An Instance Selection Algorithm for Regression and its Application in Variance Reduction

Ismael Rodríguez-Fdez, Manuel Mucientes, Alberto Bugarín
Centro de Investigación en Tecnoloxías da Información (CITIUS)
Universidade de Santiago de Compostela
{ismael.rodriguez, manuel.mucientes, alberto.bugarin.diz} @usc.es

***Abstract*—The tradeoff between bias and variance is a well-known problem in machine learning, since algorithms are expected to achieve a reduced training error without going into overfitting. In Genetic Fuzzy Systems (GFSs), overfitting is usually avoided through the control of the number of rules and/or the number of labels. However, in many machine learning approaches, variance is reduced through the use of a validation set. Inspired by this idea, we propose in this paper an Instance Selection (IS) algorithm for regression problems called Class Conditional Instance Selection for Regression (CCISR) which is based on CCIS [1]. The output of CCISR is used in a GFS to obtain Rule Bases with a low variance, as the rules are generated with an *ad hoc* data driven method guided by the selected instances, but the error is still measured with the full training dataset. The combined system has been tested over 12 publicly available datasets, and results were compared with other GFSs. Our approach is capable of achieving a reduction in the number of rules while maintaining a good accuracy.**

***Keywords*—Instance Selection, regression problems, Genetic Fuzzy Systems (GFSs), variance reduction**

I. Introduction

Machine learning algorithms try to obtain models with low bias and variance. The objective is to find a good balance between the complexity of the model and the reduction in the training error in order to, finally, get a good test error. However, when the complexity of the model exceeds a threshold, although the training error decreases, the machine learning algorithms generate models with a low bias but a high variance, i.e., the model overfits the training data. In the literature, many different techniques, such as regularization parameters, validation sets, etc. [2], have been described to reduce the variance. In particular, the validation set approach consists of learning the model with a part of the training data, and checking for overfitting with the other part of the training set.

The use of fuzzy rule base systems to model regression problems is very extended, since it combines the interpretability and expressiveness of the rules with the ability of fuzzy logic for representing uncertainty. Several machine learning algorithms have been used to learn fuzzy rules for regression, like swarm optimization [3], an hybrid with Support Vector Machines for Regression [4], [5], or the most widely used Genetic Fuzzy Systems (GFSs) [6]. GFSs are the combination of evolutionary algorithms and fuzzy logic. Evolutionary algorithms are are endowed with some features that make them suitable for learning fuzzy rules. In particular, the flexibility of evolutionary algorithms allows to codify any part of the fuzzy rule base system and, also, to manage the balance between accuracy and interpretability of the model in an effective way [7], [8].

The reduction of the variance in GFSs for regression has been addressed, principally, through the control of the complexity of the rule base and, particularly, taking into account the number of rules and/or the number of labels. Traditionally, the size of the rule base is managed following two approaches: (i) with a regularization parameter, and (ii) with a multi-objective evolutionary algorithm that takes into account both accuracy and complexity. In the first approach, the fitness function is augmented with a term that penalizes models with a high number of rules [6]. In the second approach, although multi-objective evolutionary algorithms can manage the complexity in an elegant and natural way [9]–[13], if the most accurate model of the Pareto needs to be selected, the GFS has to include other conditions to prevent overfitting. For example, the multi-objective learning in [13] also uses a threshold for limiting the maximum number of rules.

Inspired by the idea of reducing the variance with a validation set, we propose in this paper to select a subset of the examples in the training data to generate the rule base, while still measuring the training error with the full training set. In order to implement this idea, an Instance Selection (IS) algorithm is needed for getting the most adequate instances to guide the rules generation. IS is the process of selecting a subset of representative examples from the original data. The aim is to obtain a representative training set with a lower size than the original one that can accomplish tasks with little or no performance deterioration. This data reduction simultaneously faces the computational complexity, storage requirements, and noise tolerance.

Currently, there are many different methods of IS and several reviews [14]–[16] that summarize the progress in the field. More recently, [17], [18] describe a taxonomy and an empirical study of IS algorithms. Furthermore, in [19] the application of several IS methods to a GFS for classification was described. IS methods can be grouped into different categories according to their common properties and to the description presented in [18]: type of selection (position of the instances to be retained with respect to the decision boundaries), the direction in which the search can proceed (incremental, decremental, batch,...), and evaluation of search (filter or wrapper). Different evaluations methods can be selected, being nearest neighbors (NN) the most widely used to determine the criteria for adding

or removing examples.

There are a few works that apply IS to regression problems. In [20] the IS method uses the concept of mutual information to decide which examples should be selected. The proposed algorithm is tested in simple regression datasets in order to evaluate the robustness of the approximation models using noisy training inputs. More recently, in [21] a coevolutionary algorithm was presented. IS was implemented through a single-objective evolutionary algorithm executed after a multi-objective evolutionary algorithm. This single-objective algorithm aims to maximize a measure of how much the reduced training set is representative of the overall TS, while the multi-objective algorithm learns the rule base and the membership function parameters of the fuzzy sets by maximizing the accuracy and minimizing the complexity.

In this work we propose a new algorithm for IS in regression, CCISR (Class Conditional Instance Selection for Regression), which is based on Class Conditional Instance Selection (CCIS) [1]. CCIS uses a class conditional nearest neighbor relation over pairs of points in the training set. Using this relation, CCIS creates two different graphs and defines a novel scoring function over the instances by means of an information-theoretic divergence measure. The scoring function is employed to develop an effective large margin instance selection method. CCISR extends CCIS to cope with regression problems. CCISR modifies the nearest neighbor definition to construct the graphs, it includes a new error measure, and it redefines the initial size of the selected set of instances.

Moreover, the partition of the training set generated by CCISR can be combined with a GFS in order to get a fuzzy rule base with low variance. CCISR can be integrated with any GFS based on a data-driven method for the generation of the rule base. The GFS requires just a minor modification for the integration. In this paper we have selected as GFS a modified version of GLD [6], because this algorithm obtains rule bases with good accuracy. Specifically, GLD learns the Data Base (DB) using the number of labels and the lateral displacement per variable. For each DB generated by the evolutionary algorithm, an ad-hoc data driven rule generation process is used to obtain the Rule Base (RB). In summary, the fuzzy knowledge base learning system for regression has three modules: K-means, CCISR, and GFS. The first one has been included in order to improve the reduction in the number of selected instances.

The main contributions of the paper are: (i) CCISR, a new IS method for regression that has a good balance between reduction and accuracy, with a limited computational cost; (ii) the combination of CCISR with a data-driven based GFS in order to generate accurate rule bases with low variance. The paper is structured as follows: Sec. II presents the new IS method for regression, and Sec. III describes how CCISR is used within the evolutionary algorithm to learn fuzzy rule base systems for regression. Sec. IV presents the obtained results and, finally, Sec. V points out the most relevant conclusions.

II. Class Conditional Instance Selection for Regression (CCISR)

CCISR is based on CCIS [1], an instance selection method for classification problems that achieves a good accuracy and a high reduction rate, with a reasonable computational cost. In the next subsection we briefly show the main characteristics of CCIS—a more detailed description can be found in [1]. In the remaining subsections we describe the differences between CCISR and CCIS: (i) we propose a new way to construct the graphs; (ii) we define an appropriate error measure; (iii) and we redefine the size of the initial instances set.

A. Class Conditional Instance Selection (CCIS)

CCIS is based on a novelty relation called class conditional nearest neighbor (*ccnn*), defined on pairs of points from a labeled training set as follows: for a given class c, $ccnn$ associates to instance a its $nearest\ neighbor$ computed among only those instances (excluded a) in class c. Thus, this relation describes proximity information conditioned to a class label. Two different graphs can be constructed using this relation:

- Within-class directed graph (G_{wc}): consists in a graph where each instance has an edge pointing to the nearest instance of the same class.
- Between-class directed graph (G_{bc}): is a graph where each instance has an edge pointing to the nearest instance of any different class.

These graphs are used to define an instance scoring function by means of a directed information-theoretic measure (the K-divergence) applied to the in-degree distributions of these graphs. The scoring function (named `Score`) is used to develop an effective large margin instance selection method, called Class Conditional selection (Fig. 1).

The IS algorithm starts from a set of training examples:

$$E = \{e^1, e^2, \ldots, e^n\} \tag{1}$$

where n is the number of examples and each example e^l is defined as:

$$e^l = (e^l_i, e^l_o), e^l \in E \tag{2}$$

where e^l_i is a vector of the input values of e^l and e^l_o is a vector of the output values of e^l. First, an initial core of instances is used with size $k_0 = max(c, \lceil \frac{\epsilon^E}{2} \rceil)$ where c is the number of classes, and ϵ^E is the *leave-one-out* error for the set of examples in E using *1NN*. This choice is motivated (i) because there is at least one example for each class and (ii) misclassification in ϵ^E can be with equal probability either an outlier or a regular instance (no prior knowledge is assumed). After that, the instance selection method iteratively selects instances, choosing in the first place those with the highest score and storing them in a set (named S). The process terminates when the empirical error (ϵ^S) of the resulting *1NN* rule increases.

In order to further improve the storage performance of the method, a post-processing algorithm, called Thin-out selection, is used (Fig. 2). This algorithm selects points close to the decision boundary of the *1NN* rule. This is achieved by selecting instances having positive in-degree in the between-class graph of the actual training set (G^S_{bc}) and storing them in

1: $\{e^1, \ldots, e^n\} = E$ sorted in decreasing order of `Score`
2: $S = \{e^1, \ldots, e^{k_0}\}$
3: $go\_on = true$
4: $ub = n - |\{e^l \text{ s.t. } \texttt{Score}(e^l) <= 0\}|$
5: $l = k_0 + 1$
6: **while** $l < ub \wedge go\_on$ **do**
7: $\quad Temp = S \cup \{e^l\}$
8: $\quad$ **if** $\epsilon^S \leq \epsilon^E$ **then**
9: $\quad\quad go\_on = false$
10: $\quad$ **end if**
11: $\quad$ **if** $\epsilon^{Temp} < \epsilon^S \wedge go\_on$ **then**
12: $\quad\quad S = Temp$
13: $\quad\quad l = l + 1$
14: $\quad$ **else**
15: $\quad\quad go\_on = false$
16: $\quad$ **end if**
17: **end while**
18: return S

Fig. 1. Pseudocode of Class Conditional selection [1].

1: $S_f = \{e^l \in S$ with in-degree in $G_{bc}^S > 0\}$
2: $S_{prev} = S$
3: $S_1 = S \setminus S_f$
4: $go\_on = true$
5: **while** go_on **do**
6: $\quad S_t = \{e \in S_1$ with in-degree in $G_{bc}^{S_1} > 0$ and with in-degree in $G_{bc}^{S_{prev}} > 0\}$
7: $\quad go\_on = \epsilon^{S_f \cup S_t} < \epsilon^{S_f}$
8: $\quad$ **if** go_on **then**
9: $\quad\quad S_f = S_f \cup S_t$
10: $\quad\quad S_{prev} = S_1$
11: $\quad\quad S_1 = S \setminus S_f$
12: $\quad$ **end if**
13: **end while**
14: return S_f

Fig. 2. Pseudocode of Thin-out selection [1].

S_f. Also, S_1 is the subset of examples that are in S but not in S_f. Then an iterative process is done as follows: points having positive in-degree in the $G_{bc}^{S_1}$ are added to S_f if they were not isolated in the previous iteration, that is, if their in-degree was not zero (line 6). This iterative process terminates when the empirical error increases (line 7).

In regression problems the outputs are real values instead of labels and, therefore, CCIS must be modified in three different ways, leading to CCISR:

- In order to use the G_{wc} and G_{bc} graphs, a new definition of the nearest example of the same and different class is needed.
- As CCISR is going to be combined with a GFS that uses a data-driven rule base generation, we include an error measure based on the Wang & Mendel method [22], because it is the algorithm used by GDL for the rule base generation.
- Finally, k_0 (line 2, Fig. 1) must be redefined, as there are infinite classes in regression and, taking into account the definition in CCIS, this would result in an initial subset containing all the instances.

These modifications are described, in the following subsections.

B. Construction of G_{wc} and G_{bc} Graphs

This modification changes the way how a pointed example is selected in both G_{wc} and G_{bc}. Instead of using only the nearest instance to construct these graphs, we can define a neighborhood that depends on the probability density function of the examples. First, the distance between two examples is defined using the Mahalanobis distance:

$$D_M(e^l, e^h) = (e_i^h - e_i^l)^T {\Sigma_i^l}^{-1} (e_i^h - e_i^l) \tag{3}$$

where Σ_i^l is the covariance matrix calculated over the examples in S, but using e_i^l as the mean value.

Then we can define the σ-neighborhood (σN) of an example e^l as:

$$\sigma N^l = \{e^h \mid D_M(e^l, e^h) \leq \alpha_{D_M}\} \tag{4}$$

where α_{D_M} is a threshold over the Mahalanobis distance that indicates the percentage of Σ_i^l that is going to be used to define the neighborhood.

Finally, we can build the G_{wc} and G_{bc} graphs using the following definitions:

- σ-within-class directed graph (G_{wc}^{σ}): is a graph in which each instance e^l has an edge pointing to the instance that belongs to σN^l, and whose difference in the output space is the smallest.
- σ-between-class directed graph (G_{wc}^{σ}): is a graph in which each instance e^l has an edge pointing to the instance that belongs to σN^l, and whose difference in the output space is the largest one.

Given this adaptation, the graphs are constructed in the same way as in the original CCIS.

C. Error Measure

CCISR is going to be combined with a GFS with a data-driven rule base generation method. In GLD, Wang & Mendel is the algorithm that generates the fuzzy rule bases, given the definition of the linguistic labels and the set of examples. It seems appropriate to guide the instance selection process with the same error measure as the one used in the GFS. The error measure is based on the mean squared error of a fuzzy knowledge base:

$$MSE(E) = \frac{1}{2|E|} \sum_{l=1}^{|E|} (F(e_i^l) - e_o^l)^2 \tag{5}$$

where $|E|$ is the number of examples, $F(e_i^l)$ is the output of the fuzzy rule base for example e^l, and e_o^l is the desired output. Thus, the error measure for CCISR is defined as:

$$\epsilon^S = MSE(E) \tag{6}$$

where E is the original set of examples and the rule base is obtained with the Wang & Mendel algorithm over the selected

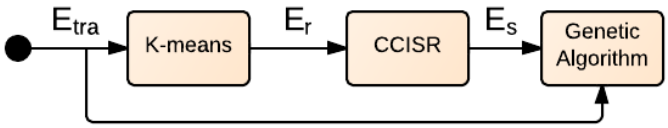

Fig. 3. Instance selection process in GLD-IS.

examples S. This error measure could be easily adapted to other rule base generation methods.

The Wang & Mendel method requires the definition of the DB. CCISR uses uniform partitions, and to select the most adequate granularity for each variable, a greedy algorithm is run prior to CCISR:

1) Set all granularities to max_g (the maximum allowed granularity) and initialize ϵ^E_{best}.
2) For each variable:
   a) Decrement the granularity by one and calculate the error ϵ^E_{new}.
   b) If $\epsilon^E_{new} \geq \epsilon^E_{best}$ then $\epsilon^E_{best} = \epsilon^E_{new}$ and go to the previous step. Else, continue with the following variable.
3) If ϵ^E_{new} has changed go to step 2. Otherwise, stop and return the best DB definition.

D. Definition of k_0

In regression, the definition of k_0 in CCIS cannot be used, as there are infinite classes, and the error is not a misclassification one, but a difference between the real output and the predicted output. Thus, in order to obtain the number of a priori outliers, first the mean error of the complete training set is calculated. Then, k_0 is defined as the number of examples that have an error larger than the average:

$$k_0 = |\{e^l \in E \mid \epsilon^l > \epsilon^E\}| \tag{7}$$

where ϵ^l is the error of example e^l.

III. Genetic Fuzzy System with Instance Selection

CCISR can be combined with any GFS with a data-driven rule base generation method. In this paper we have selected GLD [6] as the genetic algorithm for learning RBs. The combination of GLD with CCISR, called GLD-IS, consists of three different phases (Fig. 3): K-means (Sec. III-A), CCISR (described in Sec. II) and the GLD adapted to work with instance selection (Sec. III-B).

A. K-means

In order to improve the reduction capacity of CCISR, a preprocessing stage was used. This preprocessing method consists of the execution of the K-means algorithm [23] over the training examples E_{tra}. The parameter k is selected using a Quadratic Fit searching process [24] that chooses a k that minimizes the Davies-Bouldin index [25], defined as:

$$DB_{index} = \frac{1}{k}\sum_{i=1}^{k} \max_{i \neq j} \frac{s_i + s_j}{d(c_i, c_j)} \tag{8}$$

where k is the number of clusters, c_i is the centroid of cluster i, s_i is the average distance of all elements in cluster i to centroid c_i, and $d(c_i, c_j)$ is the distance between centroids c_i and c_j. Thus, clusters with low intra-cluster distances (high intra-cluster similarity) and high inter-cluster distances (low inter-cluster similarity) will have a low Davies-Bouldin index.

After obtaining the best clusters, the example closer to each cluster centroid is added to the reduced examples set E_r. Then, the CCISR algorithm (Sec. II) is executed over E_r in order to obtain a subset of examples named E_s.

B. GLD with Instance Selection

GLD [6] is a genetic algorithm that codifies the granularities and the lateral displacements of the variables to generate accurate linguistic fuzzy rule bases. First, we briefly describe the main characteristics of GLD and, at the end of the section we define the evaluation process of GLD-IS (GLD combined with instance selection), which is different from that in GLD. Figure 4 shows the pseudocode of GLD:

- *Chromosome codification*: a double-coding scheme is used: a vector of granularities for each variable (C_1) and a vector of real numbers that represent the lateral displacement associated for each label (C_2). The second part of the chromosome corresponds to a 2-tuple representation of the labels, introduced in [26]. The symbolic translation of a linguistic term is a number within the interval $[-0.5, 0.5)$ that expresses the displacement of a label when it is moving between its two lateral labels.
- *Initialization*: the initial pool of individuals is generated by a combination of two initialization procedures. A half of the individuals are generated with the same random granularity for each variable, while the other half is created with a different random granularity for each variable. The lateral displacements are initialized to 0 in all cases.
- *Rule base generation method*: an ad-hoc method is used to construct the RB from the codified DB. The Wang & Mendel algorithm is used to create the rule base for each individual. The method is quick and simple, and obtains a representative rule base given the definition of the DB and a set of examples.
- *Evaluation*: the fitness function is a linear combination of two measures:

  $$fitness = MSE(E_{tra}) + \alpha_f NR \tag{9}$$

  where MSE is the mean squared error (Eq. 5), NR is the number of rules and α_f is a parameter that determines the tradeoff between accuracy and complexity.
- *Recombination*: two crossover operations are defined: one-point crossover for exchanging the C_1 parts and, when the C_1 are equal, the parent-centric BLX (PCBLX) [27] is used for the C_2 part.

As the algorithm searches for models with a low number of rules (compact linguistic models) and the support of the final membership functions can be displaced, there could be non-covered zones in the input space. In [6], in order to

1: pop_0 = initialization()
2: **for all** $ind_i \in pop_0$ **do**
3: $RB_i = WM(DB_i)$
4: *fitness*$_i$ = evaluation(RB_i)
5: **end for**
6: **repeat**
7: pop_{new} = recombination(pop_{sel})
8: **for all** $ind_i \in pop_{new}$ **do**
9: $RB_i = WM(DB_i)$
10: *fitness*$_i$ = evaluation(RB_i)
11: **end for**
12: pop_t = replacement($pop_t \cup pop_{new}$)
13: **until** stopping criteria
14: return $\{ind_b \in pop_t \ / \ b = argmax_i$ *fitness*$_i\}$

Fig. 4. GLD pseudocode [6].

1: Given $E_s = CCISR(E_r)$
2: **for all** $ind_i \in pop$ **do**
3: RB_i = $WM(E_s, DB_i)$
4: *fitness*$_i = MSE(E_{tra})$
5: **end for**

Fig. 5. Evaluation process of GLD-IS.

consider non-covered input data, an interpolation of the two nearest rules is used to infer the output. Our implementation of GLD, instead of this new inference method, uses the standard inference in fuzzy rule base systems: when an example is not covered by any rule, the RB returns the average output. Moreover, this classic inference is more interpretable.

1) GLD with instance selection: in this section, the modifications of GLD to work with an instance selection method is presented.

Both E_s and E_{tra} are used in GLD-IS within a redefined evaluation function. Thus, the loops in lines 2-5 and 8-11 are substituted by the loop in Fig. 5. Subset E_s is used to obtain the RB from the codified DB. In this manner, those examples that are not representative are not taken into account for rules generation, thus both avoiding the creation of too specific rules and reducing the time needed to create the RB.

After obtaining the RB, the fitness is calculated as the mean squared error (Eq. 5):

$$fitness = MSE(E_{tra}) \tag{10}$$

where E_{tra} is the full training dataset. Using all the examples in E_{tra} for evaluation can be seen, in some way, as a validation process, as the rule base was constructed with a subset of them.

IV. Results

In order to analyze the performance of CCISR and its integration with a GFS (GLD-IS), we have used 12 real-world regression problems with different complexity. These problems were obtained from the KEEL project [28]. Table I shows the characteristics of the datasets.

A. Performance of CCISR

In order to evaluate the performance of CCISR, two measures are considered:

TABLE I. Characteristics of the datasets

Dataset	Abbr.	Variables	Examples
Electrical Maintenance	ELE	4	1056
Auto MPG6	MPG6	5	398
Auto MPG8	MPG8	7	398
Anacalt	ANA	7	4052
Abalone	ABA	8	4177
Stock	STO	9	950
Weather Izmir	WIZ	9	1461
Weather Ankara	WAN	9	1609
Forest Fires	FOR	12	517
Mortgage	MOR	15	1049
Treasury	TRE	15	1049
Baseball	BAS	16	337

Available at `http://www.keel.es` [28]

TABLE II. Best data bases obtained by the greedy algorithm (Sec. II-C)

Dataset	Granularities	MSE_{tra}
ELE	[4, 7, 7, 7, 7]	38,388.806
MPG6	[5, 6, 6, 7, 7, 7]	2.669
MPG8	[6, 6, 6, 7, 6, 6, 7, 7]	2.181
ANA	[7, 1, 2, 2, 2, 7, 2, 7]	0.012
ABA	[7, 6, 7, 5, 5, 7, 7, 7, 6]	2.703
STO	[6, 6, 6, 6, 6, 6, 6, 7, 7, 7]	0.505
WIZ	[7, 7, 5, 4, 6, 5, 6, 5, 6, 7]	1.789
WAN	[6, 6, 7, 5, 5, 6, 6, 7, 6, 7]	2.373
FOR	[5, 7, 4, 7, 2, 6, 6, 4, 6, 6, 6, 1, 7]	169.588
MOR	[5, 4, 5, 5, 7, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 7]	0.050
TRE	[4, 6, 6, 6, 6, 6, 5, 6, 6, 7, 7, 5, 6, 5, 7, 7]	0.112
BAS	[7, 5, 6, 5, 6, 6, 6, 7, 6, 7, 6, 6, 2, 2, 2, 1, 7]	28,627.247

- Reduction (red): is the percentage of reduction in the number of examples, defined as:

$$red_{E_{sub}} = 1 - \frac{|E_{sub}|}{|E_{tra}|} \tag{11}$$

where $|E_{sub}|$ is the number of examples in the subset E_{sub} and $|E_{tra}|$ is the original number of examples in the training set.

- Increase of error (inc): is the increment of the error after applying the instance selection process. In order to compare the different configurations of CCISR, the MSE error defined in Eq. 5 was used. Thus, Wang & Mendel was applied to obtain the rules from the examples, before and after selection, and the granularities were obtained by the same greedy algorithm as in CCISR (sec. II-C). Finally, the increase in the error is defined as:

$$inc_{E_{sub}} = \frac{MSE_{sub}}{MSE_{tra}} \tag{12}$$

where MSE_{sub} is the MSE obtained when the RB is built from the examples in E_{sub}, and MSE_{tra} when all the examples in the training set are used to construct the RB. The MSE_{tra} obtained for each dataset, together with the selected granularities for each variable, are shown in table II.

Table III shows the average values of *inc* and *red* with 30 runs for each dataset. The K-means column shows the results obtained after applying K-means to E_{tra}, while the CCISR column represents the results when CCISR is applied directly to E_{tra}, without preprocessing the dataset with K-Means. Finally, the K-means + CCISR column shows the

TABLE III. AVERAGE RESULTS FOR K-MEANS AND CCISR (SEC. II-C)

Dataset	K-means		CCISR		K-means + CCISR		
	red_{E_r}	inc_{E_r}	red_{E_s}	inc_{E_s}	red_{E_s}	inc_{E_s}	time (s)
ELE	0.498	1.551	0.672	1.776	0.802	3.610	529.926
MPG6	0.499	1.986	0.555	2.714	0.776	4.730	58.498
MPG8	0.429	2.242	0.588	4.421	0.768	6.449	75.499
ANA	0.446	2.200	0.380	2.693	0.633	4.703	613.526
ABA	0.535	1.084	0.585	1.167	0.806	1.214	14,223.440
STO	0.521	1.264	0.520	1.575	0.779	2.832	470.400
WIZ	0.475	1.723	0.646	4.158	0.823	5.841	1,527.727
WAN	0.512	2.010	0.637	4.310	0.847	6.937	2,049.829
FOR	0.444	287.874	0.550	410.772	0.779	621.148	371.793
MOR	0.528	1.404	0.539	2.568	0.790	6.997	1,025.008
TRE	0.536	1.183	0.535	1.710	0.786	4.615	907.456
BAS	0.472	19.006	0.812	38.816	0.893	48.886	148.269

TABLE IV. AVERAGE RESULTS OF THE DIFFERENT ALGORITHMS

Dataset	GLD-IS			GLD			FSMOGFS [13]			FSMOGFS^e [13]		
	Rules	Training	Test	Rules	Training	Test	Rules	Training	Test	Rules	Training	Test
ELE	31	17,743.610	19,784.872	31	13,041.375	**15,147.656**	10	16,018.000	16,083.000	**9**	16,153.000	16,338.000
MPG6	**32**	3.158	**4.375**	69	2.560	4.995	38	3.850	4.820	39	3.894	4.866
MPG8	**38**	3.029	5.036	121	1.932	4.876	40	3.827	**4.453**	40	3.827	**4.453**
ANA	78	0.016	0.017	100	0.006	0.008	25	0.006	**0.006**	**23**	0.006	**0.006**
ABA	69	2.487	2.593	29	2.457	**2.516**	17	2.682	2.697	**15**	2.670	2.708
STO	113	0.401	0.600	195	0.353	**0.499**	44	1.361	1.460	**43**	1.393	1.456
WIZ	70	1.073	1.747	95	0.944	**1.249**	23	1.469	1.567	**17**	1.519	1.571
WAN	71	1.610	2.543	142	1.227	**1.658**	12	1.810	1.823	**11**	1.897	2.151
FOR	51	1,771.497	3,131.715	356	80.093	58,376.609	**34**	1,873.000	**2,254.000**	35	1,892.000	2,449.000
MOR	62	0.032	0.056	83	0.023	**0.028**	12	0.032	0.033	**11**	0.033	0.034
TRE	56	0.043	0.061	68	0.035	**0.047**	17	0.046	0.049	**15**	0.046	0.052
BAS	**18**	402,978.818	475,295.087	250	30,954.190	870,956.207	33	167,300.000	257,500.000	28	170,600.000	**248,300.000**

results when the whole process of instance selection in GLD-IS (Fig. 3) is performed. The values of the parameters for these experiments were: $\alpha_{D_M} = 0.05$ and a relative tolerance with respect to the Davies-Bouldin index to declare convergence in K-Means equal to 10^{-6}. Moreover, the results were obtained in an AMD Opteron 6262 HE 2.10 GHz processor with 128 GB RAM, using only one of the eight cores. The time column in K-means + CCISR shows the runtime of the whole process —including the selection of the number of clusters.

The percentage of reduction achieved by K-means is, in general, under 0.5, while CCISR gets an average reduction over 0.5 in most of the datasets. Also, the combination of both methods shows the best reduction rate, with values between 0.7 and 0.9 in all the datasets but ANA. Furthermore, the reduction rate does not depend neither in the size of the dataset, nor in the number of variables, but on the complexity of the data.

On the other hand, the increase in error of K-means is low, so this phase does not modify significantly the information contained in the examples set. CCISR produces an increase in the error with respect to K-means, as the reduction is also higher. Generally, the increase in the error is not very high, except for FOR and BAS. These two datasets have a low number of examples combined with a high dimensionality (table I) and, therefore, reducing the number of examples has a more direct effect in the increase of the error. However, this misbehavior does not have a great impact in the subsequent learning phase with GLD-IS, as we show in the next section.

B. Performance of GLD-IS

In order to analyze the performance of GLD-IS we have compared the results obtained with different algorithms:

- GLD: an implementation of the algorithm presented in [6] (sec. III). The only difference with the original GLD is that we used the standard fuzzy inference method.
- FSMOGFS [13]: a multi-objective evolutionary algorithm (based on SPEA2) that performs an embedded

TABLE V. AVERAGE RANKINGS FOR THE NUMBER OF RULES OF TABLE IV

Algorithm	Ranking
FSMOGFS^e	1.458
FSMOGFS	2.042
GLD-IS	2.583
GLD	3.917
Friedman p-value: 0.00003	

TABLE VI. POST HOC COMPARISON FOR THE NUMBER OF RULES OF TABLE IV AND $\alpha = 0.05$

i	Comparison	z	p	α/i	Hypothesis
4	GLD-IS vs GLD	2.53	0.011	0.013	Rejected
3	FSMOGFS^e vs GLD-IS	2.135	0.033	0.017	Accepted
1	FSMOGFS vs GLD-IS	1.028	0.304	0.05	Accepted

TABLE VII. AVERAGE RANKINGS FOR THE TEST ERROR OF TABLE IV (FRIEDMAN)

Algorithm	Ranking
FSMOGFS	2.083
GLD	2.083
FSMOGFS^e	2.583
GLD-IS	3.25
Friedman p-value: 0.085	

TABLE VIII. POST HOC COMPARISON FOR THE TEST ERROR OF TABLE IV AND $\alpha = 0.05$

i	Comparison	z	p	α/i	Hypothesis
6	FSMOGFS vs GLD-IS	2.214	0.027	0.008	Accepted
5	GLD vs GLD-IS	2.214	0.027	0.01	Accepted
4	FSMOGFS^e vs GLD-IS	1.265	0.206	0.013	Accepted

genetic DB learning including feature selection, granularities and lateral displacement of fuzzy partitions in order to control the dataset dimensionality and obtain a reduced KB. For each DB definition the Wang & Mendel method is used to obtain the RB with a cropping system that stops the method when the RB reaches a limit of 50 rules. Two minimization objectives are used: *MSE* and number of rules.

- FSMOGFS^e [13]: a modification of FSMOGFS, where a fast error-computation mechanism is used to reduce the long time consumed by error computation in large-scale datasets. This algorithm takes a small percentage of the training examples to estimate the error of bad solutions, and only uses all the examples to evaluate good candidate solutions.

There is also a version of FSMOGFS, called FSMOGFS+ TUNthat includes a post-processing stage. This second stage consists in a fine tunning of the fuzzy labels parameters and a rule selection through an evolutionary process. In this paper we are only interested in the learning stage and, although GLD-IS could be run with a tuning stage, for comparison purposes we have not considered the post-processing stage in the algorithms.

The following values were used for the parameters of GLD and GLD-IS: population size of 61, 50,000 evaluations, and the maximum granularity equal to 7. Moreover, for GLD: $\alpha_f = 0.5$. Furthermore, in the case of the multi-objective algorithms, we have considered a standard population of 200 and an external population of 61. For the fast error-computation, the percentage of examples used is 0.2. For each dataset, we executed six times the algorithm (with different seeds) and, for each of them, a five fold cross-validation was done, giving a total of 30 runs.

Table IV shows the average values for the 30 runs for each algorithm and dataset. The information shown in the columns is: the number of rules of the RB, and the training and test error measured by Eq. 5. Additionally, in order to compare the experimental results, non-parametric tests of multiple comparisons have been applied. The Friedman test [29] with Holm post-hoc test [30] was used as the method for detecting significant differences among the results. In order to analyze the comparisons in which GLD-IS was involved, the multiple comparison approach for the Holm test was used. The test was performed both for the test error and the number of rules of table IV. The results are shown in tables V to VIII.

In the case of number of rules (table V), both multi-objective algorithms have the best performance, as these algorithms limit the size of the learned RB to 50 rules. However, the post-hoc results (table VI) show that there are no statistically significant differences comparing the multi-objective algorithms with GLD-IS. In particular, in the case of STO (table IV), the high number of rules in GLD-IS is a consequence of the complexity of the problem, and for FSMOGFS and FSMOGFS^e the limit in the number of rules produces a bad test error. Moreover, GLD has the worst ranking in Friedman, and compared with GLD-IS the difference in number of rules is statistically significant.

Taking into account the test error (table VII), the algorithms that calculate the error with all the training data (FSMOGFS and GLD) have the highest ranking, while FSMOGFS^e and GLD-IS have the worst ranking. However, the p-value of the test in table VII is over 0.05, showing that the significance level of the Friedman test does not have a high level of confidence. Furthermore, the post-hoc method (table VIII) shows that there are no significant differences between GLD-IS and all the other algorithms.

As a summary of the tests, we can conclude that GLD-IS offers an alternative to reduce the variance in evolutionary learning, producing rule bases with a low number of rules. Moreover, the accuracy of the obtained rule bases is also good, showing no statistically significant differences with other methods that control the variance through a regularization parameter (GLD) or with a multi-objective approach combined with a threshold in the number of rules (FSMOGFS and FSMOGFS^e).

In relation with the effect of CCISR combined with GLD, even when using a low number of examples to generate the RB, the results obtained by GLD-IS are good (table IV). This is due to three reasons. First, the reduced examples set contains only the meaningful examples and, therefore, the rule base does not contain very specific rules or rules generated from outliers. In second place, the search space is smaller, as the number of rules bases generated by the examples set is also reduced. Finally, the use of all the training examples to evaluate a RB that was generated by a subset of the examples allows to measure the generalization ability of the learned RB during the evolutionary process.

V. Conclusions

In this paper we have presented a new IS method for regression based on CCIS, called CCISR. In order to adapt CCIS to work with regression problems, three modifications of the algorithm have been done: (i) the construction of the graphs, (ii) the error measure and (iii) the k_0 parameter. Moreover, we have integrated CCISR with a GFS in order to obtain fuzzy rule bases with a low variance. The full system, called GLD-IS, has three different stages: a preprocessing phase based on K-means, CCISR and, finally, a genetic algorithm to learn the knowledge base. The RBs obtained during the evolutionary process were constructed by an *ad hoc* data driven method guided by the selected instances from CCISR, while the error measure was calculated using the full training dataset.

CCISR has been tested with 12 regression problems, showing a good reduction rate, while keeping the most meaningful examples. Also, we have compared GLD-IS, the combination of GLD with instance selection, with three GFS approaches. The results show that GLD-IS obtains a low number of rules as RB generation is based on the selected instances, while still having a good accuracy. Therefore, GLD-IS offers a different and elegant way to reduce the variance of GFSs, and is an alternative to the use of regularization parameters or thresholds in the number of rules.

Acknowledgments

This work was supported by the Spanish Ministry of Economy and Competitiveness under grants TIN2011-22935 and TIN2011-29827-C02-02. I. Rodriguez-Fdez is supported by the Spanish Ministry of Education, under the FPU national

plan (AP2010-0627). M. Mucientes is supported by the Ramón y Cajal program of the Spanish Ministry of Economy and Competitiveness. This work was supported in part by the European Regional Development Fund (ERDF/FEDER) under the projects CN2012/151 and CN2011/058 of the Galician Ministry of Education.

REFERENCES

[1] E. Marchiori, "Class conditional nearest neighbor for large margin instance selection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 32, no. 2, pp. 364–370, 2010.

[2] J. Friedman, T. Hastie, and R. Tibshirani, *The elements of statistical learning*. Springer Series in Statistics, 2001, vol. 1.

[3] K. Ying, S. Lin, Z. Lee, I. Lee *et al.*, "A novel function approximation based on robust fuzzy regression algorithm model and particle swarm optimization," *Applied Soft Computing*, vol. 11, no. 2, pp. 1820–1826, 2011.

[4] C. Juang, R. Huang, and W. Cheng, "An interval type-2 fuzzy-neural network with support-vector regression for noisy regression problems," *IEEE Transactions on Fuzzy Systems*, vol. 18, no. 4, pp. 686–699, 2010.

[5] C. Juang and C. Hsieh, "A fuzzy system constructed by rule generation and iterative linear svr for antecedent and consequent parameter optimization," *IEEE Transactions on Fuzzy Systems*, vol. 20, no. 2, pp. 372–384, 2012.

[6] R. Alcalá, J. Alcalá-Fdez, F. Herrera, and J. Otero, "Genetic learning of accurate and compact fuzzy rule based systems based on the 2-tuples linguistic representation," *International Journal of Approximate Reasoning*, vol. 44, no. 1, pp. 45–64, 2007.

[7] O. Cordon, F. Gomide, F. Herrera, F. Hoffmann, and L. Magdalena, "Ten years of genetic fuzzy systems: current framework and new trends," *Fuzzy sets and systems*, vol. 141, no. 1, pp. 5–31, 2004.

[8] F. Herrera, "Genetic fuzzy systems: taxonomy, current research trends and prospects," *Evolutionary Intelligence*, vol. 1, no. 1, pp. 27–46, 2008.

[9] M. Fazzolari, R. Alcala, Y. Nojima, H. Ishibuchi, and F. Herrera, "A review of the application of multi-objective evolutionary fuzzy systems: Current status and further directions," *IEEE Transactions on Fuzzy Systems*, 2012.

[10] J. Casillas, P. Martínez, and A. Benítez, "Learning consistent, complete and compact sets of fuzzy rules in conjunctive normal form for regression problems," *Soft Computing-A Fusion of Foundations, Methodologies and Applications*, vol. 13, no. 5, pp. 451–465, 2009.

[11] R. Alcalá, P. Ducange, F. Herrera, B. Lazzerini, and F. Marcelloni, "A multiobjective evolutionary approach to concurrently learn rule and data bases of linguistic fuzzy-rule-based systems," *IEEE Transactions on Fuzzy Systems*, vol. 17, no. 5, pp. 1106–1122, 2009.

[12] M. Antonelli, P. Ducange, B. Lazzerini, and F. Marcelloni, "Learning concurrently data and rule bases of mamdani fuzzy rule-based systems by exploiting a novel interpretability index," *Soft Computing-A Fusion of Foundations, Methodologies and Applications*, vol. 15, no. 10, pp. 1981–1998, 2011.

[13] R. Alcalá, M. Gacto, and F. Herrera, "A fast and scalable multi-objective genetic fuzzy system for linguistic fuzzy modeling in high-dimensional regression problems," *IEEE Transactions on Fuzzy Systems*, vol. 19, no. 4, pp. 666–681, 2011.

[14] D. Wilson and T. Martinez, "Reduction techniques for instance-based learning algorithms," *Machine learning*, vol. 38, no. 3, pp. 257–286, 2000.

[15] J. Bezdek and L. Kuncheva, "Nearest prototype classifier designs: An experimental study," *International Journal of Intelligent Systems*, vol. 16, no. 12, pp. 1445–1473, 2001.

[16] N. Jankowski and M. Grochowski, "Comparison of instances seletion algorithms i. algorithms survey," *Proceedings of Artificial Intelligence and Soft Computing (ICAISC) 2004*, pp. 598–603, 2004.

[17] I. Triguero, J. Derrac, S. Garcia, and F. Herrera, "A taxonomy and experimental study on prototype generation for nearest neighbor classification," *IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews*, vol. 42, no. 1, pp. 86–100, 2012.

[18] S. Garcia, J. Derrac, J. Cano, and F. Herrera, "Prototype selection for nearest neighbor classification: Taxonomy and empirical study," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 34, no. 3, pp. 417–435, 2012.

[19] B. Giglio, F. Marcelloni, M. Fazzolari, R. Alcala, and F. Herrera, "A case study on the application of instance selection techniques for genetic fuzzy rule-based classifiers," in *Proceedings of IEEE International Conference on Fuzzy Systems (FUZZ-IEEE) 2012*, 2012, pp. 1–8.

[20] A. Guillen, L. Herrera, G. Rubio, H. Pomares, A. Lendasse, and I. Rojas, "New method for instance or prototype selection using mutual information in time series prediction," *Neurocomputing*, vol. 73, no. 10, pp. 2030–2038, 2010.

[21] M. Antonelli, P. Ducange, and F. Marcelloni, "Genetic training instance selection in multiobjective evolutionary fuzzy systems: A coevolutionary approach," *IEEE Transactions on Fuzzy Systems*, vol. 20, no. 2, pp. 276–290, 2012.

[22] L. Wang and J. Mendel, "Generating fuzzy rules by learning from examples," *IEEE Transactions on Systems, Man and Cybernetics*, vol. 22, no. 6, pp. 1414–1427, 1992.

[23] J. MacQueen *et al.*, "Some methods for classification and analysis of multivariate observations," in *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*, vol. 1, no. 281-297. California, USA, 1967, p. 14.

[24] D. G. Luenberger and Y. Ye, *Linear and nonlinear programming*. Springer, 2008, vol. 116.

[25] D. L. Davies and D. W. Bouldin, "A cluster separation measure," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. PAMI-1, no. 2, pp. 224–227, 1979.

[26] F. Herrera and L. Martnez, "A 2-tuple fuzzy linguistic representation model for computing with words," *IEEE Transactions on Fuzzy Systems*, vol. 8, pp. 746–752, 1999.

[27] F. Herrera, M. Lozano, and A. Sánchez, "A taxonomy for the crossover operator for real-coded genetic algorithms: An experimental study," *International Journal of Intelligent Systems*, vol. 18, no. 3, pp. 309–338, 2003.

[28] J. Alcalá-Fdez, L. Sánchez, S. García, M. Del Jesus, S. Ventura, J. Garrell, J. Otero, C. Romero, J. Bacardit, V. Rivas *et al.*, "Keel: a software tool to assess evolutionary algorithms for data mining problems," *Soft Computing*, vol. 13, no. 3, pp. 307–318, 2009.

[29] M. Friedman, "The use of ranks to avoid the assumption of normality implicit in the analysis of variance," *Journal of the American Statistical Association*, vol. 32, no. 200, pp. 675–701, 1937.

[30] S. Holm, "A simple sequentially rejective multiple test procedure," *Scandinavian journal of statistics*, pp. 65–70, 1979.