Hybrid Rough Neural Stock Market Prediction System

Wael Abd El-kader Awad
Math and Comp. Dept, Faculty of Science, Suez Canal, Egypt.

Abstract

This paper proposes a rough neural predicting system with modified definitions of rough set approximations for knowledge discovery and data prediction from complex high dimensional data for stock market price index. Before using the historical data to train the network, we use rough sets as a tool for reducing and choosing the most relevant sets of internal states for predicting the index price over 7 days in the future, then we use this selected data to train a back propagation neural network.
Keywords: Neural network, Rough set, Stock market prediction.

1. Introduction

This paper deals with the application of hybridized soft computing techniques for automated stock market forecasting and trend analysis. We make use of rough neural system for analyzing the trend of the predicted stock values. We analyzed the 50 weeks stock data for the main index of Egypt Stock Exchange. Input data were preprocessed using principal component analysis and fed to a rough set system for stock forecasting. The predicted stock values are further fed to a neural system to analyze the trend of the market.
In order to get an accurate prediction, it is important to collect all the observations, and to understand which factors have most influence on the price index movement. The basic task behind discovery of knowledge from raw data is to divide the data set in the attribute space into different classes, most of the real world data with large number of attributes come with noises some of which are irrelevant to the decision under consideration. In general, the first major step is to find which attributes are important.

Rough set is an effective knowledge discovery approach at feature reduction which provides an efficient way to solve the problems in the dynamic stock exchange environment, and to remove unuseful and redundant features. So unuseful features fed into neural network adaptation process. Therefore, we can get an optimally associated linkage a minimal set of factors with hidden association concerned.

2. Rough Set theory

The rough set theory is an extension of set theory for the study of intelligent system with indiscernible, ambiguous and imperfect data information. Rough set theory appeared two decades ago. Its main thrust is used in the attribute reduction, rule-generation and prediction. For instance, by applying a mathematical approach to solve the data reduction problem, it is good at figuring out discernity, vagueness and uncertainty in data analysis. The rough set concept was introduced by Pawlak and his colleagues. This theory became very popular among scientists around the world in the last decade, and it is now one of the most developed AI methods. The primary goal of this new emerging theory is to classify the information from incomplete information (data). Its successful application to a variety of problems has shown its usefulness and versatility [2].

The information system of rough set is represented as a table which is called decision table (DT), it is the quadruple T = $\{U; A; C; D\}$, where U is a nonempty finite set of objects called the universe, A is a nonempty finite set of primitive attributes, and $C, D \subseteq A$ are two subsets of attributes that are called the condition and decision attributes [10].

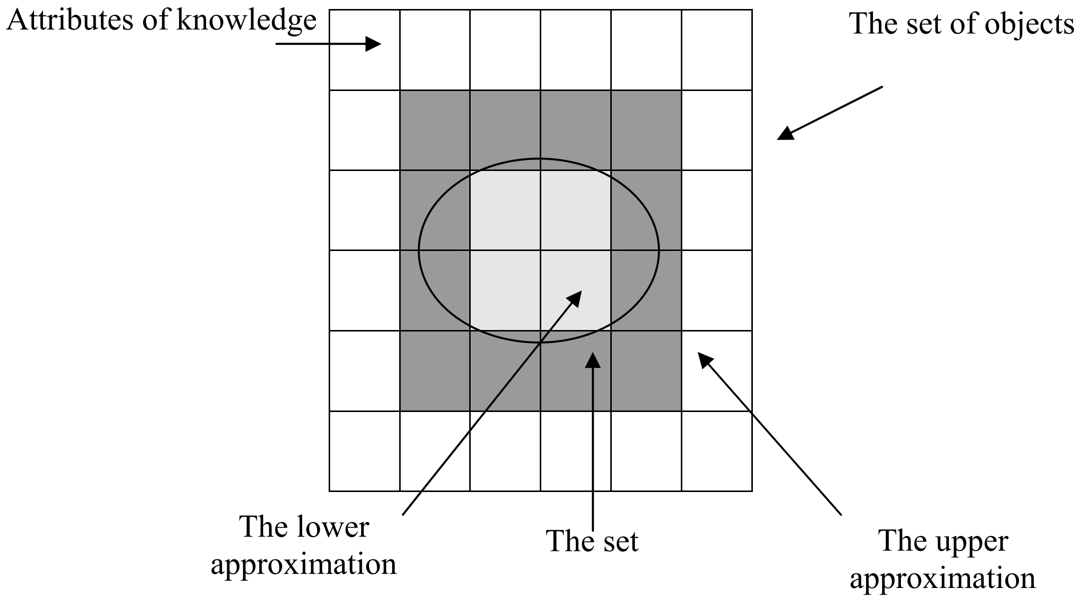


Fig1. Information system

Let $P \subset A$ be a subset of attributes. The indiscernibility relation, denoted by *IND(P)*, is an equivalence relation defined as:

$IND(P) = \{(x, y) \in U \times U : \forall a \in P, a(x) = a(y)\}$, where *a(x)* denotes the value of feature a of object x. If $(x, y) \in IND(P)$, x and y are said to be indiscernible with respect to *P*.

The family of all equivalence classes of *IND(P)* (Partition of *U* determined by *P*) is denoted by *U/IND(P)*. Each element in *U/IND(P)* is a set of indiscernible objects with respect to *P*.

For any concept $X \subseteq U$ and attribute subset $R \subseteq A$, X could be approximated by the *R*-lower approximation and *R*-upper approximation using the knowledge of *R*. The lower approximation of *X* is the set of objects of *U* that are surely in *X*, defined as:

$$R_*(X) = \bigcup\{E \in U / IND(R) : E \subseteq X\}.$$

The upper approximation of *X* is the set of objects of *U* that are possibly in *X*, defined as:

$$R^*(X) = \bigcup\{E \in U / IND(R) : E \bigcap X \neq \phi\}.$$

The boundary region is defined as:

$$BND_R(X) = R^*(X) - R_*(X).$$

If the boundary region is empty, that is, $R_*(X) = R^*(X)$, concept *X* is said to be *R*-definable. Otherwise *X* is a rough set with respect to *R* [9].

3. Artificial Neural Network

Artificial neural networks are relatively crude electronic networks of "neurons" based on the neural structure of the brain. They "learn" by comparing their prediction of the output with the known actual target.

Roughly speaking, a neuron in an artificial neural network is:

1- A set of input values (x_i) and associated weights (w_i) .

2- A function (g) that sums the weights and maps the results to an output (y).

3.1 Training an Artificial Neural Network

In the training phase, the correct class for each input is known (this is termed supervised training), and the output nodes can therefore be assigned "correct" values -- "1" for the node corresponding to the correct class, and "0" for the others. (In practice it has been found better to use values of 0.9 and 0.1, respectively.) It is thus possible to compare the calculated values of the network for the output nodes to these "correct" values, and calculate an error term for each node (the "Delta" rule). These error terms are then used to adjust the weights in the hidden layers so that, hopefully, the next time around the output values will be closer to the "correct" values[1].

3.2 The backpropagation learning method

The backpropagation learning algorithm is a supervised learning method used to train feed forward neural networks based on a set of data that contains a number of input and output pairs. In this learning method, input data are processed in the network and the resulting outputs are compared with actual outputs and the deviations or errors are used to update connection weights in a way that guarantees minimization of the sum of squared errors [6,7].

The feed forward neural network shown in Fig. 2 is used to illustrate the backpropagation learning method. Given an input vector x_p, $p=1,2,\ldots,N$ and its target output, where N is the size of the training sample, the objective of the learning process is to compute a set of weights w_{ij} and w_{jk} for the neural network connections that map inputs into the corresponding outputs as to minimize the sum of squared errors between target outputs T_k and the outputs produced by the network during the learning process O_k. The objective of the learning process can be stated as:

$$\min \quad E = \sum_k (T_k - O_k)^2$$

The objective of minimizing the error function E is achieved by continually changing the values of the weights w_{ij} and w_{jk} until a set of weights that minimize E has been reached. The backpropagation learning method includes a number of steps:

1. The weights w_{ij} and w_{jk} are randomly initialized.

2. A training vector $\boldsymbol{x}_p$ is presented to the input layer and, by using an activation function (e.g., the sigmoid function), the output of each neuron in the hidden layer is calculated as follows:

$$Z_j = f(net_j)$$

$$net_j = \sum_i w_{ij} X_i \qquad i = 1,2,..,I; \;\; j = 1,2,..,J$$

The outputs of the neurons in the hidden layer are used to calculate the outputs of the neurons in the output layer which are the outputs of the neural network:

$$O_K = f(net_k)$$

$$net_k = \sum_j w_{jk} Z_j \qquad k = 1,2,..,K$$

3. The connection weights of the neural network are adjusted in order to minimize the error calculated at each node in the output layer. This error is the difference between what the neural network actually produces using the weights w_{ij} and w_{jk} and what the network should produce based on actual outputs. The weights connecting the neurons in the hidden layer and in the output layer are adjusted first using the following equation:

$$w_{jk}(t+1) = w_{jk}(t) + \Delta w_{jk}(t)$$

where

$$\Delta w_{jk}(t) = \eta \, \theta_k \, Z_j + \alpha w_{jk}(t-1)$$

$$\theta_k = (T_k - O_k) \frac{\partial O_k}{\partial net_k}$$

The term $w_{jk}(t+1)$ is the set of weights in iteration $t+1$ and $w_{jk}(t)$ is the set of weights in iteration *t* of the training process whereas the term $\frac{\partial O_k}{\partial net_k}$ is the derivative of the activation function used in the output layer. If the sigmoid function is used, the derivative will be $O_k(1-O_k)$. The terms η and α are constants between 0 and 1 used to control and improve the quality of the training process. The constant η is the learning rate whereas α$\alpha$is the learning momentum and will be briefly discussed in the next subsection.

4. After all the weights connecting neurons in the hidden layer and the output layer have been adjusted, the weights connecting the neurons in the input and hidden layers have to be adjusted as follows:

$$w_{ij}(t+1) = w_{ij}(t) + \Delta w_{ij}(t)$$

where

$$\Delta w_{ij}(t) = \eta \, \theta_j \, X_i + \alpha w_{ij}(t-1)$$

$$\theta_j = \frac{\partial Z_j}{\partial net_j} \sum_k \theta_k w_{jk}$$

The term $\frac{\partial Z_j}{\partial net_j}$ is the derivative of the activation function for the nodes in the hidden layer and for the sigmoid function it will be $Z_j(1-Z_j)$.

There are two main activities performed in this learning method: forward pass and backward pass. In the forward pass stage, inputs to the network are received through the neurons in the input layer and based on these inputs, the outputs of the neurons in the hidden layer are determined. Similarly, the outputs of the neurons in the hidden layer are used as inputs to the neurons in the output layer and by using an activation function, the outputs of the neurons in the output layer of the network are determined [10].

In the reverse pass stage, deviations between target and network outputs are calculated and back propagated to update the weights where the weights connecting hidden and output layers are updated first then the weights connecting input and hidden layers. Forward and reverse passes in the backpropagation learning method are repeated continuously for each data pair until the weights that minimize the deviations between the outputs produced by the network and target outputs are obtained [10].

4. Hybrid Rough Neural Prediction System

Hybrid model proposed in this paper is composed of rough set component and neural network component. Neural network shows us its strong ability to solve complex problems for medical image processing. But neural network cannot tell the redundant information from huge amount of data, which will easily lead to some problems such as too complex network structure, long training time, low converging speed and much computation[8].

By rough set, some rules are extracted from the information system. Using rough set tool, we can discover knowledge in two kinds of rules: deterministic and non-deterministic. Sometimes, the rules generated by rough set approach fail to predict newly entered object because of non-deterministic rules. To handle this situation, some researchers reported that reduced data set (horizontally or vertically) is fed into neural network for complementing the limitation of rough set, which finally produces full prediction of new case data[1].

To increase the efficiency of the prediction process, rough sets with Boolean reasoning discretization algorithm is used to discretize the data. Rough set reduction technique is applied to find all the reducts of the data, which contains the minimal subset of attributes that are associated with a class label for prediction. So useless and reduct features removed from the dataset, and uncertainty prediction rule applied on dataset to find the best predictor from the features.

The main procedure is based on the principle of indifference combined with the maximum entropy principle, where the entropy estimates the mean number of comparisons minimally necessary to retrieve the equivalence class information of a randomly chosen element $x \in U$ and we can define it as $H(P) \overset{def}{=} \sum_{i=0}^{k} \frac{r_i}{n} \log_2(\frac{n}{r_i})$, where P is a partition of U with class $x_i, i \leq k$, each have cardinality r_i. The prediction approach is based on the assumption that structure and amount of uncertainty can be estimated by the interaction of *d* and *C*.

The estimator $H^{loc}(d \mid C)$ measures the uncertainty to predict membership in a class of θ_d given a class of θ_C, and the importance of $H^{loc}(C \to d)$ is that it enables the researcher to compare different attribute sets C_i in terms of a common unit of measurement which cannot be done by a conditional measure of prediction success like $H^{loc}(d \mid C)$.

In order to be able to compare different entropies within a model, we define a normalized entropy measure – bounded within [0,1] – as follows: If $H(d) = \log_2(n)$, i.e. if θ_d is the identity, then

$$S^{loc}(C \to d) \overset{def}{=} \begin{Bmatrix} 1, & if\ \theta_C = \theta_d, \\ 0, & otherwise. \end{Bmatrix}$$

Uncertainty in the sense of this model is not a feature of the predictor set *C,* but a local feature of the intersection of equivalence classes $X \in \theta_C$ and $Y \in \theta_d$.

The maximum normalized entropy measure votes for the best predicted indicators[3].

The output from the rough set indicator prediction system will act as the input of backpropagation network in our design.

The input datasets with all features are filtered by rough-sets logic and subsequently the models based on the minimum feature sets are built with the integration of rough neurons into Neural Rough Networks. Out network architecture is consisting of three layer feed forward rough neural network for processing, $N_{input} \rightarrow N_{hidden} \rightarrow 1_{output}$. During this process, the error in the output neuron is propagated back from one layer to the previous layer by adjusting the weights between connected neurons.

A rule of thumb known as the Baum-Haussler rule[4], is used to determine the number of hidden neurons to be used:

$$N_{hidden} \leq \frac{N_{train} \cdot E_{tolerance}}{N_{pts} + N_{output}}$$

Where N_{train} is the number of hidden neurons, $E_{tolerance}$ is the error tolerance, N_{pts} is the number of data inputs during training phase, N_{output} is the number of output neurons. In our application, N_{output}=1. and let $E_{tolerance}$=0.002. so the hidden neuron depends on number of inputs and number of training iteration.

5. Implementation of Rough Neural Stock Market Prediction System

Based on the technical analysis, past information will affect the future. So, there should be some relationship between the stock prices of today and future. The relationship can be obtained through a group of mapping of constant time interval.

In our implementation 50 weeks' stock data of E-F-G Hermes company as day-to-day stock data cannot be predicted only from the stock price itself, we consider several factors which give rise to a multidimensional data set. The factors considered are:

1- Recent change of index price per week.
2- Variation of Egypt stock market index (case 30).
3- New York stock market index (Daw Jones) average.
4- Four dummy variables which are instant rate, economic growth rate, dollar price and petroleum price

These factors are used as inputs to the rough system to remove unuseful and reduct features from the dataset. They applied the uncertainty prediction rule to find the best predictor from the features. Predicted factors from rough set system are used as the inputs to a neural network and the index of stock price is used to supervise the training process, in order to discover implicit rules governing the price movement of Egypt Stock Exchange Price Index (ESEPI). Finally, the trained neural network is used to predict the future levels of ESEPI. The technical analysis method is used commonly to forecast the ESEPI, buying and selling point. Neural network could be used to recognize the patterns of the chart and the value of index.

There are two principal phases in neural network analysis, "learning" and "predicting". During the learning, or training, phase the network "learns" by adjusting the weights between it nodes. The input data must be presented to the network many times. Data are split into two groups. The first is used to train the network and the second file is used as a test of the networks predictive ability. During the training phase the network weights are saved at many intervals and tested to see how well the network can predict outcomes using weights it has learned up to that point. Following thousands of iterations, convergence occurs and the best weights for each element of the network can be derived.

The system architectural framework for stock market price index prediction is shown in figure 3.

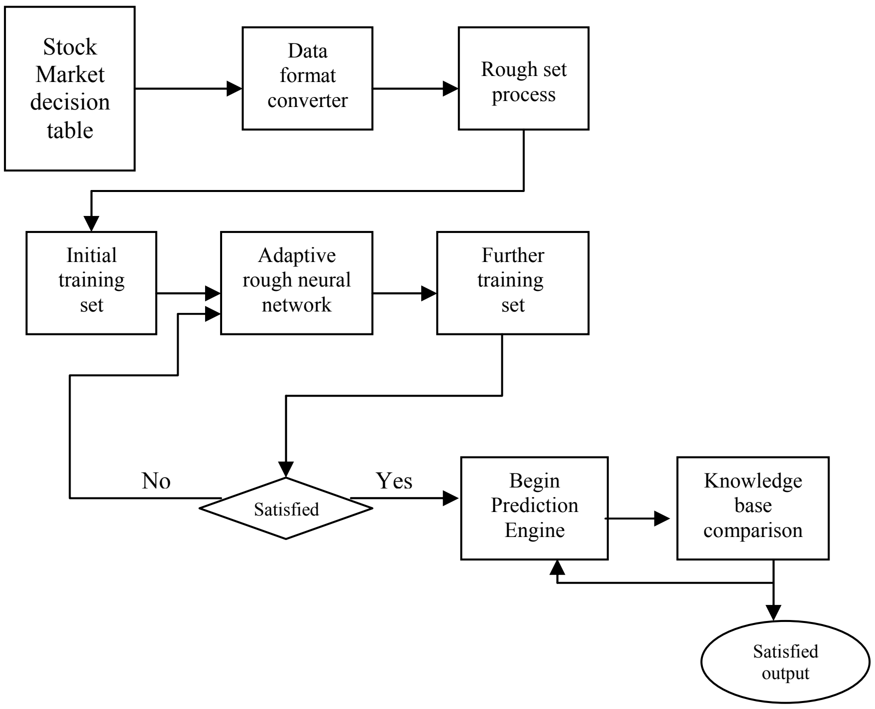


Fig3. illustrative Schematic for Stock Prediction System

5.1 Implementation Algorithm

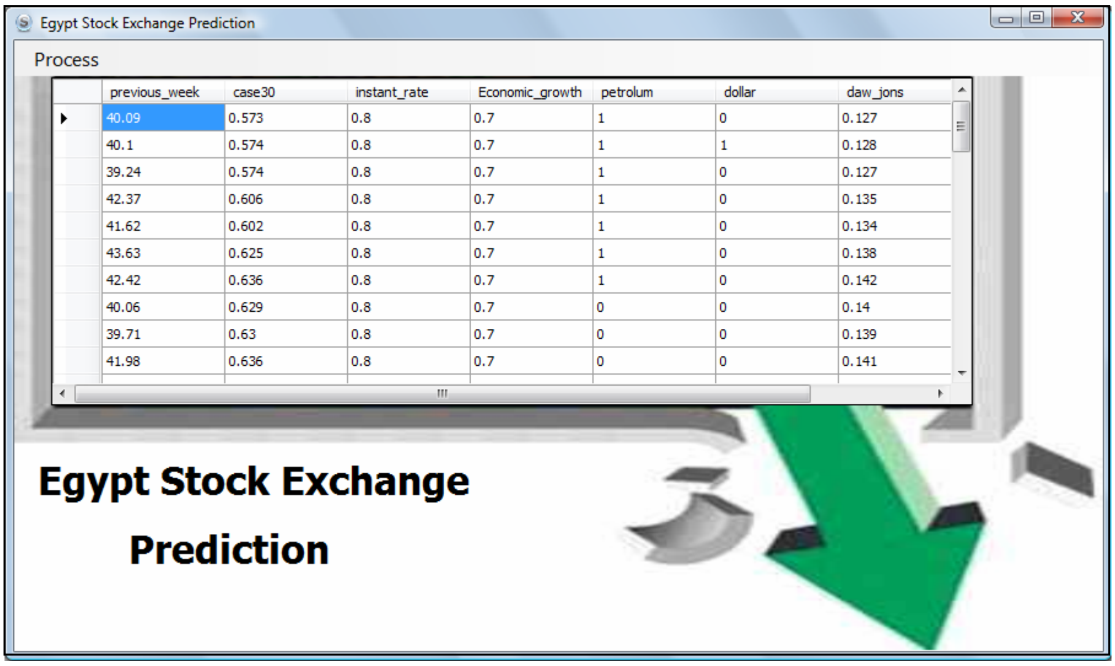
1. Initialize decision table initially and refine set of possible data of stock price index.

2. Apply rough uncertainty prediction rule and find the normalized entropy measure to find the reduct and best predictors[3].

3. Build the rough neural network over the predicted features.

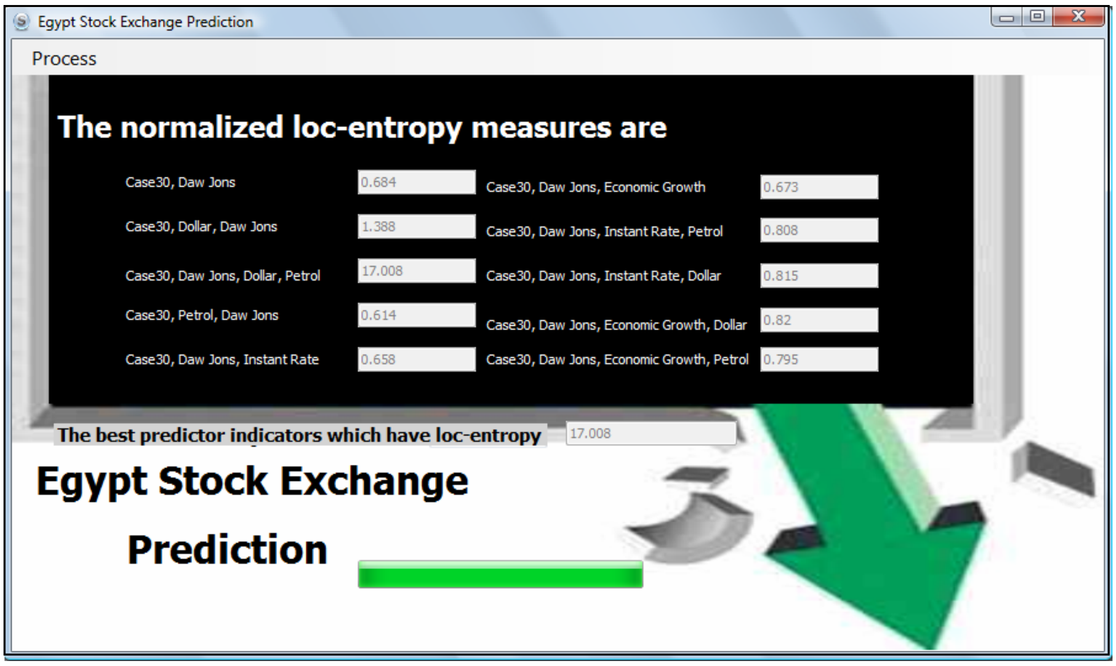
4. Adopt the structure of neural network by applying backpropagation algorithm, and compute number of hidden neuron.
5. Perform training stage on the predicted features.
6. If the network output is not satisfied, repeat step 5 until the classification error of the network is satisfied.
7. Compare the output of network with historical data so as to find any errors.

This algorithm translate into program code using C# language, which consists of two phases, the first on is to compute the best predictors by rough system, the second phase applied backpropagtation to find the satisfied output. Figure 4 shows the first phase in the program, it is allocating the historical data which will be the input in the training phase, it includes 50 weeks as training set and 15 weeks as testing set.



previous_week	case30	instant_rate	Economic_growth	petrolum	dollar	daw_jons
40.09	0.573	0.8	0.7	1	0	0.127
40.1	0.574	0.8	0.7	1	1	0.128
39.24	0.574	0.8	0.7	1	0	0.127
42.37	0.606	0.8	0.7	1	0	0.135
41.62	0.602	0.8	0.7	1	0	0.134
43.63	0.625	0.8	0.7	1	0	0.138
42.42	0.636	0.8	0.7	1	0	0.142
40.06	0.629	0.8	0.7	0	0	0.14
39.71	0.63	0.8	0.7	0	0	0.139
41.98	0.636	0.8	0.7	0	0	0.141

Fig 4 Stock decision table



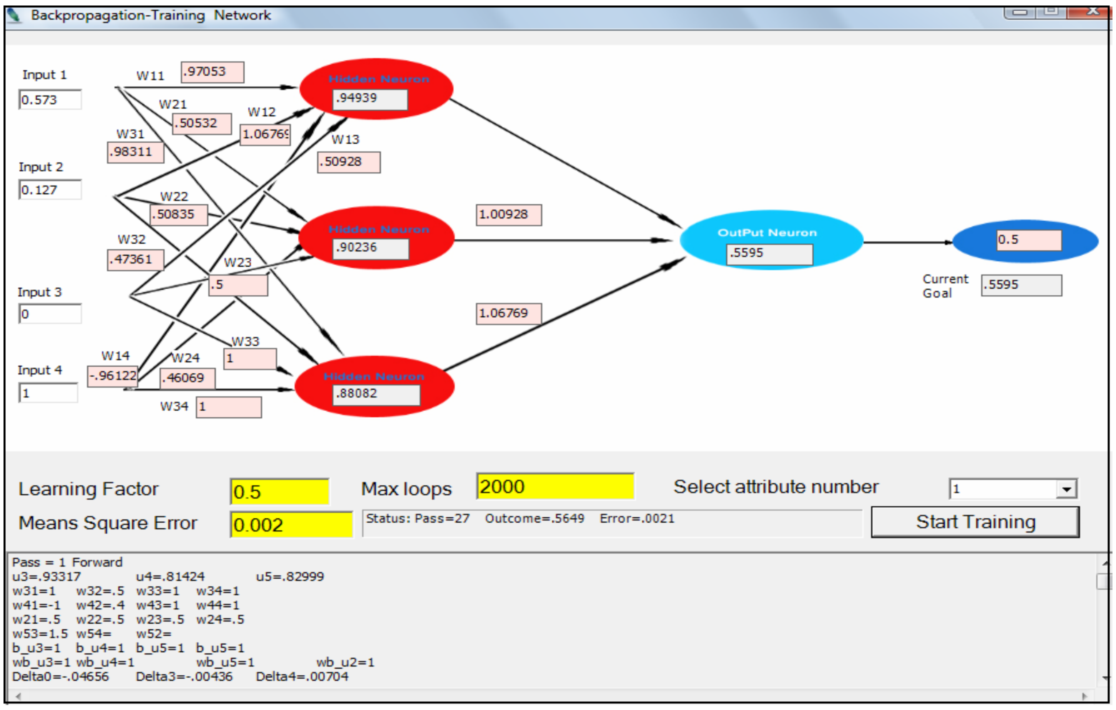
The normalized loc-entropy measures are

Case30, Daw Jons	0.684	Case30, Daw Jons, Economic Growth	0.673
Case30, Dollar, Daw Jons	1.388	Case30, Daw Jons, Instant Rate, Petrol	0.808
Case30, Daw Jons, Dollar, Petrol	17.008	Case30, Daw Jons, Instant Rate, Dollar	0.815
Case30, Petrol, Daw Jons	0.614	Case30, Daw Jons, Economic Growth, Dollar	0.82
Case30, Daw Jons, Instant Rate	0.658	Case30, Daw Jons, Economic Growth, Petrol	0.795

The best predictor indicators which have loc-entropy: 17.008

Fig 5. Find the best predicted indicators with rough system

The next phase shows the output from the rough indicator prediction system, here the best predictor is taken due to the maximum normalized loc-entropy measure, so as shown in figure 5.2 the best predicted indicators are Case30, Daw Jones, Dollar, Petrol.



Learning Factor: 0.5
Max loops: 2000
Select attribute number: 1
Means Square Error: 0.002
Status: Pass=27 Outcome=.5649 Error=.0021

```
Pass = 1 Forward
u3=.93317          u4=.81424          u5=.82999
w31=1     w32=.5   w33=1    w34=1
w41=-1    w42=.4   w43=1    w44=1
w21=.5    w22=.5   w23=.5   w24=.5
w53=1.5   w54=     w52=
b_u3=1    b_u4=1   b_u5=1   b_u5=1
wb_u3=1 wb_u4=1             wb_u5=1            wb_u2=1
Delta0=-.04656     Delta3=-.00436     Delta4=.00704
```

Fig 6. Training phase

The next phase is training the backpropagation neural network, by taking the inputs from rough set system and apply backpropagation training algorithm to find the best weights for our prediction engine.

Figure 7 shows comparison between actual and predicted values of 21 weeks of testing data, the figure shows the difference between the actual and predicted value. As shown the maximum error is 0.1.

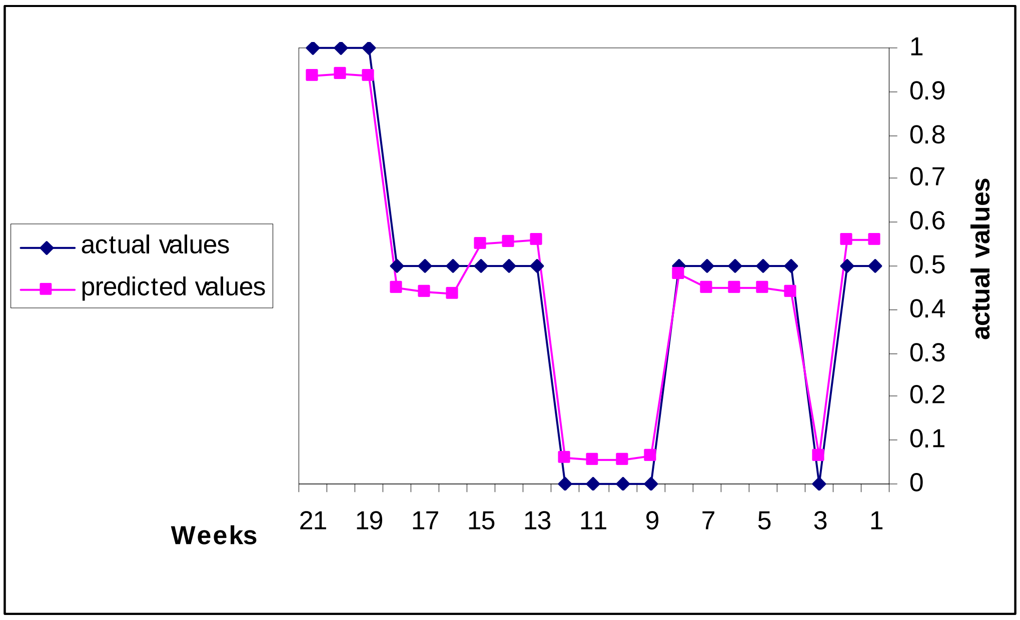


Fig7. Comparison between the actual and predicted values

6. Conclusion

A rough neural prediction system is proposed for knowledge discovery and data prediction from complex high dimensional data for stock market price index. Predicting stock price index is a complex process and does not depend on the past stock values only. There are other factors which require analysis of multidimensional data with different characteristics of the attributes.

In our research we used concepts from rough set theory and proposed some modifications to handle real world data to find best predicted indicators. This facilitates dividing the data into smaller subspaces. Finally from the training set of neural networks the final decision can be achieved by predicting of the outputs from the network. The simulation results show that the proposed rough neural predicting system is a practical approach to handle problems with complex high dimensional data like stock price analysis.

References

[1] AhnB.S; Cho.S.S; Kim.C.Y, The Integrated Methodology of Rough Set Theory and Artificial Neural Network for Business Failure Prediction, Expert Systems with Applications, Vol. 18 , 2000 P. 65– 74

[2] Al-Qaheri. H; Hassanien. A. E; Abraham. A, Discovering Stock Price Prediction Rules using Rough Sets, International Journal on Non-standard Computing and Artificial Intelligence, Vol.18, 2008, P 181-198.

[3] Düntsch. I; Gediga. G, Uncertainty Measures of Rough Set Prediction, Artificial Intelligence, Vol. 106, 1998, P 109-137.

[4] E.Baum; D. Haussler, What size net gives valid generalization?, Neural Computation, 1989, P 151-160.

[5] Hassanien. A. E, Selzak.D, Rough Neural Intelligent Approach for Image Classification: A Case of Patients with Suspected Breast Cancer; International Journal of Hybrid Intelligent Systems; Vol.4, P 205-218.

[6] Herbert. J, Yao. J. T, Time-Series Data Analysis with Rough Sets, 4th International Conference on Computational Intelligence in Economics and Finance (CIEF), Salt Lake City, USA, July 21-26, 2005, P 908-911.

[7] Hindi. A; A Neural Network Approch to Forcasting Stock Prices of Saudi Companies, J. King Saud Univ, Vol. 14, Admin Sci. (1), 2002, P 65-81.

[8] Khan. A; Motwant. M; Sharma.S, Stock Rate Prediction Using Backpropagation Algorithm: Results with Different Number of Hidden Layers, Gournal of Software Engineering, 2007, P13-27.

[9] Watanabe. H; Chakraborty. B; Chakraborty. G, Rough Neuro Voting System for Data Mining: Application to Stock Price Preiction, Springer-Verlag Berlin Heidelberg, 2007, P 558-565.

[10] Zhong. N; Skowron. A, A Rough Set-based Knowledge Discovery process, Int. J. Appl. Math. Comput. Sci., Vol.11; No.3, 2001,P 603-619.