ARE EXCHANGE RATES PREDICTABLE? EXCHANGE RATE PREDICTION MODEL WITH ANN FOR TURKEY

Fuat Sekmen

Sakarya University,Faculty of Political Science, Turkey, sekmen@sakarya.edu.tr

Haşmet Gökırmak, Istanbul Sabahattin Zaim University, İstanbul, Turkey
Murat Kürkcü, Republic of TurkeyMinistry of Treasury and Finance, Turkey
Şükrü Apaydın, Nevsehir Hacı Bektas Veli University, Turkey
Hasan Memiş, Balikesir University, Turkey

ABSTRACT

The forecasting concept can be expressed as predicting a variable's future values using various methods under certain assumptions. In the estimations made using the time series, the value of a variable in the current period and the past period values are used. In the time series analysis, past time values of the variable to be estimated constitute the model's independent variables. Learning from data, generalizing, working with very large samples, etc. are essential Artificial Neural Networks (ANN) methodology features. ANN methodology, which provides significant advantages thanks to these features, has vast usage possibilities in prediction modeling as in other fields. ANN has become one of the time series estimations' methods since the late 1980s. This study handles an artificial neural network learning paradigm to estimate the exchange rate for economic targets. This paradigm has not prior conditions such as normal distribution or stationary. Therefore it has been frequently preferred in time-series analyses.

The present study predicts the Turkish Lira's value against the U.S. Dollar by using an ANN methodology. Our study is built upon purchasing power parity and the different researches dealing with exchange rate predictions. Our primary purpose is to minimize the error between the expected output of the network and the output it produces. This study uses an Artificial Neural Network model that builds upon the MLP to estimate the exchange rates. We first define the MLP with one hidden layer for this model, using monthly data from 2000 to 2019.

In contrast, the independent variables are interest rates, Gross Domestic Product, and Consumer Price Index data for Turkey and the United States of America. We prefer batch training to train the ANN in SPSS Neural Networks 17.0 software. Sixty percent of the data were used in training, 25 percent in the testing, and 10 percent in the holdout, of the artificial neural networks, respectively. According to research results, the performance of ANN is relatively higher. While the relative error in the training set is 1 percent, the test set's relative error is below 1 percent (0,007).

Keywords: Artificial Neural Networks, Forecasting Methods, Multilayer Perceptron, Learning Algorithms

1. INTRODUCTION

Forecasting is predicting the future values of a variable by various methods, given some assumptions. The time series uses a variable's value in the current period and the past period values in the estimations. The past time values of the dependent variable are the model's independent variables.

Artificial Neural Networks (ANN), which operate like a human brain, have an important place in Artificial Intelligence Technologies. ANN methodology can learn from data, generalize results, and work with an unlimited number of variables. ANN methodology is widely used in prediction modeling in many fields of study.

The artificial neurons have firstly been modeled by Warren McCulloch and Walter Pitts in 1943. But it is possible to see new and more sophisticated models of artificial neurons in the present day. Perceptron was first used by Rosenblatt (1962) to model visual perception (retina). This error propagation model or backpropagation algorithm developed by Rumelhart et al. is an essential solution tool, especially for problems that require classification, recognition, and generalization. This model uses a learning method called "Delta Learning Rule." Its primary purpose is to minimize the error between the expected output of the network and the output it produces. This network is called an error propagation network since it does this by spreading the network's error.

ANN has been used in time series estimations since the late 1980s. ANN methodology has been generally used in classification problems and time-series predictions in the economics literature for forecasting foreign exchange rates, stock market indexes, etc. For example, Nag and Mitra (1999), Franck and Schmied (2003), Çelik and Karatepe (2007), and Fioramanti (2008) have used an ANN model to compare with different models' results or estimate financial crises. Roh (2007), Zhu et al. (2008), Yudong and Lenan (2009), Bildirici and Ersin (2009), Guresen et al. (2011) examined stock market indexes by using ANN. Also, Pekkaya and Hamzaçebi (2007), Preminger and Franck (2007), and Zhang and Wan (2007) have used an ANN model for forecasting exchange rates.

Yu, Wang, and Lai (2007) reviewed 45 articles on exchange rates forecasting with ANN and concluded that ANNs perform better than the other models in 60% of these articles. According to Swanson and White (1997), ANN improves macroeconomic variables' prediction results. Ataseven (2013) concluded that ANN models resulted in lower errors than the PETKİM and Box-Jenkins models. Adewole et al. (2011) forecasted foreign exchange rates with an ANN model and concluded that their model achieved 81.2% accuracy.

The present study aims to predict the Turkish Lira's value against the U.S. Dollar by using an ANN Model with a multi-layer perceptron (MLP). Section 2 presents the ANN literature and the main properties of ANNs. Section 3 describes the MLP methodology and data. Section 4 describes the empirical analysis. Section 5 summarizes our findings.

2. LITERATURE REVIEW

Simplified biological neural network models form the essence of the neural networks. According to Graupe (2007), neurons called nerve cells are the essential biological neural network elements. Nervous systems, which perform different functions, composed of neural cells or neurons. The neural networks model information processing by nervous systems. The models explaining the structure of the human brain and the nervous systems of animals have different features. Providing control by transmitting information is based on the functional design of neural networks. Neurons are highly slow compared to ANN, but the human brain is also better than any digital computer at solving problems. ANN resembles the brain in point of acquiring and storing knowledge. ANN model is used to test the relationship between dependent and independent variables. The model's structure is generally determined by external or internal information provided by the network at the learning phase. The essential biological and ANN elements are listed in Table 1.

Table 1: The Basic Elements of Biological and Artificial Neural Networks

Biological	**Artificial**
Soma	Unit
Axon and Dendrite	Connections
Synapse	Weight
Signal	Activation

The biological model's basic structure consists of dendrites, synapses, cell bodies, and axons. The connection points between different neurons, synapses, store information at the biological neural networks. Similarly, artificial neurons have input channels and an output channel. Artificial neurons were first used in the 1940s and 1950s. Mc Culloch and Pitts (1943) and Rosenblatt (1959) used the perceptron algorithm, which is also termed the single-layer perceptron. In the 1960s, the single-layer perceptron was found to be insufficient to solve many problems. Thus, the applications of ANN started to lose its popularity for a while. However, ANN became popular with the backpropagation algorithm, a new learning procedure introduced by Rumelhart et al. (1986). This algorithm's main feature is that the connection

weights are changed continuously to reduce the difference between the desired and actual outcomes.

The perceptron and adaptive linear neuron (ADALINE) can be the most well-known feedforward network. Hopfield (1982), Kohonen (1984), and Anderson (1995) have presented the first examples of recurrent networks. Recurrent networks have a more dynamic structure than feedforward networks.

Although ANN does not explain the degree of relationship between variables better than the time series models, it has a strong predictive ability. Since the relationship between parameters cannot be fully explained in the model, The ANN methodology is also referred to as a black box methodology in the literature. The ANN methodology can also produce incorrect results due to excessive fitting.

There is a hierarchical multi-layered structure in the neural networks. ANN consists of three different layers, including input, hidden, and output layers. Independent variables called inputs are the main elements of the input layer. Independent variables from the input layer to the hidden layer units are passed to the hidden layer by weighting. All of the inputs are summed altogether by being modified through the weights. Then, the output layer's estimation values are obtained by converting them into output through activation functions. Figure 1 presents the architectural structure of an artificial neuron graphically.

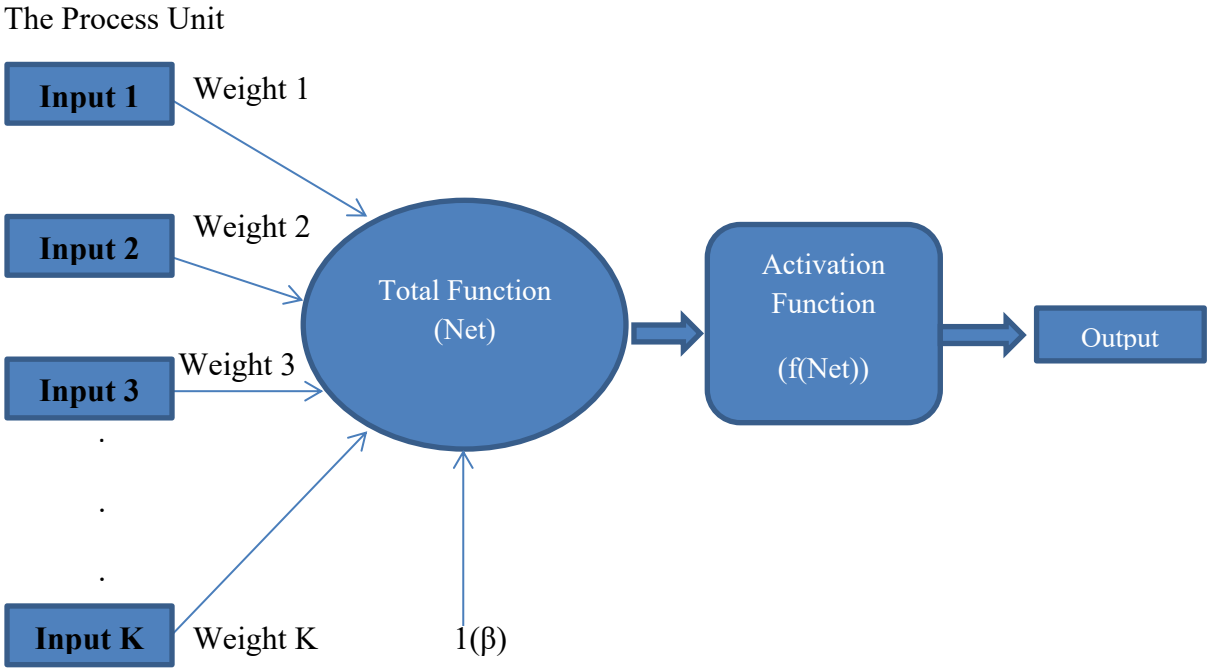

Figure 1: The architectural structure of the typical artificial neuron

Input1, K: The inputs of the neuron

Weight1, K: The connection weights

1(β): It is the bias of the network output

$$\text{Net} = \sum_{k=1}^{n} (W_k I_k) + \beta \qquad (1)$$

The neuron's output is determined according to the activation function's outcome using the Net's value. The connection pattern can be represented mathematically in Equation 1. The connection weights represent the strengths of the connections. The inputs and outputs of the network reveal the directions. The input of each neuron has a different weight in artificial neural networks. This structure is comprised of a summation and a logistic (sigmoid or linear) function f(Net). Many artificial neural network architectures have been proposed in the literature. The feedforward neural network (FNN) is the most predominant of the artificial neural network architectures.

The reliability of the results is one of the most determining ANN issues of the network's learning ability. A training set consisting of different examples to the network is presented for achieving the network's learning ability. The learning algorithm performed by changing the connection weights enables the network to reproduce a correct result with the correct input values. As a black-box model, ANN has two main difficulties, generalizing the results and deciding how long the network should be trained (Alavala, 2008).

In recent years, ANN's interest has covered different research areas, from psychology to computer science, engineering to economics, etc. Unlike other machine learning techniques, ANN can be modified according to the conditions. This situation has brought flexibility to neural networks in quickly solving problems of different sizes. To demonstrate nonlinear system models is an essential capability of neural network architectures. They can handle large amounts of data, perform collective processing, and quickly learn from the data (Lin and Hoft, 1994).

3. METHODOLOGY AND DATA

A simple network can represent a relationship between the output and the input. The MLP with one hidden layer used in this paper has the following form. Perceptron learning rule assumes that we have a set of learning samples consisting of an input vector and the desired output. The perceptron learning rule can be summarized as follows:

1. Start with random weights for the connections;

2. Select randomly or sequentially an input vector from the set of training samples;

3. If the MLP network gives an undesired output, modify all connections;

4. And go back to the second step.

It is used with the MLP for the proposed forecasting of the foreign exchange rate. The data has been obtained from the Central Bank of the Republic of Turkey and the International Monetary Fund (IMF), International Financial Statistics datasets. The data set consists of 237 monthly data from January 2000 to September 2019. Inputs included the gross domestic product, interest rate, and consumer price index for Turkey and the U.S. The model uses 146 data points for training, 34 data for validation, and 57 data for testing. Table 2 presents a summary of the data.

Table 2: Case Processing Summary

		N	Percent
Sample	Training	146	61,6%
	Testing	57	24,1%
	Holdout	34	14,3%
Valid		237	100,0%
Excluded		3	
Total		240	

The foreign exchange rates generally have uncertainty and instability in nature. Nevertheless, the present statistical models cannot present effective enough solutions for forecasting foreign exchange rates. In this study, an ANN model was designed for predicting foreign exchange rates. The model has a two-phase structure. The first phase of the model is the training of the artificial neural network, and the second phase is the forecasting of the foreign exchange rate through the trained ANN. In the training phase, the backpropagation algorithm was used for the network to learn and achieve appropriate outcomes. The learning weights were randomly determined to achieve a consistent and reliable desired outcome. A feedforward network was used to improve the efficiency of the backpropagation. The desired outcome of the MLP Network designed for forecasting was obtained by using SPSS. The network consists of three layers: an input layer, a hidden layer, and an output layer.

3.1. Multi-layer Perceptron Network

The MLP can be defined as a function of predictors minimizing the dependent variable's prediction error. Principe et al. (1999) expressed the MLP has two essential characteristics. These are nonlinear processing elements and their massive interconnectivity. According to Principe et al. (1999), MLPs are trained with the backpropagation algorithm. This algorithm deals with the errors in the network by allowing the adaptation of the hidden processing elements. The MLP is trained with error correction. The error correction is used for the network to produce the desired response.

According to Fine (1999), the neural network methodology is especially useful in explaining the input-output relationships in large samples. The most common multi-layer ANN is the Multi-layer Perceptron. Networks with one or several hidden layers are generally referred to as multi-layer ANNs. It is possible to classify ANN according to their structures and learning

algorithms. In this context, artificial neural network architectures can be examined in two separate categories, feedforward and recurrent networks. The directions of the connections between the nerves or the flow directions of the network's signs determine these categories.

In feedforward networks, the processor elements are usually divided into layers. The signals are transmitted from the input layer to the output layer with one-way connections. In feedforward ANN, neurons are arranged in layers, and the outputs of cells in one layer are given as input to the next layer via weights. The input layer transmits the information it receives from outside environments to the hidden layer cells without any changes. The network output is determined by processing the information in the middle and output layer.

Although an artificial neural network structure and the number of nerve cells vary, there are no accepted rules for forming an artificial neural network. While ANN with less than the required hidden layers is insufficient in solving complicated problems, ANN with too many hidden layers encounters undesirable instability. After determining the number of hidden layers, the issue came across when deciding how many neurons will be present in each layer. The neuron number in the input layer is equal to the number of entries in the system. Likewise, the output layer can be determined by the desired output number. The main problem is to determine the number of neurons in hidden layers. Unfortunately, there is no mathematical test of how many neurons will most efficiently be found in the hidden layer. The decision must be taken by applying the trial and error method.

There are various learning concepts in the literature. It can be expressed as symbolic processing, connectionist systems, statistical pattern recognition, genetic algorithms and evolutionary programming, and case-based learning. ANN are learning methods based on learning from examples. It is aimed to understand the relationships between inputs and outputs of an event by using examples realized in such learning methods and these relations in determining the outcomes of new instances that will occur later. MLP networks have adopted the teacher learning strategy.

The backpropagation algorithm is widely used in neural network training. A momentum term is often added to the backpropagation to accelerate and stabilize the learning procedure. The momentum term improves the convergence speed by some Eigen components of the system closer to critical damping.

Notation:

$X^{(m)} = (x_1^{(m)}, \ldots\ldots\ldots x_p^{(m)}$	Inputs
$Y^{(m)} = (y_1^{(m)}, \ldots\ldots\ldots y_R^{(m)}$	Output
I	Number of Layers
J_i	Number of units in layer i.

$a_{i:j}^{m}$	Unit j layer i.
$w_{i:j,k}$	Weight leading from layer i-1, unit j to layer i, unit k.
$y_i(c)$	Activation function for layer i.
w	Weight
E	Error

These neurons are organized in layers, as shown in Figure 2. Scaled data enters the network at neurons of the input layer and is propagated to the output through intermediate layers.

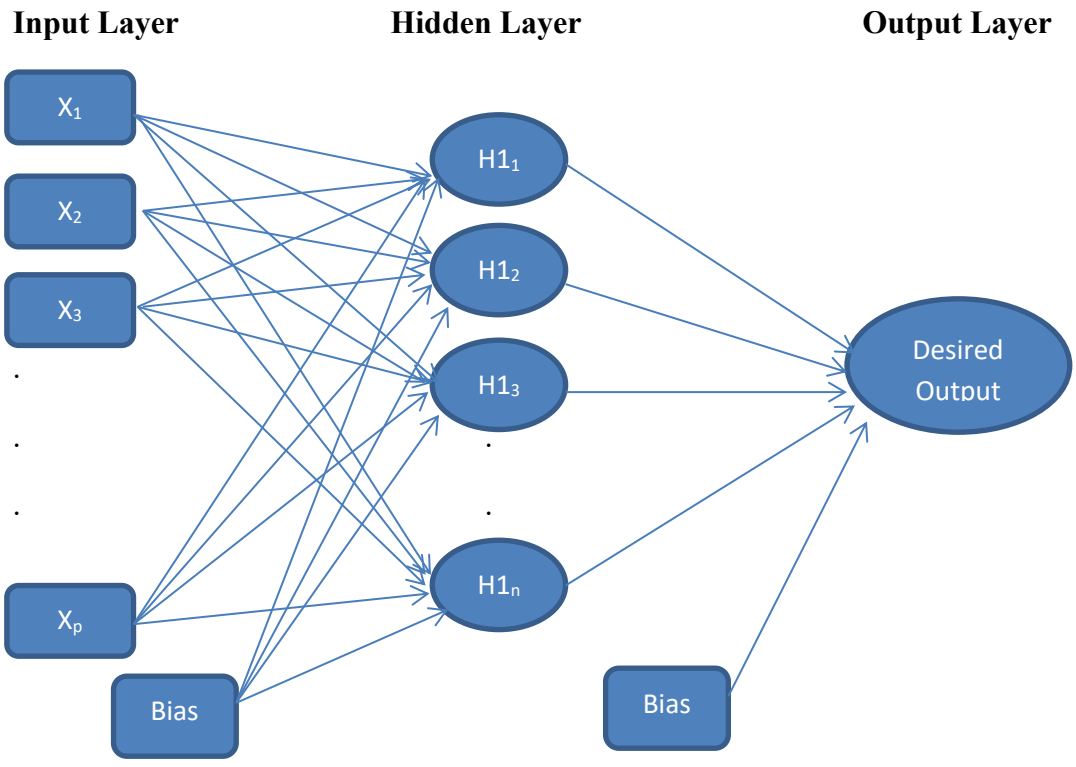

Figure 2: The General Architecture of the Multilayer Perceptron:

The general structure of the MLP network with a single hidden layer is as in figure 2. The layers that make up the general structure of the network can be expressed as follows:

Input Layer:

$$j_0 = P\ units, a_{0:1}, \ldots., a_{0:j_0}; with\ a_{0:j} = x_j. \quad (2)$$

Hidden Layer:

$$j_i\ units, a_{i:1}, \ldots., a_{i:j_i}; with\ a_{i:k} = y_i(c_{i:k})\ and\ c_{i:k} = \sum_{j=0}^{j_{i-1}} w_{i:j}, k^{a_{i-1:j}}\ where\ a_{i-1:0} = 1 \quad (3)$$

Output Layer: The pattern index and each layer's bias term are not counted in the total number of units for that layer.

$$j_I = R\ units, a_{I:1}, \ldots., a_{I:j_I}; with\ a_{I:k} = y_I(c_{I:k})\ and\ c_{I:k} = \sum_{j=0}^{j_1} w_{I:j}, k^{a_{i-1:j}}\ where\ a_{i-1:0} = 1 \quad (4)$$

The cases in the input layer can be presented to the MLP network randomly or sequentially. These covariates in the input layer can be delivered to the network by scaling or without any scaling. While scaling, the following methods can be used: Standardized, normalized, and adjusted normalized.

Activation Functions (MLP algorithms)

Hyperbolic Tangent

$$y(c) = tan_h(c) = \frac{e^c - e^{-c}}{e^c + e^{-c}} \quad (5)$$

Error Functions (MLP algorithms)

Sum-of-Squares

$$E_T(w) = \sum_{m=1}^{M} E_m(w) \quad (6)$$

where

$$E_m(w) = \frac{1}{2}\sum_{r=1}^{R}(y_r^{(m)} - a_{I:r}^m)^2 \quad (7)$$

Error Backpropagation (MLP algorithms)

Error-back propagation is used to compute the first partial derivatives of the error function with respect to the weights.

First note that $y'(c) = 1 - [y(c)]^2 \quad \tanh \quad (8)$

The backpropagation algorithm follows:

For each i, j, k, $\frac{\partial E_T}{\partial w_{i:k,J}} = 0$

For each m in group T; For each p=1,..., J.I., let

$\delta_{I:p}^m = \frac{\partial E_m}{\partial c_{I:p}^m} = a_{I:p}^m - y_p^{(m)}$ if cross-entropy error is used

$\delta_{I:p}^m = \frac{\partial E_m}{\partial c_{I:p}^m} = y'(c_{I:p}^m)(a_{I:p}^m - y_p^{(m)})$ otherwise

For each i=I,...,1 (start from the output layer); For each j=1,...,J_i; For each k=0,...,J_{i-1}

Let $\frac{\partial E_m}{\partial w_{i:k,j}} = \delta_{i:j}^m \, a_{i-1:k}^m, where \; \delta_{i:j}^m = \frac{\partial E_m}{\partial c_{i:j}^m}$

Set $\frac{\partial E_T}{\partial w_{i:k,j}} = \frac{\partial E_T}{\partial w_{i:k,j}} + \frac{\partial E_m}{\partial w_{i:k,j}}$

If k > 0 and i > 1, set $\delta_{i-1:k}^m = y'_{i-1}(c_{i-1:k}^m) \sum_{j=1}^{j_i} \delta_{i:j}^m w_{i:k,j}$

This gives us a vector of elements that form the gradient of E. T. (w_k).

$$\sum_{i=0}^{I-1} (j_i + 1) j_{i+1}$$

The feedforward MLP will predict the U.S. dollar/T.L. exchange rate; hence, only the MLP operation will be handled in this study.

3.2 Training of Artificial Neural Networks

Each connection has associated with it a weighting which acts to modify the signal strength. There are various techniques for optimizing criterion functions to train the neural network. A critical characteristic of neural network classifiers is that their training usually requires iterative techniques.

The backpropagation classifier is the most popular technique for training by using the gradient descent method. This method's advantage is that it is simple and easy to understand, but the disadvantage is that convergent speed is slow. The backpropagation classifier is not so robust. The backpropagation algorithm is now introduced.

In the training phase, the weights are updated using the standard backward error propagation algorithm, also known as the generalized delta rule. The system first employs the input vector to produce its output vector and then compares this with the desired output or target vector. If there is no difference, no learning takes place. Otherwise, the weights are changed to reduce the difference. Now the problem is to minimize the total mean-squared error *E.T.* given by E

$$E_T(w) = \frac{1}{2}\sum_{r=1}^{R}(y_r^{(m)} - a_{l:r}^m)^2$$

This error can be minimized by taking partial derivatives of *E.T.* with respect to each weight.

Many successful results have been obtained by applying neural network techniques for the problems in different areas to be modeled and forecasted. There has been a major historical breakthrough in using the backpropagation method for learning the interconnection weights either off-line or online from patterns after the introduction of MLP architectures (Suykens, 2002).

During the training, examples can be shown to the network in two ways. Either a pattern is presented to the network, and the weights are adapted, or after all, patterns are shown, the weights are updated. This is called batch learning. The process continues until some stopping criterion is met. The validation set is used to decide when to stop training (Suykens, 2002). The test set not used within the training and validation process is generally used to check the trained model's performance. According to Haykin (1999), the ANN training should be stopped when the neural network's performance declines in the validation set. According to Principe et al. (1999), various methods, such as normalizing training data, the hyperbolic tangent instead of the logistic function, the larger step size, more sophisticated learning methods, or more (training) patterns than weights, can be used to reduce the training times or to increase the performance of the network.

The training algorithm is used to specify how the network should be trained. The types of training and the optimization algorithms determine which training options are available. The kind of training determines how the network processes the records. We can select one of the batches, online, and mini-batch training types of training the ANN in the SPSS ANN 17.0 software.

3.3 The Learning Process in ANN

The learning situations in ANN can be classified into two distinct categories as supervised and unsupervised learning. In the supervised learning known as associative learning, the input and output patterns provided by a teacher or the system are used by matching these input-output pairs during ANN training. In unsupervised learning, the system is supposed to discover statistically salient features of the input population. According to Alavala (2008), the patterns have no initial set of categories classified as dissimilar to the supervised learning paradigm.

The learning process determines how the relationships between the dependent and independent variables should be. According to a linear or nonlinear relationship between the dependent and independent variables, the correct model structure will be automatically determined by the neural network. However, we need to keep in mind that this flexibility has the disadvantage

that synaptic weights are not easily interpreted. Thus, whether to use a more traditional model or a neural network should be decided based on the importance of model reliability or explaining the relationship between dependent and independent variables.

The multi-layered network's learning rule is the generalized version of the "Delta Learning Rule" based on the least-squares method. Therefore, it is also called as "Generalized Delta Rule." A set consisting of examples, called a training set, is needed for the network to learn. In this set, both the network's inputs for each sample and the outputs are determined.

The operation of perceptron-related models using threshold units can be broken down into two modes: processing and learning. In the processing mode, an input pattern is presented to the perceptron. The output is determined by the input signals, current weights, and thresholds. The weights and thresholds, therefore, implement a functional mapping between input patterns and output signals. In the learning mode, the input weights and threshold arc are modified in response to the target signal. The specifics of the weight modification process constitute the learning rule.

The generalized delta rule (Rumelhart et al., 1986) specifies a family of rules that can be used to learn the weights in a perceptron. The generalized delta rule differs from the classical perceptron in treating the target or desired response signal. The target signal can be given or derived from some credit assignment strategy, such as backward error propagation. The source of the target signal is irrelevant here. In the classical perceptron, the target signal is compared to the actual output to generate the error signal. The generalized delta rule compares the target signal to the activity to create the error signal. The activity is defined as the weighted sum of the inputs.

The generalized delta rule is a mathematically derived formula used to update a neural network during a (backpropagation) training step. A neural network learns a function that maps an input to an output based on given example pairs of inputs and outputs. A set number of input and output pairs are presented repeatedly in random order during the training. The generalized delta rule is used repeatedly during training to modify weights between node connections. Before training, the network has connection weights initialized with small, random numbers. The weight modifications aim to reduce the overall network error, reducing the difference between the actual and expected output. The generalized delta rule is important in creating useful networks capable of learning complex relations between inputs and outputs. This rule also has mathematical utility because it is mathematically derived, which is an advantage over other early learning rules.

The generalized delta rule consists of two stages: Feed Forward and Back Propagation

I) Feed Forward:

At this stage, information processing begins with the demonstration of an example from the input layer to the network. The incoming entries are sent to the middle layer without any changes.

The output (O_{ki}) of the kth Process element in the input layer is determined as follows:

$O_{ki} = I_k$

Each process element in the intermediate layer receives the information from all process elements in the input layer by affecting their connection weights (w_1, w_2,....,w_n). The net input (c_j^a) that comes to the process elements in the interlayer first is calculated by using the formula:

$$c_j^a = \sum_{k=1}^{n} w_{kj} o_k^i$$

w_{kj} shows the weight value of the connection that connects the K^{th} input element to the J^{th} middle layer element.

The output of the J^{th} interlayer element is calculated by passing this net-input through the activation function. The output of the J^{th} interlayer element varies depending on the activation function used.

The formula used for sigmoid activation function:

$$o_j^a = \frac{1}{1 + e^{-(c_j^a + \beta_j^a)}}$$

The formula used for hyperbolic tangent activation function:

$$o_j^a = tan_h(c) = \frac{e^{(c_j^a + \beta_j^a)} - e^{-(c_j^a + \beta_j^a)}}{e^{(c_j^a + \beta_j^a)} + e^{-(c_j^a + \beta_j^a)}}$$

βj shows the weight of the bias element connected to the Jth element in the middle layer. The output of this threshold unit is constant and equals 1. During training, the network determines this value itself.

The outputs of all process elements of the intermediate layer and the output layer are determined in the same way by calculating the input that came to them and passing it through the activation function. Thus, when the output layer's desired output values are found, the network's forward calculation process is completed.

II) Back Propagation

The network's output is compared with the network's expected outputs for the input presented to the network. The difference between them is considered an error. This error is distributed to the network's weight values, reducing the error in the next iteration. The error of the M^{th} process element in the output layer, E_m;

$E_m = D_m - A_m$

The total error (T.E.) for the output layer can be calculated as follows:

$TE=1/2(\Sigma E^2_m)$

This error should be distributed to the process elements that cause it to minimize the total error. When changing the web's weights, both the weights between the intermediate layer and the output layer and the weights between the intermediate layers or between the intermediate layer and the input layer are changed. If the change in weight of the connection connecting the j^{th} process element in the intermediate layer to the m^{th} process element in the output layer is called Δw^a; The amount of change in weight at any time t is calculated as:

$$\Delta w_{jm}^{a}(t) = \lambda \delta_m o_j^a + \alpha \Delta w_{jm}^{a}(t-1)$$

Here λ indicates the learning coefficient, α is the momentum coefficient, and δ_m is the error of the m^{th} output unit. The momentum coefficient ensures that the weight change value is added to the next change at a specific rate so that the network does not get stuck at a local optimum point during learning. After calculating the amount of change, the new values of the weights in the t^{th} iteration can be calculated as follows:

$$w_{jm}^{a}(t) = w_{jm}^{a}(t-1) + \Delta w_{jm}^{a}(t)$$

Similarly, it is necessary to change the weight of the threshold unit. All the network weights will be changed by changing the values between the input layer and the intermediate layer. And so the feedforward and backpropagation processes will be completed.

4. EMPIRICAL ANALYSIS ANN OF U.S. DOLLAR/TL EXCHANGE RATE SERIES

It has been developed an MLP network to predict the U.S. Dollar/T.L. exchange rate by using the monthly data from the era 2000m1- 2019m9. In the input layer, six variables thought to have significant effects on exchange rate movements were used. These variables are composed of interest rate (IRT), inflation (CPIT) and GDP (GDPT) in Turkey and interest rate, inflation, and GDP data in the United States as a foreign country. The variables were scaled (adjusted normalized: 0.02) to estimate the foreign exchange rate more consistently. This network has one hidden layer. There are five processing units in this layer, excluding the bias value unit.

The cases were randomly assigned to the network based on relative numbers of cases, as shown in Table 2. It has been used as the hyperbolic tangent as activation function among layers. Batch learning has been used as a type of training and scaled conjugate gradient as an optimization algorithm during the learning. The general structure of the MLP network composing for USD/TL exchange rate prediction is summarized in Table 3.

Table 3: The Multilayer Perceptron Network Information for USD/TL Exchange Rate Prediction

Input Layer	Covariates	1	IRT
		2	IRUS
		3	CPIUS
		4	CPIT
		5	GDPT
		6	GDPUS
		Number of Units[a]	6
		Rescaling Method for Covariates	Adjusted normalized
Hidden layer (s)		Number of Hidden Layers	1
		Number of Units in Hidden Layer 1[a]	5
		Activation Function	Hyperbolic tangent
Output Layer	Dependent Variables	1	USD ER
	Number of Units		1
	Rescaling Method for Scale Dependents		Adjusted normalized
	Activation Function		Hyperbolic tangent
	Error Function		Sum of Squares

a. Excluding the bias unit

The dependent variable USD ER in the output layer of the MLP network has been determined by the processing units in the hidden layer. These units form the output layer by subjecting them to specific weights and activation functions. The overall performance of the MLP network is measured by the error function “Sum of Squares.”

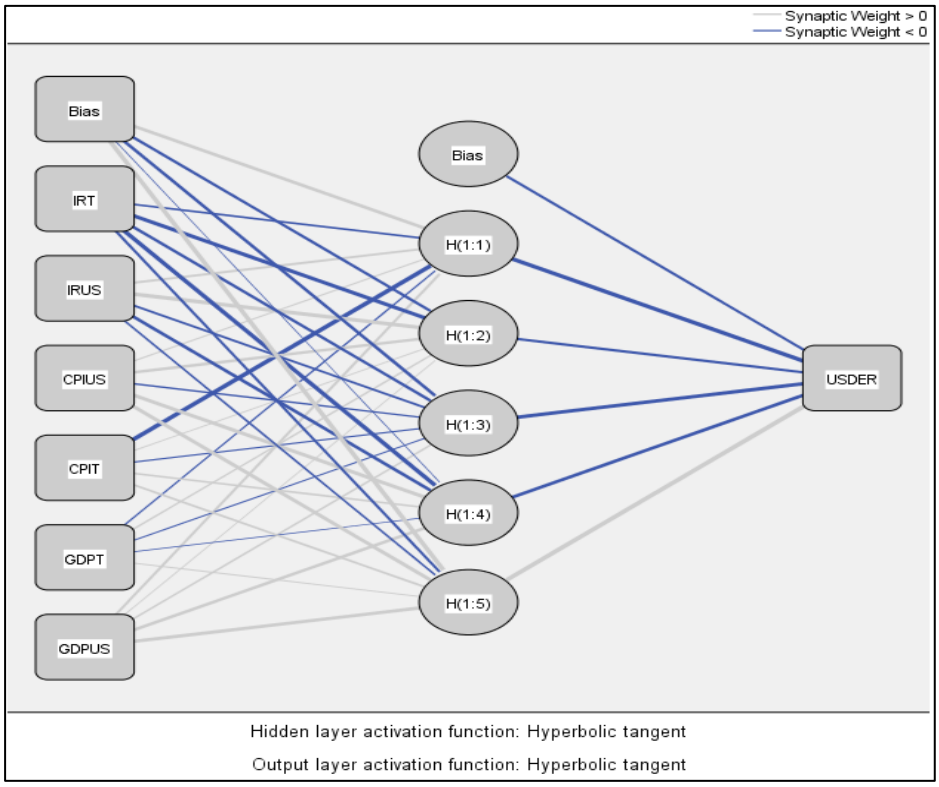

Figure 3: The distribution of synaptic weights among layers

Two essential determinants explain the relationship between the covariates in the input layer and the hidden layer's processing elements. The first of these is the connection weights known as synaptic weight. The latter is the activation function. The distribution of synaptic weights that can take positive and negative values between the input, hidden, and output layers is shown in Figure 3.

The conclusions of the USD\TL exchange rate prediction model has been indicated in table 4. With this model, significant and reliable results were obtained.

Table 4: Model Summary

Training	Sum of Squares Error	,104
	Relative Error	,008
	Stopping Rule Used	Maximum number of epochs (1200) exceeded
	Training Time	0:00:00.174
Testing	Sum of Squares Error	,032
	Relative Error	,007
Holdout	Relative Error	,012

Dependent Variable: USD Exchange Rate

Also, the level of error regarding the estimation results is very low. The most successful results in the model were achieved in 1200 epochs. The relative error in training, test, and holdout sets was 0.008, 0.007, and 0.012, respectively. Both the training set and test set produced similar results in terms of error percentages. In this respect, it can be stated that the model does not create a problem like overfitting.

The main goal of the model is to predict the desired output with the least error. The parameters estimated in this direction are as in table 5. Artificial neural networks, also known as the black box model, are insufficient to explain the relationship between dependent and independent variables while minimizing the estimation error. Hence, the predicted parameters cannot correctly describe the relationship between the ANN model's inputs and output. It can be said that this situation is undoubtedly one of the most critical constraints of the model. However, this situation can be ignored thanks to the strong predictive ability of the ANN model.

Table 5: Parameter Estimates

Predictor		Predicted					
		Hidden Layer 1					Output Layer
		H(1:1)	H(1:2)	H(1:3)	H(1:4)	H(1:5)	USDER
Input Layer	(Bias)	,835	-,585	-,751	-,021	1,853	
	IRT	-,374	-1,156	-,685	-1,482	-,497	
	IRUS	,478	1,360	-,404	-,803	-,338	
	CPIUS	,145	,728	-,285	1,018	,925	
	CPIT	-1,893	,102	-,233	,339	,397	
	GDPT	-,261	,282	-,116	-,057	,109	
	GDPUS	,482	,065	,333	,774	,896	
Hidden Layer 1	(Bias)						-,518
	H(1:1)						-1,882
	H(1:2)						-,585
	H(1:3)						-1,125
	H(1:4)						-,847
	H(1:5)						2,199

Using these parameters, the error is propagated between layers and minimized to obtain the best estimation result. These parameters are usually expressed as the weights of the ANN model. These parameters are updated during the learning process of the network according to the training algorithm used.

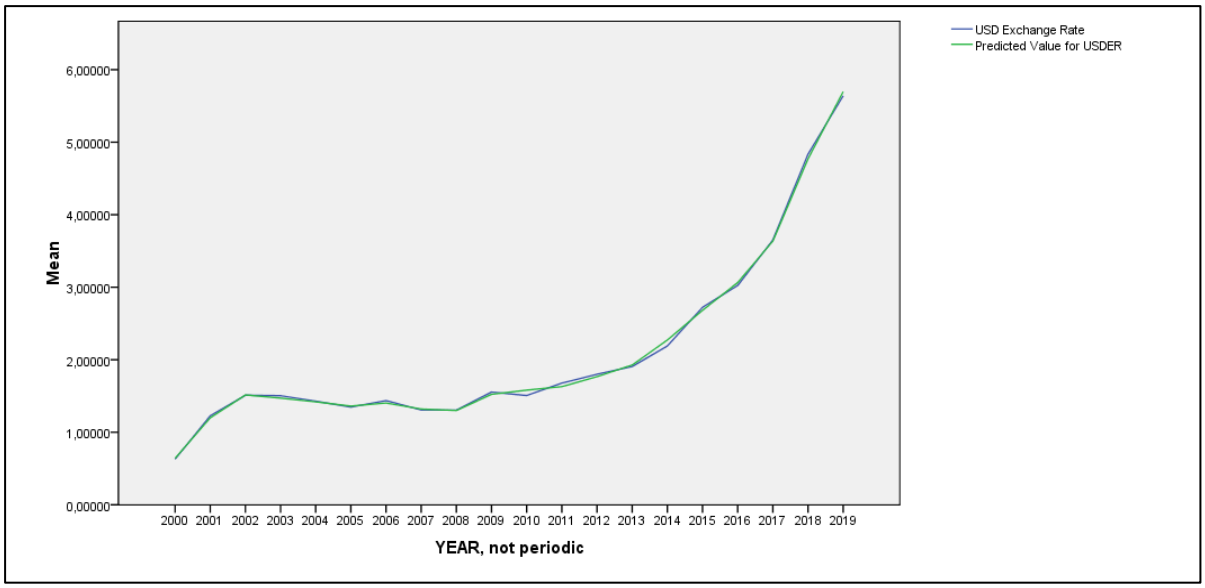

Figure 4: The expected and the realized exchange rate in the 2000-2019 period in Turkey

Extremely successful results have been achieved through ANN's used for forecasting the Turkish TL/US dollar exchange rate series, as seen in Figure 4. The expected and actual exchange rate series for the 2000-2019 periods are very close to each other.

5. CONCLUSION

It was developed an ANN model that builds upon the MLP to estimate USD/TL E.R. in this paper. This paper has used the forward feed model with one hidden layer through the MLP methodology. This methodology has been chosen mostly for its successful results in the literature. We defined an MLP with one hidden layer for this model. There are six parameters in the input layer of the model. It has five neurons on its single hidden layer and one output unit in the output layer. A threshold neuron, which has a constant input that is equal to 1, is also defined. It has been preferred batch learning as a type of training and scaled conjugate gradient as an optimization algorithm during the learning. It used monthly data from January 2000 to September 2019 for Turkey and the USA.

The USD/TL exchange rate series were estimated with a relatively low relative error of 0.007 through the applied ANN model. Compared to conventional methods, the estimation results of the model have a very low error rate. Also, considering the low sum of squares error, it can be said that the overall performance of the model is quite good. Suppose the explanatorily of the model's parameters is not important. In that case, it can be said that ANNs are a suitable model for the estimation of the exchange rate series due to their fast and reliable results.

REFERENCES

Adewole, A. P., Akinwale, A. T., & Akintomide, A. B. (2011). An artificial neural network model for forecasting foreign exchange rate. World of Computer Science and Information Technology Journal (WCSIT). Vol. 1, No. 3,110-118.

Alavala, C.R., 2008. Fuzzy logic and neural networks: Basic concepts and applications. New Age international.

Anderson, J.A., 1995. An introduction to neural networks. MIT press.

Ataseven, B. 2013. Yapay sinir ağlari ile öngörü modellemesi. Öneri Dergisi, 10(39), 101-115.

Bildirici, M., & Ersin, Ö. Ö. (2009). Improving forecasts of GARCH family models with the artificial neural networks: An application to the daily returns in Istanbul stock exchange. Expert Systems with Applications, 36, 7355–7362.

Celik, A. E., & Karatepe, Y. (2007). Evaluating and forecasting banking crises through neural network models: An application for the Turkish banking sector. Expert systems with Applications, 33, 809–815.

Fine, T.L., 1999. Feedforward neural network methodology. Springer-Verlag.

Fioramanti, M., 2008. Predicting sovereign debt crises using artificial neural networks: A comparative approach. Journal of Financial Stability, 4(2): 149-164.

Franck, R. and A. Schmied, 2003. Predicting currency crisis contagion from East Asia to Russia and brazil: An artificial neural network approach. AMCB Working Paper. No. 2. Bar-Ilan University.

Guresen, E., Kayakutlu, G., and Daim, T. U. (2011). Using artificial neural network models in stock market index prediction. Expert Systems with Applications, 38(8), 10389-10397.

Graupe, D., 2007. Principles of ANN2nd Edn., World Scientific.

Haykin, S., 1999. Neural networks: A comprehensive foundation. Prentice-Hall.

Hopfield, J.J., 1982. Neural networks and physical systems with emergent collective computational abilities. Proceedings of National Academy of Science, 79(8): 2554-2558.

Kohonen, T., 1984. Self-organization and associative memory. Springer-Verlag.

Lin, B. R., & Hoft, R. G. (1994). Neural networks and fuzzy logic in power electronics. Control engineering practice, 2(1), 113-121.)

McCulloch, W., and W. Pitts (1943), "A Logical Calculus of the Ideas Immanent in Nervous Activity," Bulletin of Mathematical Biophysics, Vol. 5, pp. 115–133.

Nag, A.K. and A. Mitra, 1999. Neural networks and early warning indicators of a currency crisis. Reserve Bank of India Occasional Papers, 20(2): 183-222.

Pekkaya, M., and Hamzaçebi, C. (2007). Yapay sinir ağları ile döviz kuru tahmini üzerine bir uygulama. In Proceedings of the 27th YA/EM National Congress (pp.973–978). Izmir, Turkey.

Preminger, A., & Franck, R. (2007). Forecasting exchange rates: A robust regression approach. International Journal of Forecasting, 23, 71–84.

Principe, J.C., W.C. Lefebvre, and N R. Euliano, 1999. Neural and adaptive systems: Fundamentals through simulations. John Wiley & Sons, Inc.

Rumelhart, D.E., G.E. Hinton, and W. R.J., 1986. Learning internal representation by error propagation (in parallel distributed processing) MIT Press.

Roh, T. H. (2007). Forecasting the volatility of stock price index. Expert Systems with Applications, 33, 916–922.

Rosenblatt, F., 1959. Two theorems of statistical separability in the perceptron. In mechanization of thought processes. Proceeding of a Symposium Held at the National Physical Laboratory (1): 421-456.

Rosenblatt, F., 1962. Principles of Neurodynamics: Perceptrons and the Theory of Brain Mechanisms, Spartan Books, Washington.

Suykens, J.A.K., 2002. Least squares support vector machines. World Scientific.

Swanson, N., and H. White, 1997. A model selection approach to real-time macroeconomic forecasting under linear models and artificial neural networks. The Review of Economics and Statistics, 79(4): 540-550.

Yudong, Z., & Lenan, W. (2009). Stock market prediction of S& P 500 via a combination of improved BCO approach and B.P. neural network. Expert Systems with Applications, 36, 8849–8854.

Yu, L., Wang, S., & Lai, K. K. (2007). Foreign-exchange-rate forecasting with ANN(Vol. 107). Springer Science & Business Media.

Zhang, Y., & Wan, X. (2007). Statistical fuzzy interval neural networks for currency exchange rate time series prediction. Applied Soft Computing, 7, 1149–1156.

Zhu, X., Wang, H., Xu, L., & Li, H. (2008). Predicting stock index increments by neural networks: The role of trading volume under different horizons. Expert Systems with Applications, 34, 3043–3054.