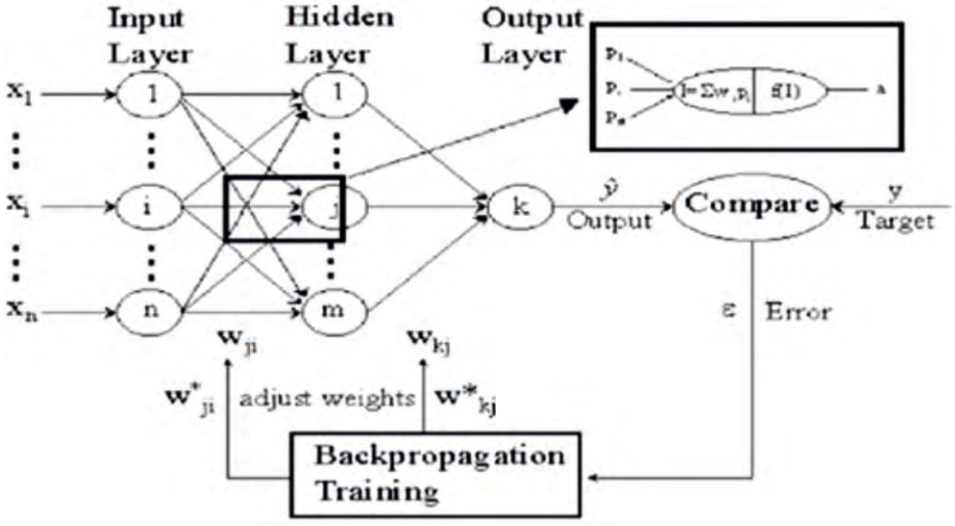
Multilayer networks can estimate any linear or non-linear relationship and can approximate any sensible/sane function promptly well. Theoretically, a trained network is capable of performing well, but back-propagation and gradient descent might not always find a minima.

The error surface for a non-linear network is more obscure than the error surface for a linear network. The biggest hurdle is that the non-linear transfer functions lead to many local minima in the error surface of multilayer networks. Since the gradient descent operates on the error surface, given the initial starting conditions, the network's solution can get trapped in one of these local minima. The seriousness of the situation depends on the closeness of local minima to the global minima and how much smaller error is required.

Neural Network Input Output Fitting Models

We can solve regression and classification problems when working with neural networks (Figure 1). The data fitting models of neural networks require the network to learn and map the input data set with the corresponding target data set. Though a simple two-layered neural network can solve multidimensional mapping problems, there are some intricate and complex cases where one needs to delve into deep learning. With the increase in the number

Figure 1. ***Simple Neural Network and Gradient Descent Back-Propagation***



Source : Google images.

of neurons and the number of hidden layers, the model's predictive ability gets immensely enhanced. But the drawback is that the models become more complex and lead to a slowing down of training due to non-accomplishment of minima in back-propagation.

Back- Propagation

It is a prominent and preferred method used in the training of neural networks. Back-propagation revamps the weights in numerous back passes and assists the neural network in correctly mapping the data inputs to their respective outputs. For exemplification, we can consider a simple neural network having one input layer, one hidden layer, and one output layer. We consider supervised machine learning in this illustration. The training of the network involves feeding the data set to the input layer, forward pass through the hidden layer, and getting the output of one pass through the output layer. The data set in its path is crushed at both the hidden layer and the output layer according to their respective activation functions. The output of the network is compared with the target output fed initially with the dataset. The difference between the two is tossed back as an error through the path: output layer – hidden layer – input layer. This is called the backward pass, and with this, the weights of the network are adjusted to make the output of the network nearer to the pre-specified target. The back-propagation uses the delta rule (gradient descent rule) for this error revision.

For a neuron '*m*', with activation function *f(a),* the delta rule (gradient descent) for *m*'s n^{th} weight w_{mn} is given as follows :

$$\Delta w_{mn} = r(t_m - y_m) f'(h_m) a_n \qquad (1)$$

where, *r* is the learning rate

f(a) is the activation function of neuron *'m'* in question.

t_m is the target output,

y_m is the produced output,

a_n is the n^{th} input,

h_m is the weighted sum of neuron's inputs.

After each cycle of forward and backward pass (the weights get upgraded), the output of the network gets closer to the target values. There can be various methods to terminate the training process. The choice of method depends on the type of research and utilized data, the requirement of accuracy or precision, the time taken for training, etc. The different methods range from limiting the number of iterations or epochs, the minimum learning rate, the relative rate of change in training error, etc. All these cases give different models of networks with differing performance efficiency. The backpropagation technique can be used in various network types: multilayer networks, dynamic networks, and radial basis networks.

Hyperparameters

Hyperparameters are defined as those parameters whose value is defined before the start of the training process. All the other parameters, on the other hand, get their values from training and gradient descent. Some examples of hyperparameters are the number of hidden layers, the number of nodes, the learning rate, the number of iterations and epochs, etc. (Goodfellow et al., 2016).

Incremental Training

Here the targets and the inputs are presented as elements of cell arrays to the network. The network's gradient biases and weights are updated after every single input/target presentation. When using dynamic neural networks for autoregressive time series prediction, we train the network by summing previous and current inputs. We also assign the first term of the sequence as the initial condition for the delay.

Training Algorithms

Training algorithms (Table 1) are the ultimate training facilitators that help update the network biases and weights in the direction in which the performance gradient decreases most speedily. The process continues through multiple iterations until the network converges. The training algorithms prominently use Jacobian calculations or gradient basis.

Table 1. ***Training Algorithms Used in the Study***

Function short form	Algorithm
LM	Levenberg – Marquardt
BR	Bayesian Regularisation
RP	Resilient Backpropagation
SCG	Scaled conjugate gradient
GDX	Variable learning rate gradient descent
GD	Gradient descent

Network Appraisal

The research's main parameters used to assess the network performance are MSE (mean squared error) and *R*-value.

Mean Squared Error

MSE is defined as the arithmetic mean of the squared difference between outputs and targets. The lower the value, the better the network performance.

$$MSE = \frac{1}{N} \sum_{i=1}^{N} (t_i - a_i)^2$$

Regression R-value

The *R* values, obtained after regression, measure the degree of correlation between the network obtained outputs and the actual targets. A value of *R* close to 1 means a good relationship, while the degree of relationship decreases to completely random as the value of *R* approaches 0.

Training, Validation, and Test Sets

The data set used for the research is divided randomly into three sets of predefined proportions. The training set consists of the portion of the data set which is used to train the network. The validation set is used to test the generalizability of the network. Once the network has been trained and validated, the test data set is used to appraise this network's performance.

Literature Review

Despite ANN-based modeling getting a place in financial applications (Fischer & Krauss, 2018; Huck, 2019; Kim et al., 2020) for a decade, not much work has been done in the case of exchange rate modeling. The previous literature was studied thoroughly to get a view of where the prediction concept stands on date and where things have been lacking.

Ghiassi and Saidane (2005) attempted a model parameter estimation in their research with artificial neural networks. They used the entire data set simultaneously for model training and compared this validated model with conventional feed-forward networks. Both the models were appraised using marketing data set, and it was confirmed that the newly designed model gave better results.

A comparative study was performed by Ghiassi et al. (2005), wherein they studied the performance of the time series forecasting model using conventional feedforward networks trained with iterative backpropagation and contrasted it with the dynamic model of artificial neural networks. It was found that the latter method performed much better and gave more accurate predictions.

Chang et al. (2009) assembled neural network models, case-based reasoning, and dynamic time windows to form a composite integrated system for forecasting in stock markets. It was found that the prediction of sell/buy judgment points was better than when any of the assembled three methods were used separately.

The two conventional methods of direct and iterative training of artificial neural networks for time series prediction were analyzed by Hamzaçebi et al. (2009). In the iterative method, each successive period value is

forecasted from its immediate previous period value. In the direct method, all the values of all the successive periods are forecasted in one go. Hamzaçebi et al. (2009) used grey relational analysis to compare these two methods and established that the direct method was superior to the iterative method.

An ensemble of methods (fundamental analysis, technical analysis, artificial neural networks, and set theory) was used by Cheng et al. (2010) in their research to develop a strategy model for timing the investment decisions. Forecasting accuracy and returns from investments were used for appraising the model.

Fluctuations in the Chinese Stock Index were investigated by Liao and Wang (2010). They introduced stochastic time effective function to enhance the prediction ability of existing neural network models. They proposed that the greater the closeness between past data and current time, the more intense is its effect on the forecasting model. The model constructed was appraised by different volatility parameters.

Guresen et al. (2011) tried to break through the traditional linear and nonlinear approaches to stock market rates forecasting and inspected three new models: Multi-layer perceptron (MLP), dynamic artificial neural network (DAN), and a hybrid neural network. The MSE (mean squared error) was used to appraise the models, and it was demonstrated that the MLP model made the best forecasts when used on the same data set.

Gupta and Dalal (2022) studied price discovery on select Indian stocks using VAR and VECM approaches. Agarwal (2019) studied this aspect of mutual funds. Joshi (2021) investigated the effect of trading volume and open interest on futures contract prices using the EGARCH model. Seth and Sidhu (2021) attempted price discovery in Indian energy markets using VECM, ARDL, and GJR-GARCH models. They found decent results with the models they used, however, accuracy was still on the slop side.

An examination of the stock forecasting ability of artificial neural networks was done by Moghaddam et al. (2016). For their research, they used two types of input datasets of the NASDAQ Stock Exchange, one set with four prior days data and the other with nine prior days data. It was found that both methods were equally commendable.

Singh et al. (2021) used ANN for the pricing of IPOs. A multilayer perceptron was trained and settled on the best predictive model after 20 iterative constructions. Selvamuthu et al. (2019) tested support vector machines, support vector regression, and backpropagation neural networks for forecasting financial time series. They acknowledged that every algorithm has its way of learning patterns and making predictions. They found better results using tick data.

Villada-Duque et al. (2020) applied ANN models to two shares and three commodities in the Colombian Stock Market for forecasting. They used data of five months as a training set and one-month data as a test set. The results showed the good performance of neural networks with predictive accuracies.

Shahvaroughi Farahani and Razavi Hajiagha (2021) used ANN to predict stock price indices by training them with a metaheuristic algorithm like social spider optimization and bat algorithm. They compared the results with time series models of ARMA and ARIMA and justified the greater accuracy of the ANN model.

In the majority of the studies, either a conventional ANN model has been used, or a default model has been examined using various software. But, as researchers, we should not get convinced by just one solution and accept it as a final result; rather, rigorous hit and trials through variations should be done. The best way to get a robust model is by experimenting with hyperparameters. A simple code can be advanced by altering the basic properties of the network. The present study aims to find the best predictive model for exchange rate forecasting by altering the training algorithms, the number of nodes in the hidden layer, and the number of hidden layers.

Research Objectives

All the major financial variables of the economy, like exports, imports, FDI, FII, inflation, etc., are affected due to the volatile nature of the forex rate. A lot of researchers have tried to develop different models with different

accuracies of prediction over the years. It appears that the full potential of deep learning has not been explored to study the FX rates. The current study, therefore, explores the proficiency of deep neural networks in predictive modeling. Empirical experimentation was done by hyperparameter tuning. I tested different models and finally concluded on the best model for the data set.

Research Methodology

Exploratory research is conducted to identify the best-fit models for exchange rate prediction. The various supervised learning models used 3-year daily data of exchange rate from January 1, 2016 to December 31, 2018. The data were collected from Investing.com, processed for any missing values, and then aligned in rows in MS Excel to help suit the data layout format of the software. The software used for analysis was MATLAB R2018b.

The input data set and the output data set consisted of 849 samples of the exchange rate (the variable under consideration in this study). So, the data format for the software is a 1×849 matrix. For the analysis, I allowed random division of data into three parts, 60% for training the network, 20% for carrying out validation checks, and 20% for testing the model. All the models were trained using incremental training. The default number of validation checks has been set as 6. The stopping criteria for training were taken as the minimum error between target and output followed by six validation checks. The activation function for hidden layer (s) was sigmoid, and the activation function for the output layer was linear. The number of training iterations and epochs were allowed to take their due course through back-propagation via gradient descent. They were, therefore, not fixed but rather allowed to reach the minimum error criterion of stopping.

To arrive at the best predictive model of the exchange rate, three hyperparameters were changed in the research:

- The training algorithm of the network.
- The number of hidden nodes (neurons).
- The number of hidden layers.

The performance of the models was appraised by *R*-value and MSE (mean squared error). The number of iterations/epochs used to give the best performance was also taken into consideration. Along with this, the time taken for training the network in different models was also considered. The above three experiments of change in hyperparameters were made using the network input-output fitting.

Analysis and Results

The data was fed into the different neural networks. The first type of testing done was by changing the training algorithms used for the network. The default setting of one hidden layer with 10 hidden neurons, 60% training set, 20% validation set, and 20% test set was considered. The activation function of the hidden layer was sigmoid, and of the output layer was linear. The performance results obtained are depicted in Table 2 and Figure 2.

From Table 2 and Figure 2, we can infer that the best performance was reached in minimum epochs/iterations (9) when the training algorithm used was LM. With GD, the network did not give results (negative *R* values). Hence, GD is not suitable for our data set. A huge gradient was found after training and validation with GDX. The *R* - values obtained were not satisfactory (approx. 0.85). With RP, the time taken was 69 iterations for getting the least error, but at the same time, the gradient after validation checks was quite large. With BR algorithm training, the network, though converged and showed very less gradient after validation, the time taken for converging was

Figure 2. ***The Performance Metrics of Neural Networks with Different Training Functions***

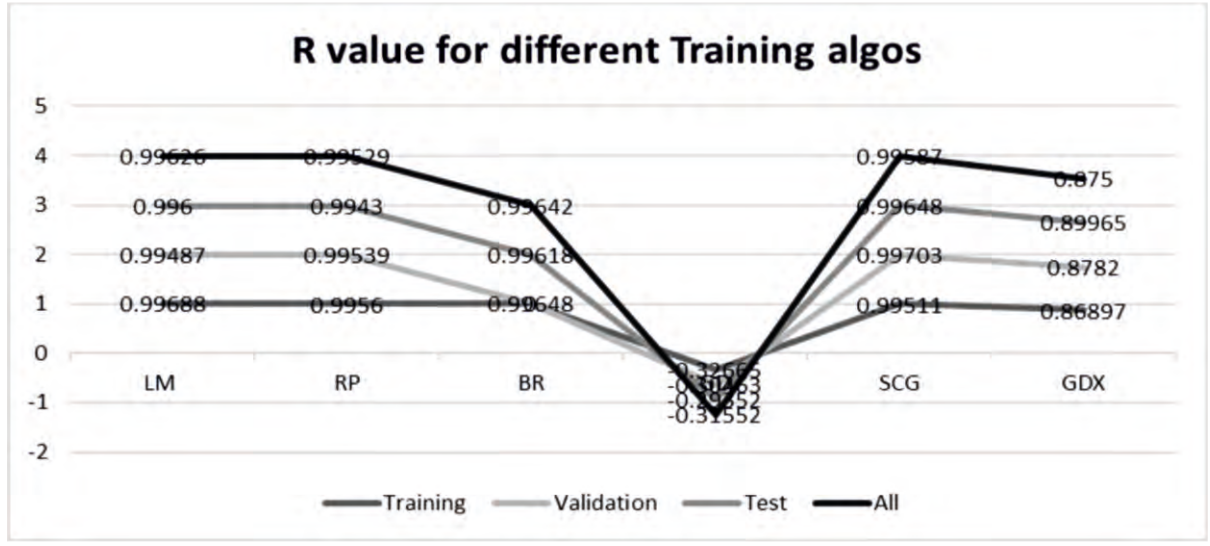


Table 2. ***Performance Metrics of Neural Networks with Different Training Functions***

S.No.	Training Algorithm	Best Validation Value	Number of Epochs	Training (*R*-value)	Validation (*R*-value)	Test (*R*-value)	All (*R*-value)	Gradient after six validations
1.	*LM*	0.058013	9	0.99688	0.99487	0.996	0.99626	0.0015646 @ epoch 15
2.	*RP*	0.053722	69	0.9956	0.99539	0.9943	0.99529	0.18658 @ epoch 75
3.	*BR*	0.041844	471	0.99648		0.99618	0.99642	0.00013351 @ epoch 600
4.	*GD*	179.5839	0	–0.32665	–0.30463	–0.29352	–0.31552	248357.1488 @ epoch 6
5.	*SCG*	0.045513	29	0.99511	0.99703	0.99648	0.99587	0.066163 @ epoch 35
6.	*GDX*	1.9034	27	0.86897	0.8782	0.89965	0.875	8.9112 @ epoch 33

Note. *LM* is Levenberg – Marquardt ; *BR* is Bayesian Regularisation ; *RP* is Resilient Backpropagation ; *SCG* is Scaled conjugate gradient; *GDX* is variable learning rate gradient descent ; *GD* is gradient descent.

very high (600 epochs). Training of the neural network with SCG converged to best performance after 29 epochs/iterations with good *R* - values (approx. 0.995) for the test validation. But when I compared this model with the one trained with the LM algorithm, I found comparable *R* values. But the LM algorithm took nine iterations for convergence with a significantly less gradient (0.0015) after validation checks compared to SCG training (gradient 0.066). Thus, it can be concluded that LM is the best-suited training algorithm for forex prediction.

Next, the suitable number of hidden nodes and hidden layers is determined. The previous defaults are upheld, and additionally, the training algorithm is set to LM for all the new model testings (Figure 3). Table 3 shows that the minimum training time taken was in networks with 15 hidden neurons (one iteration) and 17 hidden neurons (two iterations). Also, in the network with 16 hidden neurons, the gradient obtained after validation checks was extremely low (0.00083), but the training time taken was five iterations.

When we increased the number of hidden layers to two (with 15 and 5 hidden nodes), the number of iterations

Figure 3. ***Performance Metrics for Changing the Number of Hidden Layers and Hidden Neurons (LM Training Algorithm)***

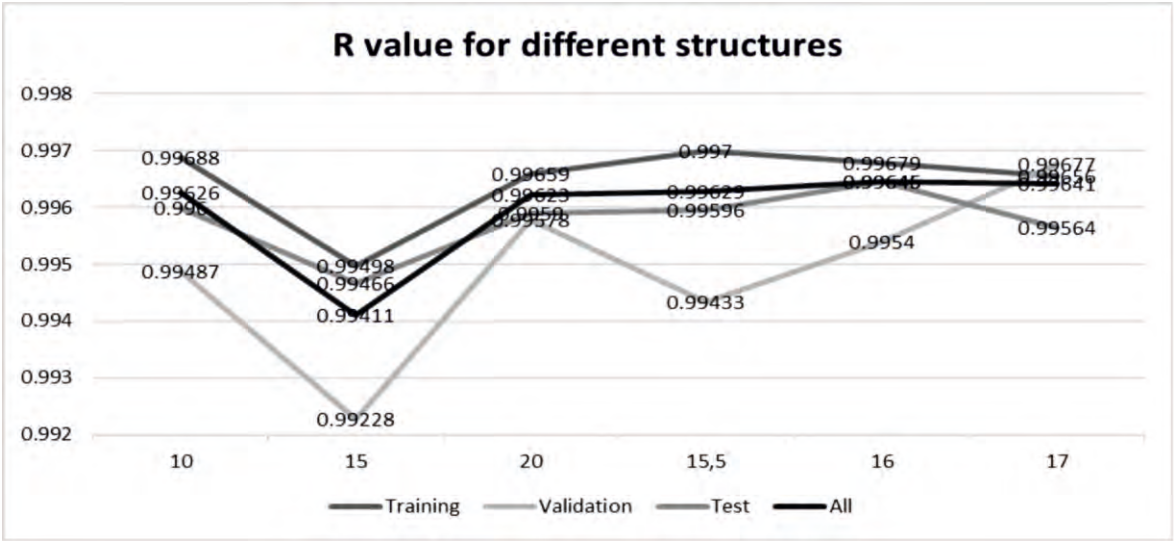


Table 3. ***Performance Metrics for Changing the Number of Hidden Layers and Hidden Neurons (LM Training Algorithm)***

S.No.	Structure	Training Algorithm	Best Validation value	Number of Epochs	Training	Validation	Test	All	Gradient after six validations	Mu (at last validation epoch)
1.	10	Lm	0.058013	9	0.99688	0.99487	0.996	0.99626	0.0015646 @ epoch 15	
2.	15	Lm	0.12414	1	0.99498	0.99228	0.99466	0.99411	0.0026093 @ epoch 7	0.0001
3.	20	Lm	0.053045	3	0.99659	0.99578	0.9959	0.99623	0.010848 @ epoch 9	1e–05
4.	15,5	Lm	0.060948	6	0.997	0.99433	0.99596	0.99629	0.0077449 @ epoch 12	0.0001
5.	16	Lm	0.047334	5	0.99679	0.9954	0.99645	0.99646	0.00083372 @ epoch 11	0.0001
6.	17	Lm	0.038465	2	0.99656	0.99677	0.99564	0.99641	0.004508 @ epoch 8	1e–05

Note. LM is Levenberg – Marquardt.

consumed in training also increased to 6. The gradient after validation also increased many folds compared to the ones with a single layer of 15 or 17 nodes. The *R* values were almost comparable in all the models tested above (approx. 0.995).

From the detailed discussion of Table 3, the models LM 15 (neural network trained with LM algorithm and having a single layer of 15 hidden neurons), LM 16 (neural network trained with LM algorithm and having a single layer of 16 hidden neurons), and LM 17 (neural network trained with LM algorithm and having a single layer of 17 hidden neurons) are suitable candidates for best models. Next, I examine the performance plots and the MSE values for training, validation, and test sets.

The performance graph (Figure 4) shows the values for the performance function with the increasing number of iterations. Three lines are visible in the graphs in Figure 4. The grey line represents the training errors, the light

grey line represents the validation errors, and the black line represents the test errors of the network. It can be observed that the validation and the training curves are quite similar. The best performance epoch displays the iteration at which the error during the validation check reached a minimum value. The graphs show that the best validation performance is achieved at epochs 1, 5, and 2, respectively, in neural net models LM15, LM16, and LM17. If the test curve increases significantly before the increase in the validation curve, then it indicates the possibility of some overfitting. No overfitting is seen in the graphs depicted in Figure 4.

From Table 4, it can be inferred that the MSE of training, validation, and test sets is highest for the model having 15 hidden neurons compared to the other two models. The model with LM 16 took more training time and had a higher MSE (training, validation, and test) than LM 17. Hence, it can be concluded that the best model for the prediction of forex rate is the one using the LM training algorithm, having 17 neurons in the single hidden layer of the artificial neural network.

Figure 4. ***Performance Plots of LM 15, LM 16, and LM 17***

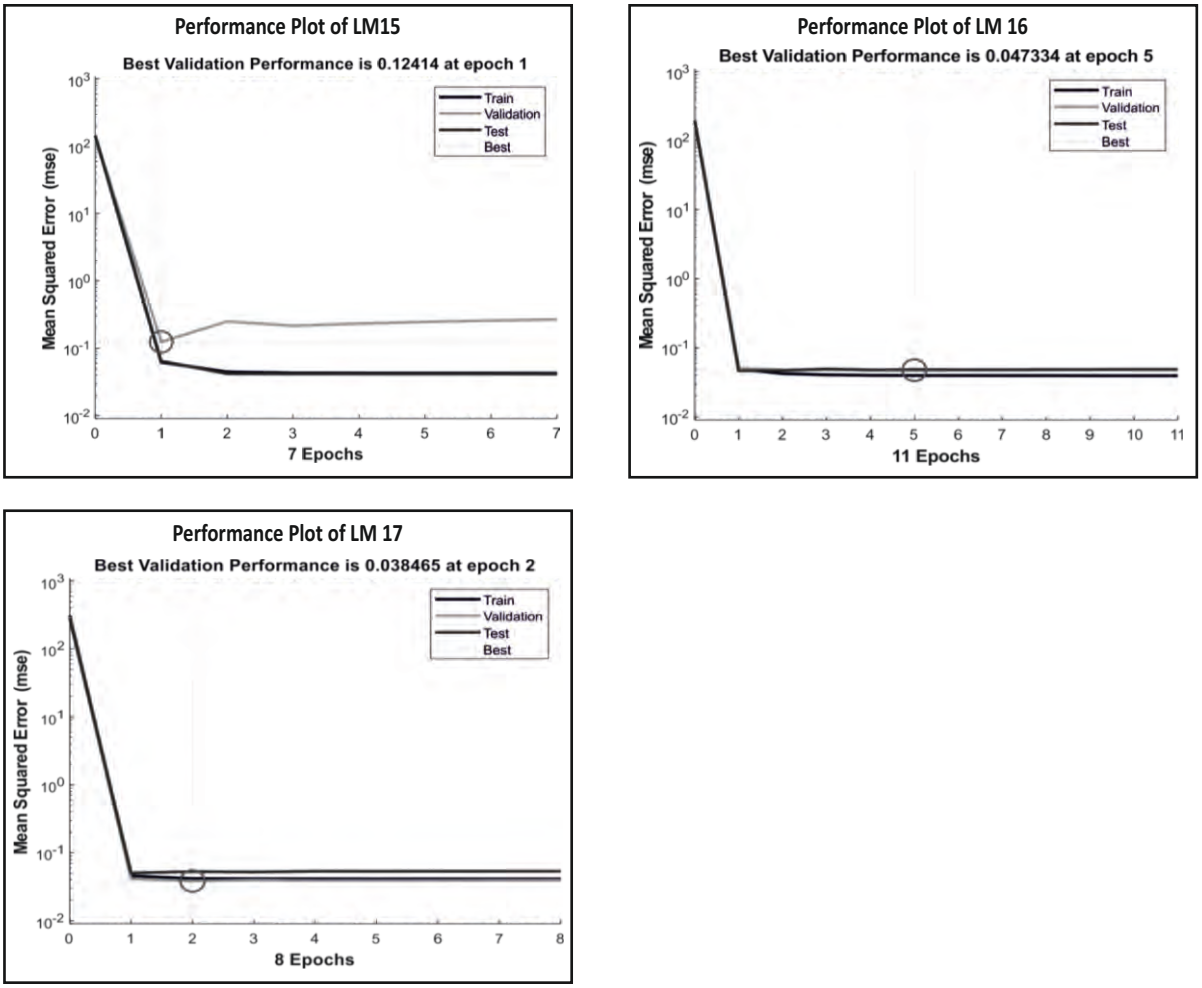
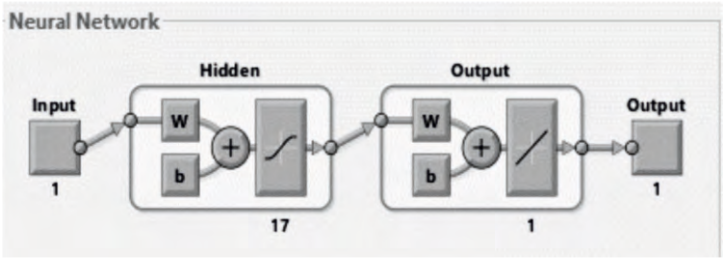


Table 4. ***Performance Comparison of LM 15, LM 16, and LM 17***

	LM 15	LM 16	LM 17
Training MSE	6.17755e–2	4.97215e–2	4.18121e–2
Validation MSE	1.24140e–1	4.73340e–2	3.84645e–2
Test MSE	6.44529e-2	6.23849e–2	5.28636e–2
Training *R*-value	0.99498	0.99679	0.99656
Validation *R*-value	0.99228	0.9954	0.99677
Test *R*-value	0.99466	0.99645	0.99564

Figure 5. ***Best Fit Model LM 17 from Input-Output Fitting***



The Best Fit Model LM 17

The best network for predicting the exchange rate for the Indian rupee is the one with a single hidden layer with 17 nodes and is trained with the Levenberg – Marquardt algorithm. The structural representation of the same is depicted in Figure 5.

The training stops when validation error increases (for six iterations), which occurred here at iteration number 2. To check the network performance, some analysis of network response is done. Linear regression is conducted between network outputs and the target outputs. Figure 6 depicts the regression results for all the data divisions: the training set, the validation set, and the test set. The overall regression results of the network are also shown.

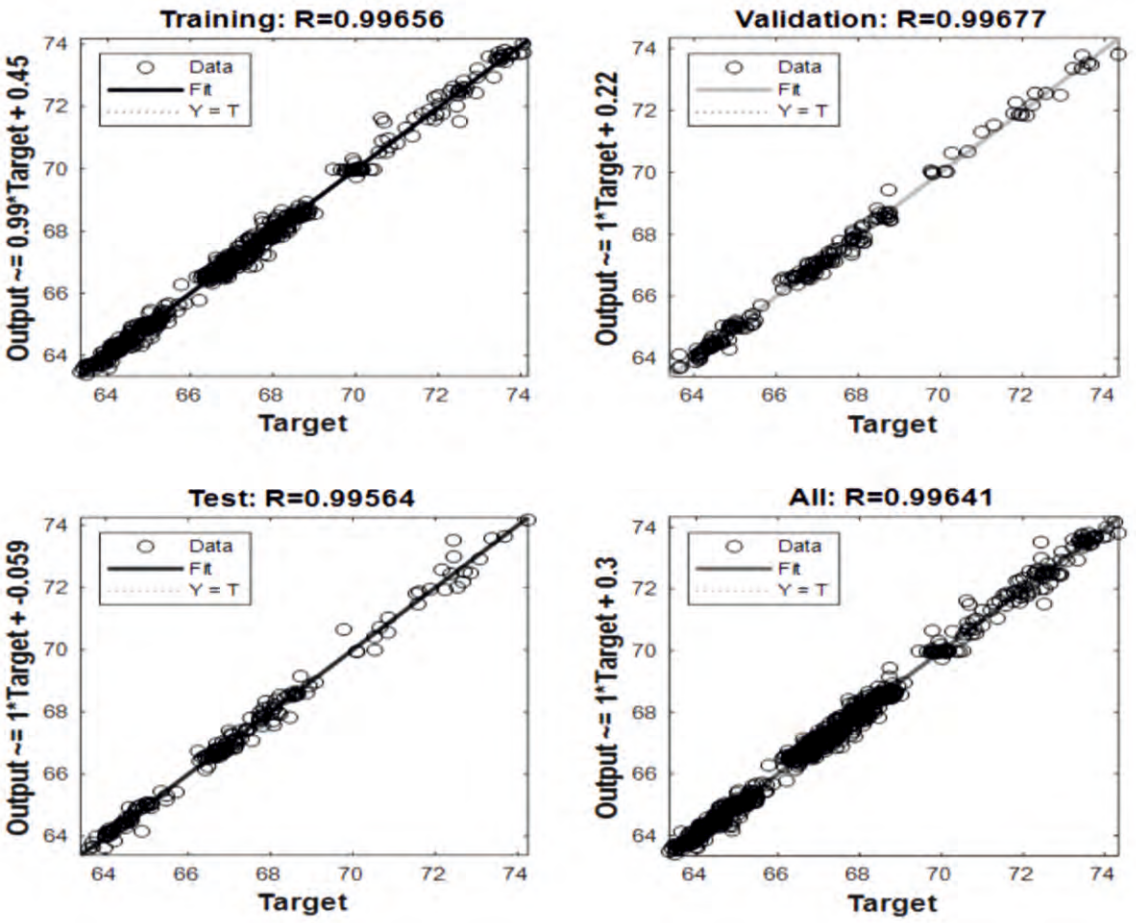
The regression plots (Figure 6) show the regression results between the network's outputs and target outputs. It helps to analyze the response of the designed network. Figure 6 shows four plots. The three axes seen represent the training data, the validation data, and the testing data. The dashed line in each axis shows the perfect result: (outputs = Targets). The solid line through the center represents the best fit line of the linear regression between outputs and targets.

In the case of perfect training, the regression plot shows a straight line with all points on the line, since, in this situation, the network outputs and the target outputs coincide. But this is a rare case. For a perfect regression fit, all the data points must fall on a 45^0 degree line, indicating that the network outputs are exactly equal to the target outputs.

R-value, used as a measure of appraisal for regression, gives an idea of the relationship between the targets and the outputs. A value of 1 indicates exact linear relation, while a value of 0 shows that no linear relationship exists. Thus, this scatter plot clearly shows fit and misfit data points. An outlier that looks to be a complete misfit should be included in the training set, and more new data should be added to the test set. This helps in making the model more robust.

For the case of the present study, the fit seems to be very good for all the data set divisions, with values of *R* in

Figure 6. ***Regression of Network Outputs and Target Outputs from Input-Output Fitting***



each case above 0.995. *R*-value is 0.99641 for the total network response, which is an excellent value. Hence, we can say that the network outputs trace and trail the target outputs efficiently.

The plot of the error histogram (Figure 7) shows the distribution of errors in the network. The bars in dark black present training data, the bars in grey present validation data, and the bars in light black present test data. This plot gives an idea of the outliers, that is, the data points that are the worst fit or misfit. In this plot, it can be seen that though most of the errors lie between –0.5 and +0.5, there are training points (outliers) with an error of 0.95 and –1, validation points with an error of –0.7, and test points with an error of –1,–0.8, and 0.7. These outliers can be easily figured out on the regression plot. Outliers give an idea of the data, and we need to include more similar points in the training set. In our datasets, more data is not required since the regression test plot shows good results with an *R* of more than 0.99. Both the histogram plot and regression plots validate the network's (LM 17) performance.

Figure 8 shows the input-output fitting of the network response. The plot also shows the errors in the fit. The fit seems to be more or less conclusive and absolute. All the above analytics validate that the model with 17 hidden neurons best fits the given data sequence.

Figure 7. ***Error Histogram from Training, Validation, and Test Sets in Input-Output Fitting***

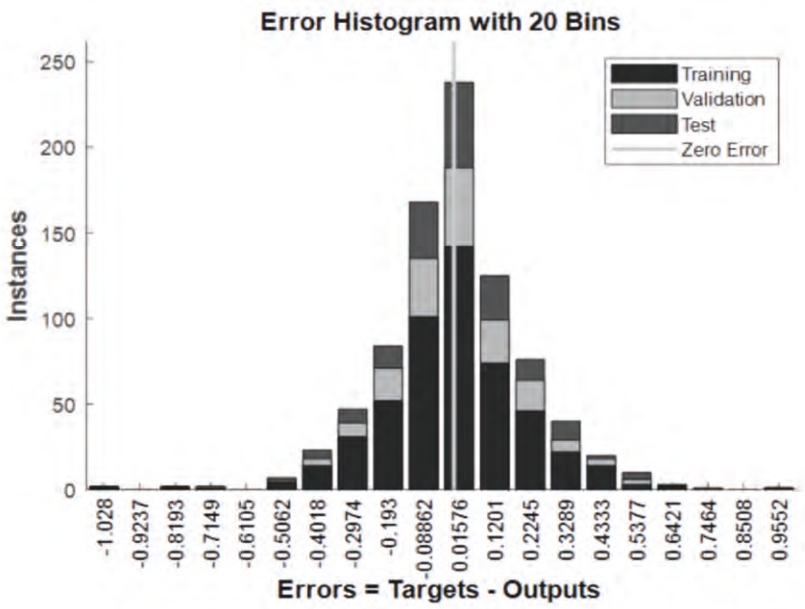
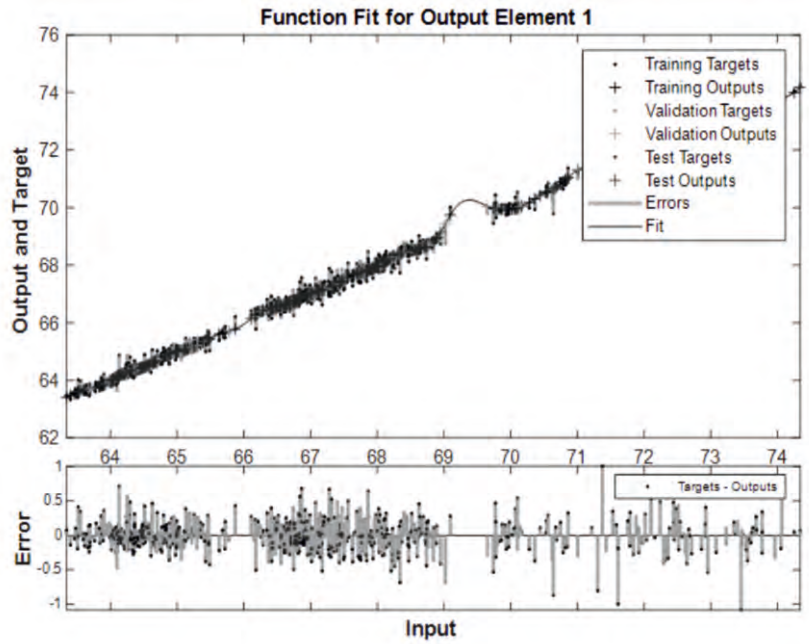


Figure 8. ***Function Fit Plot for Input-Output Fitting***



Discussion

After testing different networks, it can be concluded that the ANN with a single hidden layer having 17 nodes and

trained using the Levenberg – Marquardt algorithm gives the best performance in a minimum number of iterations. The model has a sigmoid activation function in the hidden layer and a linear activation function in the output layer. The test data set results of the model show the *R*-value of 0.9956, MSE = 5.29e-2, training epochs/iterations = 2, and validation gradient = 0.0045. These prove the generalizability of the model LM 17.

This study marks an extensive examination of ANN modeling. It is demonstrated that deep learning could help build an accurate prediction model for a challenging variable like "Forex rate." A few points to note for careful future handling of research is that sometimes if the network is re-initialized and re-trained, the results show improvement since each initialization has different network parameters. Another way to improve the results is to increase the number of hidden neurons. This increase helps make the network more flexible (since more parameters now need to be optimized). A point to note here is that an excessive increase in hidden neurons leads to the problem of overfitting. Another alternative way to improve results is to use different training functions (algorithms). One can also increase the data set. Additional data for training will produce better results. In the case of dynamic time series modeling, back-propagation takes more time to train the network since, here, the error surfaces are more complex due to feedback loops. If there exists a significant correlation among errors, the model should be improved by increasing the number of delays in tapped lines. The results can also be enhanced by taking additional factor series for prediction.

Managerial and Theoretical Implications

This model and any other improvised versions can be used by traders, investors, financiers, economists, bankers, speculators, hedgers, etc., to get an insight into future forex rates and thus make profitable decisions. This will ease the discounting of investment and help in future planning/retirement planning by common investors. The model can also help the Central banks and the government in deciding import-export policies, direct-indirect taxation rates, and also repo-reverse repo rates. The amount of liquidity to be maintained in the system and the amount of loans to be allowed by banks etc., can also be revised having the foresight of future exchange rates. Even the rates of certain commodities like oil and gold can be tweaked occasionally to help regulate the balance of trade. Various FII and FDI policies can be framed keeping in view the future expected foreign exchange rate. The minimum/maximum limit of FII can be decided with some future certainty, and this can thus help provide a cushion to the country's economic system.

Limitations of the Study and Scope for Further Research

I have developed a model for just one variable, and future studies can also be done on other variables, stock indices, commodities, etc. The model modifications can also use recurrent neural networks for better results. Future studies can test these models in simulated trading and help establish an alliance between statistical significance and economic significance.

Author's Contribution

Dr. Sonali Agarwal conceived the idea of developing an accurate predictive model for the forex rate and developed a research design to undertake this empirical study. She extracted research papers with high repute, filtered these based on keywords, and shortlisted the analytical methods. She wrote the manuscript by considering the existing literature related to the topic and reporting all the empirical results. She also highlighted the importance of the research, its managerial implications, and the scope for further studies.

Conflict of Interest

The author certifies that she has no affiliations with or involvement in any organization or entity with any financial interest or non-financial interest in the subject matter or materials discussed in this manuscript.

Funding Acknowledgement

The author received no financial support for this manuscript's research, authorship, and/or publication.

References

Agarwal, S., & Khan, J. (2019). Neural nets for stock indices: Investigating effect of change in hyperparameters. *Theoretical Economics Letters, 9*(3), 511–529. https://doi.org/10.4236/tel.2019.93036

Agarwal, S. (2019). Mutual funds are subject to market risks: Empirical evidence from India. *The Journal of Wealth Management*, *22*(2), 66 – 85. https://doi.org/10.3905/jwm.2019.22.2.066

Agarwal, S. (2022). Deep learning-based sentiment analysis: Establishing customer dimension as the lifeblood of business management. *Global Business Review*, *23*(1), 119–136. https://doi.org/10.1177/0972150919845160

Chang, P.-C., Liu, C.-H., Lin, J.-L., Fan, C.-Y., & Ng, C. S. (2009). A neural network with a case based dynamic window for stock trading prediction. *Expert Systems with Applications, 36*(3), 6889–6898. https://doi.org/10.1016/j.eswa.2008.08.077

Cheng, J.-H., Chen, H-P., & Lin, Y-M. (2010). A hybrid forecast marketing timing model based on probabilistic neural network, rough set and C4.5. *Expert Systems with Applications, 37*(3), 1814–1820. https://doi.org/10.1016/j.eswa.2009.07.019

Fischer, T., & Krauss, C. (2018). Deep learning with long short-term memory networks for financial market predictions. *European Journal of Operational Research, 270*(2), 654–669. https://doi.org/10.1016/j.ejor.2017.11.054

Fletcher, D., & Goss, E. (1993). Forecasting with neural networks : An application using bankruptcy data. *Information & Management, 24*(3), 159–167. https://doi.org/10.1016/0378-7206(93)90064-Z

Ghiassi, M., & Saidane, H. (2005). A dynamic architecture for artificial neural networks. *Neurocomputing, 63,* 397–413. https://doi.org/10.1016/j.neucom.2004.03.014

Ghiassi, M., Saidane, H., & Zimbra, D. K. (2005). A dynamic artificial neural network model for forecasting time series events. *International Journal of Forecasting, 21*(2), 341–362. https://doi.org/10.1016/j.ijforecast.2004.10.008

Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep learning*. MIT Press. http://www.deeplearningbook.org

Gupta, N., & Dalal, Y. (2022). Reconnoitering price discovery and market efficiency process among Indian HRITHIK stocks using VAR causality and VECM tests. *Indian Journal of Finance, 16*(2), 37–50. https://doi.org/10.17010/ijf/2022/v16i2/162434

Guresen, E., Kayakutlu, G., & Daim, T. U. (2011). Using artificial neural network models in stock market index prediction. *Expert Systems with Applications, 38*(8), 10389–10397. https://doi.org/10.1016/j.eswa.2011.02.068

Hamzaçebi, C., Akay, D., & Kutay, F. (2009). Comparison of direct and iterative artificial neural network forecast approaches in multi-periodic time series forecasting. *Expert Systems with Applications, 36*(2), 3839–3844. https://doi.org/10.1016/j.eswa.2008.02.042

Hong, Y., & Lee, T.-H. (2003). Inference on predictability of foreign exchange rates via generalized spectrum and nonlinear time series models. *The Review of Economics and Statistics, 85*(4), 1048–1062. https://doi.org/10.1162/003465303772815925

Hu, M. Y., Zhang, G., Jiang, C. X., & Patuwo, B. E. (1999). A cross-validation analysis of neural network out-of-sample performance in exchange rate forecasting. *Decision Sciences, 30*(1), 197–216. https://doi.org/10.1111/j.1540-5915.1999.tb01606.x

Huck, N. (2019). Large data sets and machine learning: Applications to statistical arbitrage. *European Journal of Operational Research, 278*(1), 330–342. https://doi.org/10.1016/j.ejor.2019.04.013

Joshi, N. (2021). Volatility, open interest, and trading volume in Indian futures markets. *Indian Journal of Finance, 15*(11), 41–54. https://doi.org/10.17010/ijf/2021/v15i11/166831

Kayacan, E., Ulutas, B., & Kaynak, O. (2010). Grey system theory-based models in time series prediction. *Expert Systems with Applications, 37*(2), 1784–1789. https://doi.org/10.1016/j.eswa.2009.07.064

Kiani, K. M. (2005). Detecting business cycle asymmetries using artificial neural networks and time series models. *Computational Economics, 26,* 65–89. https://doi.org/10.1007/s10614-005-7366-2

Kim, A., Yang, Y., Lessmann, S., Ma, T., Sung M.-C., & Johnson, J. E. (2020). Can deep learning predict risky retail investors? A case study in financial risk behavior forecasting. *European Journal of Operational Research*, 283(1), 217–234. https://doi.org/10.1016/j.ejor.2019.11.007

Kuan, C.-M., & White, H. (1994). Artificial neural networks: An econometric perspective. *Econometric Reviews, 13*(1), 1–91. https://doi.org/10.1080/07474939408800273

Lenard, M. J., Alam, P., & 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), 206–227. https://doi.org/10.1111/j.1540-5915.1995.tb01426.x

Liao, Z., & Wang, J. (2010). Forecasting model of global stock index by stochastic time effective neural network. *Expert Systems with Applications, 37*(1), 834–841. https://doi.org/10.1016/j.eswa.2009.05.086

Lyons, R. K. (2001). New perspective on FX markets: Order-flow analysis. *International Finance, 4*(2), 303–320. https://doi.org/10.1111/1468-2362.00075

Medeiros, M. C., Veiga, A., & Pedreira, C. E. (2001). Modeling exchange rates: Smooth transitions, neural networks, and linear models. *IEEE Transactions on Neural Networks, 12*(4), 755–764. https://doi.org/10.1109/72.935089

Moghaddam, A. H., Moghaddam, M. H., & Esfandyari, M. (2016). Stock market index prediction using artificial neural network. *Journal of Economics, Finance and Administrative Science, 21*(41), 89–93. https://doi.org/10.1016/j.jefas.2016.07.002

Sager, M. J., & Taylor, M. P. (2006). Under the microscope: The structure of the foreign exchange market. *International Journal of Finance & Economics, 11*(1), 81–95. https://doi.org/10.1002/ijfe.277

Salchenberger, L. M., Cinar, E. M., & Lash, N. A. (1992). Neural networks: A new tool for predicting thrift failures. *Decision Sciences, 23*(4), 899–916. https://doi.org/10.1111/j.1540-5915.1992.tb00425.x

Selvamuthu, D., Kumar, V., & Mishra, A. (2019). Indian stock market prediction using artificial neural networks on tick data. *Financial Innovation, 5*, Article 16. https://doi.org/10.1186/s40854-019-0131-7

Seth, N., & Sidhu, A. (2021). Price discovery and volatility spillover for Indian energy futures market in the pre-and post-crisis periods. *Indian Journal of Finance, 15*(8), 24–39. https://doi.org/10.17010/ijf/2021/v15i8/165816

Shahvaroughi Farahani, M., & Razavi Hajiagha, S. H. (2021). Forecasting stock price using integrated artificial neural network and metaheuristic algorithms compared to time series models. *Soft Computing*, *25*, 8483 – 8513. https://doi.org/10.1007/s00500-021-05775-5

Singh, A., Jain, M., Jain, S., & Gupta, B. (2021). A new modus operandi for determining post - IPO pricing : Analysis of Indian IPOs using artificial neural networks. *Indian Journal of Finance, 15*(1), 8–22. https://doi.org/10.17010/ijf/2021/v15i1/157011

Swanson, N. R., & 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 & Economic Statistics, 13*(3), 265–275. https://doi.org/10.1080/07350015.1995.10524600

Swanson, N. R., & White, H. (1997a). A model selection approach to real-time macroeconomic forecasting using linear models and artificial neural networks. *The Review of Economics and Statistics, 79*(4), 540–550. https://doi.org/10.1162/003465397557123

Swanson, N. R., & White, H. (1997b). Forecasting economic time series using flexible versus fixed specification and linear versus nonlinear econometric models. *International Journal of Forecasting, 13*(4), 439 – 461. https://doi.org/10.1016/S0169-2070(97)00030-7

Tyree, E. W., & Long, J. A. (1995). Forecasting currency exchange rates: Neural networks and the random walk model. In, *Proceedings of the Third International Conference on Artificial Intelligence Applications*, New York. http://citeseer.nj.nec.com/131893.html

Villada-Duque, F., López-Lezama, J. M., & Barrientos-Marín, J. (2020). Forecasting prices in financial markets using artificial neural networks. *International Journal of Engineering Research and Technology, 13*(11), 3247–3250. https://doi.org/10.37624/IJERT/13.11.2020.3247-3250

Zhang, G., & Hu, M. Y. (1998). Neural network forecasting of the British pound/US dollar exchange rate. *Omega, 26*(4), 495–506. https://doi.org/10.1016/S0305-0483(98)00003-6

Zurada, J. M., Foster, B. P., Ward, T. J., & Barker, R. M. (1999). Neural networks versus logit regression model for predicting financial distress response variables. *Journal of Applied Business Research, 15*(1), 21–30. https://doi.org/10.19030/jabr.v15i1.5685

About the Author

Dr. Sonali Agarwal is currently an Assistant Professor at USHSS, Economics department. She has done a Ph.D. in finance. Her area of expertise is finance, econometrics, machine learning, statistics, and mathematics. She has 22 research publications in various national and international journals. She has guided many master's students in their research projects. She is always enthusiastic about taking up new research projects and exploring new avenues related to finance.