International Journal of Engineering & Technology, 7 (4.10) (2018) 282-289

International Journal of Engineering & Technology

Website: www.sciencepubco.com/index.php/IJET

Research paper



Improvement of ANN Models via Data Envelopment Analysis for Stock Prices Forecasting

Raja Das[1] and Jitendra Kumar Jaiswal[2*]

[1,2]Vellore Institute of Technology, Vellore – 632014, Tamilnadu, India.
**Corresponding author E-mail: jit.frenz@gmail.com*

Abstract

The implementation of artificial neural network techniques has become quite prevalent in the field of nonlinear data modelling and forecasting in this era. The only application of ANN models for model fitting may not be sufficient for close and satisfactory performances; hence the researchers are adopting hybrid models of ANN with different statistical and machine learning approaches such as support vector machines, particle sworm optimization, principal component analysis, etc. We have also developed a hybrid model in this paper with ANN and data envelopment analysis (DEA) techniques for stock prices forecasting in share market. The efficient decision making units have been selected with help of DEA approach and provided it as input to the Lavenberg-Marquardt technique based ANN model in sliding window manner. Further a closed performance of our hybrid model has been achieved by carrying out our experimentation with different number of nodes in the hidden layer of ANN model. Since the prices of stocks follow numerous factors such as demand and supply, political environments, economy and finance, buy and sell, etc., the historical prices for stocks may be convenient for the further forecasting.

Keywords: *Decision Making Units; Forecasting; Neurons; Back-propagation; and Sliding Window.*

1. Introduction

Data Envelopment Analysis (DEA) is a nonparametric and non-stochastic, linear programming approach, which has been considered as one of the highly competent tool in the recent decades for classification and estimation of efficiency and yield of Decision Making Units (DMUs) as frontiers. Its vital application has been observed in both private and public sectors such as airlines, hospitals, banks, universities, etc., for mostly production and manufacturing. Many applications of DEA can be explored in [1, 2, 3, 4, 5, 6]. DEA was not preferably considered earlier due to its occupancy of memory and CPU time in the computing for large datasets comprising many attributes and DMUs, but since those constraints are no more barriers these days, it is being implemented frequently in different disciplines.

ANN has been applied very much successfully in numerous forecasting premises in recent decades [7, 8, 9, 10, 11, 12]. Many researchers have preferred ANN over statistical multivariate regression models since ANN does not consider any structure as in regression models [13]. Even though, this characteristics of ANN, weakens its forecasting ability in the cases of monotonic curves. An example may be considered from the point that increased individual's income may lead to demand of assets [9, 14].

The monotonic properties of curves support a lot for forecasting. Researchers [9] have emphasized about biased individual preferences for predicting market behaviour through the stocks having monotonic attributes. Wang [13] has also expressed that nonlinear would hardly undergo for any over-fitting scenarios when they preserve monotonicity. For a multi-attribute data with input vector $x_i = (x_{1i},...,x_{mi}), \forall i = 1,...,n$ and y_i if there exists a mapping function $f(\bullet)$ that maps x_i to y_i, then the monotonic property could be established for any two x_1 and x_2, and their respective values y_1 and y_2 with following conditions:

1. $(x_{j1} - x_{j2}) > 0 \Leftrightarrow y_1 > y_2, \Leftrightarrow \exists$ at least one $j = 1,...,m$ such that $x_{j1} > x_{j2}$ and $x_{j1} < x_{j2}$ with no $j = 1,...,m$, and
2. $(x_{j1} - x_{j2}) = 0 \Leftrightarrow y_1 = y_2 \forall j \in 1,...,m.$

For forecasting in the cases of nonlinear functions, monotonicity can be retained by reformulating the algorithm or by applying the training data that itself preserves monotonicity. The data envelopment analysis (DEA) technique is quite capable to follow all these requirements. The DEA is a nonparametric model that has a characteristics of piecewise linearity (almost nonlinearity), which works on measurement of data, based on productivity. In this section, we have followed the data screening technique to generate subsample training data with "weak" monotonicity. This approach will works as preprocessing of data before providing it as input to our ANN models.

In this paper we have explained the concept of DEA with data preprocessing approach in Section-2 and 3. Further we have discussed about ANN structure in Section-4. Mathematical concept about iterative methods for nonlinear least square problems has be described in Section-5. In the Section-6, experimentation and result analysis has been discussed. Finally, the paper has been concluded with future scope in the Section-7.

Copyright © 2018 Authors. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

2. Concept of DEA

The basic application of DEA technique is to calculate the performance of decision making units (DMUs) (or observations), those actually play a vital role in developing the model. Let us consider that there are *n* - DMUs with *m* - input and *s* – output variables, then the score for relative inefficient $t^{th} \in \{1,...,n\}$ - DMU can be calculated by the linear programming as follows:

minimize θ_t^c

subject to $\sum_{i=1}^{n} \lambda_i x_{ji} - \phi_{jt} \le 0 \quad \forall j = 1,...,m,$

$$\sum_{i=1}^{n} \lambda_i y_{ki} - y_{kt} \ge 0 \quad \forall k = 1,...,s, \tag{1}$$

$\lambda_i \ge 0 \quad \forall i = 1,...,n,$

where θ_t^c is the inefficiency score for t^{th}-DMU, given by [15] (CCR). DMU applies x_{ji} inputs and produces y_{ki} outputs, and λ_k are dual variables. The CCR model asserts that a DMU, *t* is inefficient when it can reflect similar output levels while consuming less number of inputs.

The CCR model follows a constant returns to scale (CRS) cases, that is, it requires exactly double costs for doubling outputs, however, [16] (BCC) model performs on variable returns to scale (VRS) cases:

minimize θ_t^B

subject to $\sum_{i=1}^{n} \lambda_i x_{ji} - \phi_{jt} \le 0 \quad \forall j = 1,...,m,$

$$\sum_{i=1}^{n} \lambda_i y_{ki} - y_{kt} \ge 0 \quad \forall k = 1,...,s, \tag{2}$$

$\lambda_i \ge 0$ and $\sum_{i=1}^{n} \lambda_i = 1, \quad \forall i = 1,...,n$

where θ_t^B is the inefficiency score for t^{th}-DMU, given by [16] (BCC). Under VRS cases, there is nonlinear decreasing or increasing conditions with inputs and outputs. For instance, decreasing scale cases may require a higher proportion of increase in inputs for getting increase in outputs.

For input and output vectors $x = (x_j), j = 1,...,m$ and $y = (y_k), k = 1,...,s$, a standard data envelop function *f (x)* with a set of DMUs, can be established with following equation [13]:

$$y = f(x) - p - q \tag{3}$$

where *p* and *q* are inner and outer deviations. The DEA technique performs on assumption of monotonicity, and hence *q = 0*.

3. Data Preprocessing by DEA for Input to ANN

The functionality of ANNs essentially depends on the structure and quality of input data which determine the performance on ANN models [17]. Here we have followed DEA-based technique to sort out the input data. For univariate cases, we have firstly observed the application of DEA for partitioning the input data. Further a generalized mathematical structure has been produced for a multivariate input data [18].

3.1. Case of Univariate Input Data

The assumption of monotonic property has played a vital role for decision making scenarios. In a simple way, mathematically, a monotonic property for univariate input is defined as follows:

$$x_1 > x_2 \Leftrightarrow f(x_1) > f(x_2) \tag{4}$$

DEA techniques have worked on finding capable frontiers which are compatible with assumption of monotonicity. Actually every capable DMU (also called as reference DMU) stays on capable frontiers which satisfy monotonicity. Figure 1 can be observed as an example for a set of frontiers for a combination of univariate input and single output scenario. Frontier-1 appears as the most capable while Frontier-2 and #3 performs in gradually decreasing capability of frontiers. Existing 9-DMUs lie on different frontiers as per the efficiencies gradually decreasing from Frontier-1 to #3 but each 3 DMUs have the same efficiency on a single frontier.

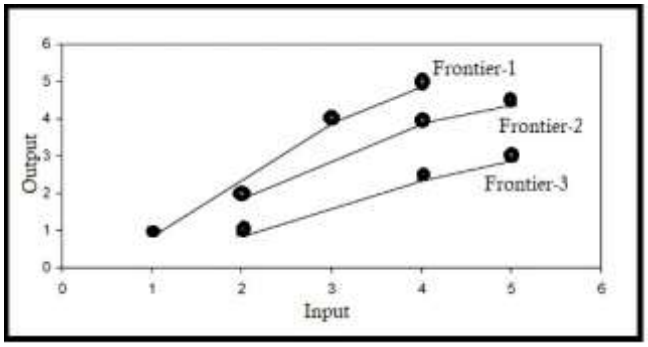


Figure 1: A set of linear frontiers in piecewise manner

ANN applies the method of least square to minimize errors in nonlinear predicting models. If ANN is trained with the DMUs lying only on a single frontier, it will be preserving monotonic property, but if all the 9 DMUs are involved, it will certainly lead to inconsistency for the model with monotonicity assumptions.

It can also be understood in a way that the probability of a monotonicity assumption based predicting function learned by an ANN decreases gradually, when there is involvement of training data having lesser DEA-based efficiencies. Let us represent *DMUs* by DMU_{s1}, DMU_{s2}, and DMU_{s3} lying on Frontiers-1 to #3 respectively, following monotonic property (*MP*), the respective conditional probabilities $P(MP|DMU_{si})$ reflect the following relation:

$$\begin{aligned} P(MP \mid DMU_{s1}) &> P(MP \mid DMU_{s1}, DMU_{s2}) \\ &> P(MP \mid DMU_{s1}, DMU_{s2}, DMU_{s3}) \end{aligned} \tag{5}$$

3.2. Case of Multivariate Input Data

Let us proceed with a case of multiple input and single output data for *k*-observations training data set $\{(x_1,y_1),...,(x_k,y_k)\}$, where x_i represents *m*-dimensional input multivariate vector of i^{th}-observation with y_i as output value. The forecasting function $f(x_i) > 0$ with a variable $\zeta_i = |y_i/f(x_i)|$, is selected in such a way that the following condition is optimized:

$$\text{maximize} \quad \sum_{i=0}^{k} \zeta_i \tag{6}$$

subject to $0 < \zeta_i < 1, i = 1,...,k$

It is obvious that this formulation has an optimal value for *k* as upper bound that provides a perfect fit for the function $f(x_i)$ on training data. Nevertheless, for cases when perfect fit are not possible, the selection of $f(x_i)$ should be in such way that all the values

of ζ_i are close to 1, for all i, that is, the probability density function for ζ_i is monotonically increasing. [19] had proposed a class of monotonically increasing densities on [0,1] with mode 1 as follows:

$$h(\zeta) = c_\alpha e^{\alpha\zeta}, \alpha > 0 \tag{7}$$

Lemma 3.1.
Let us suppose that ζ_i are independent and identically distributed (iid) with monotonically increasing densities, as in the equation-(7), similar values of ζ_i in the sample are also maximized in the equation-(6).

Lemma 3.2.
In the processed data, the outliers (external and internal deviations) are also minimized by equation-(6). [18] can be referred for proofs of Lemma 3.1 & 3.2.

4. Artificial Neural Network Structure

The concept of artificial neural networks is very much relevant and analogous to natural nervous systems in human brain. It can be considered as a substantially parallel adaptive networks of neurons. Neurons are here cognitive to simple nonlinear computing components. The prime intention of neural networks is to perform both analysis and establishment of such substantive parallel computing systems.

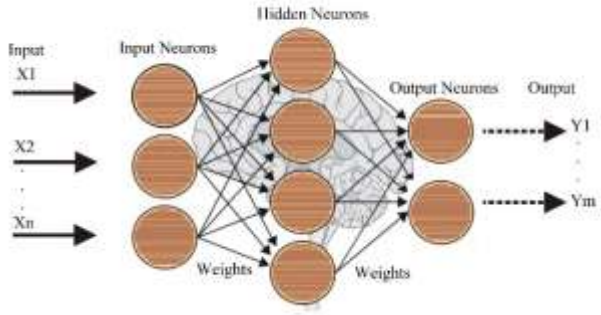


Figure 2: Artificial Neural Network Model

Neurons are categorized into three types: input, hidden, and output. Input neurons receive inputs from external sources as an stimulant to the network. Output of neurons produces output signals of the network. The intermediate functions are calculated by hidden neurons, and these neurons are not visible from the external sites. A neural network model can be created as a weighted directed graph containing neurons as nodes and weighted edges as links. The basic structure of the artificial neural networks is sketched in Figure 2.
An artificial neural network model has the capability of adopting the change of environment that is called as learning. In this process it generates an internal model with sampled data, which represent structured weight vectors. Learning algorithms develop an architecture-based approach to assign patterns into weights to produce internal models. Learning process continues with update of connection weights.

5. Iterative Methods for Nonlinear Least Square Problems

Some of the useful definitions may be followed as preliminary context of this chapter as follows.

To minimize a real valued function f with N variables in the *unconstrained optimization scenarios*, we search a *local minimizer*, that is, a value x^* such that

$$f(x^*) \le f(x) \tag{8}$$

for all x close to x^*. In the case of *global minimizer*, it will consider simply for all x. In the *constrained optimization scenarios*, for a l*ocal minimizer* of a function f over a set $U \subset \mathrm{R}^N$ and $x^* \in U$, such that $f(x*) \le f(x)$ for all $x \in U$ near x^*. Again in case of *global minimizer*, it is simply for all $x \in U$.

5.1. Notations and Some Preliminary Definitions

In this paper, we have followed the following convention. Vectors are considered as column vectors. The vector x^* is a solution to function f, while x is a desired value. The initial iteration or sometimes preliminary guess is $x0$, and $\{x_k\}, k \ge 0$ are the series of iterations. The i^{th} element of a vector x by $(x)_i$, and the i^{th} element of x_k by $(x_k)_i$. The error is $(\varepsilon = x - x^*)$ and $\varepsilon_n = x_n - x^*$ is the error in n^{th} iteration. The gradient of f, is called *Jacobian* of f, denoted by $\nabla f(x) \in \mathrm{R}^N$ for $x \in \mathrm{R}^N$, if it exists,

$$\nabla f(x) = \left(\frac{\partial f}{\partial x_1} \frac{\partial f}{\partial x_2} \ldots \frac{\partial f}{\partial x_n} \right) \tag{9}$$

The 2^{nd} derivative is called *Hessian* of f, and if it exists, represented as

$$\nabla^2 f = \begin{pmatrix} \frac{\partial^2 f}{\partial x_1 \partial x_1} & \frac{\partial^2 f}{\partial x_1 \partial x_2} & \cdots & \frac{\partial^2 f}{\partial x_1 \partial x_n} \\ \frac{\partial^2 f}{\partial x_2 \partial x_1} & \frac{\partial^2 f}{\partial x_2 \partial x_2} & \cdots & \frac{\partial^2 f}{\partial x_2 \partial x_n} \\ . & . & \cdots & . \\ . & . & \cdots & . \\ . & . & \cdots & . \\ \frac{\partial^2 f}{\partial x_n \partial x_1} & \frac{\partial^2 f}{\partial x_n \partial x_2} & \cdots & \frac{\partial^2 f}{\partial x_n \partial x_n} \end{pmatrix} \tag{10}$$

Throughout this paper, the Euclidean norm is given as

$$\| x \| = \sqrt{\sum_{i=1}^{n} (x_i^2)} \tag{11}$$

Definition 5.1.
A positive-definite matrix A obeys the follows $x^T Ax > 0$ for every non-zero vector x. If A is positive definite and symmetric as well, it is semi-positive definite. If A consists both negative and positive eigenvalue., It is called as indefinite.

Definition 5.2.
Least Square Problem: The general function for finding local minimize

$$f(x) = \frac{1}{2} \sum_{i=1}^{m} (f_i(x))^2 \tag{12}$$

where $f_i : \mathrm{R}^N \to \mathrm{R}$, $i = 1,\ldots,m$ *are generated functions and* $m \ge n$.
Let us consider a simplest quadratic objective function as

$$f(x)=\frac{1}{2}x^T Hx-x^T b \tag{13}$$

and without loss of generality, we may assume that H is symmetric, since

$$x^T Hx=x^T\left(\frac{H+H^T}{2}\right)x \tag{14}$$

H is $\nabla^2 f(x)$, for all x, and by the symmetry

$$\nabla f(x)=Hx-b \tag{15}$$

Definition 5.3
If H is positive semidefinite, the objective quadratic function is convex.
If H is positive semi-definite, the objective quadratic function is convex.
If x^* is local minimum of a quadratic function f, the necessary terms for optimality involve that H is positive semidefinite, and $Hx^* = b$ and the global minimizer is solution of this linear system in particular. If H is not sparse, for an average length N, this linear system can be solved by *Cholesky factorization* [20] of H as $H = LL^T$, L is positive diagonal with lower triangular nonsingular matrix. The Cholesky factorization will not exist for indefinite H.

5.2. Gauss-Newton Method

Nonlinear least squares conditions may have objective functions as follows:

$$f(x)=\frac{1}{2}\sum_{i=1}^{m}||r_i(x)||^2 \tag{16}$$

or

$$f(x)=\frac{1}{2}R(x)^T R(x) \tag{17}$$

Where vector $R = (r_1,...,r_m)$ is *residual*.
In the Gauss-Newton method, $\nabla^2 f$ is simply discarded, and one calculates a step

$$s=-(R'(x_c)^T R'(x_c))^{-1}\nabla f(x_c) \tag{18}$$

Or

$$s=-(R'(x_c)^T R'(x_c))^{-1}R'(x_c)^T R'(x_c) \tag{19}$$

The Gauss-Newton next iteration is $x_+ = s + x_c$, for the close iteration x_c.

5.3. Steepest Descent Method

The direction in the steepest descent method is determined by $-\nabla f(x)$, and the current iteration x_c is updated by

$$x_+=-\lambda\nabla f(x_x)+x_c \tag{20}$$

For $\lambda = 1$, x_+ may not be more close to a solution than x_c, since the direction is scaled with f , unlike the direction in Newton's method. For the effective results, the *steplength* λ should be chosen. It may be determined as $\lambda = \gamma^m$, where $\gamma \in (0,1)$, and $m \geq 0$ is the smallest integer in such a way that can sufficiently reduce the value of f . So steepest decent method may follow

$$f(-\lambda\nabla f(x_x)+x_c)-f(x_c)<-\lambda\alpha||\nabla f(x_c)||^2 \tag{21}$$

This is called as *Armijo rule* [21], for *line search*, applied to find minimizer in a direction where f locally decreases. This testing is repeatedly continued up to some threshold and step-size is also reduced if test fails as *backpropagation*.

5.4. Levenberg-Marquardt Method

Most of the standard ANN techniques want to get the bottom of nonlinear least square problems that deal with fitting a parameterized function to install the minimization of sum of squared errors (SSE) between output of the function and actual data values by continuously updating the parameter values. The LM method is basically a combination of two minimization methods, namely, the steepest descent and Gauss-Newton (GN) method. The steepest descent gradually decreases the SSE by modifying the parameters in the steepest-descent direction whereas Gauss-Newton does the same by treating the least square function as locally quadratic and the estimate the least value of the curve. Steepest descent method deals with the global minimization while Gauss-newton method seeks the local minimum. The LM method follows the steepest-descent method when the parameters are far from the optimal point and switches to GN when the parameters are in the range of local quadratic curve [22].
The solutions of trust region problems are characterized by the theory of constrained optimization [23].

Theorem 5.4
Suppose that g $\in$ R^N and A is an N x N symmetric matrix. Let

$$m(s)=s^T As/2+g^T s \tag{22}$$

A vector s is a solution to $\min_{||s||\leq\Delta} m(s)$ *if and only if there exist* v ≥ 0, *such that* $(A+vI)s = -g$ *and either* $||s|| = \Delta$ *or* $v = 0$.

Algorithm 5.1 TrustRegionTestLM(x_t, x_c, x_+, f, v)

1. $z = x_c$
2. ***while*** $z = x_c$
i) Actual reduction
$actred = f(x_c)-f(x_t); s_t = x_t - x_c;$
ii) Predicted reduction $predred = -\nabla f(x_c)^T s_t/2$
iii) ***if*** $\frac{actred}{predred}=\mu_0$ ***then*** $z = x_c; v=\max(w_{up}v, v_0),$
and recalculate the test point with new v
iv) ***if*** $\mu_0 \leq \frac{actred}{predred} \leq \mu_{low}$ ***then***
$z = x_t; v=\max(w_{up}v, v_0),$
v) ***if*** $\frac{actred}{predred} \geq \mu_{low}$ ***then*** $z = x_t$
vi) ***if*** $\frac{actred}{predred} \geq \mu_{high}$ ***then*** $v = w_{down}v$
vii) ***if*** $v_0 > v$, ***then*** $v = 0$

3. $x_+ = z$

Now the Levenberg-Marquradt algorithm can be sketched as follows:

Algorithm 5.2 lmalgorithm(x, R, $kmax$)

1. $v = v_0$
2. ***for*** $k = 1: kmax$

suppose $x_c = x$

Calculate R, f, R', and ∇f ; Check for termination

$$x_t = -(R'(x_c)^T R'(x_c) + v_c I)^{-1} R'(x_c)^T R'(x_c) + x_c$$

Call TrustRegionTestLM(x_t, x_c ,x, f, v)

6. Experimentation with DEA and BPNN-LM Model

6.1. Data Acquisition

Since the data from financial market are frequently accessible in the both paid and free forms, we have considered the free data from NSE India website for the stocks MARUTI and TATA Motors for 200-days on the daily basis.

We have considered 85% of the data, that is, 170-days for training the model and predicted for next 15-days closing prices. Further, in the sliding window manner, we have discarded the most old 15-days data and added the 15-days data and retrained the model to predict for new 15-days closing prices (Figure. 3).

Since, we have observed that the Levenberg-Marquardt from back-propagation technique based neural network models was performing well, hence we have applied the same model here, giving it a name as BPNN-LM, for prediction of stock market prices, which would be accepting the preprocessed data by DEA, as input. As our final objective is to develop a hybrid model with subtly executing ANN models, as some of its constituents, the study of performance of different ANN models becomes important, and hence here we have focused on BPNN-LM model.

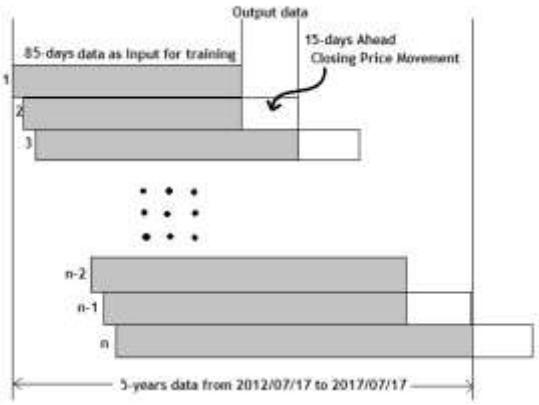
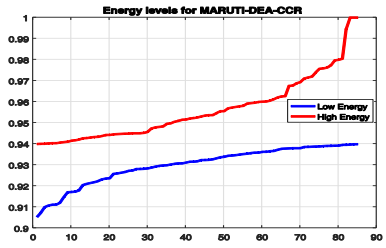


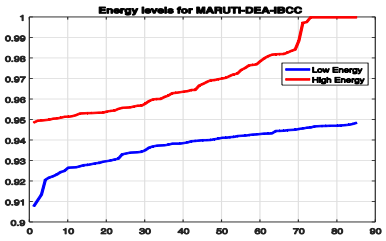
Figure 3: Windowing of the Data for Prediction of Closing Prices for Next 15-Days

6.2. Result Analysis

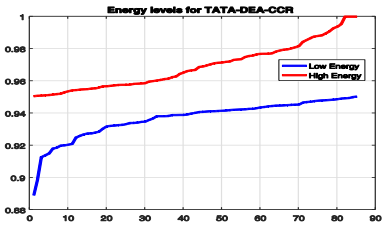
We have presented the energy level of decision making units in Figure 4, where 50% of data has been considered as low and rest as high energy data, those are also called as efficient and non-efficient data respectively. These data have been separated by CCR and BCC methods.



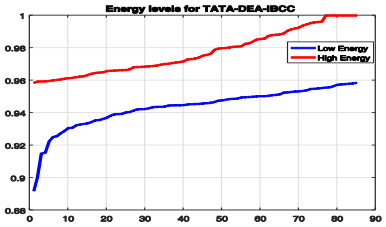
(a) MARUTI DEA CCR



(b) MARUTI DEA IBCC



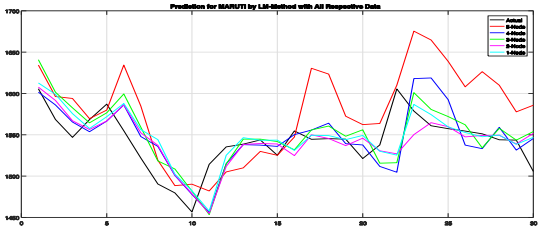
(c) TATA DEA CCR



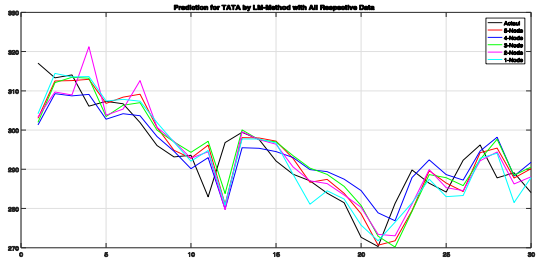
(d) TATA DEA IBCC

Figure 4: Partitioning the Data into High & Low Energy States for Maruti and TATA Motors from the Two Considered Methods for DEA

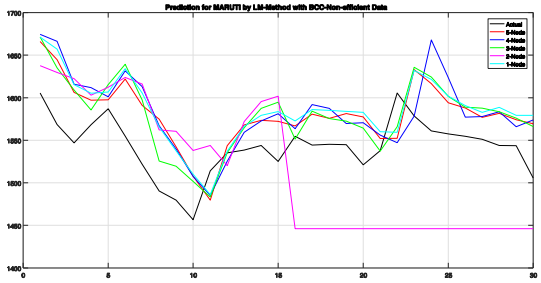
We have produced the prediction graphs in Figure 5 and Figure 6.



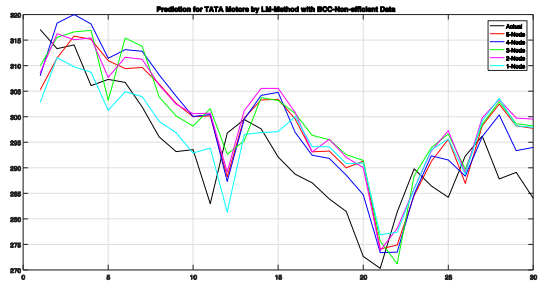
(a) lmMARUTI All Respective Data



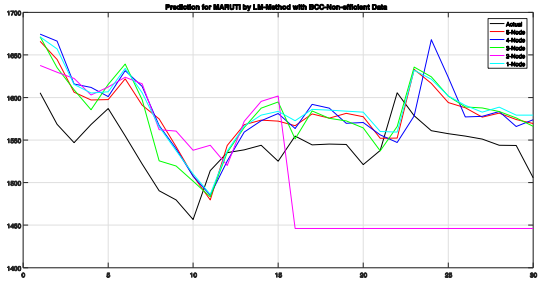
(b) lmTATA All Respective Data



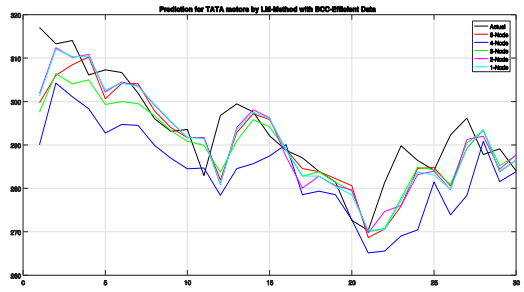
(c) lmMARUTI Nonefficient Data



(d) lmTATA Nonefficient Data

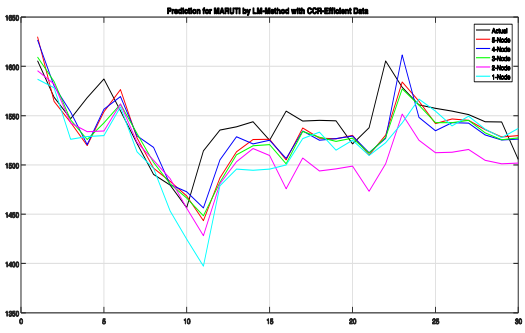


(e) lmMARUTI Efficient Data

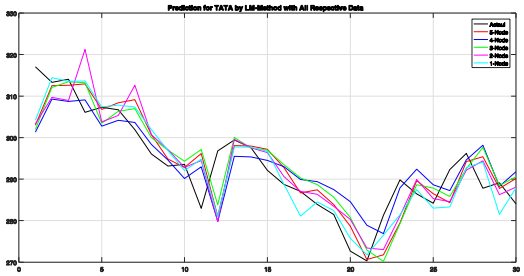


(f) lmTATA Efficient Data

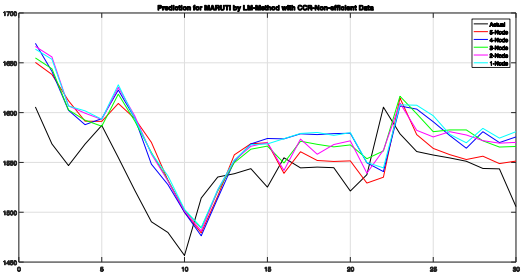
Figure 5: Prediction for Maruti and TATA Motors with All Respective, Non-efficient, and Efficient Data by CCR DEA



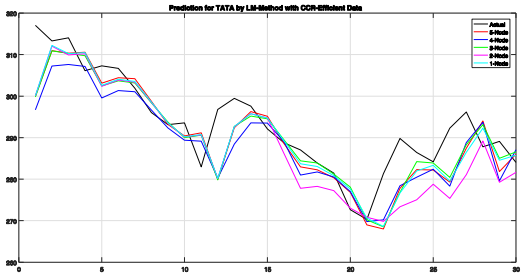
(a) lmMARUTI All Respective Data



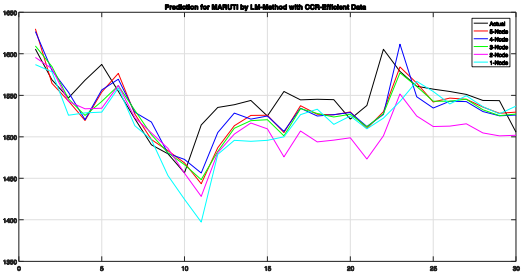
(b) lmTATA All Respective Data



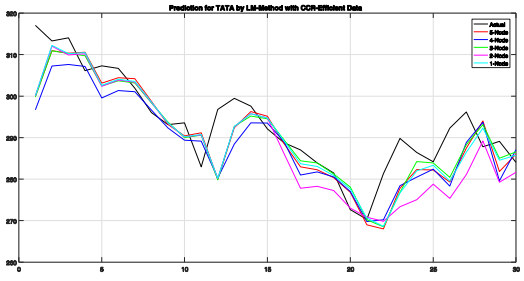
(c) lmMARUTI Nonefficient Data



(d) lmTATA Nonefficient Data



(e) lmMARUTI Efficient Data



(f) lmTATA Efficient Data

Figure 6: Prediction for Maruti and TATA Motors with All Respective, Non-efficient, and Efficient Data by BCC DEA

With the help of all considered training data, efficient and non-efficient data sets, giving them as input to BPNN-LM model with varying number of nodes from one to five in the hidden layer

of the ANN model. These data set has been segregated with CCR and BCC models of DEA approach. The performances of these approaches has been observed by root mean square error (RMSE) estimation presented in Table 1 and Table 2 respectively.

Table 1: Root Mean Square Error (RMSE) Values for Nodes and DEA CCR BPNN-LM Comparison

	5-NODE	4-NODE	3-NODE	2-NODE	1-NODE
RMSE ALL DATA CCR MARUTI	26.26	27.57	119.84	29.46	27.46
RMSE NON-EFFICIENT CCR MARUTI	47.45	53.98	46.37	86.45	51.98
RMSE EFFICIENT CCR MARUTI	38.03	29.71	27.28	31.91	28.10
RMSE ALL DATA CCR TATA	6.91	7.00	6.25	6.92	6.05
RMSE NON-EFFICIENT CCR TATA	9.11	8.56	9.66	9.57	9.56
RMSE EFFICIENT CCR TATA	6.96	11.73	7.05	6.02	6.61

We can observe in the Table 1 that for Maruti with DEA CCR BPNN-LM model, with one node at hidden layer and considering all training data, the RMSE is 27.46 whereas efficient data and three nodes in the hidden layer, the RMSE is 27.28. Similar scenario is there for TATA motors, as with one node at hidden layer and considering all training data, the RMSE is 6.05 whereas efficient data and two nodes in the hidden layer, the RMSE is 6.02. In the Table 2 that for Maruti with DEA BC BPNN-LM model, with one node at hidden layer and considering all training data, the RMSE is 27.11 whereas efficient data and four nodes in the hidden layer, the RMSE is 27.03. Similar scenario is there for TATA motors, as with one node at hidden layer and considering all training data, the RMSE is 5.97 whereas efficient data and two nodes the hidden layer, the RMSE is 5.96.

Results reflected in these two tables, express the success of ourexperiment in the perspectives that similar, rather somewhat betterperformance is being achieved with all considered data and filtered by 50% efficient data. This will reduce the processing time of the computer very much and with high volume of data, it can help us to faster our decision makings.

Table 2: Root Mean Square Error (RMSE) Values for Nodes and DEA BCC BPNN-LM Comparison

	5-NODE	4-NODE	3-NODE	2-NODE	1-NODE
RMSE ALL DATA BCC MARUTI	52.61	31.74	30.74	26.93	27.11
RMSE NON-EFFICIENT BCC MARUTI	39.05	44.00	40.76	42.63	46.34
RMSE EFFICIENT BCC MARUTI	28.30	27.03	28.71	43.07	38.48
RMSE ALL DATA BCC TATA	6.39	6.70	6.68	6.94	5.97
RMSE NON-EFFICIENT BCC TATA	8.15	8.04	8.77	9.05	9.17
RMSE EFFICIENT BCC TATA	7.09	7.73	6.87	8.35	5.96

7. Conclusion

We have focused in this paper for data preprocessing before giving it as input to ANN models, in the perspective of decision making units. We have followed the DEA technique for selecting efficient DMUs and generated DEA-BPNN-LM model for forecasting. We had performed our experimentation with providing input data as sliding window manner and varying number of nodes in the hidden layer of ANN models from one to five and after observing the performances on different set of nodes, we selected the best combination. By reducing the number of DMUs from original data set, we achieved better performance with reduced processing time in computation.

These days data preprocessing are being focused so frequently as input data to artificial neural network model and we also followed the same in both of our models. There are numerous opportunities to regulate ANN techniques with optimization and data preprocessing methods from statistical and machine learning approaches such as principal component analysis, support vector machines, regression analysis, particle sworm optimization, etc. as the future works.

References

[1] Emrouznejad A (2003), *An alternative DEA measure: A case of OCED countries*, Applied Economic Letters, 10, 779-782.

[2] Emrouznejad A and Podinovski V. (2004), *Data envelopment analysis and performance management*, UK: Aston University Print.

[3] Emrouznejad A and Thanassoulis E (2005), *A mathematical model for dynamic efficiency using data envelopment analysis*, Journal of Applied Mathematics and Computation, 160(2).

[4] Emrouznejad A and Shale E (2009), A combined neural network and DEA for measuring efficiency of large scale datasets, Computers and Industrial Engineering, 56, 249-254.

[5] Mulwa R, Emrouznejad A and Muhammad L (2008), *Economic Efficiency of small holder maize producers in Western Kenya: a DEAmeta-frontier analysis*, International Journal of Operational Research, 3(6).

[6] Kirigia J. M., Emrouznejad A., Vaz R. G., Bastiene H. and Padayach J (2008), *A comparative assessment of performance and productivity of health centers in Seychelles*, International Journal of Productivity and Performance Management, 57 (1).

[7] Bloch D. A. and Silverman B. W (1997), *Monotone Discriminant Functions and their Rheumatology*, Journal of American Statistical Association, 92 (437), 144-153.

[8] Byrd T. A. and Marshall T. E (1997), *Relating Information Technology Investment to Organizational Performance: A Causal Model Analysis*, The International Journal of Management Science, 25(1), 43-56.

[9] Quah J K H (2000), *The monotonicity of individual and market demand*, Econometrica, 68(4), 911-930.

[10] Pradhan M K and Raja Das (2015), *Application of a general regression neural network for predicting radial overcut in electrical discharge machining of AISI D2 tool steel*, International Journal of Machining and Machinability of Materials, 17(3-4), 355-369.

[11] Pradhan M K, Raja Das and Biswas C K (2010), *Prediction of surface roughness in electrical discharge machining of D2 steel using regression and artificial neural networks modeling*, Journal of Machining and Forming Technologies, 2(1-2), 25-46.

[12] Jaiswal J. K. and Das R (2017), *Application of artificial neural networks with backpropagation technique in the financial data*, Materials Science and Engineering, 263(4).

[13] Wang S (2003), *Adoptive non-parametric efficiency frontier analysis: A neural network model*, Computers and Operations Research, 30, 279-295.

[14] Kannai Y (1989), *A characterization of monotone individual demand functions*, Journal of Mathematical Economics, 18, 87-94.
[15] Charnes A, Cooper W W and Rhodes E (1978), *Measuring the Efficiency of Decision Making Units*, European Journal of Operational Research, 2.
[16] Banker R D, Charnes A and Cooper W W (1984), *Some Models of Estimating Technical and Scale Inefficiencies in DEA*, Management Science, 20(9), 1078-1092.
[17] Bansal A, Kauffman R J and Weitz R R (1993), *Comparing the Modeling Performance of Regression and Neural Networks as Data Quality Varies: A Business Value Approach*, Journal of Management Information System, 10(1), 11-32.
[18] Pendharkar P C (2005), *A Data Envelopment Analysis-Based Approach for Data Preprocessing*, IEEE Transactions on Knowledge and Data Engineering, 17 (10), 1379-1388.
[19] Troutt M D (1995), *Maximum Decisional Efficiency Principle*, Management Science, 41(1), 76-82.
[20] Stewart G. W. (1973), *Introduction to Matrix Computations*, Academic Press, New York, USA.
[21] Armijo L (1996), *Minimization of functions having Lipschitz-continuous first partial derivatives*, Pacific Journal of Mathematics, 16, 1-3.
[22] Henri P. G. (2017), *A technical report on, The Levenberg-Marquardt method for nonlinear least squares curve-fitting problems*, Department of Civil and Environmental Engineering, Duke University, North Carolina, USA.
[23] Gill P. E., Murray W. and M. H. Wright (1981), *Practical Optimization*, Academic Press, London.