Available at
www.ComputerScienceWeb.com
POWERED BY SCIENCE DIRECT®

ELSEVIER

Information Sciences 150 (2003) 77–93

INFORMATION SCIENCES
AN INTERNATIONAL JOURNAL

www.elsevier.com/locate/ins

Learning fuzzy classification rules from labeled data

Johannes A. Roubos [a,*], Magne Setnes [b], Janos Abonyi [c]

[a] *Control Laboratory, Faculty of Information Technology and Sciences,*
Delft University of Technology, P.O. Box 5031, 2600 GA Delft, The Netherlands
[b] *Heineken Technical Services, R&D, Burgemeester Smeetsweg 1, 2382 PH Zoeterwoude,*
The Netherlands
[c] *Department of Process Engineering, University of Veszprem, P.O. Box 158,*
H-8201 Veszprem, Hungary

Received 5 August 2000; accepted 7 April 2001

Abstract

The automatic design of fuzzy rule-based classification systems based on labeled data is considered. It is recognized that both classification performance and interpretability are of major importance and effort is made to keep the resulting rule bases small and comprehensible. For this purpose, an iterative approach for developing fuzzy classifiers is proposed. The initial model is derived from the data and subsequently, feature selection and rule-base simplification are applied to reduce the model, while a genetic algorithm is used for parameter optimization. An application to the Wine data classification problem is shown.

Keywords: Compact fuzzy classifier; Linguistic model; Genetic algorithm; Similarity-driven rule-base reduction; Wine data

[*] Corresponding author. Tel.: +31-15-278-3371; fax: +31-15-278-6679.

E-mail addresses: hans@ieee.org (J.A. Roubos), magne@ieee.org (M. Setnes), abonyij@fmt.vein.hu (J. Abonyi).

0020-0255/02/$ - see front matter © 2002 Elsevier Science Inc. All rights reserved.
PII: S0020-0255(02)00369-9

1. Introduction

Rule-based expert systems are often applied to classification problems in various application fields, like fault detection, biology, and medicine. Fuzzy logic can improve such classification and decision support systems by using fuzzy sets to define overlapping class definitions. The application of fuzzy if-then rules also improves the interpretability of the results and provides more insight into the classifier structure and decision making process [1]. We focus on the extraction of fuzzy rule-based classifiers from labeled data. Data-driven identification of such classifiers has to deal with structural issues, like the selection of the relevant features and finding an effective partitioning of the input domain. Moreover, linguistic interpretability is also an important aspect of rule-based classifiers.

The automated construction of fuzzy classification rules from data has been approached by different techniques like, e.g., neuro-fuzzy methods [2,3], genetic-algorithm based rule selection [4], and fuzzy clustering in combination with other methods such as fuzzy relations [6] and genetic algorithm (GA) optimization [5]. Traditionally, algorithms to obtain classifiers have focused either on the accuracy or the interpretability aspects. Recently, some approaches to combining these properties have been reported; fuzzy clustering is proposed to derive transparent models in [7], linguistic constraints are applied to fuzzy modeling in [1] and rule extraction from neural networks is described in [8].

In this paper, we describe an approach that addresses both the accuracy and the interpretability. We show that compact, accurate and interpretable fuzzy rule-based classifiers can be obtained from labeled observation data in an iterative approach. First, an initial model is derived from the observation data. Secondly, feature selection and rule-base simplification methods [9] are applied to reduce the initial model. Finally, a real-coded GA is applied to optimize the models parameters in order to improve the classification accuracy [10,11]. The GA can also be applied in a loop together with rule-base simplification. In this case, the GA uses a multi-criterion objective to search not only for model accuracy but also for model redundancy. This redundancy is then exploited to reduce and simplify the rule base. Finally, the GA is applied with a multi-criterion function where the redundancy is suppressed in order to make the rules more distinguishable while preserving the accuracy. The result is a compact fuzzy rule base of low complexity with high a classification accuracy.

In the following, we continue with Section 2 where the initial modeling step is explained and transparency and accuracy issues are discussed. In Section 3, feature selection and rule-base simplification are described and the GA optimization is described in Section 4. Section 5 considers the Wine data classification problem known from the literature. Finally, Section 6 concludes the paper.

2. Fuzzy models for classification

2.1. The model structure

We apply fuzzy classification rules that each describe one of the N_c classes in the data set. The rule antecedent is a fuzzy description in the n-dimensional feature space and the rule consequent is a crisp (non-fuzzy) class label from the set $\{1, 2, \ldots, N_c\}$:

$$R_i: \quad \textbf{If } x_1 \text{ is } A_{i1} \textbf{ and} \ldots x_n \text{ is } A_{in} \textbf{ then } g_i = p_i, \quad i = 1, \ldots, M. \tag{1}$$

Here n denotes the number of features, $\vec{x} = [x_1, x_2, \ldots, x_n]^T$ is the input vector, g_i is the output of the ith rule and $A_{i1}, \ldots, A_{in}$ are the antecedent fuzzy sets. The **and** connective is modeled by the product operator, allowing for interaction between the propositions in the antecedent. The degree of activation of the ith rule is calculated as:

$$\beta_i(\vec{x}) = \prod_{j=1}^{n} \mu_{A_{ij}}(x_j), \quad i = 1, 2, \ldots, M, \tag{2}$$

where $\mu_{A_{ij}} \in [0, 1]$ is the membership degree of the jth feature of the data pair x to A_{ij}. The output of the classifier is then determined by the rule that has the highest degree of activation:

$$\hat{y} = g_{i^*}, \quad i^* = \arg \max_{1 \leqslant i \leqslant M} \beta_i. \tag{3}$$

In the following we assume that the number of rules corresponds to the number of classes, i.e., $M = N_c$. The certainty degree of the decision is given by the normalized degree of firing of the rule:

$$\mathrm{CF} = \beta_{i^*} \Big/ \sum_{i}^{M} \beta_i. \tag{4}$$

2.2. Data-driven initialization

From the K available input–output data pairs $\{\vec{x}_k, y_k\}$ we construct the n-dimensional pattern matrix $X^T = [\vec{x}_1, \ldots, \vec{x}_K]$, and the corresponding label vector $\vec{y}^T = [y_1, \ldots, y_K]$. The fuzzy antecedents $A_{ij}(x_j)$ in the initial rule base are now determined by a three-step algorithm. In the first step, M multivariable membership functions are defined in the product space of the features. Each describes a region where the system can be approximated by a single fuzzy rule. This partitioning can be realized by iterative methods such as clustering [10]. Here, given the labeled data, a one-step approach, similar to the one in [12], is proposed. This assumes that each class is described by a single, compact construct in the feature space. If this is not the case, other methods such as, e.g., relational classification [6], can be applied. Similar to the fuzzy clustering algorithm in [13], the approach proposed here also assumes that the shape of the fuzzy sets can be approximated by ellipsoids. Hence, each class prototype is represented by a center $\vec{v}_i$ and its covariance matrix Q_i:

$$\vec{v}_i = \frac{1}{K_i} \sum_{k|y_k=i} \vec{x}_k, \tag{5}$$

$$Q_i = \frac{1}{K_i} \sum_{k|y_k=i} (\vec{x}_k - \vec{v}_i)^{\mathrm{T}} (\vec{x}_k - \vec{v}_i). \tag{6}$$

Here i denotes the index of the classes, $i = 1, \ldots, N_{\mathrm{c}}$, and K_i represents the number of samples that belong to the ith class. In the second step, the algorithm computes the fuzzy partition matrix U whose ikth element $u_{ik} \in [0, 1]$ is the membership degree of the data object $\vec{x}_k$ in class i. This membership is based on the distance between the observation and the class center:

$$D_{ik}^2 = (\vec{x}_k - \vec{v}_i)^{\mathrm{T}} Q_i^{-1} (\vec{x}_k - \vec{v}_i). \tag{7}$$

Using this distance, the membership becomes:

$$u_{ik} = 1 \Bigg/ \sum_{j=1}^{K} \left(\frac{D_{ik}}{D_{jk}} \right)^{2/(m-1)}, \tag{8}$$

where m denotes a weighting exponent that determines the fuzziness of the obtained partition ($m = 1.8$ is applied in the example).

The rows of U now contain the pointwise representations of the multidimensional fuzzy sets describing the classes in the feature space. In the third step, the univariate fuzzy sets A_{ij} in the classification rules (1) are obtained by projecting the rows of U onto the input variables x_j and subsequently approximate the projections by parametric functions [14]. In the example we apply triangular fuzzy sets for simplicity:

$$\mu(x; a, b, c) = \max\left(0, \min\left(\frac{x-a}{b-a}, \frac{c-x}{c-b}\right)\right). \tag{9}$$

If more smooth membership function constructs are used, e.g., Gaussian or exponential functions, the resulting model will in general have a higher accuracy in fitting the training data.

3. Model reduction

3.1. Ensuring transparency and accuracy

Fixed membership functions are often used to partition the feature space [4]. Membership functions derived from the data, however, explain the data-patterns in a better way. Typically less sets and fewer rules result than in a fixed partition approach. The initial rule base constructed by the proposed method fulfills many criteria for transparency and good semantic properties [1,11]: moderate number of rules, distinguishability, normality and coverage. The transparency and compactness of the rule base can be further improved by model reduction methods. Two methods are presented here. The first method is an open-loop feature selection algorithm that is based on Fisher's interclass separability criterion [15], calculated from the covariances of the clusters. The other method is the similarity-driven rule-base simplification proposed by Setnes et al. [9].

3.2. Feature selection based on interclass separability

Using too many features results in difficulties in the prediction and interpretability capabilities of the model due to redundancy, non-informative features and noise. Hence, feature selection is usually necessary. We apply the Fischer interclass separability method that is based on statistical properties of the labeled data. This criterion is based on the *between-class* and *within-class* scatter or covariance matrices, called Q_b and Q_w, respectively, which sum up to the *total scatter matrix* Q_t which is the covariance of the whole training data containing K data pairs.

$$Q_\mathrm{t} = \frac{1}{K}\sum_{k=1}^{K}(\vec{x}_k - \vec{v})^\mathrm{T}(\vec{x}_k - \vec{v}), \tag{10}$$

where

$$\vec{v} = \frac{1}{K}\sum_{k=1}^{K}\vec{x}_k = \frac{1}{K}\sum_{i=1}^{N_\mathrm{c}}K_i\vec{v}_i, \tag{11}$$

with K_i the number of cases in each class. The total scatter matrix can be decomposed as

$$Q_\mathrm{t} = Q_\mathrm{b} + Q_\mathrm{w}, \tag{12}$$

where

$$Q_\mathrm{b} = \sum_{i=1}^{N_\mathrm{c}}K_i(\vec{v}_i - \vec{v})^\mathrm{T}(\vec{v}_i - \vec{v}), \tag{13}$$

$$Q_{\mathrm{w}} = \sum_{i=1}^{N_{\mathrm{c}}} Q_i. \tag{14}$$

The feature interclass separability selection criterion is a trade-off between Q_{b} and Q_{w}. A feature ranking is made iteratively by leaving out the worst feature in each step and is exploited for the open-loop feature selection:

$$J_j = \det(Q_{\mathrm{b}})/\det(Q_{\mathrm{w}}), \tag{15}$$

where det is the determinant and J_j is the criterion value including j features.

3.3. Similarity-driven rule-base simplification

The similarity-driven rule-base simplification method [9] uses a similarity measure to quantify the redundancy among the fuzzy sets in the rule base. A similarity measure based on the set-theoretic operations of intersection and union is applied:

$$S(A,B) = \frac{|A \cap B|}{|A \cup B|} \tag{16}$$

where $|\cdot|$ denotes the cardinality of a set, and the $\cap$ and $\cup$ operators represent the intersection and union, respectively. If $S(A,B) = 1$, then the two membership functions A and B are equal. $S(A,B)$ becomes 0 when the membership functions are non-overlapping. The complete rule-base simplification algorithm is given in [9].

Similar fuzzy sets are merged when their similarity exceeds a user defined threshold $\theta \in [0,1]$ ($\theta = 0.5$ is applied). Merging reduces the number of different fuzzy sets (linguistic terms) used in the model and thereby increases the transparency. If all the fuzzy sets for a feature are similar to the universal set U, $\mu_U(x) = 1, \forall x$, or if merging led to only one membership function for a feature, then this feature is eliminated from the model. The method is illustrated in Fig. 1.

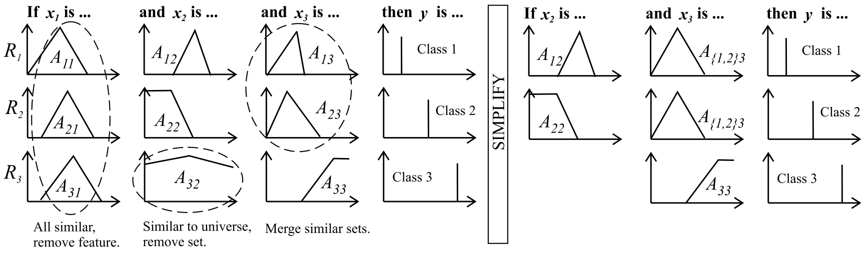


Fig. 1. Similarity-driven rule-base simplification.

3.4. Genetic multi-objective optimization

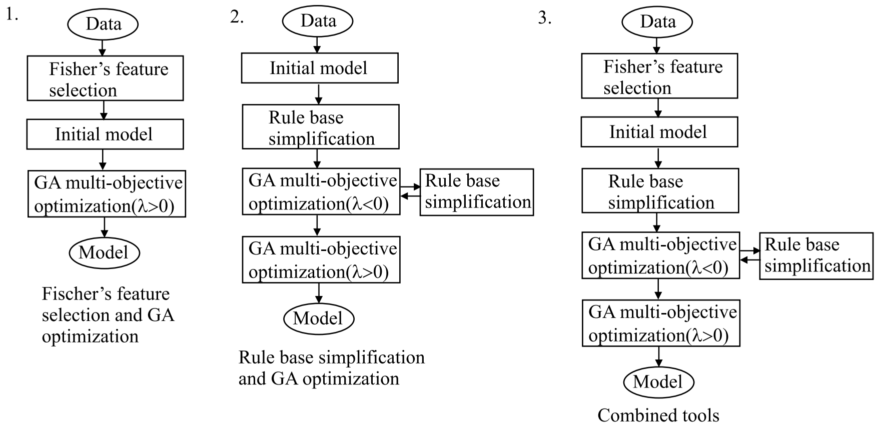
Many steps in the modeling process are sub-optimal. For instance, the projection of the clusters onto the input variables, and their approximation by triangular fuzzy sets introduces a structural error since the resulting premise partition differs from the cluster partition matrix. Another example is the separated identification of the models antecedent and consequent parts. To improve the classification capability of the rule base, we apply a GA-based optimization method that was developed in [5,11]. Also other model properties can be optimized by applying multi-objective functions, like, e.g., search for redundancy [10].

When an initial fuzzy model has been obtained from data, it is simplified and optimized in an iterative fashion. Combinations of the GA with the model reduction tools described above can lead to various modeling schemes. Three different approaches are shown in Fig. 2.

The model accuracy is measured in terms of the number of misclassifications. To further reduce the model complexity, the misclassification rate is combined with a similarity measure in the GA objective function. Similarity is rewarded during the iterative process, that is, the GA tries to emphasize the redundancy in the model. This redundancy is then used to remove unnecessary fuzzy sets in the next iteration. In the final step, the accuracy is optimized while similarity among fuzzy sets is penalized as to obtain a distinguishable term set suitable for inspection and linguistic interpretation.

The GAs is subject to minimizing the following multi-objective function:

$$J = (1 + \lambda S^*) \cdot \mathrm{MCE}, \tag{17}$$


1.
Data
Fisher's feature selection
Initial model
GA multi-objective optimization(λ>0)
Model
2.
Data
Initial model
Rule base simplification
GA multi-objective optimization(λ<0)
Rule base simplification
GA multi-objective optimization(λ>0)
Model
3.
Data
Fisher's feature selection
Initial model
Rule base simplification
GA multi-objective optimization(λ<0)
Rule base simplification
GA multi-objective optimization(λ>0)
Model

Fischer's feature selection and GA optimization

Rule base simplification and GA optimization

Combined tools

Fig. 2. Modeling schemes resulting from a combination of tools.

where MCE is the mean classification error:

$$\mathrm{MCE} = \frac{1}{K}\left(\sum_{k=1}^{K}(y_k \neq \hat{y}_k)\right), \tag{18}$$

with c the class and $\hat{c}$ the predicted class. $S^* \in [0,1]$ is the average of the maximum pair-wise similarity that is present in each input, i.e. S^* is an aggregated similarity measure for the total model:

$$S^* = \frac{1}{n}\left(\sum_{i=1}^{n}\frac{\max(S(A_{ij},A_{ik}))}{\eta_i - 1}\right), \quad j,k \in \{1,2,\ldots,\eta_i\}, \quad j \neq k, \tag{19}$$

where n is the number of inputs and η_i the number of sets for each input variable. The weighting function $\lambda \in [-1,1]$ determines whether similarity is rewarded ($\lambda < 0$) or penalized ($\lambda > 0$). In the example a fixed λ of -0.2 and 0.2 is applied for the two cases.

4. Real-coded genetic algorithm

A real-coded GA [16] is used for the optimization of the parameters of the antecedent membership functions. GAs are attractive for multi-objective optimization since they perform a parallel search for multiple solutions in parallel and are able to handle problems with complex objectives and constraints. The main aspects of the proposed GA-based optimization are discussed in the subsections below and the implementation is then summarized in Section 4.5.

4.1. Fuzzy model representation

Chromosomes are used to describe the solutions. With a population size L, we encode the parameters of each fuzzy model (solution) in a chromosome $\vec{s}_l$, $l = 1,\ldots,L$, as a sequence of elements describing the fuzzy sets in the rule antecedents. A classifier with M fuzzy rules is encoded as:

$$\vec{s}_l = (\mathrm{ant}_1,\ldots,\mathrm{ant}_M), \tag{20}$$

where $\mathrm{ant}_i = (a_{i1}, b_{i1}, c_{i1}, \ldots, a_{in}, b_{in}, c_{in})$, contains the parameters of the antecedent fuzzy sets A_{ij}, $j = 1,\ldots,n$, according to (9). In the initial population $\mathrm{S}^0 = \{\vec{s}_1^{\,0},\ldots,\vec{s}_L^{\,0}\}$, $\vec{s}_1^{\,0}$ is the initial model, and $\vec{s}_2^{\,0},\ldots,\vec{s}_L^{\,0}$ are created by random variation with a uniform distribution around $\vec{s}_1^{\,0}$.

4.2. Selection function

The *roulette wheel* selection method [16] is used to select n_C chromosomes for operation. The chance on the roulette wheel is adaptive and is given as $P_l/\sum_{l'} P_{l'}$, where

$$P_l = \left(\frac{1}{J_l}\right)^2, \quad l, l' \in \{1, \ldots, L\}, \tag{21}$$

and J_l is the performance of the model encoded in chromosome $\vec{s}_l$. The inverse of the selection function (P_l^{-1}) is used to select chromosomes for deletion. The best chromosome is always preserved in the population (*Elitist* selection). The chance that a selected chromosome is used in a crossover operation is 90% and the chance for mutation is 10% (in this paper). When a chromosome is selected for crossover (or mutation), one of three crossover (or mutation) operators are applied with equal probability.

4.3. Genetic operators

Two classical operators, *simple arithmetic crossover* and *uniform mutation*, and four special real-coded operators are used in the GA. In the following, $r \in [0, 1]$ is a random number (uniform distribution), $t = 0, 1, \ldots, T$ is the generation number, $\vec{s}_v$ and $\vec{s}_w$ are chromosomes selected for operation, $k \in \{1, 2, \ldots, N\}$ is the position of an element in the chromosome, and $v_k^{\min}$ and $v_k^{\max}$ are the lower and upper bounds, respectively, on the parameter encoded by element k: Single chromosomes $(\vec{s}_v)$ are selected for mutation and pairs of chromosomes $(\vec{s}_v, \vec{s}_w)$ are selected for crossover:

(1) *Uniform mutation*: a random selected element v_k, $k \in \{1, 2, \ldots, N\}$ is replaced by v'_k which is a random number in the range $[v_k^{\min}, v_k^{\max}]$. The resulting chromosome is $\vec{s}_v^{t+1} = (v_1, \ldots, v'_k, \ldots, v_m)$.

(2) *Multiple uniform mutation*: uniform mutation of n randomly selected elements, where n is also selected at random from $\{1, \ldots, N\}$.

(3) *Gaussian mutation*: all elements of a chromosome are mutated such that $\vec{s}_v^{t+1} = (v'_1, \ldots, v'_k, \ldots, v'_m)$ where $v'_k = v_k + f_k$, $k = 1, 2, \ldots, N$. Here f_k is a random number drawn from a *Gaussian* distribution with zero mean and an adaptive variance $\sigma_k = ((T - t)/T)((v_k^{\max} - v_k^{\min})/3)$. The parameter tuning performed by this operator becomes finer and finer as the generation counter t increases.

(4) *Simple arithmetic crossover*: $\vec{s}_v^t$ and $\vec{s}_w^t$ are crossed over at the kth position. The resulting offsprings are $\vec{s}_v^{t+1} = (v_1, \ldots, v_k, w_{k+1}, \ldots, w_N)$ and $\vec{s}_w^{t+1} = (w_1, \ldots, w_k, v_{k+1}, \ldots, v_N)$, where k is selected at random from $\{2, \ldots, N - 1\}$.

(5) *Whole arithmetic crossover*: a linear combination of $\vec{s}_v^t$ and $\vec{s}_w^t$ resulting in $\vec{s}_v^{t+1} = r(\vec{s}_v^t) + (1-r)\vec{s}_w^t$ and $\vec{s}_w^{t+1} = r(\vec{s}_w^t) + (1-r)\vec{s}_v^t$.
(6) *Heuristic crossover*: $\vec{s}_v^t$ and $\vec{s}_w^t$ are combined such that $\vec{s}_v^{t+1} = \vec{s}_v^t + r(\vec{s}_w^t - \vec{s}_v^t)$ and $\vec{s}_w^{t+1} = \vec{s}_w^t + r(\vec{s}_v^t - \vec{s}_w^t)$.

4.4. Constraints

The optimization performed by the GA is subjected to two types of constraints: *partition* and *search space*. The partition constraint prohibits gaps in the partitions of the input (antecedent) variables. The coding of a fuzzy set must comply with (9), i.e., $a \leqslant b \leqslant c$. To avoid gaps in the partition, pairs of neighboring fuzzy sets are constrained by $a_\mathrm{R} \leqslant c_\mathrm{L}$, where L and R denote left and right set, respectively.

The GA search space is constrained by a user defined bound-parameter α_1 that applies to the antecedent of the rules. This bound α_1 is intended to maintain the distinguishability of the models term set (the fuzzy sets) by allowing the parameters describing the fuzzy sets A_{ij} to vary only within a bound of $\pm\alpha_1|\chi_j|$ around their initial values, where $|\chi_j|$ is the length (range) of the domain on which the fuzzy sets A_{ij} are defined. The search space constraints are coded in the two vectors, $\vec{v}^{\max} = [v_1^{\max}, \ldots, v_N^{\max}]$ and $\vec{v}^{\min} = [v_1^{\min}, \ldots, v_N^{\min}]$, giving the upper and lower bounds on each of the N elements in a chromosome. During generation of the initial partition, and in the case of a uniform mutation, elements are generated at random within these bounds.

4.5. Genetic algorithm

Given the pattern matrix Z and a fuzzy rule base, select the number of generations T, the population size L, the number of operations n_C and the constraints α_1 and α_2. Let S^t be the current population of solutions $\vec{s}_l^t$, $l = 1, \ldots, L$, and let $\vec{J}^t$ be the vector of corresponding values of the evaluation function:

```
(1) Create initial chromosome $\vec{s}_1^0$ from the initial fuzzy rule base.
(2) Calculate the constraint vectors $\vec{v}^{\min}$ and $\vec{v}^{\max}$ using $\vec{s}_1^0$ and $\alpha_1$.
(3) Create the initial population $S^0 = \{\vec{s}_1^0, \ldots, \vec{s}_L^0\}$ where $\vec{s}_l^0$, $l = 2, \ldots, L$ are
    created by constrained random variations around $\vec{s}_1^0$, and the partition
    constraints apply.
(4) Repeat genetic optimization for $t = 0, 1, 2, \ldots, T-1$:
    (a) Evaluate $S^t$ and obtain $\vec{J}^t$.
    (b) Select $n_C$ chromosomes for operation.
    (c) Select $n_C$ chromosomes for deletion.
    (d) Operate on chromosomes acknowledging the search space constraints.
    (e) Implement partition constraints.
```

(f) Create new population S^{t+1} by substituting the operated chromosomes for those selected for deletion.

(5) Select best solution from S^t by evaluating $\vec{J}^t$.

5. Example: Wine data

The Wine data [1] contains the chemical analysis of 178 wines produced in the same region in Italy but derived from three different cultivars. The problem is to distinguish the three different types based on 13 continuous attributes derived from chemical analysis: alcohol, malic acid, ash, alcalinity of ash, magnesium, total phenols, flavanoids, nonflavanoids phenols, proanthocyaninsm color intensity, hue, OD280/OD315 of diluted wines and proline (Fig. 3).

Corcoran and Sen [17] applied all the data for learning 60 non-fuzzy if-then rules in a real-coded genetic based machine learning approach and Ishibuchi et al. [4] applied all the data for designing a fuzzy classifier with 60 fuzzy rules by means of an integer-coded GA and grid partitioning. In [18], they discuss several GA-based methods for classifier design and show results for the Wine data for a various number of rules between 3 and 100. The results of these approaches are summarized in Table 1.

Corcoran and Sen used a population of 1500 individuals and applied 300 generations, with full replacement, to come up with the following result for 10 independent trials: best classification rate 100%, average classification rate 99.5% and worst classification rate 98.3% which is three misclassifications. Ishibuchi et al. [19] applied all the 178 samples designing a fuzzy classifier with 60 fuzzy rules by means of an integer-coded GA and grid partitioning. Their population contained 100 individuals and they applied 1000 generations, with full replacement, to come up with the following result for 10 independent trials: best classification rate 99.4% (1 misclassifications), average classification rate 98.5% and worst classification rate 97.8% (4 misclassifications). In both approaches the final rule base contains 60 rules. The main difference is the number of model evaluations that was necessary to come to the final result. In [17] the Pittsburgh approach of GA-based learning is used where each individual in the population contains a complete fuzzy model, resulting in 150,000 model evaluations. In [19] the Michigan approach is followed were each individual contains one rule and the complete population consists of one fuzzy model and thus only 1000 model evaluations were performed.

In Ishibuchi et al. [18], both the Michigan and the Pittsburgh approach are extensively studied for fuzzy classifier design. A superior performance of the

[1] The Wine data is available from the University of California, Irvine, via anonymous ftp. ftp.ics.uci.edu/pub/machine-learning-databases/.

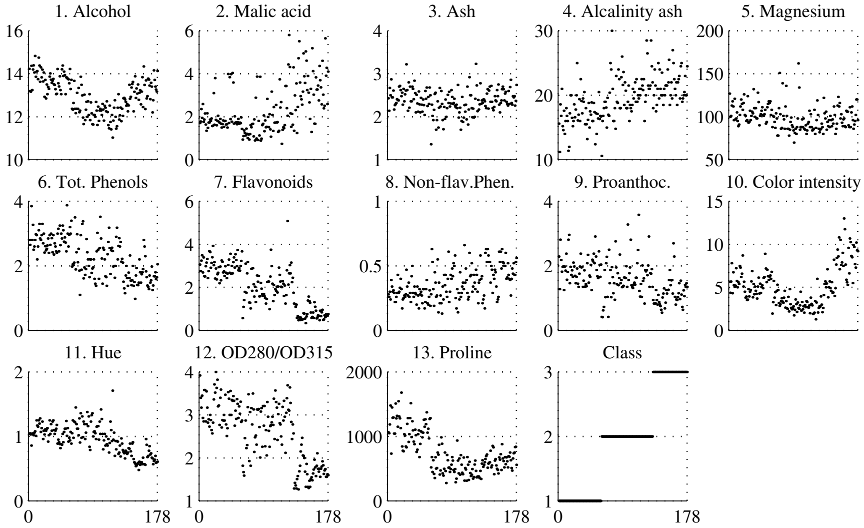


Fig. 3. Wine data: three classes and 13 attributes.

Table 1
Classification results on the Wine data for 10 independent runs

Method	Best result	Average result	Worst result	Rules	Model evaluation
Corcoran and Sen [17]	100%	99.5%	98.3%	60	150000
Ishibuchi et al. [4]	99.4%	98.5%	97.8%	60	6000
Ishibuchi et al. [18]	–%	98.5%	–%	8.0	20000
Ishibuchi et al. [18]	–%	95.5%	–%	6.9	20000
This paper	99.4%	Various schemes	98.3%	3	4000–8000

Michigan approach was found, among others for the Wine data. Rules with on average less than two inputs resulted, that are highly interpretable. However, the rules were weighted with a certainty factor that degrades the transparency. Such a weighting was necessary because fixed membership functions were applied. The best results are also summarized in Table 1.

5.1. Proposed approach

An initial classifier with three rules was constructed with the proposed covariance-based model initialization by using all samples resulting in 90.5% correct, 1.7% undecided and 7.9% misclassifications with the following average certainty factors (CFs) [82.0, 99.6, 80.5] for the three wine classes. The resulting

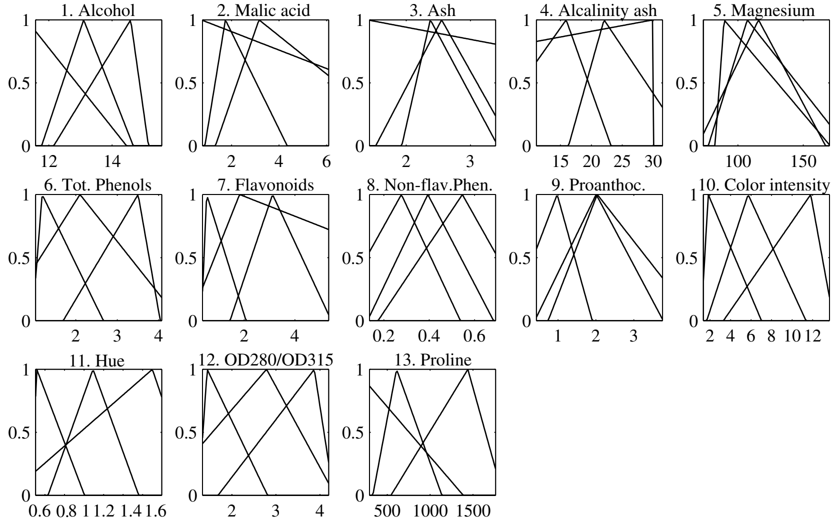


Fig. 4. Wine data: Fuzzy sets by the initialization method for the three classes and all 13 attributes.

fuzzy sets are shown in Fig. 4. Improved classifiers are developed based on the three schemes given in Fig. 2.

Scheme 1. The Fisher interclass separability criterion gives the following feature ranking $\{13, 12, 1, 4, 7, 6, 10, 9, 3, 2, 11, 5, 8\}$. Classifiers were made by adding features *one by one* and 400 iterations with the GA-optimization. The two best classifiers were obtained by using the first 5 or 7 features (15 or 21 fuzzy sets). This gave 98.9% and 99.4% correct classification with CF for the three classes $[0.95, 0.94, 0.84]$ and $[0.94, 0.99, 0.97]$, respectively. The first three-rule classifier is shown in Fig. 5 and rules are given in Table 2.

Scheme 2. The similarity-driven rule-base simplification removed the following eight inputs in three steps: (i) $\{3, 5\}$, (ii) $\{2, 4, 8, 9\}$, (iii) $\{6, 12\}$. After each reduction, 200 GA-iterations were done and 400 after the last reduction. The final three-rule classifier given in Table 3 contains only 11 fuzzy sets as shown in Fig. 6. The classification result was 99.4% correct and CF for the three wine classes was $[0.96, 0.94, 0.94]$.

Scheme 3. Five features were selected based on the feature ranking initially resulting in 5% misclassification. Successively, five fuzzy sets were removed by iterative similarity-driven rule-base simplification and GA optimization (200 iterations). After the final GA tuning (400 iterations) the classification rate was 98.3% with CFs $[0.93, 0.91, 0.91]$. The final model contains five features $\{1, 4, 7, 12, 13\}$. The three-rule classifier is shown in Fig. 7 and rules are given in

Table 2
Three rule fuzzy classifier #1 (L = low, M = medium, H = high, A = any)

	1	2	3	4	5	6	7	8	9	10	11	12	13	Class
	Alc	Mal	Ash	aAsh	Mag	Tot	Fla	nFlav	Pro	Col	Hue	OD2	Pro	
R_1	H	–	–	L	–	–	H	–	–	–	–	H	H	1
R_2	L	–	–	A	–	–	A	–	–	–	–	A	L	3
R_3	M	–	–	H	–	–	L	–	–	–	–	L	L	2

Table 3
Three rule fuzzy classifier #2 (L = low, M = medium, H = high)

	1	2	3	4	5	6	7	8	9	10	11	12	13	Class
	Alc	Mal	Ash	aAsh	Mag	Tot	Fla	nFlav	Pro	Col	Hue	OD2	Pro	
R_1	H	–	–	–	–	–	H	–	–	M	H	–	H	1
R_2	L	–	–	–	–	–	–	–	–	L	H	–	L	2
R_3	H	–	–	–	–	–	L	–	–	H	L	–	L	3

Table 4
Three rule fuzzy classifier #3 (L = low, M = medium, H = high)

	1	2	3	4	5	6	7	8	9	10	11	12	13	Class
	Alc	Mal	Ash	aAsh	Mag	Tot	Fla	nFlav	Pro	Col	Hue	OD2	Pro	
R_1	H	–	–	M	–	–	H	–	–	–	–	H	H	1
R_2	L	–	–	–	–	–	H	–	–	–	–	M	L	2
R_3	H	–	–	–	–	–	L	–	–	–	–	L	L	3

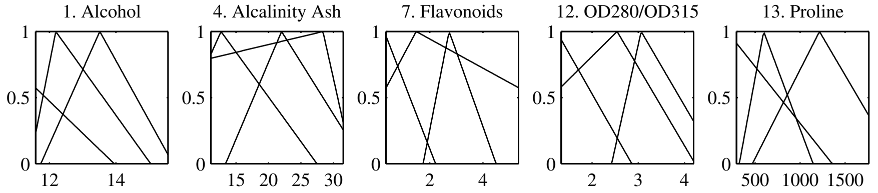


Fig. 5. The fuzzy sets of the optimized three rule wine-classifier (Scheme 1).

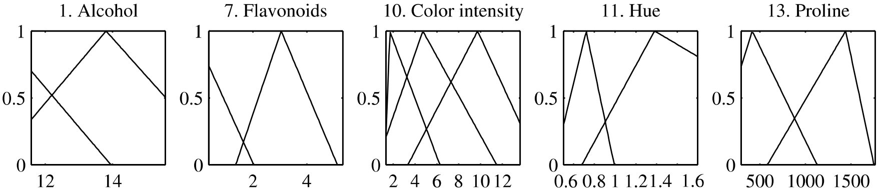


Fig. 6. The fuzzy sets of the optimized three rule wine-classifier (Scheme 2).

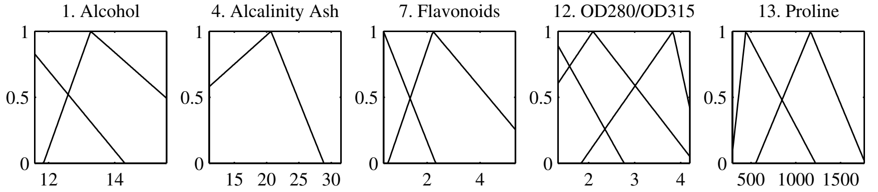


Fig. 7. The fuzzy sets of the optimized three rule wine-classifier (Scheme 3).

Table 4. The fuzzy sets obtained for $\{1, 7, 13\}$ are similar to those obtained with Scheme 2 (Fig. 6). The 'any' label for feature 4 seems not to be informative and could be removed manually resulting in one feature less and the same performance.

In this example, feature reduction was obtained by all three schemes. The resulting three rule classifiers use only five of the initial 13 features. Differences in the reduction methods are: (i) Similarity analysis results in a closed-loop feature selection because it depends on the actual model while the applied open-loop feature selection can be used beforehand as it is independent from the model. (ii) In similarity analysis, a feature can be removed from individual rules. In the interclass separability method the feature is omitted in all the rules. (iii) Similarity analysis allows for single fuzzy sets for multiple classes, e.g., in the Wine data, the combined class 1 and 2 are distinguishable from class 3 by feature 11, while this one ranks low in the interclass separability method. The similarity based-rule reduction is important for transparency as can be seen by comparing the sets and rules for the three schemes, e.g., an additional 'any' label is necessary for the result of Scheme 1. Overall, the Wine data is still relative low in dimension. It is expected that in higher dimensional problems, the two reduction method should be combined, i.e., some of the features should be removed beforehand by the interclass separability method.

Comparing the fuzzy sets in Figs. 5–7 with the data in Fig. 3 shows that the obtained rules are highly interpretable. For example in Fig. 6, the flavonoids are divided in low, medium and high, which is clearly visible in the data. Visual inspection of the data also shows that 'don't care' elements (fuzzy sets similar to a universal set) were obtained for features that contain little variation over the three classes, e.g., Ash, Mag, nFlav, Hue, etc.

Concluding, the obtained result is comparable to those in [17] and [4], but our classifiers use far less rules (3 compared to 60) and less features. The rule-bases in [18] are comparable in accuracy and number of rules. However, these models are less interpretable due to the application of fixed fuzzy sets that are applied in combination with a rule-weighting. Thus the rule-interpolation determines the final output in contrast to our approach where only one-rule determines the output due to the winner takes all strategy. Results for other data-sets with similar modeling schemes are studied in [20,21].

6. Conclusion

The design of fuzzy rule-based classifiers based on labeled data is approached by combining tools for feature selection, model initialization, model reduction and model tuning. It is shown that these can be applied in an iterative way. A covariance-based model initialization method is applied to obtain an initial fuzzy classifier. Successive application of feature selection, rule-base simplification and GA-based tuning then results in compact, interpretable and accurate fuzzy rule-based classifiers. The proposed approach was successfully applied to the Wine data. The resulting classifier is very compact in comparison with other studies, while the accuracy is very similar.

References

[1] J. Valente de Oliveira, Semantic constraints for membership function optimization, IEEE Transactions on Fuzzy Systems 19 (1) (1999) 128–138.

[2] D. Nauck, R. Kruse, Obtaining interpretable fuzzy classification rules from medical data, Artificial Intelligence in Medicine 16 (1999) 149–169.

[3] S. Mitra, Y. Hayashi, Neuro-fuzzy rule generation: Survey in soft computing framework, IEEE Transactions on Neural Networks 11 (3) (2000) 748–768.

[4] H. Ishibuchi, T. Nakashima, Voting in fuzzy rule-based systems for pattern classification problems, Fuzzy Sets and Systems 103 (1999) 223–238.

[5] M. Setnes, J.A. Roubos, GA-fuzzy modeling and classification: Complexity and performance, IEEE Transactions on Fuzzy Systems 8 (5) (2000) 509–522.

[6] M. Setnes, R. Babuška, Fuzzy relational classifier trained by fuzzy clustering, IEEE Transactions on Systems, Man, and Cybernetics—Part B: Cybernetics 29 (1999) 619–625.

[7] M. Setnes, R. Babuška, H.B. Verbruggen, Rule-based modeling precision and transparency, IEEE Transactions on Systems, Man, and Cybernetics—Part C: Applications and Reviews 29 (1) (1999) 165–169.

[8] D. Setiono, Generating concise and accurate classification rules for breast cancer diagnosis, Artificial Intelligence in Medicine 18 (1999) 205–219.

[9] M. Setnes, R. Babuška, U. Kaymak, H. van Nauta Lemke, Similarity measures in fuzzy rule base simplification, IEEE Transactions on Systems, Man, and Cybernetics—Part B: Cybernetics 28 (3) (1998) 376–386.

[10] J.A. Roubos, M. Setnes, Compact fuzzy models through complexity reduction and evolutionary optimization, in: Proceedings of the 9th IEEE International Conference on Fuzzy System, San Antonio, USA, 2000, pp. 762–767.

[11] M. Setnes, J.A. Roubos, Transparent fuzzy modeling using fuzzy clustering and GA's, in: Proceedings of the 18th International Conference of the North American Fuzzy Information Processing Society, NAFIPS, New York, USA, June 10–12, 1999, pp. 198–202.

[12] S. Abe, Dynamic cluster generation for a fuzzy classifier with ellipsoidal regions, IEEE Transactions on Systems, Man, and Cybernetics—Part B: Cybernetics 28 (6) (1998) 869–876.

[13] D.E. Gustafson, W.C. Kessel, Fuzzy clustering with a fuzzy covariance matrix, in: Proceedings of the IEEE CDC, San Diego, USA, 1979, pp. 761–766.

[14] R. Babuška, Fuzzy Modeling for Control, Kluwer Academic Publishers, Boston, USA, 1998.

[15] K.J. Cios, W. Pedrycz, R.W. Swiniarski, Data Mining Methods for Knowledge Discovery, Kluwer Academic Press, Boston, USA, 1998.

[16] Z. Michalewicz, Genetic Algorithms + Data Structures = Evolution Programs, second ed., Springer Verlag, New York, USA, 1994.

[17] A.L. Corcoran, S. Sen, Using real-valued genetic algorithms to evolve rule sets for classification, in: Proceedings of the 1st IEEE Conference on Evolutionary Computation, Orlando, USA, June 27–29, 1994, pp. 120–124.

[18] H. Ishibuchi, T. Nakashima, T. Murata, Techniques and applications of genetic algorithm-based methods for designing compact fuzzy classification systems, in: C.T. Leondes (Ed.), Fuzzy Theory Systems: Techniques and Applications, vol. 3, Academic Press, London, UK, 1999, p. 28 (Chapter 40).

[19] H. Ishibuchi, T. Nakashima, T. Murata, Performance evaluation of fuzzy classifier systems for multidimensional pattern classification problems, IEEE Transactions on Systems, Man, and Cybernetics—Part B: Cybernetics 29 (5) (1999) 601–618.

[20] J. Abonyi, J.A. Roubos, Initialization of fuzzy classification rules, in: 5th Online World Conference on Soft Computing in Industrial Applications. Available from <http://www.cranfield.ac.uk/wsc5/>, 2000, CD-ROM.

[21] J.A. Roubos, M. Setnes, ompact fuzzy models and classifiers through model reduction and evolutionary optimization, in: L. Chambers (Ed.), The Practical Handbook of Genetic Algorithms: Applications, 2nd edition, CRC Press, London, UK, 2000, pp. 31–59 (Chapter 1).