Egyptian Informatics Journal (2014) **15**, 129–147

Cairo University

Egyptian Informatics Journal

www.elsevier.com/locate/eij
www.sciencedirect.com

ORIGINAL ARTICLE

A novel Neuro-fuzzy classification technique for data mining

Soumadip Ghosh [a,*], **Sushanta Biswas** [b], **Debasree Sarkar** [b], **Partha Pratim Sarkar** [b]

[a] *Academy of Technology, Aedconagar, Hooghly, 712121 WB, India*
[b] *Department of Engineering and Technological Studies (DETS), University of Kalyani, 741235 WB, India*

Received 14 February 2014; revised 15 July 2014; accepted 25 August 2014
Available online 17 September 2014

KEYWORDS

Classification;
Data mining;
Neural network;
Neuro-fuzzy

Abstract In our study, we proposed a novel Neuro-fuzzy classification technique for data mining. The inputs to the Neuro-fuzzy classification system were fuzzified by applying generalized bell-shaped membership function. The proposed method utilized a fuzzification matrix in which the input patterns were associated with a degree of membership to different classes. Based on the value of degree of membership a pattern would be attributed to a specific category or class. We applied our method to ten benchmark data sets from the UCI machine learning repository for classification. Our objective was to analyze the proposed method and, therefore compare its performance with two powerful supervised classification algorithms Radial Basis Function Neural Network (RBFNN) and Adaptive Neuro-fuzzy Inference System (ANFIS). We assessed the performance of these classification methods in terms of different performance measures such as accuracy, root-mean-square error, kappa statistic, true positive rate, false positive rate, precision, recall, and f-measure. In every aspect the proposed method proved to be superior to RBFNN and ANFIS algorithms.

1. Introduction

Data mining has attracted many researchers and analysts in the information industry and in research organizations as a whole in the last decades, due to the availability of large amounts of data and the immediate need for transforming such data into meaningful information and knowledge. The useful knowledge gathered can be applied in many areas such as market survey, customer retention, production control, evolutionary analysis and science exploration [1,2].

Classification is an important data mining technique which involves extracting interesting patterns representing knowledge

* Corresponding author.
E-mail addresses: soumadip@rediffmail.com (S. Ghosh), biswas.su@gmail.com (S. Biswas), dsarkar70@gmail.com (D. Sarkar), parthabe91@yahoo.co.in (P.P. Sarkar).
Peer review under responsibility of Faculty of Computers and Information, Cairo University.

Production and hosting by Elsevier

1110-8665 © 2014 Production and hosting by Elsevier B.V. on behalf of Faculty of Computers and Information, Cairo University.
Open access under CC BY-NC-ND license. http://dx.doi.org/10.1016/j.eij.2014.08.001

from large real-world databases. Such analysis can provide a deep insight into the better understanding of different large-scale databases. The study related to effective knowledge development is also very popular in any research as because the decision-making process mainly depends upon the effectiveness of the classification method being utilized.

Basically data classification [3,4] is the method of discovering a model or classifier that describes and differentiates data classes so that the model could predict the class of entities with unknown class label value. It is a two-step procedure, in the first step; a classifier is constructed denoting a predefined set of concepts or data classes. This is the training phase, where a classification algorithm constructs the classifier by learning from a training data set and their associated class label attributes. In the next step the model is used for classification. In order to estimate the performance of the classifier a test set independent of the training tuples is used. Several preprocessing steps such as data cleaning, data selection and data transformation are also applied to the data set before the classification procedure takes place.

Artificial neural network (ANN) or simply neural network (NN) [5–7] is a popular data modeling tool that can perform intelligent tasks similar to the human brain. NN is well-known for high precision and high learning ability even when a very little information is available. One of the reliable methods of data classification from the neural network domain is the Multilayer Perceptron Backpropagation network (MLPBPN) [8,9] algorithm. The Radial Basis Function Neural Network (RBFNN) [10,11] is another powerful neural network model that utilizes radial basis functions as the activation functions. The output of such a neural network model is the linear combination of radial basis functions of inputs and neuron parameters. RBFNNs have many utilities, comprising classification, function approximation, system control, and prediction of time series.

Due to the presence of imprecise input information, ambiguity or vagueness in input data, overlapping boundaries among classes, and indefiniteness in defining features some uncertainties can still arise at any stage of a data classification system. The fuzzy set theory [12–14] as a generalization of the classical set theory is very flexible in handling different aspects of uncertainties or incompleteness about real life situations. In a fuzzy system the features are associated with a degree of membership to different classes. Both NNs and fuzzy systems are very adaptable in estimating the input–output relationships. Neural networks deal with numeric and quantitative data while fuzzy systems can handle symbolic and qualitative data. Neuro-fuzzy hybridization leads to a crossbreed intelligent system widely known as Neuro-fuzzy system (NFS) [15,16] that exploits the best qualities of these two approaches efficiently. The hybrid system unites the human alike logical reasoning of fuzzy systems with the learning and connectedness structure of neural networks by means of fuzzy set theory based approach.

There is another Neuro-fuzzy classification based model which comprises a set of interpretable IF-THEN rules. They consider two conflicting requirements in fuzzy modeling: interpretability against accuracy. In reality, one of the two properties persists. Therefore the rule based Neuro-fuzzy modeling research area is divided into two branches: the linguistic fuzzy modeling that focuses on interpretability, primarily the Mamdani model; and the exact fuzzy modeling that focuses on accuracy, mainly the Sugeno model or Takagi–Sugeno–Kang (TSK) model. The rule based Neuro-fuzzy classification approach normally applies the concept of adaptive neural network. An adaptive network is a network of nodes (processing elements) and directed links (weights) that is functionally equivalent to a Fuzzy Inference System and is referred to as Adaptive Neuro-fuzzy Inference System or ANFIS [15]. It normally employs the Sugeno fuzzy model to produce IF-THEN learning rules. The nodes of an adaptive network are associated with certain parameters which might have an impact on the final output. ANFIS generally utilizes a hybrid learning algorithm which is the combination of gradient descent and least square method to adapt the parameters in adaptive network. To put it simply, ANFIS is the combination of MLPBPN and Sugeno fuzzy model. A fuzzy rule in the Sugeno model has the following form

$$\text{IF } x \text{ is } P \text{ and } y \text{ is } Q \text{ THEN } z = f(x, y) \tag{1}$$

where P and Q are the fuzzy sets in the antecedent part of the given IF-THEN learning rule and $z = f(x,\ y)$ is a crisp function in the consequent part of the rule.

In our present study, we proposed a novel Neuro-fuzzy classification technique for data mining that uses a combination of MLPBPN and fuzzy set theory approach. We applied our method to ten benchmark data sets from the UCI machine learning repository for analysis and, therefore compare its performance with RBFNN and ANFIS based classification models.

This research study is arranged as follows: Section 2 includes the related works done in this area. Section 3 describes our proposed Neuro-fuzzy classification method while Section 4 explains the detailed procedure. Section 5 discusses the performance analysis and results; and Section 6 is reserved for the conclusion.

2. Related works

Neuro-fuzzy classification is a field of research that has caused a great deal of attention in the recent decades. In the Neuro-fuzzy paradigm, several attempts have been made [17–20] which led to a strong foundation for research. In that context we have presented some of the works done in this area by other researchers.

A study performed by Kuncheva [21] described how to utilize fuzzy pattern recognition concept in solving real life problems. According to her work, fuzzy pattern recognition can be related to fuzzy clustering or with fuzzy IF-THEN systems used as classifiers and it is close to any pattern classification paradigm that involves fuzzy sets. She also indicated that fuzzy systems combined with neural networks should exploit the merits of these two approaches efficiently.

Pedrycz [22] specified that artificial neural network model combined with fuzzy set theory based approaches should possess the merits of both and it should permit one to arrive at a more knowledgeable decision making systems. The research study established that Neuro-fuzzy hybridization leads to a crossbreed intelligent system that can handle real life situations reasonably well.

A research work performed by Castellano et al. [23] identified that the use of a Neuro-fuzzy system and an evolutionary fuzzy system hybridizes the approximate reasoning mechanism

of fuzzy systems with the learning abilities of artificial neural networks and evolutionary algorithms. The aim of their work was to study different hybrid soft computing based systems by considering the combined use of evolutionary algorithms and artificial neural networks in order to empower fuzzy systems with learning and adaptive capabilities. This research work vastly contributes to the evolutionary Neuro-fuzzy systems.

Abonyi et al. [24] designed Computational Intelligence (CI) based techniques by combining fuzzy rule-based expert systems and data mining algorithms founded on Soft Computing principles. They too established that such techniques could be considered for feature selection, feature extraction, rule base optimization and rule base simplification. Applications of these CI based techniques were demonstrated successfully using the benchmark Wine data classification problem.

Keleş and Keleş [25] described how to extract strong fuzzy rules using a Neuro-fuzzy classification tool named NEFCLASS. According to their work, positive predictive value of the rule base was 75% and negative predictive value was 93%.

Nauck [26] designed a combination of neural network and fuzzy rule based reasoning system. With proper demonstration they showed that such a combined approach could enhance the performance of control, decision-making and data analysis systems.

These research works [27,28] also contributed immensely in the field of Neuro-fuzzy hybridization by employing different statistical and machine learning techniques that worth mentioning.

3. Proposed Neuro-fuzzy method

In the present article, a novel Neuro-fuzzy classification method is proposed. The method extracts the feature-wise information about a set of input patterns, fuzzifies its corresponding pattern values using a membership function (MF), and provides the degree of the memberships of individual patterns to several classes. Let us assume that we have N input patterns and M classes. Let us also consider that each pattern consists of k attributes. The block diagram of the proposed classification model is shown in Fig. 1 below.

The method is divided into three steps which are described below.

Step 1 (Fuzzification process): In the first step, a membership matrix of order $N \times M$ is generated which consists of the degree of memberships of N different patterns to M different classes. Each element in this matrix is a membership function of the form $mf_{i,j}(y_i)$, where y_i is the i-th pattern value of input pattern vector y with $i = 1, 2, \ldots, N$ and $j = 1, 2, \ldots, M$. The MF can be defined as

$$mf_{i,j}(y_i) = \text{degree of membership of pattern } i \text{ to class } j \quad (2)$$

$$\text{where the } i\text{-th pattern} \quad y_i = x_{i1}, x_{i2}, x_{i3}, \ldots, x_{ik} \quad (3)$$

The input pattern vector y is thus described as

$$y = [y_1, y_2, \ldots, y_N]^T \quad (4)$$

where 'T' denotes the matrix transpose operation.

For fuzzification, we have used the generalized bell-shaped MF which depends upon three different parameters a, b, and c as given by the equation

$$mf(y; a, b, c) = \frac{1}{1 + \left|\frac{y-c}{a}\right|^{2b}} \quad (5)$$

Each of these configuration parameters has some special significance: parameter c determines the center of the MF; a denotes half-width; and b (together with a) controls the slopes at the different crossover points. Fig. 2 below shows a typical bell-shaped MF. By modifying the values of a, b, and c; we will obtain our desired MF which provides more flexibility for classification.

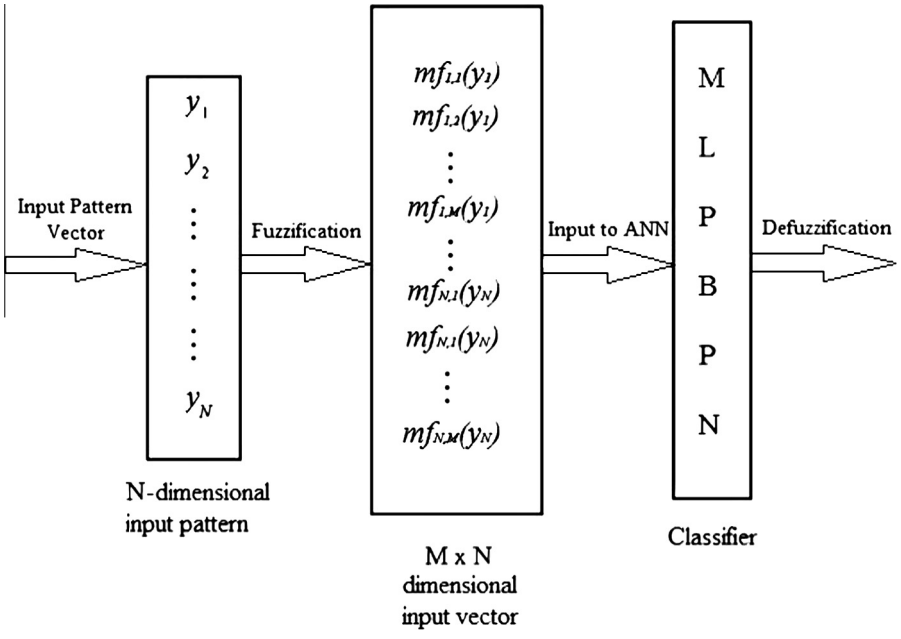

Figure 1 Proposed Neuro-fuzzy classification model.

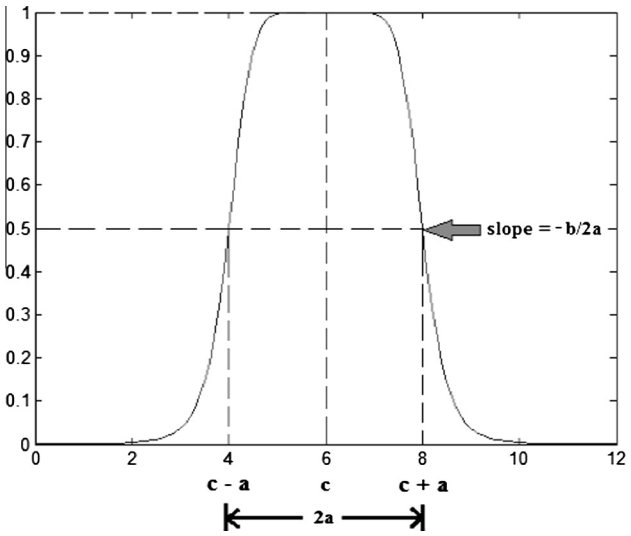

Figure 2 A typical generalized bell-shaped MF.

After the fuzzification phase the membership matrix for a particular pattern vector y would actually look like this:

$$\mathrm{MF}(y)=\begin{bmatrix} mf_{1,1}(y_1) & mf_{1,2}(y_1) & mf_{1,3}(y_1) & \cdots & mf_{1,M}(y_1) \\ mf_{2,1}(y_2) & mf_{2,2}(y_2) & mf_{2,3}(y_2) & \cdots & mf_{2,M}(y_2) \\ mf_{3,1}(y_3) & mf_{3,2}(y_1) & mf_{3,3}(y_3) & \cdots & mf_{3,M}(y_3) \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ mf_{N,1}(y_N) & mf_{N,2}(y_N) & mf_{N,3}(y_N) & \cdots & mf_{N,M}(y_N) \end{bmatrix} \tag{6}$$

where $mf_{i,j}(y_i)$ is the membership of i-th pattern of input vector y with $i = 1, 2, \ldots, N$ and $j = 1, 2, \ldots, M$.

Step 2 (Building the MLPBPN classifier): The second step of the proposed method transforms the membership matrix into an $M \times N$ vector by transposing all rows and columns. The vector is then taken as input to a MLPBPN classifier. A typical MLPBPN model has a single input layer, at least one hidden layer, and a single output layer. It exhibits two modes of operation: feedforward and backpropagation. In the network architecture, the input units are connected in a feedforward mode, with input layer units fully connected to the hidden layer units, and hidden units fully connected to the output layer units. The nodes in the input layer and in the hidden layer(s) are associated with weights (connection strengths). Initially, before training, all *weights* and *biases* are selected randomly. In backpropagation mode, the errors, as well as the learning procedure (updating the weights and biases), transmit in the backward direction starting from the output layer unit to the inner nodes. This process is repeated multiple times, and the network model keeps running to minimize the root-mean-square error between the network's predicted and target values until all the training samples are processed or the stopping criterion has been reached. To demonstrate this, a hidden layer or output layer unit (block) is shown in Fig. 3 below.

The net input to a MLPBPN unit j in the hidden or output layers is computed as a linear combination of its inputs. The predicted (activation) output of unit j is given as follows:

$$Output_j = \frac{1}{1+e^{-Net_j}} \tag{7}$$

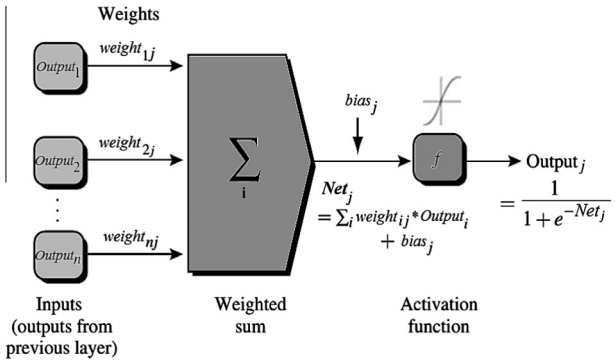

Figure 3 A hidden or output layer unit j of a MLPBPN model.

where Net_j = the net input of unit j in the MLPBPN model.

The net input can be defined as a weighted sum of the connection strengths (or weights) and the output from the previous layer:

$$Net_j = \sum_i weight_{ij} * Output_i + bias_j \tag{8}$$

where $weight_{ij}$ = the connection strength (or weight) of the link from unit i in the preceding layer to unit j, $Output_i$ = output of unit i from the preceding layer, and $bias_j$ = the bias of the unit.

We calculate the total sum of squared errors from the expected output, $Target_j$:

$$Error = \frac{1}{2}\sum_j \left(Target_j - Ouput_j\right)^2 \tag{9}$$

The backpropagation network's weights are then adjusted to reduce this overall error.

$$\Delta Weight \propto -\frac{\partial Error}{\partial Weight} \tag{10}$$

We will now consider output layer block j with a particular weight value, $weight_{ij}$.

$$\Delta weight_{ij} \propto -\frac{\partial Error}{\partial weight_{ij}} \tag{11}$$

Because the error is not directly a function of the weight, we can expand this as follows:

$$\Delta weight_{ij} = -\eta * \frac{\partial Error}{\partial Output_j} * \frac{\partial Output_j}{\partial Net_j} * \frac{\partial Net_j}{\partial weight_{ij}} \tag{12}$$

where η is a positive constant called the *learning rate*.

Following this we have the weight updating formula as

$$weight_{ij} = weight_{ij} + \Delta weight_{ij} \tag{13}$$

Similarly, we can get the formula for bias updating:

$$bias_j = bias_j + \Delta bias_j \tag{14}$$

After a rigorous analysis, we have selected a MLPBPN model with a single hidden layer. The ANN model has utilized gradient descent with momentum as a supervised learning rule, and transfer function in the hidden and output layer as tan sigmoid. The number of nodes in the input layer of the MLPBPN is equal to the number of input attributes in the data set. Similarly, the number of output layer nodes is same as the number of classes present in the data set. The choice of the number of

processing elements (PEs) present in the hidden layer is also a crucial parameter. After a proper investigation we have selected the number of PEs in the hidden layer [29], L, as given by the equation

$$L = (\text{number of input attributes} + \text{number of classes}) * 2/3 \quad (15)$$

Step 3 (Defuzzification process): In the last step, the proposed NFS classifier utilizes a hard classification by employing a maximum (max) operation to defuzzify the activation output of the MLPBPN. An input pattern is **y** designated to a particular class j with the largest class membership value if and only if

$$mf_j(y) \geq mf_i(y) \ \forall i \in (1, 2, \ldots, M) \text{ and } i \neq j \quad (16)$$

where $mf_i(y)$ is the activation value of the i-th neuron in the output layer of the MLPBPN model.

4. Detailed procedure

The broad level steps of the detailed procedure are described below.

Step 1: The following preprocessing steps are applied to each of the original data set before the classification procedure—

(1a) *Data cleaning:* This denotes the preprocessing of data for removing or reducing noise and the handling of missing values. A missing value is typically replaced by the mean value for that attribute based on statistics.

(1b) *Data selection:* Statistical correlation analysis is used to discard redundant attributes so that only the relevant attributes are taken from the given data set.

(1c) *Data transformation:* This is used to normalize the data set as because ANN based technique requires distance measurements in the training phase. It transforms attribute values to a small-scale range like −1.0 to +1.0.

Step 2: After preprocessing steps, the original data set is divided into two sub-sets namely the training data set and the test data set. We have used two-third of the whole data set for training purpose and the remaining one-third data for testing purpose.

Step 3: In the training phase, the training data set applies to our proposed Neuro-fuzzy system (NFS) for developing a classification model. The training data set is also applied to the RBFNN and ANFIS techniques separately for building other classifiers.

Step 4: In the testing phase, the three classification models (NFS, RBFNN and ANFIS) are then employed to the test set for estimating the performance of each classifier.

Step 5: The performances of these models are then compared based on the different performance measures (accuracy, root-mean-square error, tp-rate, fp-rate, precision, recall, f-measure, kappa statistic).

The broad level steps of the detailed procedure are described below in Fig. 4.

5. Experimental results and analysis

The three classification techniques namely NFS, RBFNN, and ANFIS are trained and tested on ten benchmark data sets from the University of California, Irvine (UCI) machine learning repository using MATLAB software (version R2013a). These data sets are namely *Breast Cancer Wisconsin*, *KDD Cup 1999* (*10-percent*), *Statlog Landsat Satellite*, *Mammographic Mass*, *Wilt*, *Mushroom*, *Pima Indians Diabetes*, *Iris*, *Spambase*, and *Car Evaluation*. Several comparisons are performed; a comparison of classification accuracy, root-mean-square error (RMSE), kappa statistic values; and, a comparison of True Positive Rate (TP-Rate), False Positive Rate (FP-Rate), Precision, Recall, and *F*-Measure values derived from the confusion matrix of each classifier.

5.1. Performance measures

After developing the classifiers by the above mentioned classification techniques, they are applied to the test data set for

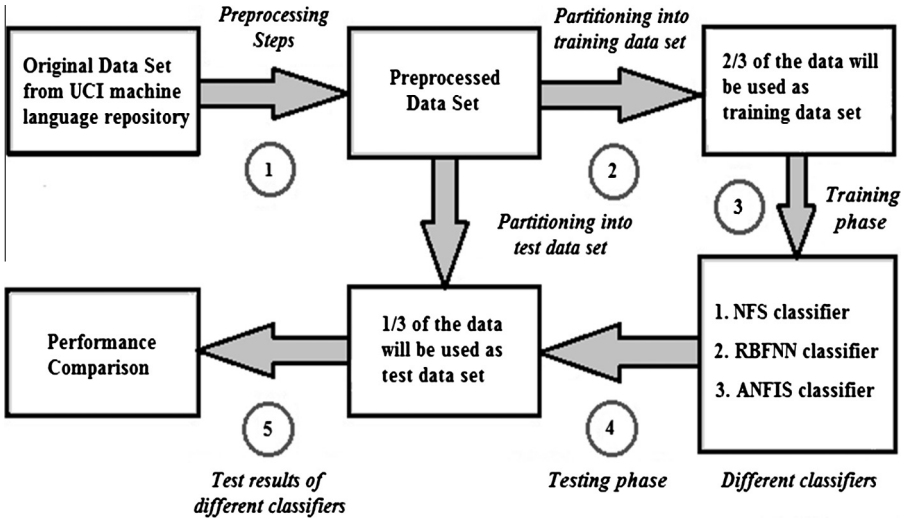

Figure 4 Broad level steps of the detailed procedure.

performance evaluation. The performances of these models are then estimated based on the different performance measures described below.

5.1.1. Root-mean-square error (RMSE)

RMSE [30] is a well-known performance measure of the difference between the values anticipated by a classifier and the values actually discovered from the surroundings of the system being modeled. The RMSE of a classifier prediction with respect to the computed variable $e_{classifier}$ is determined as the square root of the mean-squared error:

$$\text{RMSE} = \sqrt{\frac{\sum_{k=1}^{n}(e_{\text{discovered},k} - e_{classifier,k})^2}{n}} \tag{17}$$

where $e_{discovered}$ are the discovered values and $e_{classifier}$ are the predicted values for $\forall$k.

5.1.2. Kappa statistic

The Kappa statistics [31], denoted by κ, is a long-familiar statistical measure of inter-rater agreement for categorical data. In statistics, kappa is considered to be a measure of reliability among different raters or judges. The value of κ is estimated by the equation:

$$\kappa = \frac{prob(O) - prob(C)}{1 - prob(C)} \tag{18}$$

where $prob(O)$ is the probability of observed agreements among raters, and $prob(C)$ is the probability of agreements expected by chance. If $\kappa = 1$ the raters have completely agreed with each other's decision. If $\kappa = 0$ then the judges or raters are not agreed to comply with.

5.1.3. Confusion matrix

In the machine learning domain, the confusion matrix [32] is a particular tabular representation that provides visualization of the performance of a classification algorithm. Each column of the matrix denotes the examples in a predicted class, while each row indicates the examples in an actual class. This helps to find out whether a classification algorithm is commonly mislabeling one as another. It is a table layout with two rows and two columns that allows more detailed analysis than classification accuracy. Classification accuracy is not a reliable metric for assessing the performance of a classifier as it may produce misleading results when the numbers of samples in different classes vary greatly. Table 1 below presents the confusion matrix for a two class classifier with the following data entries:

(a) True positive (tp) is the number of 'positive' instances categorized as 'positive'.
(b) False positive (fp) is the number of 'negative' instances categorized as 'positive'.
(c) False negative (fn) is the number of 'positive' instances categorized as 'negative'.
(d) true negative (tn) is the number of 'negative' instances categorized as 'negative'.

Table 1 A Confusion matrix for a two class classifier.

	Predicted class	
	Positive	*Negative*
Actual class		
Positive	tp	fp
Negative	fn	tn

Several standard terms have been defined in a two-class confusion matrix. The term *accuracy* is the ratio of sum of instances that were correctly classified to total number of instances present. It is calculated using the equation:

$$accuracy = \frac{tp + tn}{tp + tn + fp + fn} \tag{19}$$

The *precision* is the ratio of the predicted positive instances that were correct, as computed using the equation:

$$precision = \frac{tp}{tp + fp} \tag{20}$$

The *fp-rate* is the ratio of negative instances that were incorrectly classified as positive, determined by the equation:

$$fp\text{-}rate = \frac{fp}{fp + tn} \tag{21}$$

The *recall* or *tp-rate* is the ratio of positive instances that were correctly discovered, as estimated using the equation:

$$recall = tp - rate = \frac{tp}{tp + tn} \tag{22}$$

In some scenarios high precision may be more important, while in other scenarios high recall may be more significant. However, in most types, we try to improve both values. The combined form of these values is called the *f-measure*, and usually expressed as the harmonic mean of both these values:

$$f\text{-}measure = \frac{2 * precision * recall}{precision + recall} \tag{23}$$

5.2. Results and analysis

NFS, RBFNN and ANFIS classifiers are applied on each of the ten UCI machine learning repository data sets for investigation and performance analysis which is reported below. We have employed the two-third of each data set for training purpose and the remaining one-third data set for testing purpose. The results described here are exclusively based on the simulation experiment that we have taken.

5.2.1. Breast Cancer Wisconsin data set

We have used the benchmark Breast Cancer Wisconsin (Original) data set for diagnosis of breast cancer. It has 699 tuples and consists of 11 attributes including the class attribute. Of them the first 10 attributes are considered as input attributes. The class attribute has exactly two values, namely Benign and Malignant. The numbers of tuples per class with percentage values are—Benign: 458 (65.5%) and Malignant: 241 (34.5%). There are 16 tuples in this data set that contain a single missing attribute value, denoted by "?".

After the training phase is over, each of the three classifiers is applied to a test set for classification. Firstly, the performance comparisons of these classifiers are done based on the different performance measures such as classification accuracy, RMSE, and the kappa statistic as shown below in Table 2.

Table 2 Performance comparisons of three classifiers.

Classifier	Classification accuracy (%)	RMSE	Kappa statistic
NFS	**98.4**	0.1536	0.9255
RBFNN	**94.2**	0.2032	0.8623
ANFIS	**95.3**	0.1728	0.8884

From Table 2 we could see that, the NFS classifier has an accuracy of 98.4%. RBFNN model has classification accuracy of 94.2%; while ANFIS model has an accuracy of 95.3%. Surely, accuracy wise NFS has performed better than RBFNN and ANFIS. Then we have analyzed the performance of each classifier based on the information on RMSE and the kappa statistic values collected from Table 2. Fig. 5 below shows a performance comparison among these classifiers.

The experiment has employed very commonly used indicator like the RMSE value which should be as low as possible. Kappa statistic is used to estimate the accuracy for distinguishing between the reliability of the classified data collected and their validity. As it is evident from Fig. 5, the kappa statistic value of the selected algorithms is around 0.81–1.0. According to the definition of kappa statistic, the accuracy of the classification methods is considered to be 'almost perfect agreement'. Based on the result, NFS comes out first with an RMSE value of 0.1536 and a kappa statistic value of 0.9255; followed by ANFIS having an RMSE value of 0.1728 and a kappa statistic value of 0.8884 and RBFNN stands last with the highest RMSE value (0.2032) and the lowest kappa statistic value (0.8623). Therefore, with regard to the performance measures such as classification accuracy, RMSE and kappa statistic, the proposed NFS classifier has performed the best.

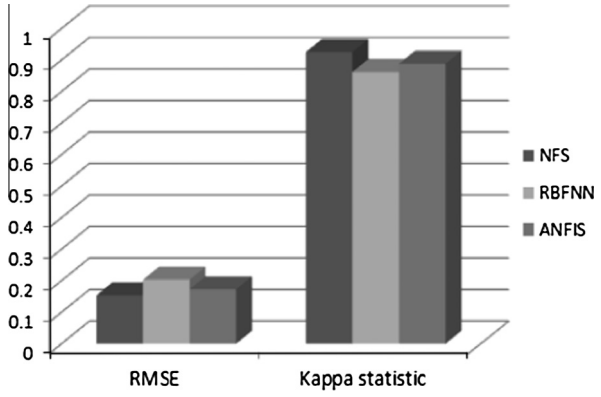

Figure 5 Comparison of RMSE and Kappa statistic values.

Next, the performances of these models are compared based on the TP-Rate (or Recall), FP-Rate, Precision, and *F*-Measure values derived from the confusion matrix of individuals with respect to the test data set. The detailed accuracy by each class (Benign and Malignant) for these classifiers is shown below in Table 3. The weighted average values are also shown in the following table. The results reported here are entirely based on simulation experiment. For evaluating the performance of a classifier, we would expect higher values for TP-Rate, Precision, Recall, *F*-Measure; and lower value for FP-Rate.

We have compared the performance of each classifier based on the information on a weighted average of different performance measures from Table 3; which is then presented in the form of a 3-D column chart as shown below in Fig. 6.

From Table 3 we could discover that the weighted average values of TP-Rate (or Recall), FP-Rate, Precision, and *F*-Measure for proposed NFS classifier are 98.4%, 3.2%, 98.4%, and 98.4%, respectively; whereas for RBFNN classifier the values are 94.2%, 6.8%, 94.2%, and 94.2% respectively. For ANFIS these values are 95.3%, 5.4%, 95.3%, and 95.3% respectively. Surely, the NFS model has the highest weighted average values for TP-Rate, Precision, Recall, and *F*-Measure and the lowest weighted average value for FP-Rate. Regarding *F*-Measure as the best performance measure derived from a confusion matrix; NFS has the highest value for the *F*-Measure as 98.4%, followed by ANFIS having an *F*-Measure value of 95.3% and RBFNN with an *F*-Measure value of 94.2%. Fig. 6 also supports this observation.

5.2.2. KDD Cup 1999 (10-percent) data set

The classification of network anomaly is an effervescent research area. We have used this data set for analyzing network anomaly based classification problem with regard to the Intrusion Detection Systems (IDSs). The data set has 400,000 records and consists of 42 attributes including the class attribute. We here classify the network by two classes, namely *Normal* and *Anomaly*. We have chosen this data set because it is free from any redundant record, so the classifiers will not be biased towards more frequent data records. Moreover, this data set contains no missing values of any attribute.

After the training phase is over, each of the three classifiers is applied to a test set for classification. Firstly, the

Table 3 Detailed accuracy by each class for three classifiers.

Classifier	Class	TP-rate /recall (%)	FP-rate (%)	Precision (%)	*F*-measure (%)
NFS	Benign	99.1	5.1	98.9	98.7
	Malignant	98.7	1.8	98.5	97.3
	Weighted average	98.4	3.2	98.4	98.4
RBFNN	Benign	95.2	8.7	95.3	95.2
	Malignant	92.9	4.2	92.4	92.1
	Weighted average	94.2	6.8	94.2	94.2
ANFIS	Benign	96.9	7.3	95.5	96.2
	Malignant	91.3	3.1	94.1	92.6
	Weighted average	95.3	5.4	95.3	95.3

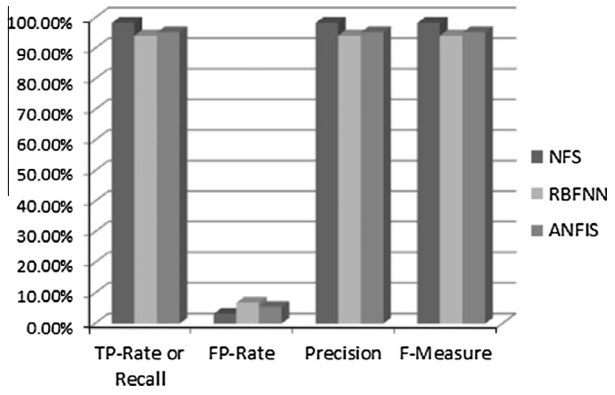

Figure 6 Comparison of three classifiers on weighted average values.

Table 4 Performance comparisons of three classifiers.

Classifier	Classification accuracy (%)	RMSE	Kappa statistic
NFS	**98.7**	0.1033	0.9676
RBFNN	**95.8**	0.1595	0.8981
ANFIS	**96.3**	0.1371	0.9368

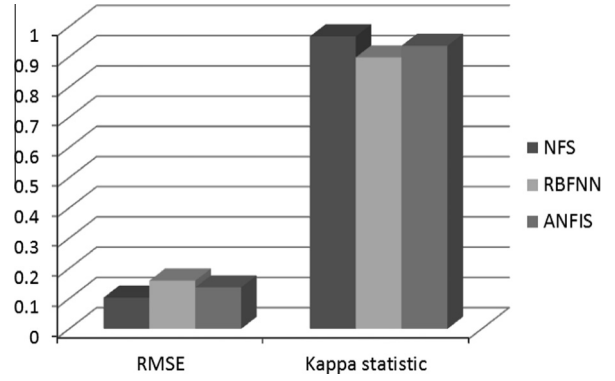

Figure 7 Comparison of RMSE and Kappa statistic values.

performance comparisons of these classifiers are done based on the different performance measures such as classification accuracy, RMSE, and the kappa statistic as shown below in Table 4.

From Table 4 we could see that, NFS classifier has an accuracy of 98.7%. RBFNN model has classification accuracy of 95.8%; while ANFIS model has an accuracy of 96.3%. Surely, accuracy wise NFS has performed better than RBFNN and ANFIS. Then we have analyzed the performance of each classifier based on the information on RMSE and the kappa statistic values collected from Table 4. Fig. 7 below shows a performance comparison among these classifiers.

The experiment has employed very commonly used indicator like the RMSE value which should be as low as possible. Kappa statistic is used to estimate the accuracy for distinguishing between the reliability of the classified data collected and their validity. As it is evident from Fig. 7, the kappa statistic value of the selected algorithms is around 0.81–1.0. According to the definition of kappa statistic, the accuracy of the classification methods was considered to be 'almost perfect agreement'. Based on the result, NFS comes out first with an RMSE value of 0.1033 and a kappa statistic value of 0.9676; followed by ANFIS having an RMSE value of 0.1371 and a kappa statistic value of 0.9368 and RBFNN stands last with the highest RMSE value (0.1595) and the lowest kappa statistic value (0.8981). Therefore, with regard to the performance measures such as classification accuracy, RMSE and kappa statistic, the proposed NFS classifier has performed the best.

Next, the performances of these models are compared based on the TP-Rate (or Recall), FP-Rate, Precision, and *F*-Measure values derived from the confusion matrix of individuals with respect to the test data set. The detailed accuracy by each class (Normal and Anomaly) for these classifiers is shown below in Table 5. The weighted average values are also shown in the following table. The results reported here are entirely based on simulation experiment. For evaluating the performance of a classifier, we would expect higher values for TP-Rate, Precision, Recall, *F*-Measure; and lower value for FP-Rate.

We have compared the performance of each classifier based on the information on a weighted average of different performance measures from Table 5; which is then presented in the form of a 3-D column chart as shown below in Fig. 8.

From Table 5 we have discovered that the weighted average values of TP-Rate (or Recall), FP-Rate, Precision, and *F*-Measure for proposed NFS classifier are 98.6%, 2.9%, 98.6%, and 98.6%, respectively; whereas for RBFNN classifier these values are 95.7%, 4.7%, 95.7%, and 95.7% respectively. For ANFIS these values are 96.3%, 3.9%, 96.3%, and 96.3% respectively. Surely, the NFS model has the highest weighted

Table 5 Detailed accuracy by each class for three classifiers.

Classifier	Class	TP-rate/recall (%)	FP-rate (%)	Precision (%)	*F*-measure (%)
NFS	Normal	98.6	3.9	98.9	99.5
	Anomaly	98.9	1.4	98.4	98.4
	Weighted average	98.6	2.9	98.6	98.6
RBFNN	Normal	95.2	7.7	95.3	95.2
	Anomaly	93.9	4.2	93.4	93.1
	Weighted average	95.7	4.7	95.7	95.7
ANFIS	Normal	97.9	6.7	95.7	96.2
	Anomaly	94.4	3.1	94.3	92.6
	Weighted average	96.3	3.9	96.3	96.3

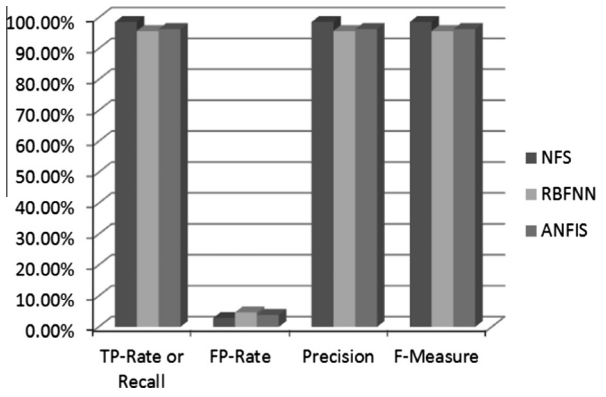

Figure 8 Comparison of three classifiers on weighted average values.

average values for TP-Rate, Precision, Recall, and *F*-Measure and the lowest weighted average value for FP-Rate. Considering *F*-Measure as the best performance measure derived from a confusion matrix; the NFS model has the highest precision value for the *F*-Measure as 98.6%, followed by the ANFIS model which has an *F*-Measure value of 96.3% and RBFNN with an *F*-Measure value of 95.7%. Fig. 8 also provisions this reflection as well.

5.2.3. Statlog Landsat Satellite data set

We have used the Statlog Landsat Satellite data set of Australian agricultural land in predicting the land classes constituting dissimilar soil types. The current database is constructed choosing only a small section (82 rows × 100 columns) from the original Landsat Multispectral Scanner (MSS) imagery data having 4 spectral bands in one single image frame. Such interpretation is quite useful for the integrated approach to remote sensing. In a satellite image, 3 × 3 (=9) square neighborhood of pixels is selected and the corresponding 4 spectral values of pixels are calculated. This multivariate data set has 6435 tuples and contains 36 (=4 spectral bands × 9 pixels in the neighborhood) input attributes and one class label attribute. The input attributes lie quantitative with their values in the range of 0–255. The classification approach is associated with the central pixel in each neighborhood area. Therefore we need to consider only 4 input attributes. The class label attribute contains 6 values—red (class 1), cotton (class 2), gray (class 3), damp gray (class 4), vegetation stubble (class 5), and very damp gray (class 7). There is no example of class 6 (mixture type) in the data set.

After the training phase is over, each of the three classifiers is applied to a test set for classification. Firstly, the performance comparisons of these classifiers are done based on the different performance measures such as classification accuracy, RMSE, and the kappa statistic as shown below in Table 6.

From Table 6 we could see that, NFS classifier has an accuracy of 92.3%. RBFNN model has classification accuracy of 87.2%; while ANFIS model has an accuracy of 85.4%. Surely, accuracy wise NFS has performed better than RBFNN and ANFIS. Then we have analyzed the performance of each classifier based on the information on RMSE and the kappa statistic values collected from Table 6. Fig. 9 below shows a performance comparison among these classifiers.

Table 6 Performance comparisons of three classifiers.

Classifier	Classification accuracy (%)	RMSE	Kappa statistic
NFS	**92.3**	0.1346	0.9181
RBFNN	**87.2**	0.1698	0.8785
ANFIS	**85.4**	0.2363	0.8263

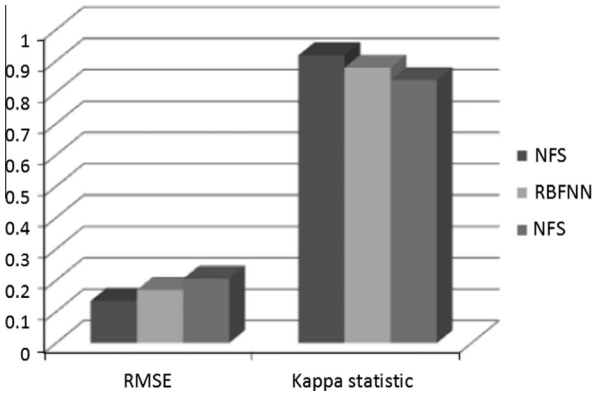

Figure 9 Comparison of RMSE and Kappa statistic values.

The experiment has employed very commonly used indicator like the RMSE value which should be as low as possible. Kappa statistic is used to estimate the accuracy for distinguishing between the reliability of the classified data collected and their validity. As it is evident from Fig. 9, the kappa statistic value of the selected algorithms is around 0.81–1.0. According to the definition of kappa statistic, the accuracy of the classification methods is considered to be 'almost perfect agreement'. Based on the result, NFS comes out first with an RMSE value of 0.1346 and a kappa statistic value of 0.9181; followed by RBFNN having an RMSE value of 0.1698 and a kappa statistic value of 0.8785 and ANFIS stands last with the highest RMSE value (0.2363) and the lowest kappa statistic value (0.8263). Therefore, with regard to the performance measures such as classification accuracy, RMSE and kappa statistic, the proposed NFS classifier has performed the best.

Next, the performances of these models are compared based on the TP-Rate (or Recall), FP-Rate, Precision, and *F*-Measure values derived from the confusion matrix of individuals with respect to the test data set. The detailed accuracy by each of the six classes for these classifiers is shown below in Table 7. The weighted average values are also shown in the following table. The results reported here are entirely based on simulation experiment. For evaluating the performance of a classifier, we would expect higher values for TP-Rate, Precision, Recall, *F*-Measure; and lower value for FP-Rate.

We have compared the performance of each classifier based on the information on a weighted average of different performance measures from Table 7; which is then presented in the form of a 3-D column chart as shown below in Fig. 10.

From Table 7 we have discovered that the weighted average values of TP-Rate (or Recall), FP-Rate, Precision, and *F*-Measure for proposed NFS classifier are 92.3%, 1.9%, 92.3%, and 92.3%, respectively; whereas for RBFNN classifier these values are 87.2%, 3.2%, 87.2%, and 87.2% respectively. For ANFIS these values are 85.4%, 4.3%, 85.4%, and 85.4%

Table 7 Detailed accuracy by each class for three classifiers.

Classifier	Class	TP-rate/recall (%)	FP-rate (%)	Precision (%)	*F*-measure (%)
NFS	1	97.5	2.2	93.4	95.4
	2	97.5	0.1	99.2	98.3
	3	96.6	2.3	92.1	94.3
	4	66.7	0.2	77.4	71.7
	5	93.6	0.9	92.8	93.2
	7	91.5	1.6	94.5	93.0
	Weighted average	92.3	1.9	92.3	92.3
RBFNN	1	97.8	1.3	96.8	97.3
	2	96.0	0.6	95.6	95.8
	3	89.9	2.5	90.8	90.4
	4	64.9	3.3	71.0	67.8
	5	79.7	1.9	85.1	82.4
	7	89.1	5.6	83.6	86.3
	Weighted average	87.2	3.2	87.2	87.2
ANFIS	1	99.3	1.4	96.4	97.9
	2	93.3	0.7	96.3	94.8
	3	94.0	3.6	86.7	90.2
	4	42.2	4.3	54.6	47.6
	5	75.5	1.8	84.8	79.9
	7	83.8	7.7	78.2	80.9
	Weighted average	85.4	4.3	85.4	85.4

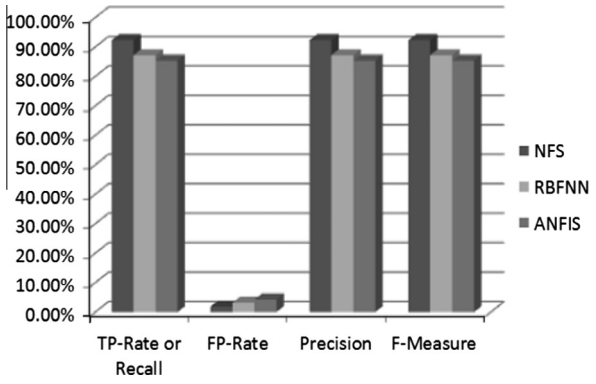

Figure 10 Comparison of three classifiers on weighted average values.

respectively. Surely, the NFS model has the highest weighted average values for TP-Rate, Precision, Recall, and *F*-Measure and the lowest weighted average value for FP-Rate. Considering *F*-Measure as the best performance measure derived from a confusion matrix; the NFS model has the highest precision value for the *F*-Measure as 92.3%, followed by the RBFNN model which has an *F*-Measure value of 87.2% and ANFIS with an *F*-Measure value of 85.4%. Fig. 10 also supports this observation.

5.2.4. Mammographic Mass data set

Then, we have used the benchmark Mammographic Mass data set for breast cancer detection. The given UCI data set is the outcome of research conducted on human breast tissue masses by using Breast Imaging-Reporting and Data System (BI-RADS) combined with digital mammography. This data set has 961 numbers of tuples and consists of 6 attributes comprising the BI-RADS assessment, Age, three BI-RADS features, namely Shape, Margin, and Density; and lastly the class attribute named Severity. The implications of each of the attributes in the MM data set are described below.

The first attribute in this data set is BI-RADS assessment which ranges from 1 to 5. The second attribute Age denotes the age of the patient age in years. The third, fourth and fifth attributes are regarded to be BI-RADS features associated with a mammographic mass lesion and they are namely Shape, Margin, and Density. Shape indicates the shape of a mammographic mass and it ranges from 1 to 4. Margin is the mass margin and it takes on values from 1 to 5. Density specifies the mammographic mass density and it ranges from 1 to 4. The class attribute has got only two values, namely Benign (normal) and Malignant (cancerous) denoted by numeric values '0' and '1' respectively.

After the training phase is over, each of the three classifiers is applied to a test set for classification. Firstly, the performance comparisons of these classifiers are done based on the different performance measures like classification accuracy, RMSE, and the kappa statistic as shown below in Table 8.

From Table 8 we could see that, NFS classifier has an accuracy of 84.4%. RBFNN model has classification accuracy of 80.2%; while ANFIS model has an accuracy of 81.7%. Surely, accuracy wise NFS has performed better than RBFNN and ANFIS. Then we have analyzed the performance of each classifier based on the information on RMSE and the kappa statistic values collected from Table 8. Fig. 11 below shows a performance comparison among these classifiers.

Table 8 Performance comparisons of three classifiers.

Classifier	Classification accuracy (%)	RMSE	Kappa statistic
NFS	**84.4**	0.3459	0.6798
RBFNN	**80.2**	0.3853	0.6025
ANFIS	**81.7**	0.3623	0.6354

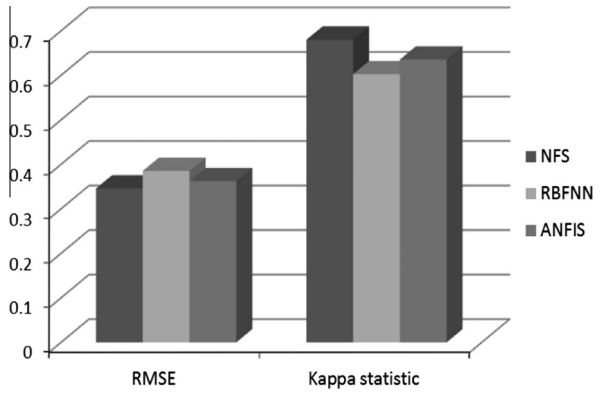

Figure 11 Comparison of RMSE and Kappa statistic values.

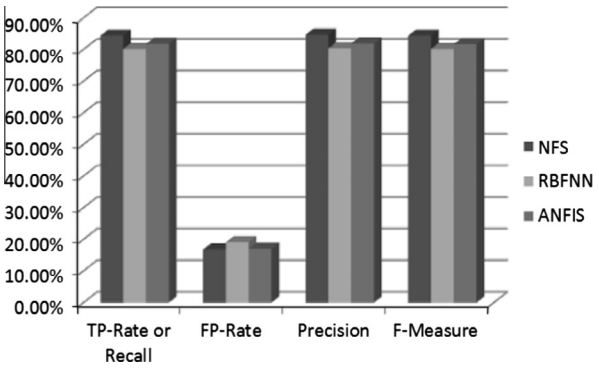

Figure 12 Comparison of three classifiers on weighted average values.

The experiment has employed very commonly used indicator like the RMSE value which should be as low as possible. As it is evident from Fig. 11, the kappa statistic value of the selected algorithms was around 0.61–0.80. According to the definition of kappa statistic, the accuracy of the classification methods is considered to be 'substantial'. Based on the result, NFS comes out first with an RMSE value of 0.3459 and a kappa statistic value of 0.6798; followed by ANFIS having an RMSE value of 0.3623 and a kappa statistic value of 0.6354 and RBFNN stands last with the highest RMSE value (0.3853) and the lowest kappa statistic value (0.6025). Therefore, with regard to the performance measures such as classification accuracy, RMSE and kappa statistic, the proposed NFS classifier has performed the best.

Next, the performances of these models are compared based on the TP-Rate (or Recall), FP-Rate, Precision, and *F*-Measure values derived from the confusion matrix of individuals with respect to the test data set. The detailed accuracy by each class (benign and malignant) for these classifiers is shown below in Table 9. The weighted average values are also shown in the following table. The results reported here are entirely based on simulation experiment. For evaluating the performance of a classifier, we would expect higher values for TP-Rate, Precision, Recall, *F*-Measure; and lower value for FP-Rate.

We have compared the performance of each classifier based on the information on a weighted average of different performance measures from Table 9; which is then presented in the form of a 3-D column chart as shown below in Fig. 12.

From Table 9 and Fig. 12, we could discover that the weighted average values of TP-Rate (or Recall), FP-Rate, Precision, and *F*-Measure for the NFS classifier are 84.4%, 16.8%, 84.7%, and 84.4%, respectively; whereas for RBFNN classifier these values are 80.2%, 19.1%, 80.4%, and 80.2% respectively. For ANFIS these values are 81.7%, 17.0%, 81.9%, and 81.7% respectively. Surely, the NFS model has the highest weighted average values for TP-Rate, Precision, Recall, and *F*-Measure and the lowest weighted average value for FP-Rate. Considering *F*-Measure as the best performance measure derived from a confusion matrix; the NFS model has the highest precision value for the *F*-Measure as 84.4%, followed by the ANFIS model which has an *F*-Measure value of 81.7% and RBFNN with an *F*-Measure value of 80.2%.

5.2.5. *Wilt data set*

This multivariate data set represents a high-resolution remotely sensed data set from a remote sensing survey in the Quickbird imagery. The data set consists of image segments from the pan-chromatic image band (Pan Band) and usually applied in predicting the soil type of diseased trees. It has 4889 tuples and consists of 6 attributes. The first attribute denotes the Gray Level Co-Occurrence Matrix with respect to the Pan band. The second, third and fourth columns indicate mean green value, mean red value, and mean Near Infrared value respectively. The fifth attribute denotes the Standard deviation value with respect to the Pan band. The class attribute has two values namely 'w' denoting diseased trees and class 'n' indicating

Table 9 Detailed accuracy by each class for three classifiers.

Classifier	Class	TP-rate/recall (%)	FP-rate (%)	Precision (%)	*F*-measure (%)
NFS	Benign	88.1	20.5	85.0	86.5
	Malignant	79.5	11.9	83.5	81.5
	Weighted average	84.4	16.8	84.7	84.4
RBFNN	Benign	78.0	16.9	85.9	81.7
	Malignant	83.1	22.0	74.2	78.4
	Weighted average	80.2	19.1	80.4	80.2
ANFIS	Benign	78.0	13.3	88.5	82.9
	Malignant	86.7	22.0	75.0	80.4
	Weighted average	81.7	17.0	81.9	81.7

other land cover. There are 74 tuples for the 'w' class and 4265 for 'n' class.

After the training phase is over, each of the three classifiers is applied to a test set for classification. Firstly, the performance comparisons of these classifiers are done based on the different performance measures like classification accuracy, RMSE, and the kappa statistic as shown below in Table 10.

From Table 10 we could see that, NFS classifier has an accuracy of 98.9%. RBFNN model has classification accuracy of 94.8%; while ANFIS model has an accuracy of 93.7%. Surely, accuracy wise NFS has performed better than RBFNN and ANFIS. Then we have analyzed the performance of each classifier based on the information on RMSE and the kappa statistic values collected from Table 10. Fig. 13 below shows a performance comparison among these classifiers.

As it is evident from Fig. 13, the kappa statistic value of the NFS method was around 0.81–1.0; which, according to the definition of kappa statistic is considered to be 'almost perfect agreement'. While the kappa statistic value of the RBFNN and ANFIS methods is around 0.61–0.80 and was considered to be 'substantial'. Based on the result, NFS comes out first with an RMSE value of 0.1105 and a kappa statistic value of 0.8787; followed by RBFNN having an RMSE value of 0.2247 and a kappa statistic value of 0.7876 and ANFIS stands last with the highest RMSE value (0.2549) and the lowest kappa statistic value (0.7589). Therefore, with regard to the performance measures such as accuracy, RMSE and kappa statistic, the NFS classifier has performed the best.

Next, the performances of these models are compared based on the TP-Rate (or Recall), FP-Rate, Precision, and *F*-Measure values derived from the confusion matrix of individuals with respect to the test data set. The detailed accuracy by each of the two classes ('w' and 'n') for these classifiers is shown below in Table 11. The weighted average values are also shown in the following table.

We have compared the performance of each classifier based on the information on a weighted average of different performance measures from Table 11; which is then presented in the form of a 3-D column chart as shown below in Fig. 14.

From Table 11 we have discovered that the weighted average values of TP-Rate (or Recall), FP-Rate, Precision, and *F*-Measure for proposed NFS classifier are 98.9%, 11.4%, 99.0%, and 98.9%, respectively; whereas for RBFNN classifier these values are 94.7%, 35.7%, 95.1%, and 94.8% respectively. For ANFIS these values are 93.7%, 49.6%, 94.1%, and 93.7% respectively. Surely, the NFS model has the highest weighted average values for TP-Rate, Precision, Recall, and *F*-Measure and the lowest weighted average value for FP-Rate. Considering *F*-Measure as the best performance measure derived from a confusion matrix; the NFS model has the

Table 10 Performance comparisons of three classifiers.

Classifier	Classification accuracy (%)	RMSE	Kappa statistic
NFS	**98.9**	0.1105	0.8787
RBFNN	**94.8**	0.2247	0.7876
ANFIS	**93.7**	0.2549	0.7589

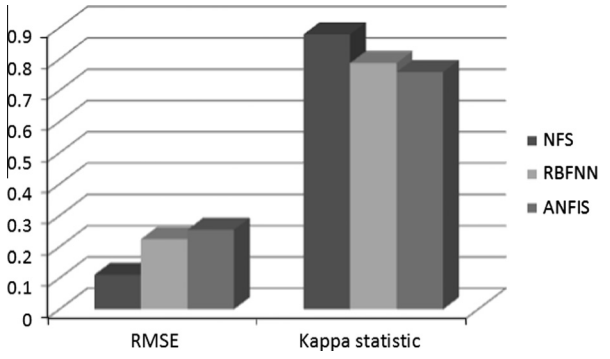

Figure 13 Comparison of RMSE and Kappa statistic values.

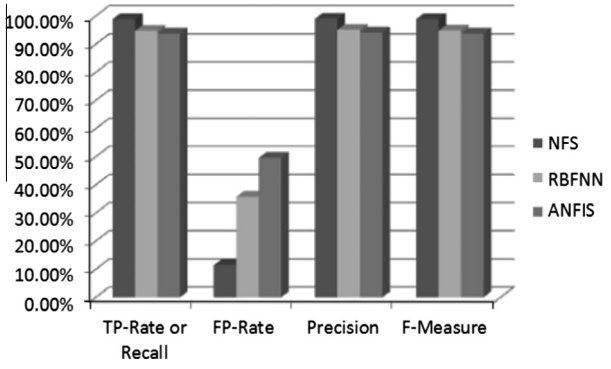

Figure 14 Comparison of three classifiers on weighted average values.

Table 11 Detailed accuracy by each class for three classifiers.

Classifier	Class	TP-rate/recall (%)	FP-rate (%)	Precision (%)	*F*-measure (%)
NFS	*w*	88.4	0.4	94.8	90.5
	n	99.6	11.6	99.2	99.4
	Weighted average	98.9	11.4	99.0	98.9
RBFNN	*w*	60.9	1.5	71.6	65.8
	n	98.5	39.1	97.5	98.0
	Weighted average	94.7	35.7	95.1	94.8
ANFIS	*w*	46.0	1.0	74.1	56.7
	n	99.0	54.0	96.6	97.8
	Weighted average	93.7	49.6	94.1	93.7

highest precision value for the *F*-Measure as 98.9%, followed by the RBFNN model which has an *F*-Measure value of 94.8% and ANFIS with an *F*-Measure value of 93.7%. Fig. 14 also supports this observation.

5.2.6. Mushroom data set

The Mushroom data set consists of hypothetical samples regarding a set of 23 gilled mushroom species in the Agaricus and Lepiota family of mushrooms. This data set is the result of a research study carried out with records gathered from the Audubon Society Field Guide to mushrooms in North America. It has 8124 tuples and consists of 22 attributes including the class attribute. All the attributes are nominally valued. The class attribute has exactly two values namely 'edible' and 'poisonous'. The numbers of tuples per class with percentage values are—edible: 4208 (51.8%) and poisonous: 3916 (48.2%). There are 2480 tuples in this data set that contain a single missing attribute value, for the attribute number 11.

After the training phase is over, each of the three classifiers is applied to a test set for classification. Firstly, the performance comparisons of these classifiers are done based on the different performance measures like classification accuracy, RMSE, and the kappa statistic as shown below in Table 12.

From Table 12 we could see that, NFS classifier has an accuracy of 99.8%. RBFNN model has classification accuracy of 98.2%; while ANFIS model has an accuracy of 95.4%. Surely, accuracy wise NFS has performed better than RBFNN and ANFIS. Then we have analyzed the performance of each classifier based on the information on RMSE and the kappa statistic values collected from Table 12. Fig. 15 below shows a performance comparison among these classifiers.

The experiment has employed very commonly used indicator like the RMSE value which should be as low as possible. Kappa statistic is used to estimate the accuracy for distinguishing between the reliability of the classified data collected and their validity. As it is evident from Fig. 15, the kappa statistic value of the selected algorithms is around 0.81–1.0.

Table 12 Performance comparisons of three classifiers.

Classifier	Classification accuracy (%)	RMSE	Kappa statistic
NFS	**99.8**	0.0792	0.9956
RBFNN	**98.2**	0.1316	0.9631
ANFIS	**95.4**	0.1873	0.9073

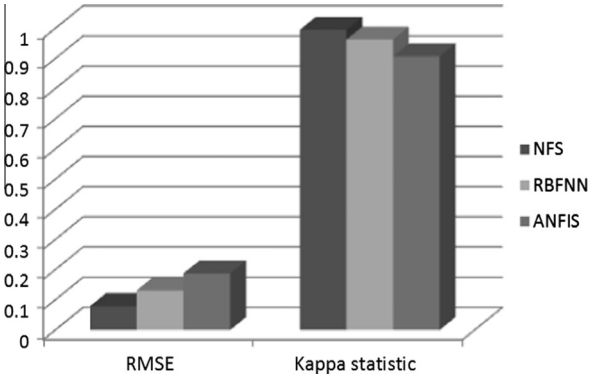

Figure 15 Comparison of RMSE and Kappa statistic values.

According to the definition of kappa statistic, the accuracy of the classification methods is considered to be 'almost perfect agreement'. Based on the result, NFS comes out first with an RMSE value of 0.0792 and a kappa statistic value of 0.9956; followed by RBFNN having an RMSE value of 0.1316 and a kappa statistic value of 0.9631and ANFIS stood last with the highest RMSE value (0.1873) and the lowest kappa statistic value (0.9073). Therefore, with regard to the performance measures such as classification accuracy, RMSE and kappa statistic, the proposed NFS classifier has performed the best.

Next, the performances of these models are compared based on the TP-Rate (or Recall), FP-Rate, Precision, and *F*-Measure values derived from the confusion matrix of individuals with respect to the test data set. The detailed accuracy by each class (edible and poisonous) for these classifiers is shown below in Table 13. The weighted average values are also shown in the following table. The results reported here are entirely based on simulation experiment.

We have compared the performance of each classifier based on the information on a weighted average of different performance measures from Table 11; which is then presented in the form of a 3-D column chart as shown below in Fig. 16.

From Table 13 and Fig. 16, we have discovered that the weighted average values of TP-Rate (or Recall), FP-Rate, Precision, and *F*-Measure for proposed NFS classifier are 99.8%, 1.2%, 99.8%, and 99.8%, respectively; whereas for RBFNN classifier these values are 97.3%, 3.1%, 97.2%, and 97.2% respectively. For ANFIS these values are 95.4%, 4.9%, 95.6%, and 95.4% respectively. Surely, the NFS model has the highest weighted average values for TP-Rate, Precision, Recall, and *F*-Measure and the lowest weighted average value for FP-Rate. Considering *F*-Measure as the best performance measure derived from a confusion matrix; the NFS model has the highest precision value for the *F*-Measure as 99.8%, followed by the ANFIS model which has an *F*-Measure value of 97.2% and RBFNN with an *F*-Measure value of 95.4%.

5.2.7. Pima Indians diabetes data set

This multivariate data set is used for diabetes detection, and is the result of a research survey carried out in the National Institute of Diabetes and Digestive and Kidney Diseases, United States on the female patients of Pima Indian heritage having age greater than 21. It has got 768 tuples and contains 9 numeric-valued attributes including the class. The class attribute has got two values, namely "tested negative for diabetes" and "tested positive for diabetes" and denoted by values '0' and '1' respectively. Many constraints were added for selecting the tuples from a large database. They are namely—

(1) the number of times a woman become pregnant,
(2) the plasma glucose concentration for a 2 h in an oral glucose tolerance test,
(3) the diastolic blood pressure (mmHg),
(4) the triceps skin fold thickness (mm),
(5) the 2-h serum insulin (muU/ml),
(6) the body mass index (weight in kg/[height in meter]^2),
(7) the diabetes pedigree function,
(8) the age in years, and
(9) the class variable.

Table 13 Detailed accuracy by each class for three classifiers.

Classifier	Class	TP-rate/recall (%)	FP-rate (%)	Precision (%)	*F*-measure (%)
NFS	Edible	99.9	2.1	99.9	99.9
	Poisonous	99.8	1.3	99.8	99.8
	Weighted average	99.8	1.2	99.8	99.8
RBFNN	Edible	98.9	2.7	97.5	98.2
	Poisonous	97.3	1.1	98.8	98.1
	Weighted average	97.3	3.1	97.2	97.2
ANFIS	Edible	99.3	8.8	92.4	95.7
	Poisonous	91.2	0.7	99.2	95.0
	Weighted average	95.4	4.9	95.6	95.4

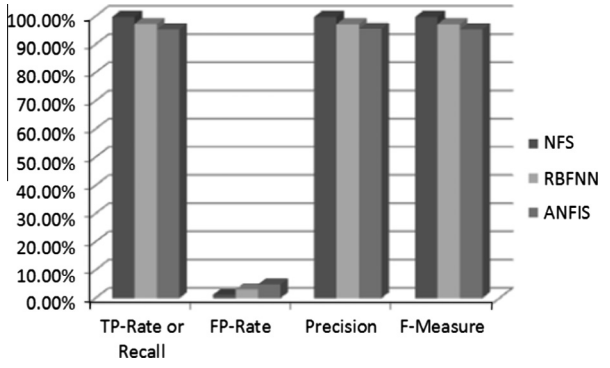

Figure 16 Comparison of three classifiers on weighted average values.

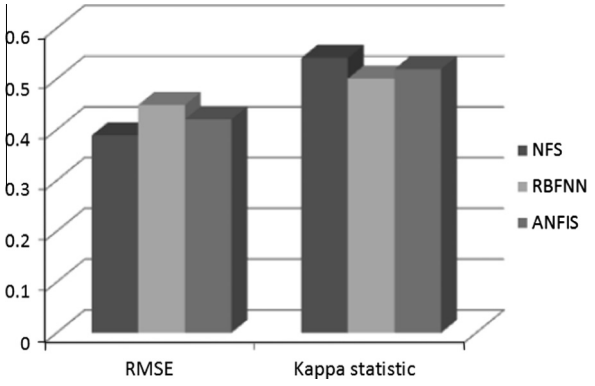

Figure 17 Comparison of RMSE and Kappa statistic values.

After the training phase is over, each of the three classifiers is applied to a test set for classification. Firstly, the performance comparisons of these classifiers are done based on the different performance measures such as classification accuracy, RMSE, and the kappa statistic as shown below in Table 14.

From Table 14 we could see that, NFS classifier has an accuracy of 82.1%. RBFNN model has classification accuracy of 79.7%; while ANFIS model has an accuracy of 80.5%. Surely, accuracy wise NFS has performed better than RBFNN and ANFIS. Then we have analyzed the performance of each classifier based on the information on RMSE and the kappa statistic values collected from Table 14. Fig. 17 below shows a performance comparison among these classifiers.

The experiment has employed very commonly used indicator like the RMSE value which should be as low as possible. Kappa statistic is used to estimate the accuracy for distinguishing between the reliability of the classified data collected and their validity. As it is evident from Fig. 17, the kappa statistic value of the selected algorithms was around 0.41–0.60. According to the definition of kappa statistic, the accuracy of the classification methods is considered to be 'moderate'. Based on the result, NFS comes out first with an RMSE value of 0.3886 and a kappa statistic value of 0.5413; followed by ANFIS having an RMSE value of 0.4205 and a kappa statistic value of 0.5197 and RBFNN stands last with the highest RMSE value (0.4487) and the lowest kappa statistic value (0.5007). Therefore, with regard to the performance measures such as classification accuracy, RMSE and kappa statistic, the proposed NFS classifier has performed the best.

Table 14 Performance comparisons of three classifiers.

Classifier	Classification accuracy (%)	RMSE	Kappa statistic
NFS	**82.1**	0.3886	0.5413
RBFNN	**79.7**	0.4487	0.5007
ANFIS	**80.5**	0.4205	0.5197

Next, the performances of these models are compared based on the TP-Rate (or Recall), FP-Rate, Precision, and *F*-Measure values derived from the confusion matrix of individuals with respect to the test data set. The detailed accuracy by each class ('0' and '1') for these classifiers is shown below in Table 15. The weighted average values are also shown in the following table. The results reported here are entirely based on simulation experiment. For evaluating the performance of a classifier, we would expect higher values for TP-Rate, Precision, Recall, *F*-Measure; and lower value for FP-Rate.

We have compared the performance of each classifier based on the information on a weighted average of different performance measures from Table 11; which is then presented in the form of a 3-D column chart as shown below in Fig. 18.

From Table 15 we have discovered that the weighted average values of TP-Rate (or Recall), FP-Rate, Precision, and *F*-Measure for proposed NFS classifier are 81.5%, 32.4%, 81.7%, and 81.1%, respectively; whereas for RBFNN classifier these values are 79.9%, 33.3%, 79.3%, and 78.9% respectively. For ANFIS these values are 80.5%, 31.9%, 80.0%,

Table 15 Detailed accuracy by each class for three classifiers.

Classifier	Class	TP-rate/recall (%)	FP-rate (%)	Precision (%)	*F*-Measure (%)
NFS	0	94.3	44.9	81.8	87.6
	1	55.1	5.7	81.8	65.9
	Weighted average	81.5	32.4	81.7	81.1
RBFNN	0	91.4	44.9	81.4	86.1
	1	55.1	8.6	75.0	63.5
	Weighted average	79.9	33.3	79.3	78.9
ANFIS	0	91.4	42.9	82.1	86.5
	1	57.1	8.6	75.7	65.1
	Weighted average	80.5	31.9	80.0	79.7

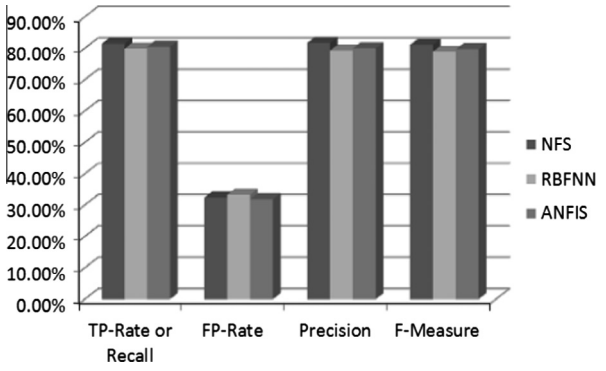

Figure 18 Comparison of three classifiers on weighted average values.

and 79.7% respectively. Surely, the NFS model has the highest weighted average values for TP-Rate, Precision, Recall, and *F*-Measure and the lowest weighted average value for FP-Rate. Considering *F*-Measure as the best performance measure derived from a confusion matrix; the NFS model has the highest precision value for the *F*-Measure as 81.1%, followed by the ANFIS model which has an *F*-Measure value of 79.7% and RBFNN with an *F*-Measure value of 78.9%. Fig. 18 also supports this observation.

5.2.8. Iris data set

This data set is frequently used in the research literature for classification. It has got 150 tuples and 5 attributes including the class. The four input attributes are sepal length (in cm), sepal width (in cm), petal length (in cm), and petal width (in cm) respectively. The class attribute has got three values, namely 'Iris-setosa', 'Iris-versicolor', and 'Iris-virginica'. The data set contains 3 classes of 50 tuples each, where each class refers to a type of Iris plant. Each class is linearly separable from the other two classes.

After the training phase is over, each of the three classifiers is applied to a test set for classification. Firstly, the performance comparisons of these classifiers are done based on the different performance measures like classification accuracy, RMSE, and the kappa statistic as shown below in Table 16.

From Table 16 we could see that, NFS classifier has an accuracy of 96.7%. RBFNN model has classification accuracy of 83.3%; while ANFIS model has an accuracy of 95.5%. Surely, accuracy wise NFS has performed better than RBFNN and ANFIS. Then we have analyzed the performance of each

Table 16 Performance comparisons of three classifiers.

Classifier	Classification accuracy (%)	RMSE	Kappa statistic
NFS	**96.7**	0.1415	0.9499
RBFNN	**83.3**	0.3333	0.7512
ANFIS	**95.5**	0.2897	0.9131

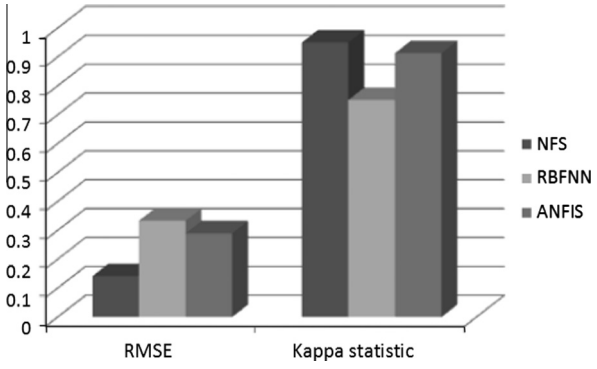

Figure 19 Comparison of RMSE and Kappa statistic values.

classifier based on the information on RMSE and the kappa statistic values collected from Table 16. Fig. 19 below shows a performance comparison among these classifiers.

As it is evident from Fig. 19, the kappa statistic value of the NFS and ANFIS algorithms is around 0.81–1.0. According to the definition of kappa statistic, the accuracy of the classification methods is regarded as 'almost perfect agreement'. While the kappa statistic value of RBFNN is 'substantial' as it lies within the range 0.61–0.80. Based on the result, NFS comes out first with an RMSE value of 0.1415 and a kappa statistic value of 0.9499; followed by ANFIS having an RMSE value of 0.2897 and a kappa statistic value of 0.9131 and RBFNN stood last with the highest RMSE value (0.3333) and the lowest kappa statistic value (0.7512). Therefore, with regard to the performance measures such as classification accuracy, RMSE and kappa statistic, the proposed NFS classifier performed the best.

Next, the performances of these models are compared based on the TP-Rate (or Recall), FP-Rate, Precision, and *F*-Measure values derived from the confusion matrix of individuals with respect to the test set. The detailed accuracy by each class ('Iris-setosa', 'Iris-versicolor', and 'Iris-virginica') for these classifiers is shown below in Table 17. The weighted

Table 17 Detailed accuracy by each class for three classifiers.

Classifier	Class	TP-rate/recall (%)	FP-rate (%)	Precision (%)	*F*-measure (%)
NFS	Iris-setosa	99.9	0.7	98.9	99.9
	Iris-versicolor	99.8	1.5	90.9	95.2
	Iris-virginica	88.9	2.9	89.9	94.1
	Weighted average	96.7	2.3	97.0	96.6
RBFNN	Iris-setosa	99.9	0.7	99.9	99.9
	Iris-versicolor	52.9	0.5	99.9	66.7
	Iris-virginica	99.9	23.8	64.3	78.3
	Weighted average	83.3	7.1	89.3	82.4
ANFIS	Iris-setosa	99.9	0.7	99.9	99.9
	Iris-versicolor	99.8	6.9	88.9	94.1
	Iris-virginica	86.7	0.5	99.9	92.9
	Weighted average	95.6	3.7	96.0	95.5

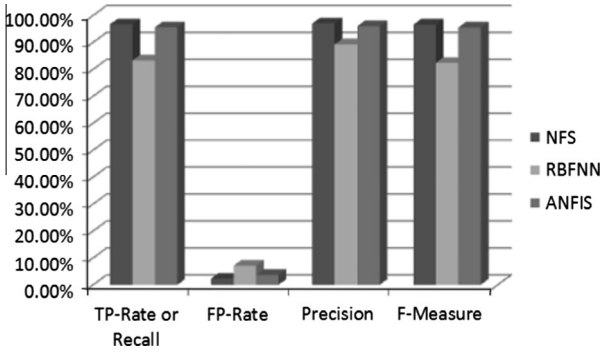

Figure 20 Comparison of three classifiers on weighted average values.

average values are also shown in the following table. The results reported here are solely based on simulation experiment. For evaluating the performance of a classifier, we would expect higher values for TP-Rate, Precision, Recall, *F*-Measure; and lower value for FP-Rate.

We have compared the performance of each classifier based on the information on a weighted average of different performance measures from Table 17; which is then presented in the form of a 3-D column chart as shown below in Fig. 20.

From Table 17 we have discovered that the weighted average values of TP-Rate (or Recall), FP-Rate, Precision, and *F*-Measure for proposed NFS classifier are 96.7%, 2.3%, 97.0%, and 96.6%, respectively; whereas for RBFNN classifier these values are 83.3%, 7.1%, 89.3%, and 82.4% respectively. For ANFIS these values are 95.6%, 3.7%, 96.0%, and 95.5% respectively. Surely, the NFS model has the highest weighted average values for TP-Rate, Precision, Recall, and *F*-Measure and the lowest weighted average value for FP-Rate. Considering *F*-Measure as the best performance measure derived from a confusion matrix; the NFS model has the highest precision value for the *F*-Measure as 96.6%, followed by the ANFIS model which has an *F*-Measure value of 95.5% and RBFNN with an *F*-Measure value of 82.4%. Fig. 20 also supports this observation.

5.2.9. Spambase data set

This is a very popular data set in the classification domain. The data set is used for classifying electronic mail (email) as spam or non-spam. It has got 4601 tuples and consists of 58 attributes including the class attribute. The class attribute has got two values, namely *non-spam*, and *spam* denoted by values '0' and '1' respectively.

Table 18 Performance comparisons of three classifiers.

Classifier	Classification accuracy (%)	RMSE	Kappa statistic
NFS	**92.3**	0.1536	0.8412
RBFNN	**87.6**	0.3191	0.7423
ANFIS	**89.5**	0.3047	0.7767

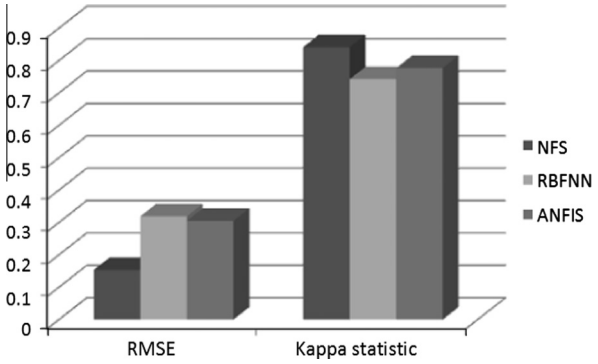

Figure 21 Comparison of RMSE and Kappa statistic values.

After the training phase is over, each of the three classifiers is applied to a test set for classification. Firstly, the performance comparisons of these classifiers are done based on the different performance measures such as classification accuracy, RMSE, and the kappa statistic as shown below in Table 18.

From Table 18 we could see that, NFS classifier has an accuracy of 92.3%. RBFNN model has classification accuracy of 87.6%; while ANFIS model has an accuracy of 89.5%. Surely, accuracy wise NFS has performed better than RBFNN and ANFIS. Then we have analyzed the performance of each classifier based on the information on RMSE and the kappa statistic values collected from Table 18. Fig. 21 below shows a performance comparison among these classifiers.

The experiment has employed very commonly used indicator like the RMSE value which should be as low as possible. Kappa statistic is used to estimate the accuracy for

Table 19 Detailed accuracy by each class for three classifiers.

Classifier	Class	TP-rate/recall (%)	FP-rate (%)	Precision (%)	*F*-measure (%)
NFS	0	93.6	9.6	93.4	93.5
	1	90.4	6.4	90.6	90.5
	Weighted average	92.3	8.3	92.3	92.3
RBFNN	0	90.3	16.3	89.0	89.6
	1	83.7	9.7	85.5	84.6
	Weighted average	87.6	13.6	87.6	87.6
ANFIS	0	95.8	17.8	87.6	91.5
	1	80.2	4.2	92.9	86.1
	Weighted average	89.5	12.5	89.7	89.3

distinguishing between the reliability of the classified data collected and their validity. As it is evident from Fig. 21, the kappa statistic value of the proposed NFS method is around 0.81–1.0. According to the definition of kappa statistic, the accuracy of this classification method is considered to be 'almost perfect agreement'. While the kappa statistic values of RBFNN and ANFIS algorithms are 'substantial' as they lie within the range 0.61–0.80. Based on the result, NFS comes out first with an RMSE value of 0.1536 and a kappa statistic value of 0.8412; followed by ANFIS having an RMSE value of 0.3047 and a kappa statistic value of 0.7767 and RBFNN stands last with the highest RMSE value (0.3191) and the lowest kappa statistic value (0.7423). Therefore, with regard to the performance measures like classification accuracy, RMSE and kappa statistic, the proposed NFS classifier has performed the best.

Next, the performances of these models are compared based on the TP-Rate (or Recall), FP-Rate, Precision, and *F*-Measure values derived from the confusion matrix of individuals with respect to the test data set. The detailed accuracy by each class ('0' and '1') for these classifiers is shown below in Table 19. The weighted average values are also shown in the following table. The results reported here are entirely based on simulation experiment.

We have compared the performance of each classifier based on the information on a weighted average of different performance measures from Table 19; which is then presented in the form of a 3-D column chart as shown below in Fig. 22.

From Table 19 and Fig. 22, we have discovered that the weighted average values of TP-Rate (or Recall), FP-Rate, Precision, and *F*-Measure for the NFS classifier are 92.3%, 8.3%, 92.3%, and 92.3%, respectively; whereas for RBFNN the values are 87.6%, 13.6%, 87.6%, and 87.6% respectively. For ANFIS these values are 89.5%, 12.5%, 89.7%, and 89.3% respectively. Surely, the NFS model has the highest weighted average values for TP-Rate, Precision, Recall, and *F*-Measure and the lowest weighted average value for FP-Rate. The NFS model has the highest precision value for the *F*-Measure as 92.3%, followed by the ANFIS model which has an *F*-Measure value of 89.3% and RBFNN with an *F*-Measure value of 87.6%.

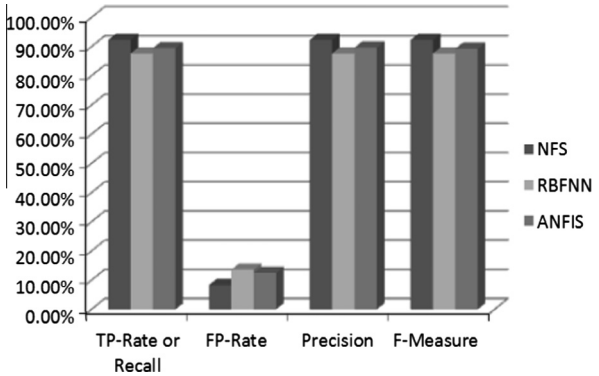

Figure 22 Comparison of three classifiers on weighted average values.

5.2.10. Car Evaluation data set

The Car Evaluation data set is constructed from a simple hierarchical decision model originally developed for demonstration purpose. The model actually comprises three intermediate concepts: Price, Technology, and Comfort. The database has got 1728 tuples and consists of 7 attributes including the class attribute. It contains tuples which directly relate the cars to six input attributes—buying, maint, doors, persons, lug_boot, and safety respectively. The class attribute has got four values, namely 'unacceptable', 'acceptable', 'good', and 'very good'.

After the training phase is over, each of the three classifiers is applied to a test set for classification. Firstly, the performance comparisons of these classifiers are done based on the different performance measures like classification accuracy, RMSE, and the kappa statistic as shown below in Table 20.

From Table 20 we could see that, NFS classifier has an accuracy of 91.2%. RBFNN model has classification accuracy of 89.8%; while ANFIS model has an accuracy of 86.1%. Surely, accuracy wise NFS has performed better than RBFNN and ANFIS. Then we have analyzed the performance of each classifier based on the information on RMSE and the kappa statistic values collected from Table 20. Fig. 23 below shows a performance comparison among these classifiers.

As it is evident from Fig. 23, the kappa statistic value of the selected algorithms is around 0.61–0.80. According to the

Table 20 Performance comparisons of three classifiers.

Classifier	Classification accuracy (%)	RMSE	Kappa statistic
NFS	**91.2**	0.1778	0.7887
RBFNN	**89.8**	0.1882	0.7695
ANFIS	**86.1**	0.2032	0.7053

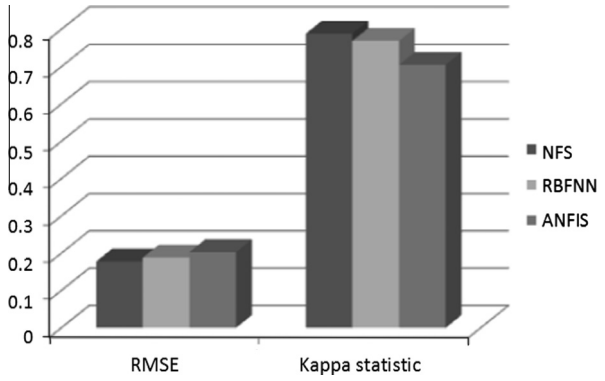

Figure 23 Comparison of RMSE and Kappa statistic values.

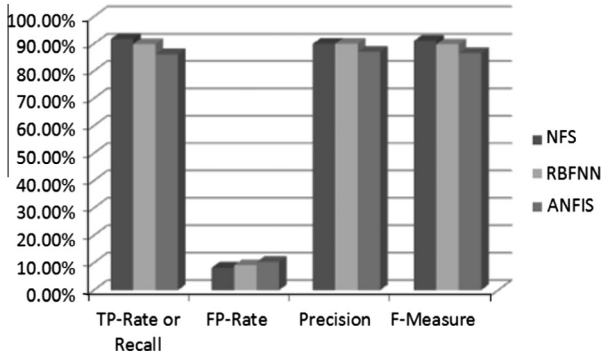

Figure 24 Comparison of three classifiers on weighted average values.

definition of kappa statistic, the accuracy of the classification methods is considered to be 'substantial'. Based on the result, NFS comes out first with an RMSE value of 0.1778 and a kappa statistic value of 0.7887; followed by RBFNN having an RMSE value of 0.1882 and a kappa statistic value of 0.7695 and ANFIS stands last with the highest RMSE value (0.2032) and the lowest kappa statistic value (0.7053). Therefore, with regard to the performance measures like classification accuracy, RMSE and kappa statistic, the proposed NFS classifier has performed the best.

Next, the performances of these models are compared based on the TP-Rate (or Recall), FP-Rate, Precision, and *F*-Measure values derived from the confusion matrix of individuals with respect to the test data set. The detailed accuracy by each of the 4 classes ('unacceptable', 'acceptable', 'good', and 'very good') for these classifiers is shown below in Table 21. The weighted average values are also shown in the following table.

We have compared the performance of each classifier based on the information on a weighted average of different performance measures from Table 21; which is then presented in the form of a 3-D column chart as shown below in Fig. 24.

From Table 21 we have discovered that the weighted average values of TP-Rate (or Recall), FP-Rate, Precision, and *F*-Measure for proposed NFS classifier are 91.8%, 8.2%, 90.0%, and 91.1%, respectively; whereas for RBFNN classifier these values are 89.9%, 9.1%, 90.0%, and 89.8% respectively. For ANFIS these values are 86.1%, 10.3%, 87.0%, and 86.5% respectively. Surely, the NFS model has the highest weighted average values for TP-Rate, Precision, Recall, and *F*-Measure and the lowest weighted average value for FP-Rate. Considering *F*-Measure as the best performance measure derived from a confusion matrix; the NFS model has the highest precision value for the *F*-Measure as 91.1%, followed by the RBFNN model which has an *F*-Measure value of 89.8% and ANFIS with an *F*-Measure value of 86.5%. Fig. 24 also supports this observation.

With regard to the different performance measures used for the ten UCI data sets; we have got better results on average for NFS compared to RBFNN and ANFIS. An algorithm having lower error rates will be considered effective as because it has the more powerful classification capability and predictive ability in the data mining field. Considering this, our proposed NFS based method has performed better than the RBFNN and ANFIS algorithms.

6. Conclusion

As a conclusion, we have taken on our objective which is to investigate and compare the proposed NFS method with

Table 21 Detailed accuracy by each class for three classifiers.

Classifier	Class	TP-rate/recall (%)	FP-rate (%)	Precision (%)	*F*-measure (%)
NFS	Unacceptable	99.9	9.3	95.5	97.2
	Acceptable	77.2	2.7	86.9	82.3
	Good	41.7	0.9	63.5	48.5
	Very good	58.3	1.5	59.3	58.3
	Weighted average	91.8	8.2	90.0	91.1
RBFNN	Unacceptable	94.6	10.5	95.4	95.0
	Acceptable	86.1	7.5	77.3	81.4
	Good	42.9	1.2	60.0	50.0
	Very good	75.0	0.1	99.9	85.7
	Weighted average	89.9	9.1	90.0	89.8
ANFIS	Unacceptable	92.1	12.4	94.5	93.3
	Acceptable	79.7	6.4	78.8	79.2
	Good	50.0	2.1	50.0	50.0
	Very good	50.0	3.3	35.3	41.4
	Weighted average	86.1	10.3	87.0	86.5

RBFNN and ANFIS classifiers based on different performance measures like accuracy, RMSE, kappa statistic, TP-Rate, FP-Rate, Precision, Recall, and *F*-Measure. These classifiers are tested with ten benchmark UCI data sets. The results suggest that among the three classifiers studied and analyzed, the proposed NFS classifier has the potential to significantly improve the conventional classification methods for use in Data Mining research field. These classification methods, described here are powerful and effective. The results reported in this study are correct, appropriate; and completely based on simulation experiment. Nevertheless, a more widespread experimental assessment of the proposed technique will be the goal of our future research. Furthermore, the integration of other interestingness measures mentioned in the literature is also part of our intended future work.

Acknowledgments

We wish to thank the UCI Machine Learning Repository for sharing the data sets. The simulation experiment for this research study is performed using the MATLAB software (version R2013a).

References

[1] Agrawal R, Imielinski T, Swami A. Database mining: a performance perspective. IEEE Trans Knowl Data Eng 1993;4(6):914–25.
[2] Chen MS, Han J, Yu PS. Data mining: an overview from a database perspective. IEEE Trans Knowl Data Eng 1996;8(6):866–83.
[3] Han J, Kamber M. Data mining: concepts and techniques. 2nd ed. Morgan and Kaufmann; 2005. p. 285–378.
[4] Pujari AK. Data mining techniques. 1st ed. Universities Press (India) Private Limited; 2001. p. 202–25.
[5] Rojas R. Neural networks a systematic introduction. Berlin: Springer-Verlag; 1996. p. 29–151.
[6] Haykin S. Neural networks: a comprehensive foundation. 2nd ed. Prentice Hall; 1998. p. 253–77.
[7] Bose NK, Liang P. Neural network fundamentals with graphs, algorithms, and applications. McGraw-Hill; 1996. p. 119–209.
[8] Werbos PJ. The roots of backpropagation. From ordered derivatives to neural networks and political forecasting. New York: John Wiley & Sons; 1994. p. 67–163.
[9] Rumelhart DE, Hinton GE, Williams RJ. Learning representations by back-propagating errors. Nature 1986;323(6088):533–6.
[10] Broomhead DS, Lowe David. Radial basis functions, multivariable functional interpolation and adaptive networks. Royal Signals and Radar Establishment, Technical report, no. 4148; 1988. p. 1–34.
[11] Hunt KJ, Haas R, Murray-Smith R. Extending the functional equivalence of radial basis function networks and fuzzy inference systems. IEEE Trans Neural Networks 1996;7(3):776–81.
[12] Zadeh LA. Fuzzy sets. Inf Control 1965;8(3):338–53.
[13] Liu B. Uncertainty theory: an introduction to its axiomatic foundations. Berlin: Springer Verlag; 2004. p. 191–346.
[14] Dubois D, Prade H. Fuzzy sets and systems: theory and applications. New York: Academic Press; 1980. p. 255–348.
[15] Jang J-SR, Sun C-T, Mizutani E. Neuro-fuzzy and soft computing: a computational approach to learning and machine intelligence. USA: Prentice Hall; 1997, pp. 333–393.
[16] Lin C-T, Lee CSG. Neural fuzzy systems: a Neuro-fuzzy synergism to intelligent systems. Prentice Hall; 1996. p. 313–459.
[17] Keller JK, Hunt DJ. Incorporating fuzzy membership functions into the perceptron algorithm. IEEE Trans Pattern Anal Mach Intell 1985;7(6):693–9.
[18] Ghosh A, Pal NR, Pal SK. Self-organization for object extraction using a multilayer neural network and fuzziness measures. IEEE Transact Fuzzy Syst 1993;1(1):5469.
[19] Pal SK, Ghosh A. Review Neuro-fuzzy computing for image processing and pattern recognition. Int J Syst Sci 1996;27(12):1179–93.
[20] Ghosh A, Shankar BU, Meher SK. A novel approach to Neuro-fuzzy classification. Neur Network 2009;22(1):100–9.
[21] Kuncheva LI. Fuzzy classifier design. Springer-Verlag, vol. 49; 2000. p. 233–63.
[22] Pedrycz W. Fuzzy sets in pattern recognition: methodology and methods. Pattern Recogn 1990;23(1):121–46.
[23] Castellano G, Castiello C, Fanelli AM, Jain L. Evolutionary neuro-fuzzy systems and applications. Advances in evolutionary computing for system design, studies in computational intelligence. Springer-Verlag, vol. 66; 2007. p. 11–45.
[24] Abonyi J, Fei B, Abraham A. Computational intelligence in data mining. Informatica, Slovenia 2005;29(1):3–12.
[25] Keleş A, Keleş A. Extracting fuzzy rules for diagnosis of breast cancer. Turkish J Electr Eng Comput Sci 2013;21(1):1495–503.
[26] Nauck D, Klawonn F, Kruse R. Foundations of Neuro fuzzy systems. Chichester: Wiley; 1997. p. 33–171.
[27] Abraham A. Neuro fuzzy systems: state-of-the-art modeling techniques. In: Mira Jose, et al. (Eds.), Connectionist models of neurons, learning processes, and artificial intelligence. Springer-Verlag: Germany, LNCS 2084; 2001. p. 269–76.
[28] Ghosh A, Shankar BU, Bruzzone L, Meher SK. Neuro-fuzzy-combiner: an effective multiple classifier system. Int J Knowl Eng Soft Data Parad 2010;2(2):107–29, Inderscience Publishers.
[29] Elisseeff A, Paugam-Moisy H. Size of multilayer networks for exact learning: analytic approach. Advances in Neural Information Processing Systems, vol. 9. USA: MIT Press; 1997. p. 162–68.
[30] Armstrong JS, Collopy F. Error measures for generalizing about forecasting methods: empirical comparisons. Int J Forecast 1992;8:69–80.
[31] Carletta J. Assessing agreement on classification tasks: the kappa statistic. Comput Linguist 1996;22(2):249–54, Cambridge, MA, USA: MIT Press.
[32] Stehman SV. Selecting and interpreting measures of thematic classification accuracy. Remote Sens Environ 1997;62(1):77–89.