Constructing Accurate Fuzzy Rule-based Classification Systems Using Apriori Principles and Rule-weighting

S.M. Fakhrahmad[1], A. Zare[2], M. Zolghadri Jahromi[3]

[1] Member of Young Researchers Club - Shiraz branch,
Faculty member of Islamic Azad University of Shiraz, Shiraz, Iran,
[2, 3] Department of Computer Science & Engineering, Shiraz University, Shiraz, Iran
{mfakhrahmad, aminzare}@cse.shirazu.ac.ir, zjahromi@shirazu.ac.ir

Abstract. A fuzzy rule-based classification system (FRBCS) is one of the most popular approaches used in pattern classification problems. One advantage of a fuzzy rule-based system is its interpretability. However, we're faced with some challenges when generating the rule-base. In high dimensional problems, we can not generate every possible rule with respect to all antecedent combinations. In this paper, by making the use of some data mining concepts, we propose a method for rule generation, which can result in a rule-base containing rules of different lengths. As the next phase, we use rule-weight as a simple mechanism to tune the classifier and propose a new method of rule-weight specification for this purpose. Through computer simulations on some data sets from UCI repository, we show that the proposed scheme achieves better prediction accuracy compared with other fuzzy and non-fuzzy rule-based classification systems proposed in the past.

Keywords: Pattern classification, fuzzy systems, data mining, rule weighting

1 Introduction

Fuzzy rule-based systems have been widely used on control problems [1,2,3]. One key feature of fuzzy rule-based systems is their comprehensibility because each fuzzy rule is linguistically interpretable. Recently, fuzzy rule-based systems have been applied successfully on classification problems [4,5,6]. The interest in using fuzzy rule-based classification systems (FRBCS) arises from the fact that those systems consider both accuracy and comprehensibility of the classification result at the same time. In fact, error minimization and comprehensibility maximization are two conflicting objectives of these kinds of classification systems and the trade off between these two objectives has been discussed in some recent studies [7,8,9,10].

Basic idea for designing a FRBCS is to automatically generate fuzzy rules from numeric data (i.e., a number of pre-labeled training examples). Hence, rule-base construction for a classification problem always has been a challenging part of it. In this paper, a novel approach for generating a set of candidate rules of each class is presented using data mining principles in which the number of generated rules is reduced dramatically. A compact rule-base is then constructed by selecting a specified number of candidate rules from each class (using a selection metric).

In many studies, antecedent fuzzy sets were generated and tuned by numerical input data for rule-base construction to improve the classification accuracy of FRBCSs. As shown in [11,12], the modification of the membership functions of antecedent fuzzy sets can be replaced by rule weight specification to some extent. Since, the adjustment of membership functions may degrade the interpretability of a FRBCS. In this paper, a learning algorithm is proposed to adjust the weights of the rules (existing in the rule-base) by the training data. This method attends to improve the generalization of FRBCS by minimizing the classification error rate on the training data.

The rest of this paper is organized as follows. In Section 2, a FRBCS is briefly introduced. In Section 3, the process of rule-base construction and the proposed method of generating rules with different lengths are described. Section 4 is devoted to introduction of the proposed method of rule weight learning. In Section 5, the experimental results over artificial and real-life data are shown. Finally, Section 6 concludes the paper.

2 Fuzzy rule-based classification systems

Various methods have been introduced for fuzzy classification. Let us assume that we have m training patterns $x_p = (x_{p1},\ldots,x_{pn})$, $p = 1,2,\ldots,m$ from M different classes where x_p is an n-dimensional vector of attributes in which x_{pi} is the i-th attribute value of the p-th training pattern ($i = 1,2,\ldots,n$). For our M-class, n-dimensional classification problem, we use fuzzy if-then rules of the form below:

$$\text{Rule } R_q\text{: If } x_1 \text{ is } A_{q1} \text{ and } \ldots \text{ and } x_n \text{ is } A_{qn} \text{ then class } C_q \text{ with } CF_q \tag{1}$$

, where R_q is the label of the q-th fuzzy if-then rule, $\mathbf{x} = (x_1,\ldots,x_n)$ is n-dimensional vector of a pattern, A_{qi} presents an antecedent fuzzy set, C_q is a class label, and CF_q is the weight assigned to the q-th rule. In [13], fuzzy rules of other types are introduced. To calculate the compatibility grade of each training pattern $\mathbf{x}_p$ with the antecedent part of the rule $\mathbf{A}_q = (A_{q1},\ldots,A_{qn})$, we use the product operator as follows:

$$\mu_{A_q}(\boldsymbol{x}_p) = \mu_{A_{q1}}(x_{p1}) \cdot \mu_{A_{q2}}(x_{p2}) \cdot \mathrm{K} \cdot \mu_{A_{qn}}(x_{pn}), p = 1,2,\mathrm{K}\ ,m \tag{2}$$

, where $\mu_{Aqi}(x_{pi})$ is the compatibility grade of x_{pi} with fuzzy membership function A_{qi}. To determine the consequent class of the q-th rule C_q, we measure the confidence degree of the association rule "$\mathbf{A}_\mathrm{q} \Rightarrow$ Class h" from the field of data mining for each class, where $\mathbf{A}_\mathrm{q}$ is a multi-dimensional fuzzy set representing the antecedent conditions and h is a class label. Confidence of a fuzzy association rule R_q is defined as follows:

$$c(A_q \Rightarrow \text{Class h}) = \sum_{X_p \in \text{Class h}} \mu_{A_q}(\boldsymbol{x}_p) \Big/ \sum_{p=1}^{m} \mu_{A_q}(\boldsymbol{x}_p), h = 1,2,\mathrm{K}\ ,M \tag{3}$$

, where $\mu_{Aq}(X_p)$ is the compatibility grade of pattern X_p with the antecedent part of the rule R_q, m is the number of training patterns and C_q is a class label. The class with maximum confidence degree is identified to determine the consequent class C_q:

$$q = \text{argmax}\{c(A_q \Rightarrow \text{Class h}) \mid h = 1, 2, \ldots, M\} \tag{4}$$

An input pattern is classified regarding to the consequent class of the winner rule. By using rules of the form (1), a weight assigned to each rule is used to find the winner rule. Rule weighting has a profound effect on the classification ability of FRBCSs. In [14], several methods of rule weighting have been introduced. In this paper, we use a learning mechanism to find the weight of each rule.

3 Rule-base construction

For an M-class problem in an n-dimensional feature space, assume that m labeled patterns $X_p=[x_{p1}, x_{p2}, \ldots, x_{pn}]$, $p=1, 2, \ldots, m$ from M classes are given. A simple approach for generating fuzzy rules is to partition the domain interval of each input attribute using a pre-specified number of fuzzy sets (i.e., grid partitioning). Some examples of this partitioning (using triangular membership functions) are shown in Fig. 1.

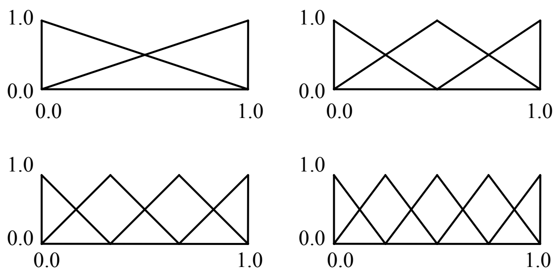


Fig. 1. Different partitioning of each feature axis.

Given a partitioning of pattern space, one approach is to consider every possible combination of antecedents to generate the fuzzy rules. The problem with grid partitioning is that an appropriate partitioning of each attribute is not usually known. One solution is to simultaneously consider different partitions, as shown in Fig. 1. That is, for each attribute, one of the 14 fuzzy sets shown in Fig. 1 can be used when generating a fuzzy rule. The problem is that for an n-dimensional problem, 14^n antecedent combinations have to be considered. It is impractical to consider such a huge number of antecedent combinations when dealing with high dimensional problems.

One solution for the above problem has already been presented by adding the fuzzy set “don’t care” to each attribute. The membership function of this fuzzy set is defined as $\mu_{don't\ care}(x) = 1$ for all values of x. The trick is not to consider all antecedent combinations (which is now 15^n) and only short fuzzy rules having a limited number of antecedent conditions are generated as candidate rules. For example, fuzzy rules having only two or less antecedent fuzzy sets (excluding don't care) are investigated.

It seems that ignoring majority of possible antecedent combinations for rule generation would degrade the accuracy of the FRBCS. On the other hand, increasing the limitation of two or less antecedent fuzzy sets may be impossible in some cases. As an instance, for a high-dimensional data set such as Sonar (available in UCI-ML repository) which is a 2-class data set, containing 60 attributes, number of all possible fuzzy rules of length 4, considering 15 fuzzy membership functions for each attribute is $(60 \times 15)^4 \times 2$ which is more than 10^{12}. Since measuring confidence and support values for each rule involves scanning all training data patterns, construction of a rule-base containing such rules seems to be very difficult or even impossible. That's why the existing classifiers are able to generate rules having at most three antecedent conditions (excluding don't care) in such cases. This limitation prevents from some useful rules, which would have positive effects on the classification accuracy, to be present in the rule-base. As mentioned in the last paragraph, by increasing the number of antecedents, the rule set grows exponentially. Moreover, within a very large number of rules, usually a small fraction of rules are acceptable. Thus, in many cases, a considerable time is devoted to useless computations.

The purpose of the solution presented in this paper, is to avoid the exponential growth of the rule sets in each step. In this approach, we do not generate rules that are hardly probable to be interesting. The method is based over two data mining principles, used for mining frequent item sets:

1) Increasing the length of an item set, the support value will not improve.
2) A set of n items is probable to be frequent (have a good support), if and only if all of its subsets of size n-1 are frequent (the Apriori principle).

The common usage of the above principles is in data warehouses, to find itemsets with good supports (i.e., set of items that have frequently occurred together). In this work, we observe them from a different viewpoint and use them to find fuzzy rules having good supports.

3.1 Generating rules with 1 or 2 antecedents

As mentioned before, A major purpose in this paper is to propose a solution that enables us to generate fuzzy rules with any number of antecedents, i.e., There would be no restriction on the number of antecedents especially for high dimensional data sets (the problem which originates from the exponential growth of rule-base by increasing the number of features). For this purpose, we consider the well-known evaluation measure, *Support* as the primary factor for rule filtering. In equation (5), a simple definition for the fuzzy aspect of the *Support* measure is presented.

$$s(A_j \Rightarrow \text{Class } h) = \frac{1}{m} \sum_{x_p \in \text{Class } h} \mu_{A_j}(x_p) \tag{5}$$

, where $\mu_j(X_p)$ is the compatibility degree of X_p with the antecedent part of the rule R_j, m is the number of training patterns and h is a class label. After determining a minimum support threshold (denoted by *MinSupp*), a set of 1-dimensional rules (containing one antecedent), is generated. This set is then filtered by selecting only rules having a support value above the *MinSupp*. Combining the rules within this set in the next step, results in the set of 2-dimensional candidate rules. The reason of this issue (that we just combine rules having good supports) refers to the first principle mentioned in Section 2. In other words, the 1-dimensional rules which are pruned through the first step because of their bad supports, can not absolutely lead to 2-dimensional rules with good supports and thus there is no need to consider them. Another key point in combination of a pair of 1-dimensional rules is the conditions under which the rules can be combined:

1) The rules must not contain similar antecedents on their left-hand sides.

2) The consequent classes of the two rules must be identical. Similarly, the resulting rule set is filtered with respect to the *MinSupp* value. However, note that the rules being selected according to their higher support values are just candidate rules and may be rejected in the next step. The rule selection metric will be discussed later.

3.2 Generating rules of higher dimensions

In order to generate rules containing more than two antecedents, a similar procedure is followed. However, in this case, both of the principles (used for mining frequent item sets) must be regarded. Generating 3-dimensional rules is accomplished using the 1 and 2-dimensional candidate rules. Any possible combination of the rules from these two sets, having the same consequent and not containing common antecedents would be a 3-dimensional candidate rule. The second principle is used to avoid the time-consuming evaluation of some useless rules (which can not have high support values). A rule resulting from a combination will be evaluated only if all of its 2-dimensional sub-rules[1] are present in the candidate set of the previous stage (i.e., all the sub-rules have good supports). Otherwise, we do not measure the support of the rule, since it can not be even a candidate rule. As an example, the support of the rule R: If X1 is A1 and X2 is A2 and X3 is A3 → C1 is computed only if all the following sub-rules have good supports:

If X1 is A1 and X2 is A2 → C1
If X1 is A1 and X3 is A3 → C1
If X2 is A2 and X3 is A3 → C1

Similarly, generating an n-dimensional rule is performed by the combination of n-1 and 1-dimensional candidate rules.

An important challenge here is to find a solution for this problem: How should we control the presence of all sub-rules in order to avoid efficiency reduction. Moreover, combining 1 and n-dimensional rules (even with respect to the mentioned conditions) may lead to some repeating combinations. Another challenging problem is how to avoid generating repeating rules, which could be so helpful to the efficiency of the

[1] Rule R1 is called to be a sub-rule of R2 iff R1 has less antecedents than R2 and every antecedent of R1 is present in R2, too.

process. To achieve these two goals, we make use of the efficiency of SQL and accomplish the primary phases of the rule generation process using this language.

Following the above process, it will also be possible to generate rules having 4 and more antecedents, for any data set having arbitrary number of features.

Although the set of rules is pruned to some extent, in some cases the number of rules is still large. This problem gets more sensible as we increase the number of antecedents. In order to obtain a more concise data set, we divide the set of candidate rules into *M* distinct groups, according to their consequents (*M* is the number of classes). The rules in each group are sorted by an appropriate evaluation factor and the final rule-base is constructed by selecting *p* rules from each class, i.e., in total, *M.p* rules are selected. Many evaluation measures have already been proposed. In this work, we use the measure proposed in [15] as the rule selection metric, which evaluates the rule $A_j \Rightarrow class\ C_j$ through the following equation:

$$e(R_j) = \sum_{\mathrm{X_p} \in \mathrm{Class}\, C_j} \mu_{\mathrm{A_q}}(\boldsymbol{x}_\mathrm{p}) - \sum_{\mathrm{X_p} \notin \mathrm{Class}\, C_j} \mu_{\mathrm{A_q}}(\boldsymbol{x}_\mathrm{p}) \qquad \textbf{(7)}$$

4 The proposed method for rule weighting

Initially, all rules are assumed to have a weight of one (i.e. CF_k=1, K=1,2,...,N). In this section we propose an algorithm to assign some real numbers (in the interval [1,∞)) as the rule weights using the training patterns. The rule-weighting process for a typical rule, R_i, can be organized into the following steps:

1. Specify the center of the rule's covering area, sort the training patterns within this area in ascending order of their distance from the center (The first pattern in the sorted list is the most compatible one with the rule).
2. Scan the patterns through the sorted list until a pattern (X_n) from the negative class (any class except the rule's target class) is met (The enemy pattern with maximum compatibility degree is found).
3. Call the pattern just before the enemy pattern in the list X^* and Find its compatibility with the rule R_i ($\mu_i(X^*)$).
4. Compute the rule's weight (CF_i) using the following equation:

$$CF_i = 1/\, \mu_i(X^*) \qquad \textbf{(8)}$$

This algorithm obtains a real number in the interval [1,∞) as the weight of each rule. However, in this issue, two exceptional cases may occur:

1. The first pattern in the sorted list is an enemy pattern. For this case, we set the value of 1 to $\mu_i(X^*)$ and thus the rule's weight will not change from 1.

2. There is no enemy pattern in the covering area of the rule (i.e., an interesting case). For this case, we chose the compatibility degree of the last pattern in the sorted list

for $\mu_i(X^*)$, i.e., $\mu_i(X^*) = \mu_i$(last pattern). Since the last pattern has the minimum compatibility with the rule, a higher weight is given to such rules.

In this method, no rule is given a weight of 0. Thus, the number of rules does not change through this weighting process. As the number of partitions of each feature increases, the performance of the system approaches the performance of weighted K-NN method, while having the extra advantage of interpretability, especially for low-dimensional data sets.

4.1 Finding the optimal decision boundaries

The method proposed in Section 3, increases the classification accuracy of each fuzzy rule over training data up to 100%. This is accomplished by tuning the boundaries for the decision area of each rule through assigning a weight to it. It can be predicted that the generalization ability of the classifier will be improved, too. However, there is really no reason that we will get the optimal results for generalization accuracy, through this issue. The main reason refers to some probable noisy or exceptional patterns or in case of data sets with highly ovelapped classes. Our proposed method can easily become more flexible (for noisy data) by making a small change to it. After determining a threshold for the rule accuracy, we do not stop the scanning of the sorted list when meeting the first enemy pattern. Instead, we extend the decision boundary of the rule until we reach an enemy pattern that makes the rule's accuracy become less than the specified threshold. In other words we let a few number of enemy patterns to exist in the decision area of each rule. This can be more effective for noisy-nature data sets.

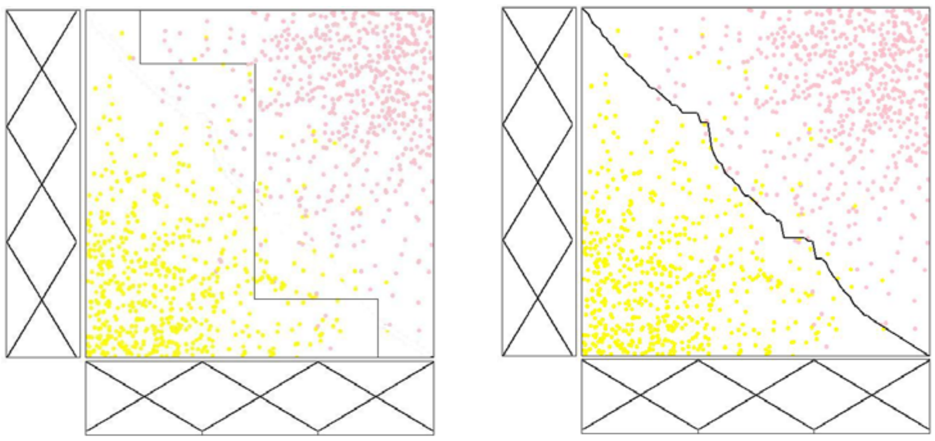
To find the best accuracy threshold for a typical data set, different threshold values can be tested through the discussed method and the optimal value will be obtained by comparing the generalization accuracies of different cases.

5 Experimental results

In order to evaluate the performance of the proposed scheme, we arranged two sets of experiment**,** over artificial and real-life data sets. The results were compared with the non-weighted rule-base classifier in each case.

5.1 Artificial data

In this experiment, we generated 1200 data points of two classes, namely 1 and 2. The data of the first class had a normal distribution, with $\mu = (0,0)$ and $\sigma = 0.3$, whereas the data of the second class had a normal distribution, with $\mu = (1,1)$ and $\sigma = 0.3$. Using 4 triangular fuzzy sets, we first run the non-weighted and then the weighted rule-base classifier over this data set and found the discriminating line in each case. It is known that the closer the discriminating line to the diagonal, the higher generalization ability of the classifier. The results of this experiment are shown in Fig. 2.



a) Non-weighted rule-base classifier b) Weighted rule-base classifier

Fig. 2. Visualized effect of the proposed rule-weighting technique compared to the non-weighted case for a 2-class, Normal-distributed data set

5.2 Real-life data sets

In order to assess the performance of the proposed scheme over some real-life data, we used the data sets shown in Table 1 available from UCI ML repository. To construct an initial rule-base for a specific data set, a number of equi-length rules (number of antecedents equaling to the number of features), having at least one training pattern in their decision areas were generated. In order to assess the effect of the proposed scheme in comparison with its alternatives, we used 10CV technique which is a case of n-fold cross validation.

In the first part of the experiment, the generalization accuracy of the initial rule-base (before rule weighting) was measured.

In the second part, our rule-weighting method was evaluated without considering any accuracy threshold, as discussed in Section 4. In other words, the threshold was set to 1. The results, shown in Table 2 narrate from a positive effect for the weighting method over the generalization ability.

Finally, in the third part, for each data set, we tried different values of the accuracy threshold by changing it from 1 down to 0.4 (by the step size of 0.05). Using the LVO (Leave One Out) technique, in each case, the error rate was measured using training data. The best threshold (leading to the best result) was then selected to evaluate the generalization ability over that data set using the 10CV technique.

The error rates of the classifier using the optimal value of θ for each data set are presented in Table 2. In this table, our proposed method is compared with another successful rule-based method as benchmark results called C4.5 reported by Elomaa and Rousu [7]. It is also compared to four different weighting methods defined by Ishibuchi in [12]. In all cases, the rule selection is performed using the single winner method. As shown in Table 2, except in one case, the proposed classifier in this paper results in better classification rates, compared to the best results already achieved by C4.5. As seen, the proposed method achieves better results in comparison with Ishibuchi metrics in all experimental cases.

Table 1. Some statistics of the data sets used in our computer simulations

Data set	Number of attributes	Number of patterns	Number of classes
Iris	4	150	3
Wine	13	178	3
Thyroid	5	215	3
Sonar	60	208	2
Bupa	6	345	2
Pima	8	768	2
Glass	9	214	6

Table 2. Classification Error rates of the proposed classifier using the optimal threshold values in comparison with threshold of 1, the non-weighted rule-base classifier, Ishibuchi weighting methods and the C4.5 method for data sets of Table 1

Data sets	Error Rates (%)							
	No Weight	proposed method		Ishibuchi Metrics				C4.5 (best results)
		$\theta = 1$	Optimal θ	Metric1	Metric2	Metric3	Metric4	
Iris	5	3.6	**3.6**	4.2	4.6	4.4	4.2	5.1
Wine	6.1	5.2	**5.1**	8.4	8.4	5.6	6.7	5.6
Pima	27.8	25.7	**24.3**	26.3	26.5	27.8	26.5	25
Bupa	39	37.8	**37.8**	39	38.4	38.8	38.8	38.2
Thyroid	8.5	4.1	**4.1**	6.7	5.8	6.3	6.1	6.7
Glass	35	34.9	**33.3**	41.1	40.2	40.6	44.4	27.3
Sonar	11.2	5	**4.1**	11	10.8	10.6	10.8	23.3

6 Conclusions

In this paper, a novel method of rule-base construction using data mining principles and a rule weighting mechanism was proposed. Using the proposed method for rule generation, it will be possible to generate rules having different lengths, efficiently. It is much more useful when dealing with high dimensional data sets, were the existing methods are not able to generate rules containing more than 2 or 3 antecedent conditions. We also proposed a new method of rule-weight specification in order to tune the classifier. As the number of partitions of each feature increases, the generalization ability of the system competes and even precedes the weighted K-NN method. Moreover, the proposed scheme is a FRBCS and has the advantage of interpretability, especially for low-dimensional data sets.

We also proposed a mechanism to find the optimal rule weights, which is much more useful in case of noisy or highly overlapped data sets, in order to prevent from overfitting of the learned classifier.

We used seven data sets from UCI-ML repository to assess the performance of the learning scheme. Simulation results on thsese data sets showed that the method can be used to construct a rule-base with a good generalization ability. The effect of rule weights could be seen, clearly, through this experiment. We also showed that the proposed method is more effective in reducing the error rate of the classifier in comparison with all weighting metrics introduced by Ishibuchi and also comparing to the C4.5 as a successful rule-based method.

References

1. Arima, M., Hara, E.H., Katzberg, J.D.: A fuzzy logic and rough sets controller for HVAC systems. In: the IEEE WESCANEX, pp. 133--138, New York (1995)
2. Cordón, O., Herrera, F., Peregrín, A.: Applicability of the fuzzy operators in the design of fuzzy logic controllers. Fuzzy Sets and Systems, Vol. 86, No. 1, 15--41 (1997)
3. Glorennec, P.Y.: Application of fuzzy control for building energy management. In: Building Simulation. In: International Building Performance Simulation Association 1. pp. 197–201, France (1991)
4. Bárdossy, A., Duckstein, L.: Fuzzy rule-based modeling with applications to geophysical. biological and engineering systems, CRC Press (1995).
5. Bezdek, J.C., Pal, S.K.: Fuzzy Models for Pattern Recognition. Methods that Search for Structures in Data. IEEE Press, Boca Raton (1992)
6. Chi, Z., Yan, H., Pham, T.: Fuzzy algorithms with applications to image processing and pattern recognition. World Scientific, New York (1996)
7. Jin, Y.: Fuzzy Modeling of High-dimensional Systems. Complexity Reduction and Interpretability Improvement, IEEE Trans. on Fuzzy Systems, 212--221 (2000)
8. Jin, Y., Von Seelen, W., and Sendhoff, B.: On Generating FC3 Fuzzy Rule Systems from Data Using Evolution Strategies, IEEE Trans. on Systems, Man and Cybernetics - Part B: Cybernetics, Vol 29, 829--845, (1999)
9. Roubos, H., and Setnes, M.: Compact and Transparent Fuzzy Models and Classifiers Through Iterative Complexity Reduction, IEEE Trans. on Fuzzy Systems, 516--24, (2001)
10. Setnes, M., Babuska, R., and Verbruggen, B.: Rule-based Modeling: Precision and Transparency, IEEE Trans. on Systems, Man, and Cybernetics - Part C: Applications and Reviews, Vol 28, 165--169, (1998)
11. Nauck D. and Kruse R.: How the learning of rule weights affects the interpretability of fuzzy systems. In: 7th IEEE International Conference on Fuzzy Systems, pp. 1235--1240, Anchorage (1998)
12. Ishibuchi H. and Yamamoto T.: Rule weight specification in fuzzy rule-based classification systems. IEEE Trans. on Fuzzy Systems, Vol. 13, No. 4, 428--435, (2005)
13. Ishibuchi H., Nozaki K. and Tanaka H.: Distributed representation of fuzzy rules and its application to pattern classification, Fuzzy Sets and Systems, Vol 52, 21--32, (1992).
14. Ishibuchi, H., Nakashima, T., and Morisawa, T.: Voting in Fuzzy Rule-Based Systems for Pattern Classification problems, Fuzzy Sets and Systems, Vol 103, 223--238, (1999)
15. Ishibuchi H., Yamamoto T.: Comparison of heuristic criteria for fuzzy rule selection in classification problems. Fuzzy Optimization and Decision Making, Vol 3, No.2, 119--139, (2004)