A Minimum-Risk Genetic Fuzzy Classifier Based on Low Quality Data

Ana M. Palacios[1], Luciano Sánchez[1], and Inés Couso[2,*]

[1] Dpto. Informática, Univ. Oviedo, Gijón, Asturias, Spain
[2] Dpto. de Estadística e I.O. y D.M, Univ. Oviedo, Gijón, Asturias, Spain

Abstract. Minimum risk classification problems use a matrix of weights for defining the cost of misclassifying an object. In this paper we extend a simple genetic fuzzy system (GFS) to this case. In addition, our method is able to learn minimum risk fuzzy rules from low quality data. We include a comprehensive description of the new algorithm and discuss some issues about its fuzzy-valued fitness function. A synthetic problem, plus two real-world datasets, are used to evaluate our proposal.

1 Introduction

In the context of GFSs, fuzzy statistics and fuzzy logic have complementary functions. On the one hand, GFSs depend on fuzzy rule-based systems (FRBS), that deal with fuzzy logic and "IF-THEN" rules [3]. These FRBSs use fuzzy sets to describe subjective knowledge about a classifier or a regression model, which otherwise accept crisp inputs and produce crisp outputs. On the other hand, fuzzy statistics use fuzzy sets to represent imprecise knowledge about the data [1]. In a broad sense, we could say that FRBSs process crisp data with fuzzy algorithms, while fuzzy statistics process fuzzy data with crisp algorithms.

In order to process fuzzy data with an algorithm which is also described in fuzzy logic terms, we need to combine fuzzy logic and fuzzy statistics [14]. Up to date, there have been few bridges between GFSs and fuzzy statistics [10]. In this paper, we continue our prior work about generalizing Genetic Fuzzy Classifiers to low quality data [8], and introduce the minimum risk problem, where the cost of misclassifying an object depends on a matrix of weights, and the classifier system does not optimize the training error but a loss function that depends on the mentioned matrix [5].

The use of GFSs for obtaining minimum risk classifiers is scarce. Some works have dealt with the concept of "false positives" [9,13] or taken into account the confusion matrix in the fitness function [15]. There are also some works in fuzzy ordered classifiers [12], where an ordering of the class labels defines, in a certain sense, a risk function different than the training error. Minimum-risk classification is also related to the use of costs in classification of imbalanced data [4]. However, up to our knowledge, the

* This work was supported by the Spanish Ministry of Education and Science, under grants TIN2008-06681-C06-04, TIN2007-67418-C03-03, and by Principado de Asturias, PCTI 2006-2009.

E. Corchado et al. (Eds.): HAIS 2009, LNAI 5572, pp. 654–661, 2009.

inclusion of the matrix of misclassification costs in the fitness function has been part of GFSs neither in crisp nor low quality datasets.

The structure of this paper is as follows: in the next section we generalize a crisp GFS, defined in [7], to the minimum risk problem. In Section 3 we extend this last algorithm to imprecise data. In Section 4 we evaluate the generalized algorithm in both crisp and imprecise datasets. The paper finishes with the concluding remarks, in Section 5.

2 Minimum Risk Genetic Fuzzy Classifiers with Crisp Data

The GFS in Figure 1 is a simple Cooperative-Competitive algorithm [6], defined in [7]. Each chromosome codifies the antecedent of a single rule; the consequents are not part of the genetic individuals. Instead, the function “assignConsequent” (line 6) determines the class label that matches an antecedent for a maximum confidence. A second function “assignFitness,” in turn, determines the winner rule for each object in the training set and increments the fitness of the corresponding individual if its consequent matches the class of the object. These last functions are described in Figures 2 and 3. The sum of the fitness of all the individuals is the number of correctly classified training examples.

```
function GFS
1      Initialize population
2      for iter in {1, . . . , Iterations}
3         for sub in {1, . . . , subPop}
4            Select parents
5            Crossover and mutation
6            assignConsequent(offspring)
7         end for sub
8         Replace the worst subPop individuals
9         assignFitness(population,dataset)
10     end for iter
11     Purge unused rules
return population
```

Fig. 1. Outline of the GFS that will be generalized [7]. Each chromosome codifies one rule. The fitness of the classifier is distributed among the rules at each generation.

Conversely, the minimum risk problem is based on a matrix of misclassification costs. We will assume that this matrix contains values between 0 and 1, and its diagonal is zero (because the cost of correctly classifying an object is null). The cost matrix needs not to be symmetric. Given a training set $\{(x_i, c_i)\}_{i=1\dots N}$, the objective of the classifier is to minimize the risk

$$\sum_{i=1}^{N} \text{loss}_i = \sum_{i=1}^{N} \text{cost}[\text{class}(x_i)][\text{c}_i] \tag{1}$$

where “class(x_i)” is the output of the FRBS in the i-th example. When all the costs but the diagonal are 1, this loss coincides with the training error.

Two changes are required in the GFS for coping with the cost matrix: (a) The assignment of the best consequent not only depends on the compatibility of the rule with the

```
function assignConsequent(rule)
1    for example in {1, ..., N}
2        m = membership(Antecedent,example)
3        weight[class(example)] =
             weight[class(example)] + m
4    end for example
5    mostFrequent = 0
6    for c in {1, ..., N_C}
7        if (weight[c]>weight[mostFrequent])
8            then mostFrequent = c
9        end if
10   end for c
11   Consequent = mostFrequent
return rule
```

```
function assignConsequentCost(rule)
1    for c in {1, ..., N_C}
2        weight[c] = 0
3        for example in {1, ..., N}
4            m = membership(Antecedent,example)
5            weight[c] = weight[c] +
                 m ∧ (1-cost[c,class(example)])
6        end for example
7    end for c
8    lowestLoss = 0
9    for c in {1, ..., N_C}
10       if (weight[c]>weight[lowestLoss])
11           then lowestLoss = c
12       end if
13   end for c
14   Consequent = lowestLoss
return rule
```

Fig. 2. Left part: Training error-based algorithm. The consequent of a rule is not codified in the GA, but it is assigned the most frequent class label, between those compatible with the antecedent of the rule [7]. Right part: Cost-based algorithm. The compatibility between antecedent and consequent is affected by the cost matrix (line 5); the consequent that produces the lowest loss in the example is chosen.

```
function assignFitness(population,dataset)
1    for example in {1, ..., N}
2        wRule = winner rule
3        if (consequent(wRule)==class(example)) then
4            fitness[wRule] = fitness[wrRule] + 1
5        end if
6    end for example
return fitness
```

```
function assignFitnessCost(population,dataset)
1    for example in {1, ..., N}
2        wRule = winner rule
3        fitness[wRule] = fitness[wRule] +
             1 - cost[consequent[wRule],class(example)]
4    end for example
return fitness
```

Fig. 3. Left: Training error-based: The fitness of an individual is the number of examples that it classifies correctly. Single-winner inference is used, thus at most one rule changes its fitness when the rule base is evaluated in an example [7]. Right: Risk-based: The fitness of an individual is the number of examples minus the loss in eq. (1), according to the cost table. In line 3 we can see how the fitness of the rule includes a value of the cost table, depending on the output of the example and the consequent of the rule.

example, but also on the value "cost[c,class(example)]," which is the cost of assigning the class "c" to the training data number "example" (see Figure 2, right part, line 5) and (b) the fitness of the winner rule is increased an amount that depends on the cost of assigning the label in the consequent to the example (see Figure 3, right part, line 3).

3 Minimum Risk Genetic Fuzzy Classifiers with Low Quality Data

In this section we will extend the algorithm defined in the preceding section to low quality data, codified by intervals or fuzzy sets. Let us consider the fuzzy case first, which we will introduce with the help of an example. Consider the FRBS that follows:

$$\begin{array}{l}\textbf{if } \text{TEMPERATURE } \textbf{is } \text{COLD } \textbf{then } \text{CLASS } \textbf{is } \text{A} \\ \textbf{if } \text{TEMPERATURE } \textbf{is } \text{WARM } \textbf{then } \text{CLASS } \textbf{is } \text{B}\end{array} \quad (2)$$

where the membership functions of "COLD" and "WARM" are, respectively, COLD(t) $= 1-t/100$ and WARM$(t) = 1-$COLD(t), for $t \in [0, 100]$. Let the actual temperature be $t = 48$, thus the degrees of truth of the two rules are 0.52 and 0.48, respectively, and the class of the object is the fuzzy set $0.52/A + 0.48/B$. Applying the winner-takes-all approach, $0.52 > 0.48$ and the output of the FRBS is "A."

Now suppose that we do not have a precise knowledge about the temperature: we perceive the triangular fuzzy set $\widetilde{T} = [45; 48; 51]$. Our information about the class of the object is limited to the fuzzy set

$$\widetilde{\text{class}}(\widetilde{T})(c) = \sup\{\widetilde{T}(u) \mid \text{class}(u) = c\}, \quad c \in \{1, \ldots, N_c\}, \quad (3)$$

thus $\widetilde{\text{class}}([45; 48; 51]) = 0.533/A+0.464/B$. In words, the set $\widetilde{\text{class}}([45; 48; 51])$ contains the output of the FRBS for all the values compatibles with the fuzzy set $[45; 48; 51]$ (see [10] for a deeper discussion on the subject). It is immediate to see that, in this case, the loss in the classification of an example is also a fuzzy number. Let $\{(\widetilde{T}_i, c_i)\}_{i=1\ldots N}$ be an imprecise training set. The membership of the risk of the FRBS in the i-th example of this set is

$$\widetilde{\text{loss}}_i(x) = \sup\{\widetilde{\text{class}}(\widetilde{T}_i)(c) \mid x = \text{cost}[c][c_i]\} \quad x \in [0, 1]. \quad (4)$$

Observe that, if the training data is represented with intervals, then the loss is either a crisp number or a pair of values, as we will see in the implementation that is explained later in this section.

3.1 Computer Algorithm of the Generalized GFS

The fuzzy risk function defined in eq. (4) imposes certain changes in the procedures `assignConsequent` and `assignFitness`, that are detailed in Figure 4.

On the one hand, the crisp assigment of a consequent consisted in adding the terms $\widetilde{A}(x_i) \wedge (1 - \text{cost}[c][c_i])$, for every example (x_i, c_i) in the training set and all the values of "c". After that, the alternative with maximum weight was selected. The same approach can be used in the imprecise case, with the help of the fuzzy arithmetic. However, for determining the most compatible rule, we need to define an order in the set of weights of the rules. This order is expressed in the pseudocode by means of the operation "dominates" used in line 11. Generally speaking, we have to select one of the values in the set of nondominated confidences and use its corresponding consequent, thus we want an operator that induces a total order in the fuzzy subsets of $[0, 1]$ [2].

On the other hand, there are three ambiguities that must be resolved for computing the fitness of a rule (Figure 4, right part): (a) some different crisp values compatible the same example might correspond to different winner rules –line 5— (see also [8]), (b) these rules might have different consequents, thus we have to combine their losses –lines 7 to 14– and (c) we must assign credit to just one of these rules –lines 15 and 16.–

There are two other parts in the original algorithm that must be altered in order to use an imprecise fitness function: (a) the selection of the individuals in [7] is based on a

```
function assignImpConsequentCost(rule)
1   for c in {1,...,N_c}
2     weight[c] = 0
3     for example in {1,...,N}
4       m̃ = fuzmship(Antecedent,example)
5       weight[c] = weight[c] ⊕
          m̃ ∧ { cost[c][class(example)] }
6     end for example
7   end for c
8   mostFrequent = {1,...,N_c}
9   for c in {1,...,N_c}
10    for c_1 in {c+1,...,N_c}
11      if (weight[c] dominates weight[c_1]) then
12        mostFrequent = mostFrequent - { c_1 }
13      end if
14    end for c_1
15  end for c
16  Consequent = select(mostFrequent)
return rule
```

```
function assignImpFitnessCost(population,dataset)
1   for ex in {1,...,N}
2     for r in {1,...,M}
3       rule.m̃ = fuzmship(Antecedent[r],ex)
4     end for r
5     setWinRule = set of indices of
        nondominated elements of rule.m̃
6     setOfCons= set of consequents of setWinRule
7     deltaFit= 0
8     if ({class(ex)} == setOfCons and
          size(setOfCons)==1) then
9       deltaFit = {1}
10    else
11      if ({class(ex)}∩ setOfCons ≠ ∅) then
12        deltaFit = deltaFit
            ∪{ 1 - cost[c][class(ex)] | c ∈ setOfCons }
13      end if
14    end if
15    Select winRule ∈ setWinRule
16    fitness[winRule] = fitness[winRule] ⊕ deltaFit
17  end for ex
return fitness
```

Fig. 4. Left part: assignment of a consequent with costs and imprecise data. We induce a total order in the fuzzy subsets of $[0, 1]$ for selecting the best weight. The function `fuzmship` in line 4 computes the intersection of the antecedent and the fuzzy input. Right part: Generalization of the function "assignFitness" to imprecise data. The set of winner rules is determined in line 5 (see [8] for further details). The set of costs of these rules is found in lines 7-14 and one rule receives all the credit in lines 15-16.

tournament, that depends on a total order on the set of fitness values. And (b) the same happens with the removal of the worst individuals. In both cases, we have used a fuzzy ranking that defines a total order [2]. We leave for future works the application of a multicriteria genetic algorithm similar to those used in our previous works in regression modeling [11].

4 Numerical Results

First, we have evaluated the crisp cost-based algorithm in a synthetic dataset, with known Bayesian risk. Second, the GFS for low quality data has been compared with the results in [8], using the two real-world interval-valued datasets proposed in that reference. All the experiments have been run with a population size 100, probabilities of crossover and mutation of 0.9 and 0.1, respectively, and limited to 200 generations. The fuzzy partitions of the labels are uniform and their size is 3, except when mentioned otherwise.

4.1 Synthetic Dataset

The decision surfaces obtained by the training error-based GFS and the crisp version of the GFS in this paper, applied to the Gauss dataset described in [8], are plotted in Figure 5.

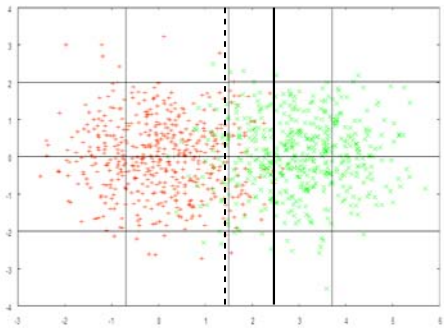

Fig. 5. Decision surfaces of the training error-based GFS (dashed) and the cost-based GFS (continuous). The cost based GFS has a small number of false positives.

Table 1. Results of the algorithm in Section 2 GFS in the synthetic dataset "Gauss"

	Train Error	Test Error	Train Risk	Test Risk
cost-based	0.441	0.437	0.013	**0.013**
training error-based	0.084	**0.086**	0.038	0.041

COST	
0	0.03
0.9	0

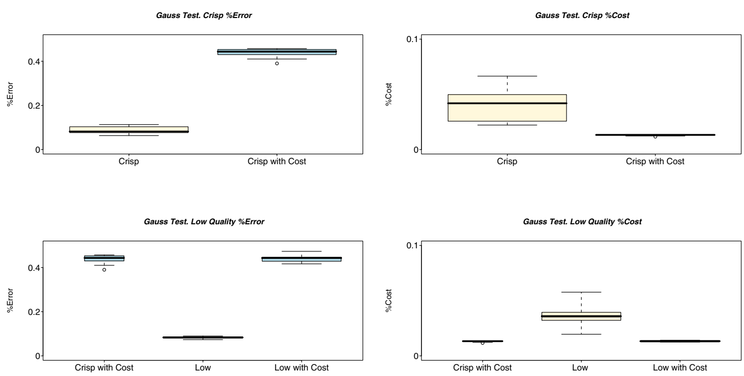

Fig. 6. Boxplots illustrating the dispersion of 10 repetitions of crisp GFS with costs, low quality data-based GFS with training error, and low quality data GFS with costs in the problem "Gauss", with 4 labels/partition. Upper row: Crisp GFS. Lower row: Low Quality-based GFS. Left parts: test error. Right parts: expected risk.

It can be observed that the cost-based classifier system (the cost matrix is defined in the right part of Table 1) produces a small number of false positives, but a much higher misclasification rate. The obtained decision surface is compatible with the theoretical surface derived from the probability density of the sample.

The first row of numbers in Table 1 contains the results of the cost-based GFS. The results of the training error-based GFS are in the second row. We have also included boxplots of the compared results for this problem and the crisp GFS with costs, the low quality data-based GFS with training error, and low quality data GFS with costs, in Figure 6.

4.2 Real World Datasets

We have used the real-world datasets "Screws-50" and "Dyslexia-12", described in [8], in combination with the cost matrices in the right parts of Tables 2 and 3. The cost matrix in the problem "Dyslexia-12" has been designed by an expert in the diagnosis of dyslexia; the costs of the problem "Screws-50" were selected at random.

Table 2. Means of 10 repetitions of the generalized GFS for the imprecise datasets "Screws-50"

% ERROR	Crisp		Low Quality	
Dataset	Train	Test	Train	Test
training error-based	0.077	**0.379**	[0.086,0.105]	**[0.350,0.379]**
cost-based	0.290	0.466	[0.189,0.207]	[0.406,0.425]
RISK	Crisp		Low Quality	
Dataset	Train	Test	Train	Test
training error-based	0.091	0.302	[0.057,0.073]	[0.214,0.236]
cost-based	0.097	**0.246**	[0.051,0.052]	**[0.190,0.207]**

COST		
0	0.90	0.90
0.15	0	0.85
0.40	0.20	0

Table 3. Means of 10 repetitions of the generalized GFS for the imprecise datasets "Dyslexia-12"

% ERROR	Crisp		Low Quality	
Dataset	Train	Test	Train	Test
training error-based	0.664	**0.675**	[0.512,0.673]	**[0.524,0.657]**
cost-based	0.648	0.726	[0.382,0.641]	[0.407,0.666]
RISK	Crisp		Low Quality	
Dataset	Train	Test	Train	Test
training error-based	0.251	0.235	[0.346,0.482]	[0.367,0.489]
cost-based	0.177	**0.225**	[0.259,0.438]	**[0.274,0.486]**

COST			
0	0.20	0.80	0.90
0.10	0	0.35	0.35
0.25	0.15	0	0.45
0.30	0.20	0.10	0

The real-world datasets show mixed results, that are mostly coherent with the cost matrix, albeit less conclusive than in the synthetic case. The average risk of the cost-based classifiers is better than the risk of the error-based GFSs. However, that difference is not significant.

5 Concluding Remarks and Future Work

The minimum risk problem posed in this paper has an immediate application to certain problems of medical diagnosis. We have briefly introduced the problem of diagnosis of dyslexia, where the professional suggested us a cost matrix. However, we plan to study more flexible cases, where the cost matrix is defined in linguistic terms.

On a different subject, our algorithm performed as expected in synthetic datasets, but the results were not conclusive in real-world datasets, and this suggests us that more work is needed to extend more modern GFSs than the simple algorithm used in this paper.

References

1. Bertoluzza, C., Gil, M.A., Ralescu, D.A. (eds.): Statistical Modeling, Analysis and Management of Fuzzy Data. Springer, Heidelberg (2003)
2. Bortolan, G., Degani, R.: A review of some methods for ranking fuzzy subsets. Fuzzy Sets and Systems 15, 1–19 (1985)
3. Cordón, O., Herrera, F., Hoffmann, F., Magdalena, L.: Genetic fuzzy systems. In: Evolutionary tuning and learning of fuzzy knowledge bases. World Scientific, Singapore (2001)
4. Fernandez, A., Garcia, S., del Jesus, M.J., Herrera, F.: A Study of the Behaviour of Linguistic Fuzzy Rule Based Classification Systems in the Framework of Imbalanced Data Sets. Fuzzy Sets and Systems 159(18), 2378–2398 (2008)
5. Hand, D.J.: Discrimination and Classification. Wiley, Chichester (1981)
6. Herrera, F.: Genetic Fuzzy Systems: Taxonomy, Current Research Trends and Prospects. Evolutionary Intelligence 1, 27–46 (2008)
7. Ishibuchi, H., Nakashima, T., Murata, T.: A fuzzy classifier system that generates fuzzy if-then rules for pattern classification problems. In: Proc. of 2nd IEEE CEC, pp. 759–764 (1995)
8. Palacios, A., Sánchez, L., Couso, I.: A baseline genetic fuzzy classifier based on low quality data. IFSA-EUSFLAT (2009) (Submitted)
9. Pulkkinen, P., Hytönen, J., Koivisto, H.: Developing a bioaerosol detector using hybrid genetic fuzzy systems. Engineering Applications of Artificial Intelligence 21(8), 1330–1346 (2008)
10. Sánchez, L., Couso, I.: Advocating the use of imprecisely observed data in genetic fuzzy systems. IEEE Transactions on Fuzzy Systems 15(4), 551–562 (2007)
11. Sánchez, L., Otero, J., Couso, I.: Obtaining linguistic fuzzy rule-based regression models from imprecise data with multiobjective genetic algorithms. Soft Computing 13(5), 467–479 (2008)
12. Van Broekhoven, E., Adriaenssens, V., De Baets, B.: Interpretability-preserving genetic optimization of linguistic terms in fuzzy models for fuzzy ordered classification: An ecological case study. International Journal of Approximate Reasoning 44(1), 65–90 (2007)
13. Verschae, R., Del Solar, J.R., Köppen, M., Garcia, R.V.: Improvement of a face detection system by evolutionary multi-objective optimization. In: Proc. HIS 2005, pp. 361–366 (2005)
14. Wu, B., Sun, C.: Interval-valued statistics, fuzzy logic, and their use in computational semantics. Journal of Intelligent and Fuzzy Systems 1–2(11), 1–7 (2001)
15. Teredesai, A., Govindaraju, V.: GP-based secondary classifiers. Pattern Recognition. 38(4), 505–512 (2005)