AN EXPERIMENTAL STUDY FOR TRANSFORMING AND DIFFERENCING EFFECTS IN MULTIPLICATIVE NEURON MODEL ARTIFICIAL NEURAL NETWORK FOR TIME SERIES FORECASTING

Damla Ilter – Elif Karaahmetoglu – Ozge Gundogdu – Ali Zafer Dalar

Abstract

Forecasting problems play an important role in time series. In recent years, to solve these problems, many good alternative methods like artificial neural networks have been proposed in the literature. Although the most used artificial neural networks type is multilayer perceptron artificial neural networks, multiplicative neuron model artificial neural networks (MNM-ANNs) have been used to obtain forecasts for a few years. Many of previous studies were used to original series without any transformations such as differencing operation, Box-Cox transformations. Although stationary is an important assumption, previous studies have shown that forecasts obtained from ANNs were employed to non-stationary time series. Box-Cox transformations have been often used to time series because of heteroscedasticity. We used particle swarm optimization (PSO) and artificial bee colony (ABC) algorithms to train MNM-ANNs, and investigated differencing effects of original and transformed data which are obtained from Box-Cox transformations. Istanbul stock exchange (IEX) data sets which are made up of five time series for years between 2009 and 2013 are analyzed. The sets have comprised of first five months for these years. The results are interpreted and discussed. It is shown that transformation operations are useful for forecasting IEX as a result of statistical hypothesis tests.

Key words: Forecasting, Artificial Neural Networks, Difference Operator, Box-Cox Power Transformations, Multiplicative Neuron Model.

JEL Code: C45, C53, C63

Introduction

Forecasting is a process of predicting the future based on past and present data. In recently, there are many studies about time series forecasting that alternative methods have been used. Especially, artificial neural networks (ANNs) are the center of interest in the literature due to

its flexibility. ANNs do not need any assumptions (such as normality distribution, linearity and homoscedasticity) contrary to conventional time series methods.

Zhang et al. (1998) and Hippert et al. (2001) were studies in neural network forecasting methods literature. In the literature, many of papers indicated that neural networks are more accurate and reliable results than traditional forecasting methods like autoregressive moving average (ARMA) models. In recent years, various ANNs models have been suggested for forecasting. Yadav et al. (2007) proposed multiplicative neuron model artificial neural networks. Although the most used artificial neural networks type is multilayer perceptron artificial neural networks, MNM-ANNs have been used to obtain forecasts for a few years. It was showed that MNM-ANNs can give better accurate forecasts because of its multiplicative structure. PSO algorithm to train MNM-ANNs was used by Zhao and Yang (2009). In literature, there are a lot of modifications of MNM-ANNs. Different ANNs which are employed multiplicative neuron model were proposed (Yolcu et al., 2013; Aladag et al., 2013; Egrioglu et al., 2014). Egrioglu et al. (2013) used multiplicative neuron model in a fuzzy time series forecasting algorithm. Karaboga and Akay (2007) used ABC algorithm to train neural networks.

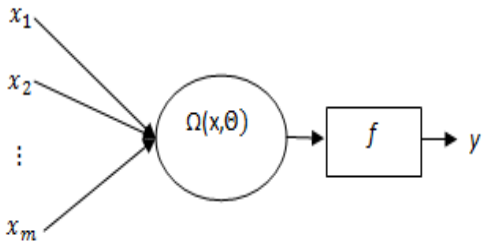
Although stationary is an important assumption in time series, previous studies have shown that forecasts obtained from artificial neural networks were employed to non-stationary time series. However, it can be discussed that transformed time series can be helpful to achieve better accurate forecasts than untransformed ones. Original series without any transformations have been used often in applications of ANNs for forecasting in the literature. Stationary assumption has been surveyed in some papers. It is shown that first order differencing is useful for forecasting with ANNs according to Chow and Leung (1996). Dalar et al. (2014) investigated stationary assumption and differencing effect for MNM-ANNs. Box-Cox power transformation that was proposed by Box and Cox (1964) has been often used to economic time series because of including heteroscedasticity. In this paper, differencing effect of original and transformed data which are obtained from Box-Cox power transformation are investigated for MNM-ANNs training with PSO and ABC algorithms by using statistical hypothesis tests.

The paper is organized as follows. Section 1 starts with the introduction of MNM-ANNs, and presents its training algorithms, differencing and Box-Cox power transformations. Section 2 presents obtained results from experimental study summarized. In conclusion, results are given and discussed.

1 Multiplicative Neuron Model Artificial Neural Networks

MNM-ANNs initially proposed by Yadav et al. (2007). The architectural structure of MNM-ANNs is shown in Fig. 1. Let be $x_1, x_2, \ldots, x_m$ are inputs of MNM-ANNs.

Fig. 1: Architectural structure of MNM-ANN



Source: Own construction

$\Omega(x,\Theta)$ is aggregation function and it has multiplicative structure. In MNM-ANNs, there is only one neuron and its output is calculated as follow:

$$net = \Omega(x,\Theta) = \prod_{i=1}^{m} (x_i w_i + b_i) \qquad (1)$$

where $\Theta = (w_1, w_2, \ldots, w_m, b_1, b_2, \ldots, b_m)$ and w_i, b_i $(i = 1,2,\ldots,m)$ are weight and bias for i^{th} input, respectively. Yadav et al. (2007) used logistic activation function as activation function (*f*) and this function given as follows:

$$f(net) = \frac{1}{1+e^{-net}} \qquad (2)$$

y=f(net) is obtained output of MNM-ANN.

1.1 ABC algorithm to training MNM-ANNs

Artificial bee colony (ABC) algorithm was firstly proposed by Karaboga (2005). The ABC algorithm for training MNM-ANNs is shown below.

Step 1. Determine limit value and the number of food sources (SN).

Step 2. Randomly generate initial food source locations from $(z^j_{\min}, z^j_{\max})$ interval.

Step 3. For each food source, calculate fitness function values.

Fitness function is mean square error (MSE) value that is calculated by using locations of the source. The locations of source can be used as weights and biases for MNM-ANNs.

Step 4. Send employed bees to the food source locations.

New food source v_i is obtained by using (3). A neighbor source (k^{th}) is randomly selected to calculate v_{ij} location. The j^{th} location of the new source is obtained from (3). Other locations of the new source are taken by i^{th} source.

$$v_{ij} = z_{ij} + \phi_{ij}(z_{ij} - z_{kj}) \tag{3}$$

The fitness function value is computed for the new source. If the fitness value of the new source is bigger than the fitness value of i^{th} source, failure counter of this source is increased by 1; on the other hand, the new source is taken as i^{th} source and the failure counter is set to 0.

Step 5. Apply the onlooker bee stage and by using (4) calculate the probability values.

$$p_i = \frac{1/f_i}{\sum_{j=1}^{SN} 1/f_i} \tag{4}$$

Onlooker bees are sent to the sources according to their probabilities.

Step 6. Determine and save the best food source.

Step 7. Check all food sources and determine exhausted sources.

If failure counter is bigger than the limit value for a source, this source can be considered as exhausted source. For each exhausted source, a scout bee is employed. The locations of new source are randomly generated from $(z_{\min}^{j}, z_{\max}^{j})$ interval instead of the exhausted source. The failure counter is set to 0 for the new source.

Step 8. Check stopping conditions.

If the stopping conditions are met, skip to step 9. Otherwise, back to step 4.

Step 9. As the solution take the best food source.

1.2 PSO algorithm to training MNM-ANNs

Firstly, Kennedy and Eberhart (1995) introduced PSO. Although Yadav et al. (2007) employed back propagation algorithms for training MNM-ANNs, Zhao and Yang (2009), Yolcu et al. (2013) and Aladag et al. (2013) used PSO algorithm, and also Alpaslan et al. (2013) used ABC algorithm to train it.

The PSO algorithm to training MNM-ANNs is shown below.

Step 1. Randomly determine positions and velocities of each m^{th} (m = 1, 2, …, *pn*) particles and kept in vectors P_m and V_m given as follows:

$$P_m = \{p_{m,1}, p_{m,2}, ..., p_{m,d}\}, m = 1,2,..., pn \tag{5}$$

$$V_m = \{v_{m,1}, v_{m,2}, ..., v_{m,d}\},\ m = 1,2,..., pn \qquad (6)$$

where $x_{m,j}$ (i=1, 2, …, d) indicates j^{th} position of m^{th} particle. pn and d demonstrate the number of particles in a swarm and positions, respectively. The initial positions and velocities of each particle in a swarm are randomly generated from uniform distribution (0, 1) and ($-v_m$, v_m), respectively. Positions of a particle are composed from weights and biases. Each particle gives a solution set for the neural network.

Step 2. Determine the parameters of PSO.

The parameters which direct the PSO algorithm are determined in the first step. These parameters are pn, v_m, c_{1i}, c_{1f}, c_{2i}, c_{2f}, w_1, and w_2. Let c_1 and c_2 demonstrate cognitive and social coefficients, respectively. w is the inertia parameter. Let (c_{1i}, c_{1f}), (c_{2i}, c_{2f}), and (w_1, w_2) be the intervals which includes possible values for c_1, c_2 and w, respectively. In each iteration, these parameters are calculated by using the formulas given in (7), (8) and (9).

$$c_1 = (c_{1f} - c_{1i})\frac{t}{\max t} + c_{1i} \qquad (7)$$

$$c_2 = (c_{2f} - c_{2i})\frac{t}{\max t} + c_{2i} \qquad (8)$$

$$w = (w_2 - w_1)\frac{\max t - t}{\max t} + w_1 \qquad (9)$$

Step 3. Compute evaluation function values and calculate evaluation function values for each particle.

Evaluation function MSE is given below.

$$MSE = \frac{1}{n}\sum_{t=1}^{n}(desired_t - output_t)^2 \qquad (10)$$

where n indicates the number of learning sample. The output value of the proposed model is calculated by PSO algorithm.

Step 4. Determine $Pbest_{\text{m}}$ (m = 1, 2, …, pn) and *Gbest* owing to evaluation function values calculated in the previous step.

$Pbest_{\text{m}}$ is a vector that it stores the positions corresponding to the m^{th} particle's best individual performance. *Gbest* is the best particle, which has the best evaluation function value, found so far.

$$Pbest_m = (pb_{m,1}, pb_{m,2}, ..., pb_{m,d}), m = 1,2,..., pn \qquad (11)$$

$$Gbest = (p_{g,1}, p_{g,2}, ..., p_{g,d}) \quad (12)$$

Step 5. Update the parameters.

The updated values of cognitive coefficient c_1, social coefficient c_2, and inertia parameter w are computed using by formulas (7), (8) and (9).

Step 6. Calculate new values of velocities and positions. New values of velocities and positions for each particle are calculated by using the formulas given in (13) and (14). If maximum iteration number is reached, the algorithm goes to Step 3; on the other hand, it goes to Step 7.

$$v_{m,j}^{t+1} = [w \times v_{m,j}^{t} + c_1 \times rand_1 \times (pb_{m,j} - p_{m,j}) + c_2 \times rand_2 \times (p_{g,j} - p_{m,j})] \quad (13)$$

$$p_{m,j}^{t+1} = p_{m,j}^{t} + v_{m,j}^{t+1} \quad (14)$$

where $m = 1, 2, ..., pn$, j=1,2,…,d

Step 7. Determine the best solution.

The elements of Gbest are taken as the best weight values of the new ANNs model.

2 Experimental Study

To test differencing and transforming effects, IEX data sets were used in the experimental study. Data consist of five subsets that are given below:

Set A. it is daily obtained from 02/01/2009 and 29/05/2009 dates.

Set B. it is daily obtained from 04/01/2010 and 31/05/2010 dates.

Set C. it is daily obtained from 03/01/2011 and 31/05/2011 dates.

Set D. it is daily obtained from 02/01/2012 and 31/05/2012 dates.

Set E. it is daily obtained from 02/01/2013 and 29/05/2013 dates.

Initially, Augmented Dickey Fuller (ADF) unit root tests whether data have stationary property is implemented for five series by using E-views Package program. As a result, all of subsets have unit roots and also stationary is obtained from first difference of theirs. Then, Box-Cox power transformations are applied to data sets. The formula of the Box–Cox power transformation is;

$$Y_t^{(\lambda)} = \begin{cases} \dfrac{Y_t^{\lambda} - 1}{\lambda}, & \lambda \neq 0 \\ \ln(Y_t), & \lambda = 0 \end{cases} \quad (15)$$

where λ is the transform parameter. Further, Box-Cox power transformation is only applied to positive data values for logarithmic transform and data sets have positive values in this study. In this paper, five Box-Cox power transformation operations for different λ values are applied to all series. Y_t turns into hyperbolic, natural logarithm (*ln*), cube root and square root when λ value is taken -1, 0, 1/3 and 1/2, respectively. In the implementation, 20% of the observations for all series are taken as test set. All series, the transformations and their first differences are analyzed by using MNM-ANNs in the experimental study. ABC and PSO algorithms are used to train MNM-ANNs, and also lag numbers are taken as 2, 3, 4 and 5. In the experimental design, all factors are listed follow:

Factor 1 is year. This factor consist of five levels which are 2009 (1), 2010 (2), 2011 (3), 2012 (4) and 2013 (5) years.

Factor 2 is number of lag variables. This factor consist of four levels which are 2 (1), 3 (2), 4 (3) and 5 (4). For all years, the lagged variables of original series can be used as inputs.

Factor 3 is type of series, and these series are original series, the transformations and their first differences. Original (1), difference (2), Ln (3), Ln difference (4), Square root (5), Square root difference (6), Cube root (7), Cube root difference (8), Hyperbolic (9) and Hyperbolic difference (10) are the levels of the factor.

Factor 4 is training algorithm. ABC (1) and PSO (2) algorithms are the levels of this factor.

Observations of dependent variable are mean absolute percentage error (MAPE) values which are obtained for test data sets in each run. MAPE values can be calculated by using equation (16).

$$MAPE = \sum_{t=1}^{n} \left| \frac{desired_t - output_t}{desired_t} \right| \qquad (16)$$

In SPSS package program, univariate general linear model section was used to obtain statistical test results. The test results are given in Tab. 1.

Tab. 1: Univariate general linear model test results

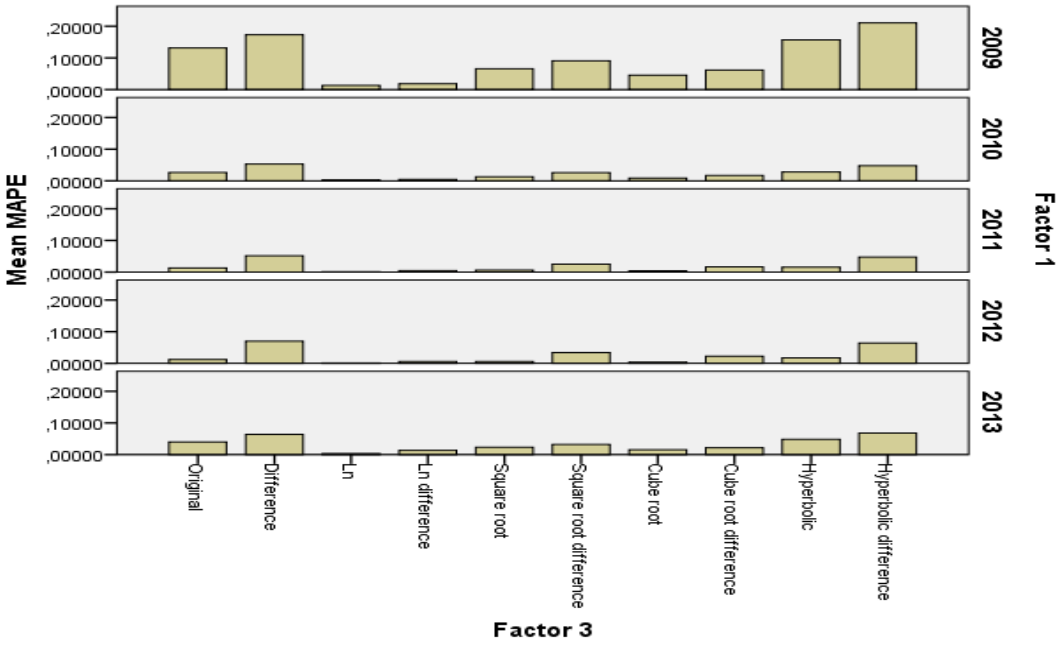
Dependent Variable: MAPE

Source	Type III Sum of Squares	df	Mean Square	F	Sig.
Corrected Model	,643[a]	17	,038	85,709	,000
Intercept	,607	1	,607	1374,887	,000
Factor 1	,343	4	,086	194,003	,000
Factor 2	8,256E-005	3	2,752E-005	,062	,980
Factor 3	,300	9	,033	75,506	,000
Factor 4	,001	1	,001	1,302	,255
Error	,169	382	,000		
Total	1,419	400			
Corrected Total	,812	399			

Source: Own calculation

According to Tab. 1, differences between levels of Factor 2 and Factor 4 are not statistically significant. Differences among the levels of Factor 1 and Factor 3 are statistically significant.

Fig. 2: Mean of MAPE values for the levels of Factor 1 and Factor 3



Source: Own calculation

Fig. 2 is given to understand differences between the levels of Factor 1 and Factor 3. According to Fig. 2, the bigger MAPE values for transformations were obtained from differenced and hyperbolic differenced series. According to MAPE values, original and transformed series are lesser than the differenced series of them. The obtained test results for year 2009 is worse than results of other years, while year 2011 is better than results of other years. Among the types of series for all years, the best MAPE values are obtained from when

ln transformation is applied to original series. On the other hand, the worst MAPE values are obtained from hyperbolic transformation series. Fig. 2 is also given to understand that there is no need to differencing operation on original and transformed series.

Conclusion

In this study, differencing and transformation effects for MNM-ANNs are investigated. An experimental study was designed to test these effects. The obtained results are showed that there is no need to difference in MNM-ANNs. MNM-ANNs can give better results for original and non-stationary time series. These conclusions are obtained for IEX data. It is clear that the different results can be obtained for different data sets. According to obtained results, it can be asserted that IEX time series can be solved with MNM-ANNs by using their transformed series, especially *ln* transformation. In the future studies, the similar experimental study can be applied for other transformations and exchange data sets of other countries.

References

Aladag, C. H., Yolcu, U., & Egrioglu, E. (2013). A New Multiplicative Seasonal Neural Network Model Based on Particle Swarm Optimization. *Neural Processing Letters, 37*(3), 251-262.

Alpaslan, F., Egrioglu, E., Aladag, C. H., Ilter, D., & Dalar, A. Z. (2013). Comparison of Single Multiplicative Neuron Artificial Neural Network Models Using ABC and BP Training Algorithms. *Anadolu University Journal of Science and Technology A- Applied Science and Engineering*, *14*(3), Accepted Paper.

Box, G. E., & Cox, D. R. (1964). An Analysis of Transformations. *Journal of Royal Statistical Society, Series B (Methodological), 26*(2), 211-252.

Chow, T. W., & Leung, C. (1996). Neural network based short-term load forecasting using weather compensation. *IEEE Transactions on Power Systems*, *11*(4), 1736-1742.

Dalar, A. Z., Egrioglu, E., Yolcu, U., Ilter, D., & Gundogdu, O. (2014). An Investigation of Differencing Effect in Multiplicative Neuron Model Artificial Neural Network for Istanbul Stock Exchange Time Series Forecasting. *American Journal of Intelligent Systems, 4*(1), 15-19.

Egrioglu, E., Aladag, C. H., Yolcu, U., Corba, B. S., & Cagcag, O. (2013). Fuzzy Time Series Method Based On Multiplicative Neuron Model and Membership Values. *American Journal of Intelligent Systems*, *3*(1), 33-39.

Egrioglu, E., Yolcu, U., Aladag, C. H., & Bas, E. (2014). Recurrent Multiplicative Neuron Model Artificial Neural Network for Non-linear Time Series Forecasting. *Procedia - Social and Behavioral Sciences*, *109*, 1094-1100.

Hippert, H. S., Pedreira, C. E., & Souza, R. C. (2001). Neural networks for short-term load forecasting: a review and evaluation. *IEEE Transactions on Power Systems, 16*(1), 44-55.

Karaboga, D. (2005). An idea based on honey bee swarm for numerical optimization. Technical Report-TR06, Erciyes University, Engineering Faculty, Computer Engineering Department.

Karaboga, D., & Akay, B. (2007). Artificial Bee Colony Algorithm on Training Artificial Neural Networks. *Signal Processing and Communications Applications*, SIU 2007, IEEE 15th, 1 – 4. doi: 10.1109/SIU.2007.4298679

Kennedy, J., & Eberhart, R. (1995). Particle swarm optimization. *In Proceedings of IEEE International Conference on Neural Networks, 4*, 1942-1948.

Yadav, R. N., Kalra, P. K., & John, J. (2007). Time series prediction with single multiplicative neuron model. *Applied Soft Computing, 7*, 1157-1163.

Yolcu, U., Egrioglu, E., & Aladag, C. H. (2013). A New Linear & Nonlinear Artificial Neural Network Model for Time Series Forecasting. *Decision Support Systems, 54*, 1340-1347.

Zhang, G., Patuwo, B. E., & Hu, Y. M. (1998). Forecasting with artificial neural networks: The state of the art. *International Journal of Forecasting, 14*, 35-62.

Zhao, L., & Yang, Y. (2009). PSO-based single multiplicative neuron model for time series prediction. *Expert Systems with Applications, 36*, 2805-2812.

Contacts

Damla Ilter
Mimar Sinan Fine Arts University, Faculty of Arts and Sciences, Department of Statistics,

34380 Istanbul, Turkey.

damla.ilter@msgsu.edu.tr

Elif Karaahmetoglu
Ondokuz Mayıs University, Faculty of Arts and Sciences, Department of Statistics, Kurupelit

Campus, Atakum 55139 Samsun, Turkey.

elif.karaahmetoglu@omu.edu.tr

Ozge Gundogdu
Cumhuriyet University, Faculty of Economics and Administrative Sciences, Department of

Econometrics, 58140 Sivas, Turkey.

ozge5gundogdu@hotmail.com

Ali Zafer Dalar
Ondokuz Mayıs University, Faculty of Arts and Sciences, Department of Statistics, Kurupelit

Campus, Atakum 55139 Samsun, Turkey.

alizafer.dalar@omu.edu.tr