Boosting of fuzzy rules with low quality data

ANA M. PALACIOS[1⋆], LUCIANO SÁNCHEZ[1†], INÉS COUSO[2‡]

[1] *Departamento de Informática, Universidad de Oviedo*
[2] *Departamento de Estadística e I.O. y D.M, Universidad de Oviedo*
33071 Gijón, Asturias, Spain

Received 07 January 2011

An extension of the Adaboost algorithm is proposed for obtaining fuzzy rule based classifiers from imprecisely perceived data. Isolated fuzzy rules are regarded as weak learners, and knowledge bases as ensembles. Rules are iteratively added to a base, and the search of the best rule at each iteration is carried out by a genetic algorithm driven by a fuzzy fitness function. The successive weights of the instances are also fuzzy, however each rule is assigned a crisp number of votes, interpreted as degrees of importance of these rules. The results of the new algorithm are compared to those of other genetic algorithms for low quality data, in both accuracy and linguistic qualilty.

Key words: Genetic Fuzzy Systems, Low Quality Data, Boosting, Genetic Algorithms, Fuzzy Rule-based Classifiers, Vague Data

1 INTRODUCTION

Boosting algorithms combine different classifiers for obtaining an ensemble that performs better than any of its components [4]. These algorithms repeatedly invoke a learning algorithm to successively generate a committee of simple, inaccurate classifiers. Each time a new simple classifier is added to this ensemble, the examples in the training set are re-weighted so that future

⋆ email: palaciosana@uniovi.es
† email: luciano@uniovi.es
‡ email: couso@uniovi.es

classifiers will focus on the most difficult examples, and a voting strength is assigned to the classifier. The number of votes a classifier is given depends on the confidence in its classification accuracy, as measured on the training set.

Fuzzy Rule Based Classifiers (FRBS) can also be regarded as ensembles, where each fuzzy rule is matched to one of the mentioned simple classifiers. When an appropriate reasoning scheme is used, it has been shown that the mechanism used for obtaining the output of a FRBS, given an input value, can be assimilated to a voting process [13], and the voting strength of each simple classifier is the product of the compatibility between the antecedent of the corresponding fuzzy rule and the degree of certainty of its consequent. This interpretation has been used before, and the Adabost algorithm has been applied to the learning of fuzzy rules from data [7][10][12], in combination with a Genetic Algorithm (GA). The resulting learning algorithm is a Genetic Fuzzy System (GFS) that shares certain properties with Iterative Rule Learning approaches [6]. Furthermore, after the work of Friedman [5], this process has been regarded as a forward stepwise estimation of the statistical parameters defining a logit transform of a Generalized Additive Model, giving rise to the LogitBoost algorithm [5] and its genetic extension for learning FRBS from data [14].

The use of boosting techniques for learning fuzzy rules from data is therefore a well known technique, whose main virtues are a fast learning and accurate results, but this use is not free from problems. The first drawback of the employment of Adaboost for learning FRBS is in the use of a voting-based inference, that makes the linguistic interpretation of the Knowledge Base (KB) harder. This problem was addressed in [19], where a new algorithm was proposed for which the degrees of confidence in the consequents of the rules were iteratively adjusted until the results produced by a "winner-takes-all" type of inference were similar to that of voting. Nevertheless, there are other difficulties that are still open. A second issue with the application of Adaboost to the learning of fuzzy rules lies in the requirements imposed to the data, that must be precisely perceived: being based in Generalized Additive Models, Adaboost or Logitboost can cope with random uncertainty, but they are not able to process data when a combination of epistemic and random uncertainties is present [8]. This means, for instance, that we cannot exploit the information in vague datasets, comprising censored, interval-valued or fuzzy instances, among others. Because of these reasons, this paper contains a new proposal for applying the Adaboost algorithm for learning FRBS from low quality data [20], including both problems with imprecisely perceived fea-

tures and problems with a partial lack of knowledge about the class labels attributed to some instances.

The organization of this work is as follows: the following section, briefly introduces the Adaboost algorithm, describes the type of fuzzy classifiers that are boosted and recall a method for boosting approximate fuzzy rule bases [3]. In Section 3, we propose an extension of this algorithm for interval and fuzzy data. In Section 4 some properties of the proposed algorithms are evaluated on a number of data sets and compared to that of other GFS for low quality data. In Section 5 some concluding remarks and future works are discussed.

2 ADABOOST AND FUZZY RULE BASED CLASSIFIERS

At this point we introduce the basic notation employed throughout the paper. Let $\mathbf{X}$ be the feature space, and let $\mathbf{x}$ be a feature vector $\mathbf{x} = (x_1, \ldots, x_n) \in \mathbf{X}$. Let p be the number of classes. The training set is a sample of m classified examples $(\mathbf{x}_i, y_i)$, where $\mathbf{x}_i \in \mathbf{X}$, $1 \leq y_i \leq p$, $1 \leq i \leq m$.

The KB of the FRBS comprises N rules. The antecedents of the rules are logical combinations of fuzzy logic asserts, whose degrees of truth are modeled by N fuzzy subsets $A^j \in \mathcal{F}(\mathbf{X})$, forming a fuzzy partition $\mathcal{A} = \{A^j\}_{j=1\ldots N}$ of the feature space. A fuzzy rule based classifier is therefore defined by means of a fuzzy relationship defined on $\mathcal{A} \times \{1, \ldots, p\}$. Values of this relationship describe the degrees of compatibility among the fuzzy subsets of the feature space collected in $\mathcal{A}$, and each of the classes. In other words, for every antecedent A^j there are p numbers between 0 and 1 that represent the confidence in the assertion "All elements in the fuzzy set A^j belong to class number k". Values close to 1 indicate "high confidence," and values close to 0 denote "absence of knowledge about the assertion."

In this paper, we will translate the former fuzzy relationship into linguistic statements by combining p terms

$$\text{compatibility}(A^j, c_k) = s_k \quad k = 1, \ldots p,$$

into a single sentence, as follows:

$$\text{if } \mathbf{x} \text{ is } A^j \text{ then truth}(c_1) = s_1^j \text{ and } \cdots \text{ and truth}(c_p) = s_p^j.$$

The antecedents A^j are decomposed in a Cartesian product of fuzzy sets defined over each feature, $A^j = A_1^j \times A_2^j \times \ldots \times A_n^j$, thus the rules are

$$\begin{gathered}\text{if } x_1 \text{ is } A_1^j \text{ and } \ldots \text{ and } x_n \text{ is } A_n^j \\ \text{then truth}(c_1) = s_1^j \text{ and } \cdots \text{ and truth}(c_p) = s_p^j.\end{gathered}$$

An instance $\mathbf{x}$ is assigned to the class

$$\arg\max_{k=1,\ldots,p} \bigvee_{j=1}^{N} A^j(\mathbf{x}) \wedge s_k^j \tag{1}$$

where "$\wedge$" is the product and "$\vee$" is the arithmetic sum, so called "maximum voting scheme" [11].

2.1 The Adaboost algorithm

Let us define a set $\{g^1, g^2, \ldots, g^N\}$ of simple, but possibly unreliable binary classifiers. Boosting consists in combining these low quality classifiers (so called "weak hypotheses" in the boosting literature) with a voting scheme to generate an overall classifier that performs better than any of its individual constituents alone. It has been shown that a fuzzy rule can be regarded as a particular case of a weak hyphothesis, and a fuzzy rule base can be interpreted as a weighted combination of weak hypotheses.

Weak hypotheses take feature values as input and produce both a class number as well as a degree of confidence in the given classification. In two-classes problems, these two outputs are encoded by a single real number, $g^j(\mathbf{x}) \in \mathbf{R}$, whose sign is interpreted as the label of $\mathbf{x}$ and whose absolute value is interpreted as the confidence in the classification. The higher this value the more confidence is given to the classification. AdaBoost is intended to produce a linear threshold of all hypotheses:

$$\text{sign}\left(\sum_{j=1}^{N} \alpha^j g^j(\mathbf{x})\right). \tag{2}$$

An outline of the Adaboost algorithm is shown in Figure 2 in the Appendix. Observe that Adaboost can operate with any learning algorithm that generates a confidence rated classifier on a given weighted data set. There are different algorithms for assigning the number of votes to a weak hypothesis, and for adjusting the weights of the examples. For example, in confidence-rated Adaboost [23] the number of votes of the weak hypothesis g^h is given by the value α^h that minimizes the following function:

$$Z(\alpha) = \sum_{i=1}^{m} w_i \exp(-\alpha y_i g^h(\mathbf{x}_i)) \tag{3}$$

and the weights w_i of the examples are updated according to

$$w_i \leftarrow w_i \exp(-\alpha^h y_i g^h(\mathbf{x}_i))/v \tag{4}$$

where v is a normalization factor such that $\sum w_i = 1$. There are analytical approximations and heuristics that may replace this formula in specific problems.

2.2 Boosting fuzzy rules in binary problems

For the sake of simplicity, we restrict the discussion for the time being to two-classes problems $y_i \in \{-1, 1\}$. The space of weak hypotheses is, in this case, the product space of the fuzzy partition and the class labels, $\mathcal{A} \times \{-1, 1\}$. Given the results in the preceding subsection, the fitness of a fuzzy rule is

$$\text{fitness}(\text{ if } \mathbf{x} \text{ is } A^j \text{ then } c^j) = \sum_i w_i \exp(-y_i c^j A^j(\mathbf{x}_i)) \tag{5}$$

and the number of votes of this rule is the value of α minimizing the following expression [22]:

$$Z(\alpha) = \sum_i w_i \exp(-\alpha y_i A^j(\mathbf{x}_i)). \tag{6}$$

$Z(\alpha)$ is convex except for the case in which the rule antecedent does not cover any negative examples; for avoiding this and other numerical instabilities, a term that penalizes large values of α is added

$$Z(\alpha) = \sum_i w_i \exp(-\alpha y_i A^j(\mathbf{x}_i)) + \sum_{i:A^j(\mathbf{x}_i)=0} w_i \exp(|\alpha\epsilon|) \tag{7}$$

where ϵ is determined by hand.

The weights w_i of all the instances are recalculated each time a new weak learner is added to the ensemble. The weights of correctly classified instances are lowered, and these of misclassified examples are increased, using the expression that follows:

$$w_i \leftarrow \frac{w_i \exp(-\alpha^j y_i A^j(\mathbf{x}_i))}{\sum_i w_i \exp(-\alpha^j y_i A^j(\mathbf{x}_i))} \tag{8}$$

Notice, that an instance is never completely removed unless $\alpha^j \to \infty$, which is prevented by the penalty term in eq. (7).

2.3 Boosting fuzzy rules in multiclass problems

We have mentioned that the space of weak hypothesis in two-classes problems is $\mathcal{A} \times \{c_1, c_2\}$. That is to say, the j-th iteration of the learning algorithm searches for a pair (A^j, c^j), which are parameters of the rule

$$\text{if } \mathbf{x} \text{ is } A^j \text{ then } c^j, \tag{9}$$

for which the fitness value is maximum. This value was defined in Eq. (5), which we rewrite here as follows:

$$\begin{array}{c}\text{fitness}(\text{if } \mathbf{x} \text{ is } A^j \text{ then } c^j) = \\ \sum_{i:y_i=c^j} w_i \exp(A^j(\mathbf{x}_i)) \\ +\sum_{i:y_i\neq c^j} w_i \exp(-A^j(\mathbf{x}_i)).\end{array} \tag{10}$$

Once the pair (A^j, c^j) maximizing this fitness function is found, the rule is assigned a number of votes s^j (see Eq. (6)), thus its final linguistic writing is:

$$\text{if } \mathbf{x} \text{ is } A^j \text{ then truth}(c^j) = s^j. \tag{11}$$

For extending this scheme to multiclass problems [7] the initial problem is transformed into p binary problems, where the instances of the k-th binary problem are re-labelled:

$$y_i^{(p)} = \begin{cases} 1 & y_i = p \\ -1 & y_i \neq p. \end{cases} \tag{12}$$

The solution of the k-th problem comprises fuzzy rules with this form:

$$\begin{array}{l}\text{if } x_1 \text{ is } A_1^j \text{ and } \ldots \text{ and } x_n \text{ is } A_n^j \\ \text{then tr}(\text{class} = c_k) = s_1^{jk} \text{ and tr}(\text{class} \neq c_k) = s_{-1}^{jk}.\end{array} \tag{13}$$

These rules can be replaced by their equivalent p-classes consequents, by substituting the "class $\neq c_k$" label with the corresponding set of class labels of the multi-class problem, i.e.

$$\begin{array}{l}\text{if } x_1 \text{ is } A_1^j \text{ and } \ldots \text{ and } x_n \text{ is } A_n^j \text{ then} \\ \text{tr}(c_1, \ldots, c_k, \ldots, c_p) = (s_{-1}^{jk}, \ldots, s_{-1}^{jk}, s_1^{jk}, s_{-1}^{jk}, \ldots, s_{-1}^{jk}).\end{array} \tag{14}$$

We obtain p fuzzy rule bases, that are merged into the overall multi-class rule base, as shown in Figure 3 in the Appendix.

3 AN EXTENSION OF THE ADABOOST ALGORITHM TO LEARN FUZZY RULE BASED CLASSIFIERS FROM LOW QUALITY DATA

We will represent the epistemic uncertainty in the data by means a nested set of confidence intervals, that provide us with the same information about the unknown value as a possibility distribution for which the α-cuts of its associated fuzzy membership function are confidence intervals with level $1-\alpha$, as explained in [20]. Representing the imprecision by means of sets of possible

values implies that the output of the FRBS might not be completely determined and generally speaking, it will be a fuzzy subset of the class labels. From the foregoing it can be deduced that the fitness of a rule should also be a set, however extending Eq. 5 is not trivial. Different decisions can be taken that influence the accuracy of the method and also its computational cost.

Let us use an illustrative example for introducing the problems that arise when we regard the computation of the output of an FRBS for imprecise data as a voting-based committe. Suppose that we are given the KB that follows:

if $x < 1.5$ then truth(class is A)= 0.4 and truth(class is B)= 0.8
if $x \in [1, 2]$ then truth(class is A)= 0.8 and truth(class is B)= 0.1
if $x > 2$ then truth(class is A)= 0.2 and truth(class is B)= 0.6

and the input is

$x < 1.8$.

If we determine first the set of compatibilities of each rule with the imprecise example and then add the sets of votes, the situation is as follows:

Rule	Votes for class 'A'	Votes for class 'B'
Rule # 1:	{0,0.4}	{0,0.8}
Rule # 2:	{0,0.8}	{0;0.1}
Rule # 3:	0	0
Total	{0, 0.4, 0.8, 1.2}	{0, 0.1, 0.8, 0.9}

In words, the output of the classifier is the set of classes {A,B}, because we cannot state that any element of {0, 0.4, 0.8, 1.2} is higher than any element of {0, 0.1, 0.8, 0.9} neither the opposite. However, this course of reasoning will not always produce the most specific answer. Observe that, if the actual value of x is lower than 1.5, rules 1 and 2 are true, thus the number of votes of classes 'A' and 'B' are 1.2 and 0.9. If the value of x is between 1.5 and 1.8, the number of votes are 0.8 and 0.1. In either case, the object should have been assigned the class 'A'.

With this example we have shown that regarding a KB as an ensemble is not immediate when the data are imprecise: the most specific output of the classifier, when the input is a crisp set, is the set of classes

$$\text{class}(\overline{X}) = \{\arg \max_{k=1\ldots p} \sum_{j=1}^{N} A^j(x) s_k^j \mid x \in \overline{X}\}. \tag{15}$$

However, if we regard the KB as an ensemble of weak classifiers, where each

one of them is assigned a set-valued, number of votes, the output is the set

$$\text{bclass}(\overline{X}) = \underset{k=1\ldots p}{\text{notdom}} \left\{ \bigoplus_{j=1}^{N} s_k^j \odot \{A^j(x) \mid x \in \overline{X}\} \right\} \tag{16}$$

where the operator "notdom" means

$$\underset{k=1\ldots p}{\text{notdom}}\{V_k\} = \{q \mid V_q \preceq V_r, r = 1, \ldots, p\} \tag{17}$$

and the precedence between set-valued votes is

$$A \prec B \iff a < b \text{ for all } a \in A, b \in B \tag{18}$$

$$A \parallel B = \neg((A \prec B) \vee (B \prec A)) \tag{19}$$

$$A \preceq B = (A \prec B) \vee (A \parallel B). \tag{20}$$

Observe that

$$\text{class}(\overline{X}) \subseteq \text{bclass}(\overline{X}) \tag{21}$$

but the equality does not hold, in general.

If the data is fuzzy, the situation is similar. The most specific output of the classifier is the fuzzy set whose membership function is as follows:

$$\begin{aligned} \text{class}(\widetilde{X})(t) = \quad & \max\{\alpha \mid \\ & t = \arg\max_{k=1\ldots p} \textstyle\sum_{j=1}^{N} A^j(x) s_k^j \\ & \text{and } \widetilde{X}(x) \geq \alpha\}. \end{aligned} \tag{22}$$

In case we regard the KB as an ensemble of weak classifiers, where each one of them is assigned a fuzzy number of votes, the output is in turn the fuzzy set

$$\text{bclass}(\widetilde{X})(t) = \max\{\alpha \mid t \in \text{bclass}([\widetilde{X}]_\alpha)\}. \tag{23}$$

Summarizing, when the data is imprecise, the most voted option of the ensemble is not known but our information about it is a normal fuzzy set. If the KB is furthermore regarded as an ensemble, thus the number of votes of each rule are independently computed, the fuzzy set describing the result of the classification is a superset of the result given by Eq. (22). This loss of specificity in the view of a KB as an ensemble has to be taken into account in the definition of the fitness function.

3.1 Fitness of a rule with interval or fuzzy data

We propose to generalize the fitness function in Eq. (10) to interval-valued data with the set-valued function that follows, which is based on the bounds given in Eq. (16):

$$\begin{aligned}
&\overline{\text{fitness}}_{\{(\overline{\mathbf{x}}_i,\overline{y}_i)\}}(\text{if } \mathbf{x} \text{ is } A^j \text{ then } c^j) = \\
&\quad \oplus_{i:\overline{y}_i=\overline{c}^j} \overline{w}_i \odot \{\exp(A^j(x)) \mid x \in \overline{\mathbf{x}}_i\} \oplus \\
&\quad \oplus_{i:\overline{y}_i\cap\overline{c}^j=\emptyset} \overline{w}_i \odot \{\exp(-A^j(x)) \mid x \in \overline{\mathbf{x}}_i\} \oplus \\
&\oplus_{i:\overline{y}_i\neq\overline{c}^j,\, \overline{y}_i\cap\overline{c}^j\neq\emptyset} \overline{w}_i \odot \{\exp(\{A^j(x), -A^j(x)\}) \mid x \in \overline{\mathbf{x}}_i\}
\end{aligned} \tag{24}$$

Observe that we allow the use of set valued weights $\overline{w}_i$. In turn, applying the extension principle, the same function can be extended to fuzzy data. The membership function of the fuzzy fitness of a rule is

$$\begin{aligned}
&\widetilde{\text{fitness}}_{\{(\widetilde{\mathbf{x}}_i,\widetilde{y}_i)\}}(\text{if } \mathbf{x} \text{ is } A^j \text{ then } c^j)(t) = \\
&\max\{\alpha \mid t \in \overline{\text{fitness}}_{\{([\widetilde{\mathbf{x}}_i]_\alpha,[\widetilde{y}_i]_\alpha)\}}(\text{if } \mathbf{x} \text{ is } A^j \text{ then } c^j)\},
\end{aligned} \tag{25}$$

where t is a real number.

3.2 Issues with the fitness of a rule in multiclass problems

When extending the procedure seen in Section 2.3 for solving multi-class problems, the intermediate binary sets of data might contain unlabeled instances. Let us explain this with an illustrative example.

Consider the imprecise dataset that follows, comprising three crisp examples with set-valued labels. It is remarked that, in this context, an instance labelled "$\{c_1, c_2\}$" means that we are sure that the true class of the object is not c_3, but this knowledge cannot be further precised:

$(x_1, y_1) = (1, \{c_1, c_2\})$
$(x_2, y_3) = (2, \{c_1, c_3\})$
$(x_3, y_3) = (3, \{c_2, c_3\})$.

Following the procedure described in Section 2.3, this dataset will be decomposed in three binary problems. Let us generalize that procedure to imprecise instances by using the most specific set of labels that is compatible with the data, as follows:

Problem # 1
$(1, \{1, -1\})$
$(2, \{1, -1\})$
$(3, -1)$

Problem # 2
$(1, \{1, -1\})$
$(2, -1)$
$(3, \{1, -1\})$

Problem # 3
$(1, -1)$
$(2, \{1, -1\})$
$(3, \{1, -1\})$

Each one of these three datasets has two elements whose classes are $\{-1, 1\}$ and therefore the fitness does not depend on how they are labelled: for instance, for Problem # 1, no matter which rule between

$$\text{if x=1 then t(class = 1)=}s_1^{1,1}$$

and

$$\text{if x=1 then t(class } \neq \text{ 1)=}s_{-1}^{1,1}$$

is chosen, the same fitness is obtained (see Eq. 24).

In this case, a learning algorithm which is only guided by the optimization of the fitness function can produce any KBs formed by selecting one rule from each line that follows. We have grayed out the rules that do not appear in an arbitary selection, whose merging will be studied later.

Problem # 1
if x=1 then t(class = 1)=$s_1^{1,1}$ $\approx$ if x=1 then t(class $\neq$ 1)=$s_{-1}^{1,1}$
if x=2 then t(class = 1)=$s_1^{2,1}$ $\approx$ if x=2 then t(class $\neq$ 1)=$s_{-1}^{2,1}$
if x=3 then t(class $\neq$ 1)=$s_{-1}^{3,1}$

Problem # 2
if x=1 then t(class = 2)=$s_1^{1,2}$ $\approx$ if x=1 then t(class $\neq$ 2)=$s_{-1}^{1,2}$
if x=2 then t(class $\neq$ 2)=$s_{-1}^{2,2}$
if x=3 then t(class = 2)=$s_1^{3,2}$ $\approx$ if x=3 then t(class $\neq$ 2)=$s_{-1}^{3,2}$

Problem # 3
if x=1 then t(class $\neq$ 3)=$s_{-1}^{1,3}$
if x=2 then t(class = 3)=$s_1^{2,3}$ $\approx$ if x=2 then t(class $\neq$ 3)=$s_{-1}^{2,3}$
if x=3 then t(class = 3)=$s_1^{3,3}$ $\approx$ if x=3 then t(class $\neq$ 3)=$s_{-1}^{3,3}$

The merging of the selected rules is

if x=1 then t(1,2,3)=($s_1^{1,1}$+$s_{-1}^{1,3}$,$s_1^{1,2}$+$s_{-1}^{1,3}$,0)
if x=2 then t(1,2,3)=($s_1^{2,1}$+$s_{-1}^{2,2}$,0,$s_{-1}^{2,2}$+$s_1^{2,3}$)
if x=3 then t(1,2,3)=($s_{-1}^{3,3}$,$s_{-1}^{3,1}$+$s_1^{3,2}$+$s_{-1}^{3,3}$,$s_{-1}^{3,1}$).

Observe that the result of this arbitrary selection has assigned a non-null degree confidence to the class c_1 in the third rule, which obviously is not the best possible KB for this problem. We realize that there are other selections that achieve the objective, but our point is showing that there is a chance that the committee does not contain the proper rules unless the unlabeled instances in the intermediate problems are removed, as follows:

Problem # 1
if x=3 then t(class $\neq$ 1)=$s_{-1}^{3,1}$

Problem # 2
if x=2 then t(class $\neq$ 2)=$s_{-1}^{2,2}$

Problem # 3
if x=1 then t(class $\neq$ 3)=$s_{-1}^{1,3}$

whose merging is:

if x=1 then t(1,2,3)=($s_{-1}^{1,3}$,$s_{-1}^{1,3}$,0)
if x=2 then t(1,2,3)=($s_{-1}^{2,2}$,0,$s_{-1}^{2,2}$)
if x=3 then t(1,2,3)=(0,$s_{-1}^{3,1}$,$s_{-1}^{3,1}$).

Notwithstanding, if the unlabelled instances are removed then additional problems with the decomposition in binary problems will appear. Since the pruned individual problems do not longer contain information about where the removed instances were located, it may happen that the corresponding areas of the feature space are covered at the same time by different rules that negatively interact between themselves. This problem also happens with Iterative Rule Learning (IRL) algorithms [3], where it is solved by a simplification stage that it is not a part of the boosting algorithm. Therefore, we have decided to simplify the search and not to generate the intermediate datasets when working with multiclass problems, and propose instead to define the fitness of a rule with multiple consequents as a vector of values, and to define a lexicographic ordering between them. The fitness function we propose is as

follows:

$$\begin{aligned}
&\overline{\text{fitness}}_{\{(\overline{\mathbf{x}}_i,\overline{y}_i)\}}(\text{if } \mathbf{x} \text{ is } A^j \text{ then } (c_1,\ldots,c_p)) = \\
&\Big(\textstyle\bigoplus_{\overline{y_i}=c_1} \overline{w}_i \odot \exp(\{A^j(x) \mid x \in \overline{\mathbf{x}}_i\}) \oplus \\
&\textstyle\bigoplus_{c_1 \in \overline{y}_i, c_1 \neq \overline{y}_i} \overline{w}_i \odot \\
&\qquad \exp(\{-A^j(x), A^j(x) \mid x \in \overline{\mathbf{x}}_i\}) \oplus \\
&\textstyle\bigoplus_{c_1 \notin \overline{y}_i} \overline{w}_i \odot \exp(\{-A^j(x) \mid x \in \overline{\mathbf{x}}_i\}), \\
&\quad \vdots \\
&\textstyle\bigoplus_{\overline{y_i}=c_p} \overline{w}_i \odot \exp(\{A^j(x) \mid x \in \overline{\mathbf{x}}_i\}) \oplus \\
&\textstyle\bigoplus_{c_p \in \overline{y}_i, c_p \neq \overline{y}_i} \overline{w}_i \odot \\
&\qquad \exp(\{-A^j(x), A^j(x) \mid x \in \overline{\mathbf{x}}_i\}) \oplus \\
&\textstyle\bigoplus_{c_p \notin \overline{y}_i} \overline{w}_i \odot \exp(\{-A^j(x) \mid x \in \overline{\mathbf{x}}_i\})\Big).
\end{aligned} \tag{26}$$

In turn, the lexicographic precedence between two crisp fitness vectors $A = (a_1,\ldots,a_p)$ and $B = (b_1,\ldots,b_p)$ is as follows: let us first define two permutations of the set $\{1,\ldots,p\}$, denoted $(k_1^a,\ldots,k_p^a)$ and $(k_1^b,\ldots,k_p^b)$, such that $a_{k_1^a} \geq \ldots \geq a_{k_p^a}$ and $b_{k_1^b} \geq \ldots \geq b_{k_p^b}$. Then,

$$\begin{aligned}
A \prec B \iff &\exists m \in \{1,\ldots,p\} \mid \\
&(a_{k_q^a} = b_{k_q^b})\ \forall q \in \{1,\ldots,m-1\} \wedge (a_{k_m^a} > b_{k_m^b}).
\end{aligned} \tag{27}$$

The extension of both the fitness function and the lexicographic ordering to fuzzy values results from applying the extension principle and from replacing the comparisons between real numbers with a suitable fuzzy ranking. The expression of the fitness function, in the general case, is as follows:

$$\begin{aligned}
&\widetilde{\text{fitness}}_{\{(\widetilde{\mathbf{x}}_i,\widetilde{y}_i)\}}(\text{if } \mathbf{x} \text{ is } A^j \text{ then } (c_1,\ldots,c_p))(t) = \\
&\quad \max\{\alpha \mid t \in \\
&\quad \overline{\text{fitness}}_{\{([\widetilde{\mathbf{x}}_i]_\alpha,[\widetilde{y}_i]_\alpha)\}}(\text{if } \mathbf{x} \text{ is } A^j \text{ then } (c_1,\ldots,c_p))\},
\end{aligned} \tag{28}$$

where t denotes a p-dimensional vector of real values.

3.3 Assignment of weights to the consequent part

The extension of Eq. (7) to set-valued data and multiclass problems consists in assigning to the k-th class in the consequent of the rule a confidence equal to the value of α_k minimizing the set-valued function Z^k defined as follows:

$$\begin{aligned}
&\overline{Z}^k_{\{(\overline{\mathbf{x}}_i,\overline{y}_i)\}}(\alpha) = \\
&\quad \{ \textstyle\oplus_{i:c_k=\overline{y}_i} \overline{w}_i \odot \exp(-\alpha A^j(x)) \oplus \\
&\quad \textstyle\oplus_{i:c_k \notin \overline{y}_i} \overline{w}_i \odot \exp(\alpha A^j(x)) \oplus \\
&\quad \textstyle\oplus_{i:c_k \neq \overline{y}_i,\, c_k \in \overline{y}_i} \overline{w}_i \odot \exp(\{-\alpha A^j(x), \alpha A^j(x)\}) \\
&\quad \mid x \in \overline{\mathbf{x}}_i\}.
\end{aligned} \tag{29}$$

A numerical stabilization term

$$\bigoplus_{i:A^j(x)=0\ \forall x\in\overline{\mathbf{x}}_i} \overline{w}_i \exp(|\alpha\epsilon|) \tag{30}$$

with a suitable value of ϵ can also be added, if needed.

The extension of Eq. (29) to fuzzy data is as follows:

$$\widetilde{Z}^k_{\{(\widetilde{\mathbf{x}}_i,\widetilde{y}_i)\}}(\gamma)(t) = \max\{\alpha \mid t \in \overline{Z}^k_{\{([\widetilde{\mathbf{x}}_i]_\alpha,[\widetilde{y}_i]_\alpha)\}}(\gamma)\}. \tag{31}$$

In both the set-valued and fuzzy cases, the values of α_k can be found with a greedy algorithm that uses a precedence operator between fuzzy sets. However, this optimization is not as efficient as the Brent search used in the crisp version of the algorithm. Since the optimization must be launched each time a fitness value is computed, in this paper we have decided to approximate the value of α_k by the center of the set

$$\overline{\alpha}_k = \log(1 - E_k) - \log(E_k) \tag{32}$$

where E_k is a normalized weighted sum of the compatibilities of the antecedent of the j-th rule with the elements of the dataset whose class does not match the k-th term in the consequent:

$$E_k = K \odot \Big(\bigoplus\nolimits_{\{i:c_k\notin\overline{y}_i\}} \overline{w}_i \odot \{A^j(x) \mid x \in \overline{\mathbf{x}}_i\} \oplus \bigoplus_{\{i:c_k\neq\overline{y}_i\wedge c_k\in\overline{y}_i\}} \overline{w}_i \odot \{\{0, A^j(x)\} \mid x \in \overline{\mathbf{x}}_i\}\Big) \tag{33}$$

The normalization factor K is the inverse of the upper bound of the normalized weighted sum of the compatibilities of the antecedent of the j-th rule with all the elements of the dataset:

$$K = (\sum_i \max\{\overline{w}_i\} \cdot \max\{A^j(x) \mid x \in \overline{\mathbf{x}}_i\})^{-1}. \tag{34}$$

3.4 Assignment of weights to the examples

After the j-th rule is added, the weights of the instances are recomputed as follows:

$$\overline{w}'_i = \bigcup_{x\in\overline{\mathbf{x}}_i} w'_i(x) \tag{35}$$

where $w'_i(x) =$

$$K' \odot \overline{w}_i \odot \begin{cases} c_k = \overline{y}_i & \exp(-\alpha_k A^j(x)) \\ c_k \notin \overline{y}_i & \exp(\alpha_k A^j(x)) \\ \text{else} & \exp(\alpha_k A^j(x) \odot \{-1, 1\}) \end{cases} \tag{36}$$

and K' is a crisp normalization factor such that $\max \oplus_i \overline{w}'_i = 1$.

3.5 Some details of the genetic algorithm

Adaboost depends on a procedure that fits a weak learner to the weighted training set. If the vector-valued fitness we have proposed in this paper is to be used, learning a weak classifier reduces to finding the antecedent A^j that optimizes the fitness function explained in section 3.2, with respect to the lexicographic ordering defined in the same section. Since all possible values of the fitness can be compared between themselves, implementing this criterion amounts to redefining the meaning of the operator "less than" ($<$) in an ordinary GA; there is no need to use multicriteria techniques [21], thus we have used instead a standard generational scheme with a tournament-based selection.

Details of this algorithm can be found in [7]. Let us recall for the convenience of the reader the only part in this GA besides the fitness function (whose explanation has occupied most of this paper) which departs from a standard implementation: the coding of the fuzzy memberships.

Descriptive fuzzy rules are coded by an integer, which is the index j of the antecedent A^j in $\mathcal{A}$. This integer is encoded in turn, by a sequence of n numbers that refer to labels of the linguistic terms in the underlying fuzzy partitions. In addition to linguistic labels describing subsets of the domain of each variable, such as "LOW" and "HIGH", each linguistic variable includes a wild card label "ANY VALUE", with a membership degree of 1 across the entire universe of discourse.

The linguistic expression of a fuzzy rule that contains a wild card can be simplified, as illustrated by the following example: assume a classification problem with two features (weight,height), where height=$\{$low,high$\}$ and weight=$\{$light, heavy$\}$. The two linguistic variables are extended with a wild card term such that height=$\{$low, high, anyvalue$\}$ and weight=$\{$light, heavy, anyvalue$\}$. If the rule antecedent is described by "anyvalue$\times$heavy", the linguistic expression

if height is anyvalue and weight is heavy then t(1,2)=(0.4,0.8)

is simplified to

if weight is heavy then t(1,2)=(0.4,0.8).

This property of the genetic representation allows it to code general rules that only refer to a subset of all possible features, thus enabling the boosting algorithm to take advantage of feature selection.

The last rule is coded by the sequence (3,2), as "anyvalue" is the third linguistic term in the first linguistic variable, and "heavy" is the second term in the second linguistic variable. Observe that the class labels in the rule consequent are not part of the genetic codification, neither the confidences

degrees in the consequents are.

4 NUMERICAL RESULTS

In this section we include the first results of the implementation of Adaboost for learning fuzzy rules from low quality data, applied to the imprecise datasets "Diagnosis of the Dyslexic" [16] and "Athletics at Oviedo University" [17]. We include first a brief description of these datasets and then we discuss the compared results of the application of the new method to these problems.

4.1 Description of the datasets

The group of datasets "Diagnosis of the Dyslexic" is based on the early diagnosis (ages between 6 and 8) of schoolchildren of Asturias (Spain). These datasets are samples of a multiclass problem, where the inputs are the answers to the tests in the Table 1 and whose outputs are the diagnosis of a psychologist ("No dyslexic", "Control and revision", "Dyslexic", "Inattention, hyperactivity or other problems"). We have considered three databases, codenamed "Dyslexic-12-12", "Dyslexic-11-01" and "Dyslexic-11-12":

- "Dyslexic-12-12" has imprecision in both the input and the output, and missing values. There are 12 features and three classes; individuals of the category "control and revision", are included in class "Dyslexic".
- "Dyslexic-11-01" and "Dyslexic-11-12" have crisp inputs and imprecise outputs. Individuals of the category "control and revision" are assigned to the classes "No dyslexic" and "Dyslexic" respectively.

The group of datasets "Athletics at the Oviedo University" comprises eight different sets, whose descriptions are as follows:

1. Dataset "B200ml-I": This dataset is used to predict whether an athlete will improve certain threshold in 200 meters. All the indicators or inputs are fuzzy-valued and the outputs are sets.
2. Dataset "B200mlP": Same dataset as "B200mlI", with an extra feature: the subjective grade that the trainer has assigned to each athlete. All the indicator are fuzzy-valued and the outputs are sets.
3. Dataset "Long": This dataset is used to predict whether an athlete will improve certain threshold in the long jump. All the features are interval-valued and the outputs are sets. The coach has introduced his personal knowledge.

TABLE 1
Categories of the tests currently applied in Spanish schools for detecting dyslexia when is an expert who evaluates the children.

Category	Test	Description
Verbal comprehension	BAPAE	Vocabulary
	BADIG	Verbal orders
	BOEHM	Basic concepts
Logic reasoning	RAVEN	Color
	BADIG	Figures
	ABC	Actions and details
Memory	Digit WISC-R	Verbal-additive memory
	BADIG	Visual memory
	ABC	Auditive memory
Level of maturation	ABC	Combination of different tests
Sensory-motor skills	BENDER	visual-motor coordination
	BADIG	Perception of shapes
	BAPAE	Spatial relations, Shapes
	STAMBACK	Auditive perception, Rhythm
	HARRIS/HPL	Laterality, Pronunciation
	ABC	Pronunciation
	GOODENOU.	Spatial orientation
Attention	Toulose	Attention and fatigability
	ABC	Attention and fatigability
Reading-Writing	TALE	Analysis of reading and writing

4. Dataset “BLong”: Same dataset as “Long”, but now the measurements or inputs are defined by fuzzy-valued data, obtained by reconciling different measurements taken by three different observers.

5. Dataset “100ml”: Used for predicting whether a threshold in the 100 metres sprint race is being achieved. Each measurement was repeated by three observers. The input variables are intervals and outputs are sets.

6. Dataset “100mlP”: Same dataset as “100mlI”, but the measurements have been replaced by the subjective grade the trainer has assigned to each indicator (i.e.“reaction time is low” instead of “reaction time is 0.1 seg”).

7. Dataset “B100mlI”: Same dataset as “100mlI”, but now the measurements are defined by fuzzy-valued data.

8. Dataset “B100mlP”: Same dataset as “100mlP”, but now the measurements are defined by fuzzy-valued data.

Both groups of datasets, “Diagnosis of the Dyslexic” and “Athletics at the Oviedo University”, are available in the KEEL repository* [1, 2]. The name, the number of examples (Ex.), number of attributes (Atts.), the classes (Classes) and the fraction of patterns of each class (%Classes) for each dataset are displayed in Table 2. Observe that the proportions of the different patterns are intervals, because the class labels of some instances are imprecise.

4.2 Experimental settings

All the experiments have been run with a population size of 100, probabilities of crossover and mutation of 0.9 and 0.1, respectively, and limited to 150 generations. The fuzzy partitions of the labels are uniform and their size is 5. All the imprecise experiments were repeated 100 times with bootstrapped resamples of the training set. Each partition of test contains 1000 tests.

4.3 Differences in accuracy

The compared accuracies between the proposed algorithm (Boosting_LQD) and other GFS capable of extracting rules from low quality data (GFS [15] and MR_GFS [18]) data are shown in Table 3. The results are expressed by

* http://www.keel.es/datasets.php

TABLE 2
Summary descriptions of datasets with meta-information.

Dataset	Ex.	Atts.	Classes	%Classes
B200mlI	19	4	2	([0.47,0.73],[0.26,0.52])
B200mlP	19	5	2	([0.47,0.73],[0.26,0.52])
Long	25	4	2	([36,64],[36,64])
BLong	25	4	2	([36,64],[36,64])
100mlI	52	4	2	([0.44,0.63],[0.36,0.55])
100mlP	52	4	2	([0.44,0.63],[0.36,0.55])
B100mlI	52	4	2	([0.44,0.63],[0.36,0.55])
B100mlP	52	4	2	([0.44,0.63],[0.36,0.55])
Dyslexic-12-12	65	12	3	([0.32,0.43],[0.32,0.52], [0.12,0.30])
Dyslexic-11-01	65	11	3	([0.43,0.53],[0.23,0.35], [0.12,0.30])
Dyslexic-11-12	65	11	3	([0.32,0.43],[0.32,0.52], [0.12,0.30])

means of intervals, that are our best bounds about the mean values of the test error. These intervals are defined by the expression

$$\overline{\text{error}} = \left\{ \frac{1}{m} \sum_{i=1}^{m} e_i \mid e_i \in \overline{e}_i \right\} \tag{37}$$

where

$$\overline{e}_i = \begin{cases} 0 & \text{bclass}(\overline{x}_i) = \overline{y}_i \text{ and } \#(\overline{y}_i) = 1, \\ 1 & \text{bclass}(\overline{x}_i) \cap \overline{y}_i = \emptyset, \\ \{0,1\} & \text{else.} \end{cases} \tag{38}$$

The method "Boosting_LQD" shows a better performance than the remaining classifiers, in both binary problems (Athletics datasets) and multiclass (Dyslexic datasets). Observe that MR_GFS could not be applied to the multiclass datasets in this paper because a suitable matrix of costs was not given for these datasets. The good behavior of the algorithm in the binary problems "Long", "BLong", "B200mlI" and "B200mlP" was remarkable, as shown in the graphs of the dispersion of the results of the 100 bootstrap tests, that have been plotted in Figures 4 to 11 in the Appendix.

TABLE 3
Behaviour of "GFS", "MR_GFS" and "Boosting_LQD" in several datasets of Athletics and Dyslexia.

Dataset	GFS	MR_GFS	Boosting_LQD
100mlI	[0.176,0.378]	[0.178,0.380]	[0.170,0.376]
100mlP	[0.176,0.360]	[0.188,0.367]	[0.180,0.358]
Long	[0.321,0.590]	[0.288,0.557]	[0.231,0.499]
BLong	[0.326,0.625]	[0.286,0.586]	[0.219,0.519]
B100mlI	[0.172,0.369]	[0.188,0.385]	[0.158,0.356]
B100mlP	[0.160,0.349]	[0.161,0.350]	[0.161,0.350]
B200mlI	[0.232,0.473]	[0.178,0.418]	[0.171,0.415]
B200mlP	[0.262,0.480]	[0.215,0.433]	[0.188,0.406]
Athletics mean	[0.228,0.453]	[0.210,0.434]	[0.184,0.409]
Dyslexic-12-12	[0.386,0.557]	N/A	[0.376,0.530]
Dyslexic-11-01	[0.445,0.573]	N/A	[0.447,0.567]
Dyslexic-11-12	[0.528,0.690]	N/A	[0.458,0.595]
Dyslexic mean	[0.453,0.606]	N/A	[0.427,0.564]
Global mean	[0.340,0.529]	N/A	[0.305,0.486]

In multiclass problems there is a substantial improvement in the datasets "Dyslexic-11-12" and "Dyslexic-12-12" (Figures 12 and 13, respectively) and a slight improvement in "Dyslexic-11-01" (Figure 14). By comparing the results obtained by the boosting algorithm in "Dyslexic-12-12" and "Dyslexic-11-12" we detect that the removal of the imprecision in the input values (dataset "Dyslexic-11-12") indeed lowers the performance of the algorithm, supporting a result already found in [16]. Moreover, the boosting algorithm show us that the imprecise outputs of that dataset are significant for determining the weights of the instances and the confidences in the consequents.

4.4 Learning time and linguistic quality of the results

The dependence between the maximum number of rules in the knowledge base and the test error is summarized in Table 4. Most of times the error has only a small, random fluctuation when related to the number of rules. Nonetheless, we have not found that the error of the ensemble tends to stabilize when the number of rules is increased; the presence of almost identical antecedents does not have the same effect with imprecise data than it has with crisp data [7]. We have included two pathological examples of this behavior in Figure 1. In the left part of this figure, we have displayed the behavior of the dataset "100mlP", for which the test error increases with the number of

TABLE 4
Behaviour of "Boosting_LQD" with respect to the number of rules

Dataset	7rules	15rules	30rules	40rules	50rules	100rules
100mlI	[0.189,0.391]	[0.181,0.383]	-	[0.170,0.376]	[0.183,0.384]	-
100mlP	[0.180,0.358]	[0.206,0.385]	[0.210,0.389]	[0.212,0.391]	-	-
B100mlI	[0.174,0.372]	[0.166,0.363]	[0.169,0.367]	[0.158,0.356]	[0.168,0.366]	-
B100mlP	[0.169,0.358]	[0.161,0.350]	[0.194,0.379]	[0.184,0.373]	-	-
Long	[0.290,0.559]	[0.265,0.533]	[0.231,0.499]	[0.234,0.503]	-	-
BLong	[0.269,0.569]	[0.228,0.528]	[0.231,0.531]	[0.219,0.519]	[0.249,0.549]	-
B200mlI	[0.171,0.415]	[0.175,0.420]	[0.187,0.431]	[0.188,0.432]	-	-
B200mlP	[0.213,0.431]	[0.202,0.420]	[0.211,0.429]	[0.188,0.406]	[0.205,0.423]	-
Dyslexic-12-12	-	-	[0.428,0.579]	-	[0.391,0.543]	[0.376,0.530]
Dyslexic-11-01	-	[0.476,0.593]	[0.471,0.588]	-	[0.459,0.579]	[0.447,0.567]
Dyslexic-11-12	-	-	[0.509,0.643]	-	[0.492,0.626]	[0.458,0.595]

rules, and in the right part of the same figure the dataset "Blong", where the same happens once its optimal size is surpassed.

We have to remark that the number of rules needed for obtaining a good classification with this technique is not higher than it was with GCCL algorithms [15][18]. In Table 5 we have displayed the average number of rules needed either with boosting or GCCL algorithms, along with a linguistic assessment of the gain in accuracy, using the labels "High", "Medium", "Low" and "None". Observe that the best improvements in accuracy are related to an increase in the size of the KB, however there are some cases, like "B100mlI" or "B200mlI", were boosting produces rule bases that are both smaller and more precise than GCCL. In Table 6 we show an example of one of the KBs obtained after the application of the boosting algorithm.

4.5 Some variants of the original algorithm

In Table 7 we have included the results of two heuristic variants of the standard Adaboost algorithm that sometimes are able to improve the accuracy or the interpretability of the basic algorithm for low quality data-based problems.

The first of these heuristics (column labeled "No Duplicates") consists in disallowing the presence of rules with identical antecedents; in case a duplicate is selected, the consequents of the already present rule and the weights of the examples are altered, but a new rule is not added. This is only a partial solution to the case, mentioned before, where fuzzy weights prevented

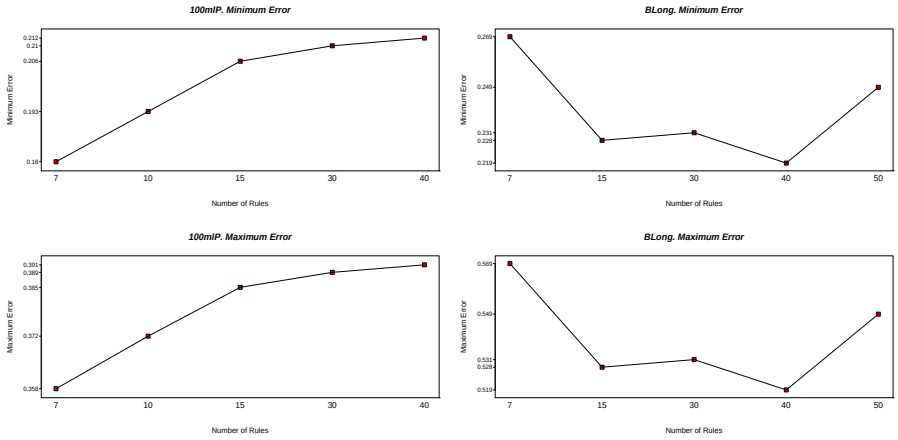

FIGURE 1
Behaviour of "Boosting_LQD" with respect to the number of rules in the datasets 100mlP and BLong (left and right parts, respectively).

TABLE 5
Number of rules obtained with "GFS", "MR_GFS" and "Boosting_LQD" in all the datasets of Athletics and Dyslexia.

Dataset	**GFS**	**MR_GFS**	**Boosting_LQD**	**Improvement of Boosting_LQD vs. GFS**
100mlI	24	24	26	"Low"
100mlP	23	25	7	"Low"
Long	16	16	26	"High"
BLong	16	16	23	"High"
B100mlI	23	23	21	"Medium"
B100mlP	23	23	14	"None"
B200mlI	13	13	7	"High"
B200mlP	13	13	24	"High"

TABLE 6
Knowledge bases obtained with Boosting_LQD in the dataset "100mlP".

Id.	Antecedentet Rule	s_0^j (Not relevant)	s_1^j (Relevant)
1	IF reaction time is Low and 20 meters speed is Medium	0.639	0.144
2	IF ratio is Very-high and reaction time is High and 40 meters speed is High	0.072	0.790
3	IF ratio is Very-high and reaction time is Medium and 20 meters speed is High	0.008	0.842
4	IF ratio is High and reaction time is Very-low and 20 meters speed is Low	0.98	0
5	IF ratio is Low and reaction time is Medium and 40 meters speed is Medium	0.615	0.142
6	IF ratio is Medium and reaction time is Very-high and 20 meters speed is Very-high and 40 meters speed is Very-high	0	1
7	IF ratio is Very-low and reaction time is Medium and 20 meters speed is High	0.626	0.12

the cancellation of the votes of almost identical rules, however it produces a small improvement only in multiclass problems.

In a second place, for improving the linguistic quality we have computed the results of using an Adaboost generated KB with a "winner takes all" type of fuzzy inference. One of the main criticisms to the use of Adaboost for learning fuzzy rules lies in the use of voting in the inference process, which is less human-readable than an inference based on the winner rule. This difference, however, is small when there are not duplicate rules and the degree of overlapping between the rules is small [9]. In the column "Alt. Inference" of Table 7 we have therefore computed the error of the KB generated by boosting when a winned-based inference is used instead of the sum of votes. The consequents of the rules are not optimal for this kind of inference (see [19] for an algorithm whose extension would produce suitable weights for this kind of inference and crisp data) however the loss of accuracy is not large and in certain cases ("Dyslexic" problems) this loss is not noticeable. Lastly, observe that even after this loss of accuracy, the boosting algorithm is still competitive with GCCL-type algorithms.

TABLE 7
Heuristic variants of the standard Adaboost algorithm that sometimes are able to improve the accuracy (column "No Duplicates") or the interpretability (column "Alt. Inference"). The column "GFS" is included as a reference of the performance of a GCCL algorithm using a "winner-takes-all" reasoning scheme.

Dataset	**Boosting_LQD**	**No Duplicates**	**Alt. Inference**	GFS
100mlI	[0.170,0.376]	[0.185,0.387]	[0.186,0.387]	[0.176,0.348]
100mlP	[0.180,0.358]	[0.182,0.361]	[0.187,0.366]	[0.176,0.360]
B100mlI	[0.158,0.356]	[0.174,0.372]	[0.171,0.369]	[0.321,0.590]
B100mlP	[0.161,0.350]	[0.168,0.357]	[0.167,0.356]	[0.326,0.625]
Long	[0.231,0.499]	[0.249,0.517]	[0.230,0.499]	[0.172,0.369]
BLong	[0.219,0.519]	[0.227,0.527]	[0.229,0.529]	[0.160,0.349]
B200mlI	[0.171,0.415]	[0.189,0.433]	[0.186,0.431]	[0.232,0.473]
B200mlP	[0.188,0.406]	[0.187,0.405]	[0.198,0.416]	[0.262,0.480]
Athletics mean	[0.184,0.409]	[0.195,0.419]	[0.194,0.419]	[0.228,0.453]
Dyslexic-12-12	[0.376,0.530]	[0.362,0.524]	[0.342,0.508]	[0.386,0.557]
Dyslexic-11-01	[0.447,0.567]	[0.456,0.572]	[0.432,0.551]	[0.445,0.573]
Dyslexic-11-12	[0.458,0.595]	[0.442,0.592]	[0.357,0.516]	[0.528,0.690]
Dyslexic mean	[0.427,0.564]	[0.420,0.562]	[0.377, 0.525]	[0.453,0.606]
Global mean	[0.305,0.486]	[0.307,0.490]	[0.285,0.472]	[0.340,0.529]

5 CONCLUDING REMARKS

In this paper we have proposed a new extension of the Adaboost algorithm for learning fuzzy rule based classifiers from interval-valued or fuzzy data. Fuzzy rules were regarded as weak learners and knowledge bases as ensembles. The number of votes of a weak learner was identified with the degree of confidence of its corresponding consequent. Both the objective function of the Adaboost algorithm and the weights of the instances were assigned interval or fuzzy values, however the number of votes each weak learning is assigned has been designed to be a crisp number thus the classifier does not introduce uncertainty of its own.

The results of the new algorithm have been compared to that of previous genetic algorithms for low quality data, in both accuracy and linguistic interpretation. The results prove that this technique is fast, its accuracy is competitive and the number of rules in the knowledge base is not higher than that of the alternatives. On the other hand, it uses a voting-based inference, whose linguistic quality is not the best, and the performance gain is not highly relevant for multiclass problems.

ACKNOWLEDGEMENTS

This study has been supported by the Spanish Ministry of Science and Technology and by European Fund FEDER (project TIN2008-06681-C06-04) and by the Principado de Asturias, PCTI 2006-2009.

REFERENCES

[1] Alcalá, J., Fernández, A., Luengo, J., Derrac, J., García, S., Sánchez, L., Herrera, F. KEEL Data-Mining software tool: data set repository, integration of algorithms and experimental analysis framework. Journal of Multivalued Logic and Soft Computing 17 (2-3) 255-287 (2011)

[2] Alcalá-Fdez, J., Sánchez, L., García, S., del Jesus, M.j., Ventura, S., Garrell, J. M., Otero, J., Romero, C., Bacardit, J., Rivas, V. M., Herrera, F. KEEL: A software tool to assess evolutionary algorithms to data mining problems Soft Computing 13 (3) 307-318 (2009)

[3] Cordón, O., del Jesus, M. J., Herrera, F. A proposal on reasoning methods in fuzzy rule-based classification systems. *International Journal of Approximate Reasoning* 20(1), 21-45. (1999).

[4] Freund, Y., Schapire, R. "Experiments with a new boosting algorithm". In Machine Learning, Proc. 13th International Conference, 148-156 (1996)

[5] Friedman, J., Hastie, T., Tibshirani, R. "Additive Logistic Regression: a Statistical View of Boosting," Annals of Statistics 28(2):337-374. (2000).

[6] González, A., Herrera, F., Multi-stage genetic fuzzy systems based on the iterative rule learning approach. *Mathware and Soft Computing* 4(3), 233-249 (1997).

[7] Del Jesus, M. J., Junco, L., Hoffmann, F., Sánchez, L., Induction of Fuzzy Rule Based Classifiers with Evolutionary Boosting Algorithms. IEEE Transactions in Fuzzy Sets. 12(3) 296-308 (2004)

[8] Dubois, D., Guyonnet, D. Risk-informed decision-making in the presence of epistemic uncertainty. International Journal of General Systems 40 (2) 145-167 (2011)

[9] González, A., Pérez, R. Completeness and consistency conditions for learning fuzzy rules. *Fuzzy Sets and Systems 96*, 37-51. (1998)

[10] Hoffmann, F., Boosting a Genetic Fuzzy Classifier. in Proc. Joint 9th IFSA World Congress and 20th NAFIPS International Conference, vol. 3, (Vancouver, Canada), pp. 1564-1569 (2001)

[11] Ishibuchi, H., Nakashima, T. and Morisawa, T., Voting in fuzzy rule-based systems for pattern classification problems. Fuzzy Sets and Systems, vol 103, no. 2, 223-239, (1999).

[12] Junco, L., Sánchez, L. Using the Adaboost algorithm to induce fuzzy rules in classification problems, Proc. ESTYLF 2000, Sevilla, 297-301. (2000)

[13] Kuncheva, L. I. Fuzzy Classifier Design. Springer-Verlag, NY, (2000).

[14] Otero, J., Sánchez, L. Induction of descriptive fuzzy classifiers with the Logitboost algorithm. Soft Computing 10(9) 825-835 (2006)

[15] Palacios, A., Sánchez, L., Couso, I. Extending a simple cooperative-competitive learning fuzzy classifier to low quality datasets. Evolutionary Intelligence 2 (1-2), 73-84 (2010)

[16] Palacios, A., Sánchez, L., Couso, I. Diagnosis of dyslexia with low quality data with genetic fuzzy systems. International Journal on Approximate Reasoning 51, 993-1009 (2010)

[17] Palacios, A. Sánchez, L., Couso, I. Future performance modelling in athletism with low quality data-based GFSs. Journal of Multivalued Logic and Soft Computing 17 (2-3) 207-228 (2011)

[18] Palacios, A., Sánchez, L., Couso, I. Linguistic Cost-Sensitive Learning of Genetic Fuzzy Classifiers for Imprecise Data. International Journal on Approximate Reasoning. Admitted.

[19] Sánchez, L. Otero. J., Boosting fuzzy rules in classification problems under single-winner inference. International Journal of Intelligent Systems 22(9) 1021-1035. (2007)

[20] 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)

[21] Sánchez, L., Couso,I., Casillas, J. Modeling vague data with genetic fuzzy systems under a combination of crisp and imprecise criteria Proceedings of the First IEEE Symposium on Computational Intelligence in Multi-Criteria Decision-Making (MCDM 2007), 30-37 Honolulu, Hawaii, USA (2007)

[22] Schapire, R., Singer, Y. Improved Boosting Algorithms Using Confidence-rated Predictions. Machine Learning 37(3): 297-336. (1999)

[23] Schapire, R. E. Theoretical views of Boosting and Applications. Lecture Notes in Artificial Intelligence, Vol. 1720, pp 13 - 25. (1999).

A PSEUDOCODES

Input data:

Training set $(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_m, y_m), \quad \mathbf{x}_i \in \mathbf{R}^n,\ y_i \in \{-1, +1\}$

Number of hypotheses $H \in \{1, \ldots, N\}$

Local Variables:

$w \in \mathbf{R}^m$ (weights of the examples in the training set)

$\alpha \in \mathbf{R}^N$ (votes of the weak hypotheses)

begin

Initialize $w_i \leftarrow 1/m$, $\alpha^j = 0$

Select the number of weak hypotheses H

Repeat H times

Identify the weak hypothesis $g^h \in \{g^1, \ldots g^N\}$
that best classifies the weighted data

Calculate the number of votes α^h of g^h

Update the weights w_i of the examples

end-repeat

Output the classifier: sign $\left(\sum_{j=1}^{N} \alpha^j g^j(\mathbf{x})\right)$

end

FIGURE 2
Outline of the generalized Adaboost algorithm. Two-class version.

Input data:

Training set $(\mathbf{x}_1, z_1), \ldots, (\mathbf{x}_m, z_m), \quad \mathbf{x}_i \in \mathbf{R}^n,\ z_i \in \{1, \ldots, p\}$

Number of fuzzy rules $H \in \{1, \ldots, N\}$

Local Variables:

$w \in \mathbf{R}^m$ (weights of the examples in the training set)

$\alpha \in \mathbf{R}^{N \times p}$ (votes of the weak hypotheses)

$s : \mathcal{A} \times \{-1, 1\} \rightarrow [0, 1]$ (fuzzy relationship = consequents of rules)

$c \in \{-1, 1\}^m$ (consequents of the weak hypotheses)

$y \in \{-1, 1\}^m$ (labels of the examples in the two-class problems)

Local Procedures:

Generate a new fuzzy rule: see text

Convert votes into confidences: **begin**

$s_k^j \leftarrow 0$

for j in $1 \ldots N$

for k in $1 \ldots p$

if $(\alpha_k^j > 0)$ $s_k^j \leftarrow s_k^j + \alpha_k^j$

else

$s_1^j \leftarrow s_1^j - \alpha_k^j,$

$\ldots$

$s_{k-1}^j \leftarrow s_{k-1}^j - \alpha_k^j,$

$s_{k+1}^j \leftarrow s_{k+1}^j - \alpha_k^j,$

$\ldots$

$s_p^j \leftarrow s_p^j - \alpha_k^j$

end-for

end-for

Normalize consequents: $s_k^j \leftarrow s_k^j / \max s_k^j$

end

begin of main algorithm

for $k = 1, \ldots, p$

Initialize $w_i \leftarrow 1/m$, $\alpha_k^j = 0$

Initialize if $(z_i = k)$ $y_i = 1$ else $y_i = -1$

repeat H/p times

Generate a new fuzzy rule " if $\mathbf{x}$ is A^h then c^h"

Calculate the number of votes of the following rule:

"votes(if $\mathbf{x}$ is A^h then c^h) $= \alpha_k^h$"

Update the weights of the examples

end-repeat

end-for

Convert votes into confidences

Output the classifier: for all j, if any $s_k^j \neq 0$ emit the rule

if x_1 is A_1^j and $\ldots x_n$ is A_n^j then $\text{tr}(c_0) = s_0^j, \ldots, \text{tr}(c_p) = s_p^j$

end of main algorithm

FIGURE 3

Adaboost algorithm applied to the induction of a descriptive, multi-class, fuzzy rule based classification system.

B GRAPHS OF THE DISPERSION OF THE PAIRED BOOTSTRAP TEST RESULTS

In these graphs, the horizontal coordinate measures the fraction of errors of the boosting algorithm, and the vertical coordinate measures the same parameter in the counterpart algorithm. Each filled circle represents a case where the boosting algorithm was the best choice, and the squares mean the opposite result. A blank circle or square means a tie (or a difference with is not significant) between both algorithms. Those figures where there is a high density of filled circles in the upper left part signal cases where the dispersion of the results is compatible with a statistically significant improvement of the bound, and those cases where the cloud is near the diagonal or in the lower, right part display situations where the differences, if exist, are not significant.

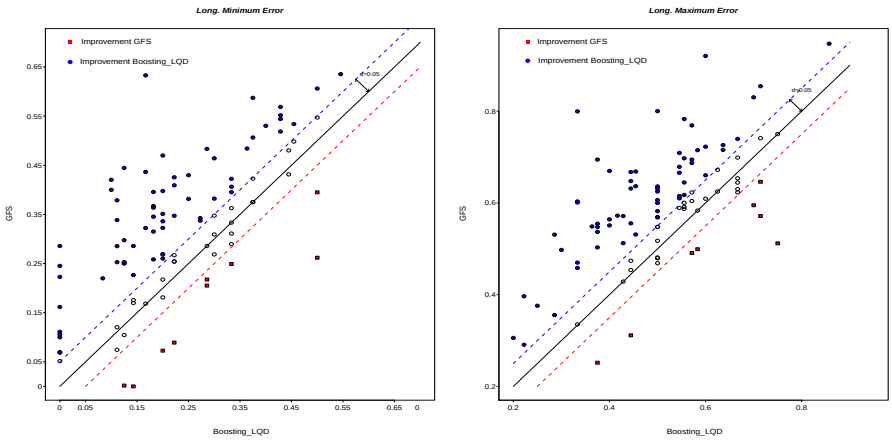

FIGURE 4
Behaviour of "GFS" and "Boosting_LQD" respect to the dataset Long. **Left figure:** Lower bounds. **Right figure:** Upper bounds.

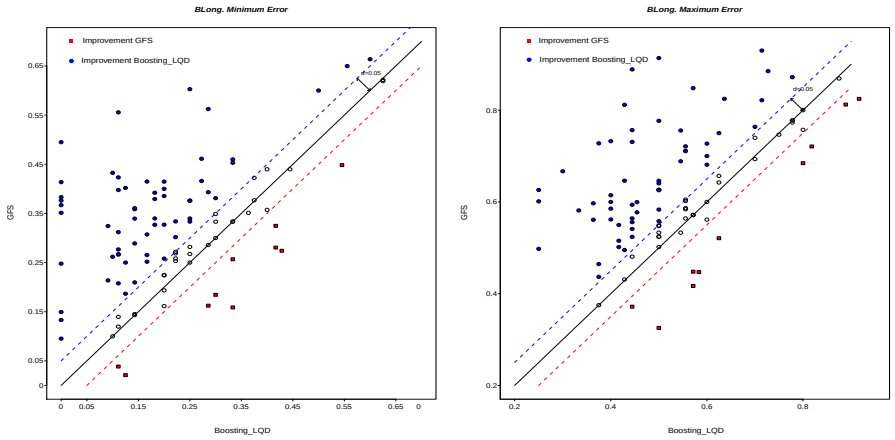

FIGURE 5
Behaviour of "GFS" and "Boosting_LQD" respect to the dataset BLong. **Left figure:** Lower bounds. **Right figure:** Upper bounds.

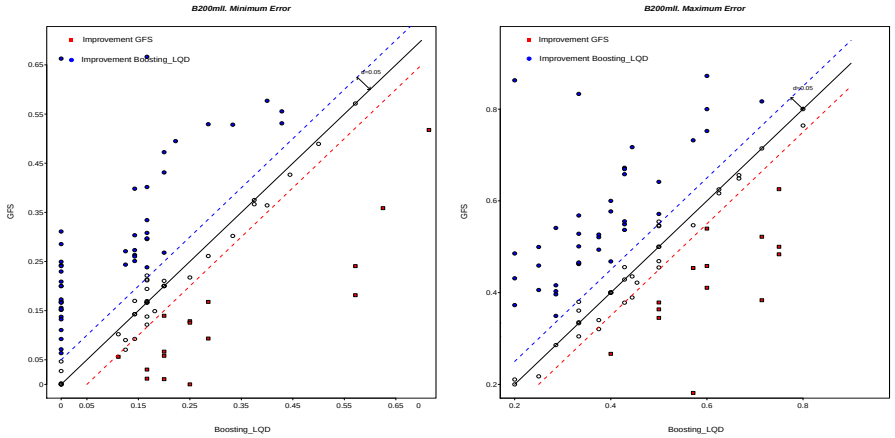

FIGURE 6
Behaviour of "GFS" and "Boosting_LQD" respect to the dataset B200mII. **Left figure:** Lower bounds. **Right figure:** Upper bounds.

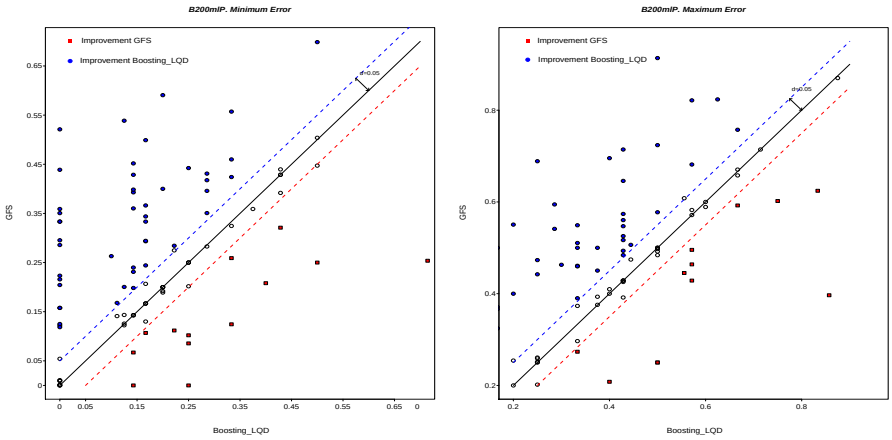

FIGURE 7
Behaviour of "GFS" and "Boosting_LQD" respect to the dataset B200mlP. **Left figure:** Lower bounds. **Right figure:** Upper bounds.

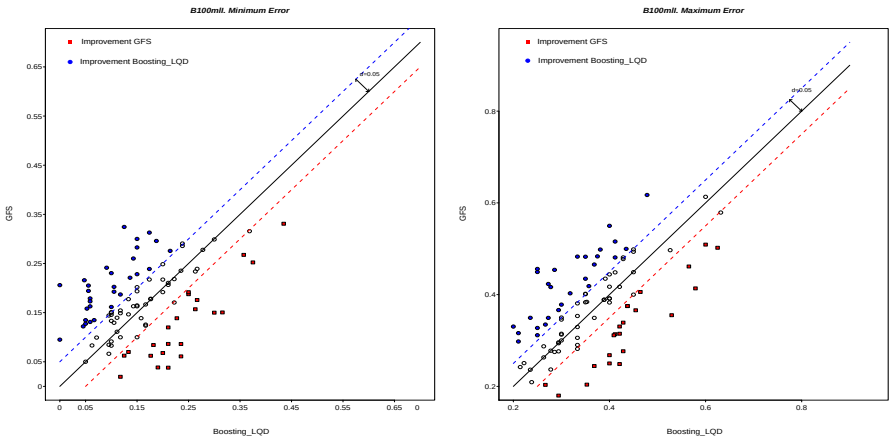

FIGURE 8
Behaviour of "GFS" and "Boosting_LQD" respect to the dataset B100mlI. **Left figure:** Lower bounds. **Right figure:** Upper bounds.

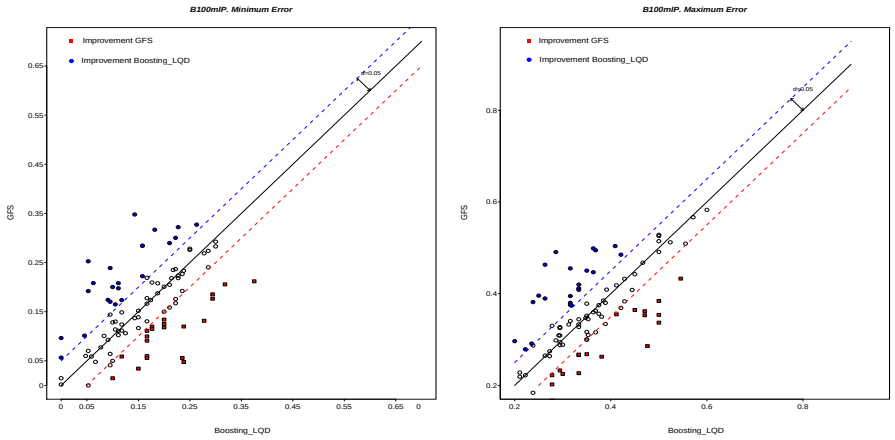

FIGURE 9
Behaviour of "GFS" and "Boosting_LQD" respect to the dataset B100mlP. **Left figure*:** Lower bounds. **Right figure:** Upper bounds.

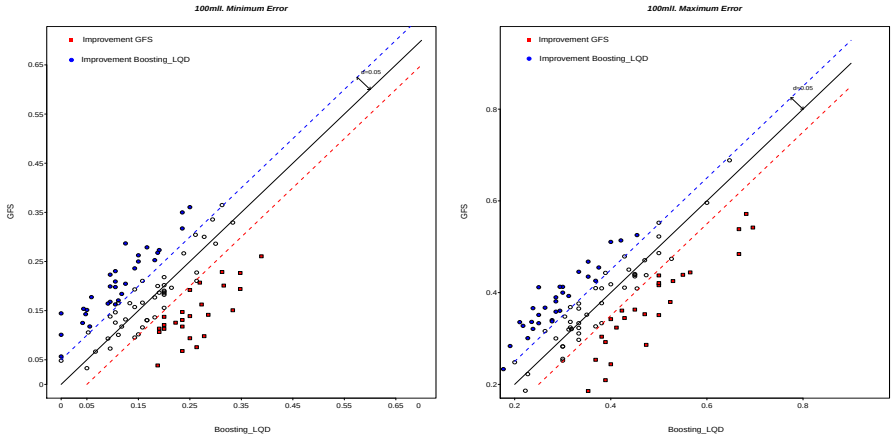

FIGURE 10
Behaviour of "GFS" and "Boosting_LQD" respect to the dataset 100mlI. **Left figure:** Lower bounds. **Right figure:** Upper bounds.

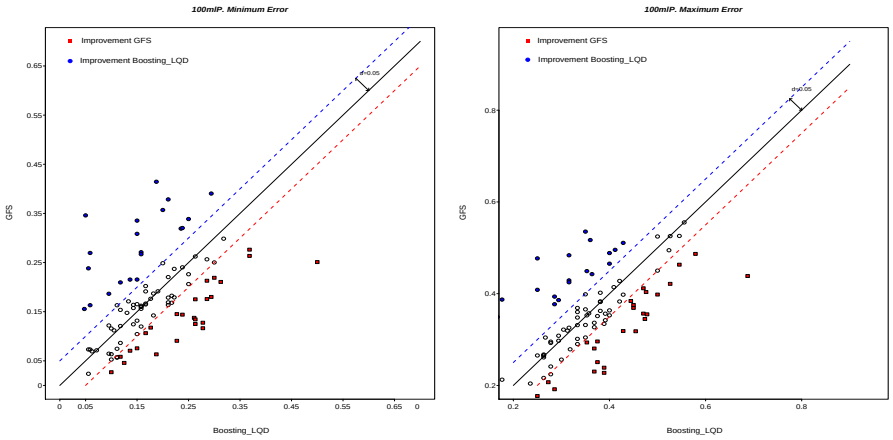

FIGURE 11
Behaviour of "GFS" and "Boosting_LQD" respect to the dataset 100mlP. **Left figure:** Lower bounds. **Right figure:** Upper bounds.

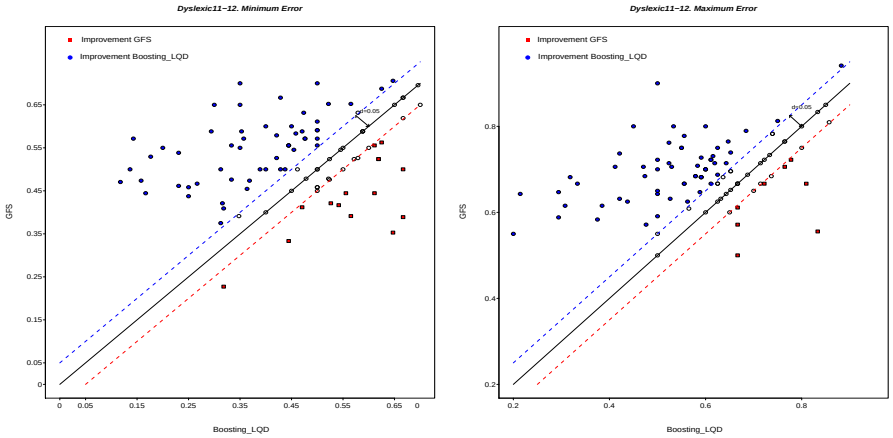

FIGURE 12
Behaviour of "GFS" and "Boosting_LQD" respect to the dataset Dyslexic11-12. **Left figure:** Lower bounds. **Right figure:** Upper bounds.

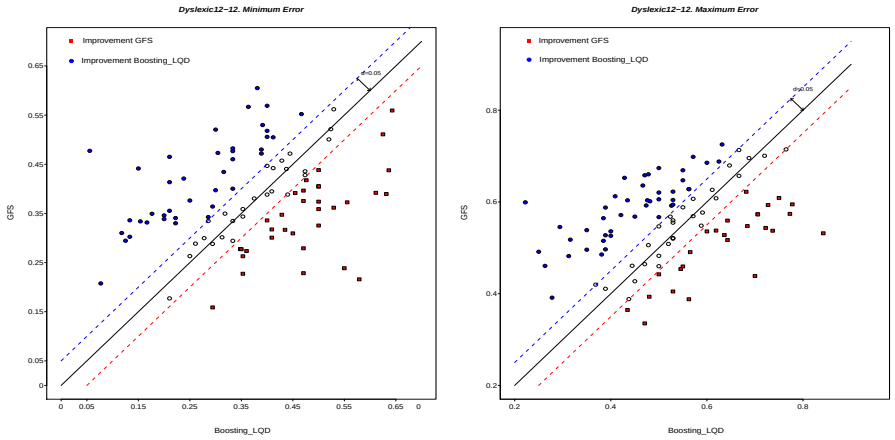

FIGURE 13
Behaviour of "GFS" and "Boosting_LQD" respect to the dataset Dyslexic12-12. **Left figure:** Lower bounds. **Right figure:** Upper bounds.

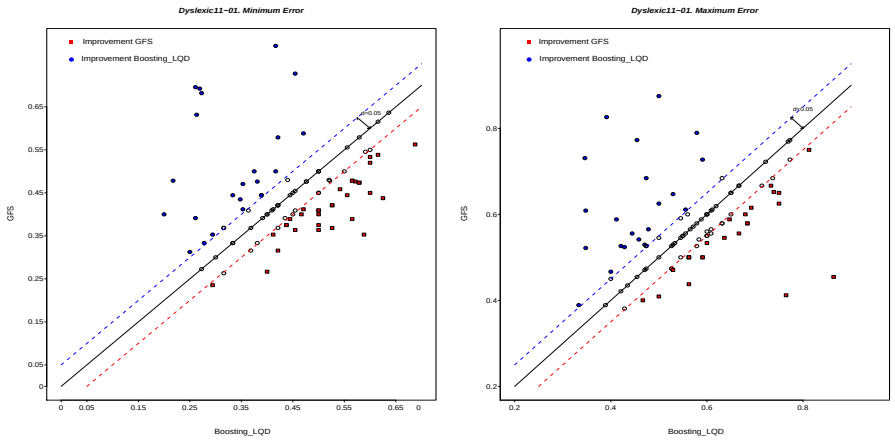

FIGURE 14
Behaviour of "GFS" and "Boosting_LQD" respect to the dataset Dyslexic11-01. **Left figure:** Lower bounds. **Right figure:** Upper bounds.