International Journal of Approximate Reasoning 52 (2011) 728–750

Contents lists available at ScienceDirect

International Journal of Approximate Reasoning

journal homepage: www.elsevier.com/locate/ijar

Intelligent data analysis and model interpretation with spectral analysis fuzzy symbolic modeling

Alexandre G. Evsukoff[a,*], Antonio C.S. Branco[b], Sylvie Galichet[c]

[a] COPPE/Federal University of Rio de Janeiro, P.O. Box 68506, 21941-972 Rio de Janeiro, RJ, Brazil
[b] EMAp/FGV-Getúlio Vargas Foundation, P.O. Box 62591, 22250-900 Rio de Janeiro, RJ, Brazil
[c] LISTIC/Polytech Annecy-Chambéry, University of Savoie, BP 80439, 74944 Annecy le Vieux, France

ARTICLE INFO

Article history:
Received 24 April 2010
Received in revised form 25 January 2011
Accepted 27 January 2011
Available online 15 February 2011

Keywords:
Fuzzy system models
Symbolic modeling
Spectral analysis
Pattern recognition
Function approximation
Learning

ABSTRACT

This paper proposes fuzzy symbolic modeling as a framework for intelligent data analysis and model interpretation in classification and regression problems. The fuzzy symbolic modeling approach is based on the eigenstructure analysis of the data similarity matrix to define the number of fuzzy rules in the model. Each fuzzy rule is associated with a symbol and is defined by a Gaussian membership function. The prototypes for the rules are computed by a clustering algorithm, and the model output parameters are computed as the solutions of a bounded quadratic optimization problem. In classification problems, the rules' parameters are interpreted as the rules' confidence. In regression problems, the rules' parameters are used to derive rules' confidences for classes that represent ranges of output variable values. The resulting model is evaluated based on a set of benchmark datasets for classification and regression problems. Nonparametric statistical tests were performed on the benchmark results, showing that the proposed approach produces compact fuzzy models with accuracy comparable to models produced by the standard modeling approaches. The resulting model is also exploited from the interpretability point of view, showing how the rule weights provide additional information to help in data and model understanding, such that it can be used as a decision support tool for the prediction of new data.

1. Introduction

One of the most widely used models in soft computing, which has been developed within many different modeling frameworks, is written as:

$$\hat{f}(\mathbf{x},\boldsymbol{\theta},\boldsymbol{\Omega}) = \sum_{i=1}^{n} g_i(\mathbf{x},\boldsymbol{\Omega}_i)\theta_i, \tag{1}$$

where $\mathbf{x} \in R^p$ is the vector of the input variables; $g_i(\mathbf{x}, \Omega_i)$, $i = 1,\ldots,n$ are the basis functions, each of which is parameterized by the set of parameters Ω_i, and $\boldsymbol{\theta} = [\theta_1,\ldots,\theta_n]$ is the output parameters vector. Usually, the interpolation function $g(\mathbf{x}, \Omega)$ is the Gaussian function:

$$g_i(\mathbf{x},\boldsymbol{\omega}_i,\sigma_i) = \exp\left(-\frac{\|\mathbf{x}-\boldsymbol{\omega}_i\|}{2\sigma_i}\right). \tag{2}$$

The set of parameters $\Omega_i = \{\boldsymbol{\omega}_i, \sigma_i\}$, where $\boldsymbol{\omega}_i \in R^p$ is the center of the Gaussian function and σ_i is the width parameter.

* Corresponding author.
E-mail addresses: alexandre.evsukoff@coc.ufrj.br (A.G. Evsukoff), antonio.branco@fgv.br (A.C.S. Branco), sylvie.galichet@univ-savoie.fr (S. Galichet).

The model defined by Eqs. (1) and (2) has been extensively studied in supervised (or prediction) learning problems in many classification (or pattern recognition) and regression (or function approximation) applications. This model is described as a radial basis function (RBF) neural network [1,2], which is equivalent to the (unbiased) support vector machine (SVM) [3,4]. These two models have also been recognized to be equivalent to certain types of fuzzy rule-based systems, both in the RBF neural network framework [5,6] and, more recently, in the SVM framework [7–9].

Several methods and algorithms have been described in the literature for the estimation of parameters given a training set of input–output data samples. Most of these methods differ in how to define the dimension of the feature space that is spawned by the set of the interpolation functions (2), i.e., the number n of interpolating functions that should be used in the model (1). There are several strategies for estimating the number of radial basis functions in the RBF neural network framework including orthogonal least squares (OLS) methods [16,17], genetic algorithms (GA) [18–20], and hybrid methods [21,22]. Using the structural risk minimization paradigm, a tolerance for the error of the model is fixed and the model structure is minimized [4], obtaining the number of support vectors as a byproduct.

In the fuzzy systems framework, the feature space dimension is related to the number of rules. In the earlier approaches, the number of rules was defined *a priori* based on expert advice [5]. Some approaches have also been proposed based on reducing the rule base through orthogonal transforms [23,24]. Since the mid 1990s, the structure optimization of fuzzy systems using GA-based strategies has being a very active research area [25–27]. More recently, some approaches for the design of fuzzy systems based on the SVM algorithm have also been proposed [7–9]. The numerical accuracy of the results is more often emphasized, but the interpretability of the rule base has recently been brought to attention [10–15], especially when using the genetic fuzzy systems approach [28–34].

A fuzzy rule described by a multidimensional Gaussian membership function (2) cannot be directly translated into a linguistic expression such as a fuzzy rule written as a set of variable-value pairs. According to the terminology proposed by Zou and Gan [12], multidimensional fuzzy rules can achieve only high-level interpretability, whereas fuzzy rules written as a set of variable-value pairs can have both high- and low-level interpretability. On the other hand, in many real world problems, the fuzzy model needs a large number of rules that relate several variable-value pairs on the premise in order to achieve the desired accuracy and, consequently, the model becomes very difficult to interpret linguistically [14,15].

The interpretability of fuzzy rules should not be regarded as an end in itself, but as a means for human experts to develop insight into complex problems. The model and the modeling methodology must be considered as a whole in the task of problem understanding. The analysis of the weights of the fuzzy rules for model interpretation can partially overcome the drawback of the lack of linguistic interpretability of multidimensional fuzzy rules. Moreover, the graphical visualization of data provides a rough interpretation of the problem that allows the analyst to obtain insight into the problem, supporting future decision making in the analysis of new data.

In this work, a methodology for design and interpretation of fuzzy models in the form (1) and (2) is proposed. The main contribution of this work is on the interpretation framework based on assigning the interpolation functions (2) to symbols that are related to prototype values. A learning methodology, compatible with the proposed interpretation framework, is also presented. The resulting model is expressed by a compact but accurate fuzzy rule base. The analysis of the weights of the fuzzy rules and the graphical visualization of the data and rules provide additional information for human experts to guide the prediction of new data.

The learning procedure is based on spectral analysis [32] to determine the number of multidimensional rules, as described by the Gaussian membership function (2). The model identification is performed in two steps: rule induction and parameter estimation. In the rule induction task, the number of relevant fuzzy rules is computed by spectral analysis, and a clustering algorithm computes the positions of their centers. In the parameter estimation task, the output parameters of the rules are computed by solving a bounded quadratic optimization problem.

This paper is organized as follows. In the next section, the symbolic approach for fuzzy modeling is introduced to set up the notation and to present the parameters that should be estimated by the identification algorithm. Section 3 presents the spectral analysis used to determine the number of rules in the model. Section 4 presents the rule induction algorithm and the formulation of the optimization problem for parameter estimation. Section 5 presents the interpretation of the model and graphical visualization. In Section 6, the performance results in classification and regression benchmark problems are presented and evaluated using nonparametric statistical tests. The interpretation and exploitation of the model is also discussed in Section 6. Concluding remarks and the directions for future research are presented in Section 7.

2. Fuzzy symbolic modeling

2.1. Problem statement

Consider a dataset composed of N input–output data samples $\mathcal{T} = \{(\mathbf{x}(t), \mathbf{y}(t)), t = 1, \ldots, N\}$, where $\mathbf{x} \in \mathcal{R}^p$ is the input variables vector and $\mathbf{y} \in \mathcal{R}^q$ is the output variables vector. A nominal valued input (output) variable is modeled as a discrete variable $x_i \in \mathcal{N} (y_i \in \mathcal{N})$, where $\mathcal{N}$ is the set of natural numbers. Consider also a set of input symbols $\mathcal{A} = \{A_i, i = 1, \ldots, n\}$ and a set of output symbols $\mathcal{B} = \{B_j, j = 1, \ldots, m\}$, of which each element refers qualitatively to the values of the input and output variable domains, respectively.

The objective of the learning problem is to compute an approximation to the unknown relationship $\mathbf{y} = f(\mathbf{x})$, which is supposed to exist among input–output data. The learning problems addressed in this work are:

1. *Classification*: there is only one ($q = 1$) categorical output variable ($y \in \mathcal{N}$) and the value $y = j$ refers to a class $B_j \in \mathbf{B}$.
2. *Regression*: the output variables are numeric and an approximation $\hat{y}_i = \hat{f}_i(\mathbf{x})$ is built for each output variable for multiple-output systems.

Only the single-output model is considered in this work, without loss of generality, because a multiple-output system can usually be modeled as a collection of single-output models. The objective of the model is to compute an approximation $\hat{y}(t) = \hat{f}(\mathbf{x}(t))$, such that it can be represented as a set of fuzzy rules $A_i \to B_j$ in the general form:

$$\textit{if } \mathbf{x}(t) \textit{ is } A_i \quad \textit{then} \quad y(t) \textit{ is } B_j. \tag{3}$$

The resulting model is a fuzzy system herein called the fuzzy symbolic model (FSM), which is computed in the three usual steps, implemented using the following operators:

- Fuzzification: $\hat{\mathbf{u}}(t) = F(\mathbf{x}(t), \mathbf{\Omega})$.
- Inference: $\hat{\mathbf{v}}(t) = I(\hat{\mathbf{u}}(t), \mathbf{\Phi})$.
- Defuzzification: $\hat{y}(t) = D(\hat{\mathbf{v}}(t), \boldsymbol{\beta})$.

The operators and their parameters are presented in the following.

2.2. Fuzzification

Fuzzification is the mapping $F : \mathcal{R}_p \to [0,1]^n$ from the p-dimensional input variable domain to an n-dimensional embedding or feature space. Each fuzzy membership function is computed using the Gaussian function (2) and is related to a symbol $A_i \in \mathcal{A}$ that represents a region on the multi-dimensional input variable domain. The membership function center $\boldsymbol{\omega}_i \in \mathcal{R}^p$ can be either a point of the training dataset or a cluster center and it represents a prototype value, i.e., the best representative of the symbol A_i.

An observed sample $\mathbf{x}(t) \in \mathcal{R}^p$ is mapped into a feature (or fuzzy) vector $\mathbf{u}(t) \in [0,1]^n$, whose components are computed as:

$$\mathbf{u}(t) = [u_1(t), \ldots, u_n(t)] = \left[\mu_{A_1}(\mathbf{x}(t)), \ldots, \mu_{A_n}(\mathbf{x}(t))\right], \tag{4}$$

where each component is computed using the Gaussian function (2), i.e., $u_i(t) = g_i(\mathbf{x}, \boldsymbol{\omega}_i, \sigma_i)$.

The result of the fuzzification is normalized before its use in the inference process as:

$$\hat{\mathbf{u}}(t) = \frac{\mathbf{u}(t)}{\sum_{i=1..n} u_i(t)}. \tag{5}$$

The Gaussian membership function (2) can be seen as a similarity function, such that the normalization divisor $s(t) = \sum_{i=1,\ldots,n} u_i(t)$ represents how much the record $\mathbf{x}(t)$ is similar to (or activates) the rules.

2.3. Inference

Inference is the mapping I: $[0,1]^n \to [0,1]^m$, where n is the number of rules and m is the number of output symbols. In classification problems, the output symbols are related to the classes. In regression problems, the output symbols are related to the output variable's fuzzy partition.

A symbolic model is represented by the fuzzy relation matrix $\mathbf{\Phi} \in [0,1]^{n \times m}$, of which each component $\varphi_{ij} = \mu_{\Phi}(A_i, B_j)$ represents the confidence of the rule $A_i \to B_j$, such that the weighted rule is written as:

$$\textit{if} \quad \mathbf{x}(t) \textit{ is } A_i \quad \textit{then} \quad y(t) \textit{ is } (B_1/\varphi_{i1}, \ldots, B_m/\varphi_{im}). \tag{6}$$

The fuzzy inference is computed using the sum-product composition operator. This produces a linear mapping in the feature space, written in the vector–matrix form as:

$$\hat{\mathbf{v}}(t) = \hat{\mathbf{u}}(t) \cdot \mathbf{\Phi}. \tag{7}$$

The output fuzzy symbols represent the classes and can be considered independently, such that $\mathbf{\Phi} = [\boldsymbol{\varphi}_1|, \ldots, |\boldsymbol{\varphi}_m]$. In this case, the general model (1) is obtained as the components of the vector $\hat{\mathbf{v}} \in [0,1]^m$, computed as:

$$\hat{v}_j(t) = \hat{\mathbf{u}}(t) \cdot \boldsymbol{\varphi}_j, \quad j = 1 \ldots m, \tag{8}$$

where $\boldsymbol{\varphi}_j$ is the vector of rule confidences related to the class B_j and $\hat{v}_j(t)$ is the class membership computed for the sample $(\mathbf{x}(t), y(t))$, such that $\hat{v}_j(t) = \mu_{B_j}(y(t))$.

The use of confidence factors to express the certainty of fuzzy rules has been investigated recently for classification problems [32–37]. Rule weights allow flexibility in the design of the model and provide additional information on the quality of the rules. In regression problems, the rule base weights can be obtained from the rule output parameters and interpreted as rule confidences. The resulting weights are also used to interpret and exploit the model, as discussed in Section 5.

2.4. Defuzzification

In classification problems, defuzzification is the mapping $D : [0,1]^m \rightarrow \mathcal{N}$ that computes the class output index, based on the output fuzzy symbol. Generally, the maximum rule is used, such that the class index is computed as the component with the greatest membership value:

$$\hat{y}(t) = j : \hat{v}_j(t) = \max(\hat{\mathbf{v}}(t)). \tag{9}$$

In regression problems, defuzzification is the mapping $D : [0,1]^m \rightarrow \mathcal{R}$, where m is the number of output symbols. The fuzzy partition of the output variable is considered to be computed by normalized and triangular-shaped membership functions, centered at prototype points $\boldsymbol{\beta} = [\beta_1 | \cdots | \beta_m], \beta_j \in \mathcal{R}$, such that $\sum_j \mu_{B_j}(y) = 1,\ \forall y$. The symbolic model output can thus be computed as:

$$\hat{y}(t) = \hat{\mathbf{v}}(t)\boldsymbol{\beta}, \tag{10}$$

where $\hat{\mathbf{v}}(t)$ is computed as in (7). Substituting (7) into (10), the model output is written as:

$$\hat{y}(t) = \hat{\mathbf{u}}(t)\boldsymbol{\Phi}\boldsymbol{\beta} = \hat{\mathbf{u}}(t)\boldsymbol{\theta}. \tag{11}$$

Model (11) is equivalent to model (1), considering normalized interpolating functions as (5). This model is referred in the literature as the Takagi–Sugeno (T–S) fuzzy model [6], and the rules are written as:

$$\textit{if} \quad \mathbf{x}(t) \textit{ is } A_i \quad \textit{then} \quad y(t) = \theta_i. \tag{12}$$

In the T–S fuzzy model, defuzzification is integrated into inference. Nevertheless, both models are equivalent when the rule output parameters are related to the rule weights as:

$$\boldsymbol{\theta} = \boldsymbol{\Phi}\boldsymbol{\beta}. \tag{13}$$

The design of the FSM is presented in the next two sections. The next section presents the spectral analysis that is used to determine the number of fuzzy rules. Section 4 presents the rule induction algorithm and the parameter estimation.

3. Spectral analysis

The spectral decomposition theorem has been used in principal component analysis (PCA), which is one of the most widely used multivariate statistic methods to extract the underlying structure of high-dimensional data. The main objective of PCA, also known as the Karhunen–Loève transform, is to eliminate redundancy by linearly projecting the observed variables into a reduced space, spanned by the eigenvectors of the covariance matrix, called the principal components. The PCA is also useful for data visualization, projecting the dataset into the 2D subspace of the two first principal components.

There are several extensions of linear PCA to the non-linear case [4]. In kernel PCA, the eigenvalue decomposition is applied to the data covariance matrix in the feature space. The number of components in kernel PCA corresponds to the number of large eigenvalues of the (centered) kernel matrix. The problem is that the meaning of “large” is generally problem-dependant. The objective of spectral analysis in this work is to provide an estimation of the number of rules, which may not have necessarily the same meaning as the number of clusters. The algorithm introduced by Ng et al. [44] has been recently used to compute the number of fuzzy rules in classification problems with good results [32].

The main tools of spectral analysis are derived from spectral graph theory [39], which is based on the definition of a graph $G(\mathcal{V},\mathcal{E})$ associated to the training dataset, of which the set of nodes $\mathcal{V}$ represents the N input variable records and the set of edges $\mathcal{E}$ is defined by the $N \times N$ adjacency (or affinity) matrix $\mathbf{A}$. The graph is undirected and weighted, such that the affinity matrix is symmetric, real valued and its elements represent the similarity between two input variable samples. In this work, the similarity metric is also computed by the Gaussian function, such that each element of the affinity matrix is computed as:

$$a_{ij} = \begin{cases} \exp\left(-\frac{\|\mathbf{x}(i)-\mathbf{x}(j)\|^2}{2\rho^2}\right), & \textit{if } i \neq j, \\ 0, & \textit{otherwise}, \end{cases} \tag{14}$$

where ρ is a width parameter that controls the spread of the similarity function.

The Gaussian function used to compute the graph affinity matrix (14) is the same as the one used in fuzzification (2). Nevertheless, they have different purposes and, therefore, the width parameter may differ. The similarity function (14) is related to the structure (i.e., the number of rules) of the model, while the membership function (2) is related to the smoothness of the model.

The $N \times N$ matrix $\mathbf{D}$ is a diagonal matrix whose elements are the degrees of the nodes of G, which are computed by the sum of similarities of the neighbors of each node:

$$d_{ii} = \sum_{j=1,\ldots,N} a_{ij}. \tag{15}$$

The spectral clustering problem is related to the graph cut problem, where the objective is to separate (cut) the set of nodes into two subsets, minimizing the number of edges between the two subsets. In recent years, spectral clustering has established itself as an efficient and meaningful alternative to traditional clustering techniques [40–43]. An excellent review of the main spectral clustering techniques has recently been presented by Luxburg [40].

The optimal solution to the graph cut problem has been proven an NP-hard problem [42], such that an approximate solution is obtained by computing the eigenvalues of a transformation of the adjacency matrix, called the Laplacian matrix, computed as$\mathbf{L} = \mathbf{D} - \mathbf{A}$. The Laplacian matrix is usually normalized, and there are some definitions for the normalized Laplacian, each one with its own properties [40]. In this work, the normalized Laplacian analyzed by Li et al. [41] has been adopted, which is computed directly by normalizing the adjacency matrix as:

$$\mathbf{L} = \mathbf{D}^{-1/2}\mathbf{A}\mathbf{D}^{-1/2}. \tag{16}$$

Ng et al. [44] used the same definition of the normalized Laplacian for spectral clustering. Their algorithm shares the same basic idea as most spectral clustering algorithms, i.e. it finds a new representation of the data based on the largest eigenvalues of the normalized Laplacian matrix and then performs the clustering on this new representation.

Spectral graph theory provides the tools to analyze the structure of a dataset by looking at the eigenvalues of the normalized Laplacian matrix, computed by the eigen decomposition:

$$\mathbf{L} = \mathbf{Z}\boldsymbol{\Lambda}\mathbf{Z}^{\mathrm{T}}, \tag{17}$$

where $\mathbf{Z}$ is the orthogonal matrix of the eigenvectors, and $\boldsymbol{\Lambda}$ is a diagonal matrix of the eigenvalues, which are all real, because the normalized Laplacian matrix is symmetric. The columns of $\boldsymbol{\Lambda}$ (and $\mathbf{Z}$) are ordered such that the eigenvalues$\lambda_1 \geqslant \lambda_2 \geqslant \cdots \geqslant \lambda_N$.

When the normalized Laplacian matrix is computed using (16), the following properties of its eigenvalues can be derived from spectral graph theory [39,41]:

1. $\lambda_1 = 1$ and $\sum_{i=1,\ldots,N}\lambda_i = 0$.
2. $-1 \leqslant \lambda_i \leqslant 1$, $i = 1,\ldots,N$.
3. If the graph is connected then $\lambda_2 < 1$.

Based on properties (1) and (2) above, it can be concluded that there will always be an integer K, such that $\lambda_i \geqslant 0, 1 < i \leqslant K$ and $\lambda_j < 0$, $K < j \leqslant N$; usually, $K \ll N$.

Moreover, for a dataset containing K disjoints clusters and if the adjacency matrix achieves perfect clustering, such that:

$$\hat{a}_{ij} = \begin{cases} 1, & \textit{if } i \neq j \textit{ and } \mathbf{x}(i) \textit{ and } \mathbf{x}(j) \textit{ are in same cluster} \\ 0, & \textit{otherwise} \end{cases} \tag{18}$$

then the eigenvalues of the normalized Laplacian computed by the adjacency matrix (18) are:

$$\begin{cases} \hat{\lambda}_i = 1, \ 1 \leqslant i \leqslant K, \\ \hat{\lambda}_i < 0, \ K < i \leqslant N. \end{cases} \tag{19}$$

If another similarity function is used to compute the adjacency matrix, for instance the one based on the Gaussian function (14), then the solution of the eigen decomposition (17) is such that $\lambda_i \to \hat{\lambda}_i, \ i = 1, \ldots, N$ [41].

The number of the positive eigenvalues of the Laplacian matrix represents the number of regular data regions in the multidimensional input variable space. It can thus be used as an estimate of the number of rules (or patches) necessary to represent the data.

The number of positive eigenvalues is a function of the neighborhood relationship among data records, which is adjusted by the dispersion parameter ρ in (14), and also the number of variables. The graph in Fig. 1 shows the number of positive eigenvalues as a function of the number of variables and the dispersion parameter ρ in (14) for a synthetic normally distributed dataset. For a fixed value of the dispersion parameter, the number of rules increases with the number of variables. For a large number of variables and a small value of the dispersion parameter the number of rules will be very low, as many elements of the affinity matrix decrease to zero. For intermediary values of the parameter ρ, the number of rules reaches a maximum and then starts to decrease. A similar pattern is obtained for datasets with non-normal distributions, but the resulting number of positive eigenvalues will be different.

The rule induction algorithm and the rule parameter estimation are presented in the next section.

4. Induction of fuzzy rules

The design of fuzzy rule-based systems has been widely studied and many approaches have been proposed, as described in recent reviews [6,10,12,31]. One of the key points of the design of fuzzy rule-based systems is the definition of the number of rules and how to avoid the “curse of dimensionality”. Although the design of fuzzy systems is a mature field, new approaches have recently been proposed in connection with the SVM algorithm [7–9]. These approaches follow a somewhat

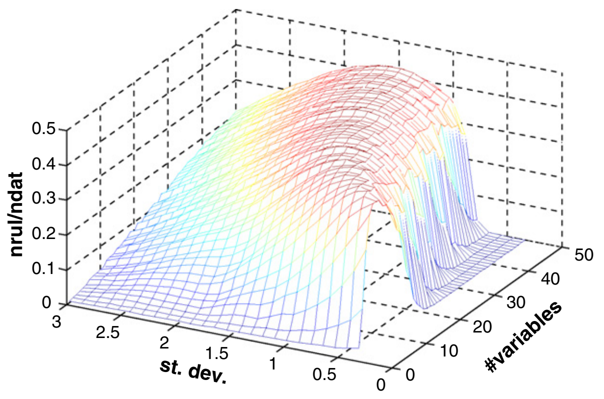

Fig. 1. The effect of the number of variables and the width parameter on the number of positive eigenvalues, as computed for a normally distributed random dataset.

similar reasoning to early neuro-fuzzy systems: a fuzzy system is defined to be equivalent to the SVM formulation and the resulting support vectors are used to design the rules. The main drawback of this approach is that the number of support vectors generated by the SVM algorithm is usually very high.

The rule base should provide an understandable and compact representation of the whole learning domain in order to provide useful information about an unknown sample. The rule induction algorithm presented in this section is designed to represent the training dataset as well as possible using a small number of rules. The approach is based on spectral analysis and on the formulation of a bounded quadratic optimization problem for parameter estimation, as described next.

4.1. The rule induction algorithm

Once the number of rules has been computed by spectral analysis, the usual way to obtain the fuzzy rules is to run a clustering algorithm on the dataset and associate a rule to each cluster. This approach for rule induction is highly flexible and a large number of clustering strategies could be used [42]. In this work, two clustering algorithms are studied:

- Spectral clustering [44] with $\boldsymbol{\omega}_i \in \mathcal{R}^p$;
- Standard *k*-means applied on the input variable space with $\boldsymbol{\omega}_i \in \mathcal{R}^p$;

The first approach employs the spectral clustering algorithm proposed by Ng et al. [44], in which the standard *k*-means algorithm is applied over the matrix $\widehat{\mathbf{Z}}$, which is computed as the first K columns of the eigenvector matrix $\mathbf{Z}$. Recall that the columns of $\mathbf{Z}$ are ordered according to the corresponding eigenvalues, such that the first columns correspond to the largest eigenvalues. Using this approach, K clusters are computed on a K-dimensional space, so that the initialization may be computed by the K-dimensional identity matrix, avoiding the instabilities due to random initialization. The coordinates of each cluster center in the input variable domain is computed by the average of the coordinates of the records assigned to the same cluster, such that the cluster center $\boldsymbol{\omega}_i \in \mathcal{R}^p$ is not a point in the training set.

In the second approach, the standard *k*-means algorithm is applied directly to the input variable space and the clusters centers are obtained as the output. The resulting cluster centers $\boldsymbol{\omega}_i \in \mathcal{R}^p$, $i = 1,\ldots,n$ are usually not in the training set. The well-known problems associated with random initialization of the *k*-means must be avoided to obtain reproducible results. In this works, the initial centers are fixed as selected points of the training set.

The rule induction algorithm is sketched out in Algorithm 1.

Algorithm 1. Rule induction

```
input: $\mathcal{X} = \{\mathbf{x}(t), t = 1,\ldots,N\},\ \sigma_i,\ \rho$
output: clusters centers $\mathbf{W} = [\boldsymbol{\omega}_1,\ldots,\boldsymbol{\omega}_n]$
01      begin
          // spectral analysis
02        compute the matrix $\mathbf{A}$     // cf. (14)
03        compute the matrix $\mathbf{D}$     // cf. (15)
04        compute $\mathbf{L} = \mathbf{D}^{-1/2}\mathbf{A}\mathbf{D}^{-1/2}$
05        compute $\mathbf{L} = \mathbf{Z}\boldsymbol{\Lambda}\mathbf{Z}^T$     // $\lambda_1 \geqslant \lambda_2 \geqslant \cdots \geqslant \lambda_N$
06        $n \leftarrow K$: $\lambda_i \geqslant \delta$, $i = 1,\ldots,K$     //estimate the number of rules
          // clustering algorithm
07        $\mathbf{W} \leftarrow clustering(\mathcal{X}, n)$
08      end
```

The number of rules in line 06 is chosen according to a parameter δ to avoid small fluctuations around zero. The value $\delta = 0.01$ was used in the examples discussed in Section 6. The matrix $\mathbf{W} \in \mathcal{R}^{p\times n}$ in line 07 stores the coordinates of the cluster centers and is computed by either one of the two approaches discussed above. Each cluster is assigned to a symbol $A_i \in \mathcal{A}$ in rule (3).

The output variable samples are not used in the rule induction algorithm, such that the set of rules represents a symbolic description of the input variable samples in the training dataset. The rule output parameters are computed according to the learning problem as described next.

4.2. Parameter estimation

The rule induction algorithm presented above provides the number of rules and the location (through clustering) of the rule prototypes. The model is parameterized by the width parameters σ_i and ρ, which are used in (2) and (14) respectively, and by the rule output parameters$\boldsymbol{\theta}$, which are used to compute the rule confidences from (13).

4.2.1. Width parameters selection

The width parameter ρ is closely related to the number of rules that will be produced in the spectral analysis. The easiest way to select this parameter is to fix it using standardized variables. Considering the variables have been standardized as the z-scores estimated from data, the parameter ρ can be set as the standard deviation of the input variables. However, this may be not the best choice for precision purposes and, in that case, the best value should be selected in cross-validation test.

There are several approaches described in the literature for setting the width parameters σ_i of the model (1) and (2) in the RBF framework [2]. The most usual ones are:

- To set a constant value such that $\sigma_i = \sigma$, $i = 1,\ldots,n$,
- To consider the half distance to the nearest rule center:

$$\sigma_i = \frac{1}{2}\|\hat{\boldsymbol{\omega}}_i - \boldsymbol{\omega}_i\|, \quad \hat{\boldsymbol{\omega}}_i = \underset{\substack{j=1,\ldots,n\\ j\neq i}}{\arg\min}\left(\|\boldsymbol{\omega}_j - \boldsymbol{\omega}_i\|\right) \tag{20}$$

- To optimize this value using a gradient-based algorithm.

In the first approach, an additional free parameter must be selected for the model and the model may become not flexible enough. The third approach may be too complex with respect to the improvement of the model result. The second approach can be considered as a trade-off between complexity and flexibility and is adopted in this work.

By setting $\rho = 1.0$ (considering that the variables were standardized as the estimated z-scores) and σ_i defined as (20), a default model is obtained where the user has no free parameter to set.

4.2.2. Rule output parameter estimation

Several heuristic approaches have been proposed to compute the rule parameters in weighted fuzzy systems [35–37]. In classification problems, the rule parameters are the rule confidence weights $\boldsymbol{\varphi}_j$ that relate the premise symbol A_i to the consequent (class) B_j. In [32], the rule parameters were constrained as $\boldsymbol{\varphi}_j \in [0,1]^n$, such that they could be interpreted as confidence values of the rules. This constraint, however, is too strong for the multidimensional rules used in this work. Nevertheless, the same formulation of the parameter estimation optimization problem may be adopted for both classification and regression problems, with different constraints.

The optimization problem for parameter estimation adopts the T–S formulation of fuzzy systems (12). The rule output parameters $\boldsymbol{\theta}$ are estimated by solving a bounded quadratic programming problem written as:

$$\begin{aligned} &\min_{\boldsymbol{\theta}} \|\mathbf{U}\boldsymbol{\theta} - \mathbf{Y}\|^2, \\ &s.t. \quad \theta_L \leqslant \theta_i \leqslant \theta_U, \quad i = 1,\ldots,n \end{aligned} \tag{21}$$

where $\mathbf{U} = [\mathbf{u}(1)|\cdots|\mathbf{u}(N)]^{\mathrm{T}}$, $\mathbf{U} \in [0,1]^{N\times n}$, is the matrix of which the lines are the fuzzy vectors computed for the data samples as in Eq. (4) and $\mathbf{Y} = [y(1),\ldots,y(N)]^{\mathrm{T}}$ is a column vector containing the observed output variable values. The constraints provide a regularization of the approximation function according to the values chosen for the bounds θ_L and θ_U.

The norm $\|\boldsymbol{\theta}\|^2 = \boldsymbol{\theta}^{\mathrm{T}}\boldsymbol{\theta}$ in the case of the regression problem is the complexity control term used in regularization and in the SVR algorithm [4]. In the optimization problem (21), the complexity control is a constraint and the empirical risk is minimized. The value of $\|\boldsymbol{\theta}\|^2$ at the solution can also be used to evaluate the quality of the model.

Considering that the variables were standardized as the estimated z-scores, the bounds in (21) are chosen such that $\|\boldsymbol{\theta}\|^2 < n$, i.e., the norm of the parameter vector cannot be greater than the number of rules. This constraint can also be viewed as a limit value for the VC dimension of the model, considering that the radius of the smallest hypersphere containing all the data is equal to unity [4].

Expanding the quadratic term in the objective function and ignoring the constant terms, the problem (21) can be reformulated as:

$$\begin{aligned} &\min_{\boldsymbol{\theta}} \frac{1}{2}\boldsymbol{\theta}^{\mathrm{T}}\mathbf{K}\boldsymbol{\theta} - \mathbf{C}_2^{\mathrm{T}}\boldsymbol{\theta}, \\ &s.t. \quad -\sqrt{n} \leqslant \theta_i \leqslant \sqrt{n}, \quad i = 1, \ldots, n \end{aligned} \tag{22}$$

where $\mathbf{K} = \mathbf{U}^{\mathrm{T}}\mathbf{U}$ is a strictly positive definite matrix as before and $\mathbf{C}_2^{\mathrm{T}} = \mathbf{Y}^{\mathrm{T}}\mathbf{U}$.

In classification problems, one quadratic optimization problem (22) is solved independently for each class B_j, $j = 1, \ldots, m$. The term $\mathbf{C}_{2j}^{\mathrm{T}} = \mathbf{V}_j^{\mathrm{T}}\mathbf{U}$ is computed for each class, where $\mathbf{V}_j = [v_j(1), \ldots, v_j(N)]^{\mathrm{T}}$, of which $v_j(t) \in \{-1, 1\}$ is a transformation of the output variable to represent the desired class information for class B_j as:

$$v_j(t) = \begin{cases} 1, & \textit{if } y(t) = j \\ -1, & \textit{otherwise.} \end{cases} \tag{23}$$

The quadratic optimization problem defined by (22) is widely known and the solution can be computed using efficient numerical algorithms.

The entire rule induction algorithm is indeed a computationally intensive task. The time and storage complexity of the affinity matrix computation is $O(N^2)$ and the full eigendecomposition of a symmetric matrix requires $O(N^3)$ floating point operations. However, only the first eigenvalues are necessary, such that efficient iterative subspace decomposition algorithms can be used, allowing the time complexity of the spectral analysis to be reduced to. The time complexity of the k-means clustering algorithm is $O(Nn)$ when the maximum number of iterations is much smaller than the number of records in the training set. This approach is suitable for small training datasets, where the learning problem is expected to be more complex. Nevertheless, recent advances in multi-core hardware technology allow efficient implementation of the algorithm by exploiting parallelism in the spectral analysis [48].

5. Model evaluation and interpretation

The prototype-based, multidimensional fuzzy membership function computed by the Gaussian function (2) cannot be directly interpreted as a linguistic concept. Nevertheless, it can be assigned to a symbol whose meaning is defined by the model. Additional features computed from the model can also be used to help human experts to gain insight into a complex problem. The interpretation and evaluation of the model are discussed in this section in classification and regression problems. The results of the modeling approach for benchmark problems are discussed in Section 6.

5.1. Model selection and evaluation

Modeling is essentially a selection task. The fuzzy symbolic modeling presented in Section 2 is completely determined by the number of rules, considering the width parameter of the membership functions as computed as (20). In the rule induction algorithm, the number of rules is computed by spectral analysis, according to the value of the parameter ρ in the affinity matrix (14). The best value for the parameter should consider the objectives of the model. In general, accuracy is the main objective and the parameter should be selected using cross-validation tests by varying the parameter ρ within a range.

The results of cross-validation tests can help in the problem understanding and in the model selection. Varying the parameter ρ within a range (and obtain the number of rules from spectral analysis) is not completely equivalent to varying the number of rules directly. The value of the parameter ρ represents a soft threshold on the neighborhood among the records and the resulting number of rules is dependent of the data structure in the input variable space. Therefore varying the parameter ρ within a range represents a scan on several different neighborhoods and the sequence of the number of rules obtained in this way can be used to compare different datasets of similar dimensions. Nevertheless this procedure may be useless if the only objective of the learning task is to select the most accurate model.

There are two results from parameter estimation that can be useful in model selection: the performance metric (i.e., the classification accuracy or the mean squared error for regression problems) and the norm of the parameter vector $\|\boldsymbol{\theta}\|$, which indicates the model size. In statistical learning terminology, the first one indicates the empirical risk and the second indicates the structural risk.

Cross-validation is a time consuming task, depending on the number of values to be tested and the size of the problem. It is thus also desirable to have a "default" value, which should be intuitive and work reasonably well in most cases. Recall that the input variables are standardized to avoid scaling effects when computing the affinity matrix, such that a very intuitive value for the parameter ρ is the average standard deviation of the input variables. This "default" selection allows also the comparison of two different datasets with the same number of variables, as the difference in the resulting number of rules will be due to internal structures revealed by the spectral analysis.

These two modeling approaches will be discussed in Section 6. After the selection of one specific model, it can be interpreted, as described next.

5.2. Symbolic interpretation: regression problems

The parameters $\boldsymbol{\theta}$ obtained as the solution of the quadratic problem (22) can be converted into symbolic rule confidences, based on the complementary property of the fuzzification and defuzzification operators. A fuzzification method *Fuz* is said to be complementary to a defuzzification method *Def* when, for any variable ξ, the following condition holds [38]:

$$Def(Fuz(\xi)) = \xi. \tag{24}$$

The output variable fuzzy partition, computed by the normalized and triangular-shaped fuzzy membership functions, is complementary to the singleton defuzzification (10), such that condition (24) can be rewritten as:

$$\mu_{B_j}(\theta_i).\beta_j = \theta_i. \tag{25}$$

The symbolic rule confidence weight φ_{ij} related to the rule $A_i \to B_j$ can thus be computed in regression problems from the components of the parameters vector $\boldsymbol{\theta}$ as:

$$\varphi_{ij} = \mu_{B_j}(\theta_i). \tag{26}$$

The resulting symbolic model depends on the definition of the prototype vector $\boldsymbol{\beta}$. The symbolic model with rule confidence weights computed as (26) will be equivalent to the T–S counterpart (w.r.t. (13)) if the prototype vector $\boldsymbol{\beta} = \left[-\sqrt{n}, \sqrt{n}\right]^{\mathrm{T}}$, because the parameters are constrained to this interval in the solution of (22). Nevertheless, this choice is not interesting for interpretability purposes because the output variable has been standardized within the $[-1, 1]$ interval and, in general, $\sqrt{n} \gg 1$, such that few rules have parameters outside that interval. Hence a good choice for symbolic interpretation of the model is $\boldsymbol{\beta} = [-1, 1]^{\mathrm{T}}$, such that the output symbols can be interpreted as "small" and "large".

It is possible to use any number m of output symbols for interpretation of the model parameters. When $m > 2$, the additional components of the prototype vector $\boldsymbol{\beta}$ may be defined by equally spaced values or they can be selected based on expert knowledge, when available. In regression problems, it is generally useful to define $m = 3$ in order to map the "medium" values such that $\boldsymbol{\beta} = [-1, 0, 1]^{\mathrm{T}}$.

5.3. Symbolic interpretation: classification problems

In classification problems, each class is considered as an independent output and one quadratic problem as (22) is computed for each class. The interpretation of parameter values as rule confidences is done as for regression problems based on the complementary property of triangular-shaped, fuzzy membership functions and singleton defuzzification.

The output prototype vector for each class is computed as $\boldsymbol{\beta}_j = [-1, 1]^{\mathrm{T}}$. These prototypes are related to two fuzzy sets representing "negative" (N) and "positive" (P) parameters values, respectively. Only the positive parameter values are used to compute the rule confidence for each class:

$$\varphi_{ij} = \mu_P(\theta_{ij}), \tag{27}$$

where $\mu_P(\theta_{ij})$ is the positive membership value computed from the component $\theta_{ij} \in \boldsymbol{\theta}_j$.

The resulting rule weights can be used to interpret the model. Typically, in the solution of (27), three types of rule weight values can occur:

- $\varphi_{ij} = 0$: this means that the rule $A_i \to \neg B_j$ is certain and that the region defined by the symbol A_i cannot be assigned to class B_j;
- $0 < \varphi_{ij} < 1$: this means that the rule is uncertain and that the symbol A_i represents a region that is partially related to the class B_j.
- $\varphi_{ij} = 1$: this means that the rule $A_i \to B_j$ is certain and that the region defined by the symbol A_i is totally related to the class B_j.

Other rule quality measures have also been studied in the domain of fuzzy association rules. The most typical measures are support and confidence, defined as [45]:

$$supp(A_i \to B_j) = \sum_{t=1,\ldots,N} \mu_{A_i}(\mathbf{x}(t)) \cdot v_j(t), \tag{28}$$

$$conf(A_i \to B_j) = \frac{\sum_{t=1,\ldots,N} \mu_{A_i}(\mathbf{x}(t)) \cdot v_j(t)}{\sum_{t=1,\ldots,N} \cdot \mu_{A_i}(\mathbf{x}(t))} \tag{29}$$

The support measure (28) is very useful, as it shows approximately the number of samples covered by each rule. Model interpretation is carried out through analysis of the rule weights and completed by the graphic visualization of the model as described next.

5.4. Graphic visualization

Rule interpretation has been related to linguistic interpretation as "if-then" sentences. The recent work on fuzzy rule base interpretability have focused on interpretability indexes or measures to quantify the quality of the "if-then" sentences generated by a given algorithm [12–15,34,46]. On the other hand, the large number of variables in complex applications usually results in very large sentences that are hard to understand linguistically.

The multidimensional fuzzy model generated by the procedure presented in the previous section cannot be interpreted linguistically as "if-then" rules. However, this kind of model can deal with a large number of input variables. It is thus necessary to develop other means of rule interpretation and data understanding. Graphic visualization of multidimensional data in two dimensions is a very intuitive way to help the model analyst to understand the model within the context defined by the data.

There are a wide variety of methods for dimensionality reduction and data projection. One of the most widely used in practice is principal components analysis (PCA). The PCA is a method for the projection of data into a low dimensional subspace in the form:

$$\hat{\mathbf{x}} = \mathbf{x}\mathbf{P}, \tag{30}$$

where $\hat{\mathbf{x}} \in \mathcal{R}^d,\ d < p$, is the low dimensional transformed variable vector and the $p \times d$ matrix $\mathbf{P}$ is a linear orthonormal transform, computed as the eigenvectors of the correlation matrix associated with the largest eigenvalues.

The PCA provides a simple and efficient way to visualize multidimensional data using the $d = 2$ principal components. Moreover, since the transform is linear and orthogonal, the inverse transform is easily computed as $\mathbf{P}^{\mathrm{T}}$ and can be used to generate data to be processed by the model and for visualization of the model results.

The idea behind model visualization is to generate a hyperplane in the directions spanned by the two first principal components and to compute and to plot the results of the model in this plane. The model visualization is sketched in the following steps:

(1) Compute the transform matrix $\mathbf{P}$ as the eigenvectors of the correlation matrix;
(2) Generate a 2D grid dataset $\mathcal{L} = \{\hat{\mathbf{z}}(t), t = 1, \ldots, l\}$;
(3) Compute the inverse transform of the grid dataset as $\mathbf{z}(t) = \hat{\mathbf{z}}(t)\mathbf{P}^{\mathrm{T}},\ t = 1, \ldots, l$;
(4) Adjust the model using the training dataset and use the transformed grid dataset as testing data;
(5) Plot the result of the symbolic inference membership vector (7), in the coordinates of the 2D grid dataset as different hues of color.

Graphic visualization of the model allows the graphic interpretation of the model in the 2D plane spanned by the principal components. This graphical visualization, in conjunction with the analysis of the model results, provides a whole view of the model behavior in the context defined by the problem.

6. Results and discussion

The evaluation of the methodology presented in this work should consider the two main objectives of the modeling task separately: performance and interpretation. The performance of the model is discussed in the first subsection, where it is verified whether the proposed methodology can provide an accurate yet understandable model. The interpretation of the model is more subjective; the idea is to verify whether the modeling methodology can provide clues for understanding the model and data.

The results discussed in this section are based on a set of benchmark classification datasets obtained from the UCI Machine Learning Repository [49]. The selected datasets for the analysis are shown in Table 1 for classification and regression problems.

6.1. Experimental setup and results

To evaluate the performance of the modeling methodology, six different models, divided into three pairs, were generated by varying the algorithms and model structure selection procedure:

- *Reference-design*

1. RD-KM-RR: *k*-means clustering and ridge regression parameter estimate,
2. RD-KM-QP: *k*-means clustering and quadratic programming parameter estimate.

- *Spectral-design*

3. SD-KM-QP: *k*-means clustering and quadratic programming parameter estimate,
4. SD-SM-QP: spectral clustering and quadratic programming parameter estimate.

Table 1
Benchmark datasets.

Classification	p	m	N	Regression	p	N
Iris	4	3	150	Voltage	4	1056
Balance	4	3	625	Electric	6	362
Diabetes	8	2	768	Auto mpg	7	392
Cancer	9	2	286	Concrete	8	1030
Glass	9	6	214	Stock	9	950
Wine	13	3	178	Housing	13	506
Heart	13	2	270	Mortgage	15	1049
Image	18	7	210			
Ionosphere	34	2	351			
Sonar	60	2	208			

• *Default-design*

5. DD–KM–QP: k-means clustering and quadratic programming parameter estimate.
6. DD–SM–QP: spectral clustering and quadratic programming parameter estimate.

For the two first pairs of models, the model structure is selected using a ten-fold cross validation test, where the best results are retained. The reference design does not consider spectral analysis, and the number of rules is varied within the range of 2% to 30% of the number of records (32 different values for the number of rules were evaluated). The model RD-KM-RR is the standard RBF model using k-means clustering with fixed initialization and ridge regression [47] to compute the rule output parameters. The model RD–KM–QP is the same as RD–KM–RR but using the constrained quadratic programming problem (22) for parameter estimation.

In spectral design, the model structure is also selected using the ten-fold cross validation test, but employs spectral analysis to compute the number of rules by varying the parameter ρ within the interval [0.1,2.0] (32 different values were evaluated). Both models (SD–KM–QP and SD–SM–QP) use the constrained quadratic programming problem for output parameter estimation, but SD–KM–QP uses the k-means clustering algorithm and SD–SM–QP uses the spectral clustering algorithm.

The third pair of models does not use a cross-validation test to select the model structure. The results presented in this set are actually an instance of the spectral design in which the parameter ρ is selected as the standard deviation of the standardized input variables. This should be regarded as the default design.

The accuracy and model size results for the classification benchmark datasets are presented in Table 2. The number of rules and the processing time (in s) for each benchmark dataset are shown in Table 3. The accuracy has been computed using the concatenation of the 10 test sets computed in the cross validation tests. For the other parameters, the value in the table is the average of the 10 runs computed in the cross validation and, for each fold, the average for all the classes.

The performance and model size results for the regression benchmark datasets are presented in Table 4. The number of rules and the processing time (in s) for each dataset are shown in Table 5. The rooted mean square (RMS) is computed by the concatenation of the test sets in the cross validation tests, and the model size is the average of the models computed in each fold. The evaluation of these results is discussed in the next section, based on the nonparametric statistical tests.

To evaluate the default design, the accuracy and the number of rules obtained by this modeling approach are compared with the accuracy and the number of support vector results obtained for the same datasets using the two SVM algorithms available in LibSVM library, accessed from the Weka[1] tool. The results shown in Table 6 were obtained using the same data partition in the cross validation so that they can be compared with the results computed by the default design. The number of rules and the number of support vectors shown in Table 6 for the default design and SVM respectively are the average of the number of rules and support vectors computed in the 10-fold cross validation. Table 7 presents the results of the default design compared with the two SVM algorithms for the regression problems, using the same experimental procedure. The evaluation of these results is discussed in the next section.

6.2. Nonparametric statistical evaluation

The evaluation of the experimental results was based on nonparametric statistical tests for multiple comparisons [50–52], performed with the Keel Tool for Statistical Analysis [53,54]. The nonparametric statistical tests consider that the null hypothesis being tested is that all the algorithms perform similarly, with no significant differences [52]. The result of a nonparametric statistical test is the p-value, which is a measure of the statistical significance of the differences observed in the evaluation. The p-value is interpreted as the probability of obtaining a result at least as extreme as the one that was actually observed, assuming that the null hypothesis is true. The lower the p-value, the less likely that the result will be observed

[1] http://www.cs.waikato.ac.nz/ml/.

Table 2
Accuracy and model size results for the classification datasets.

	Reference design				Spectral design			
	RD-KM-RR		RD-KM-QP		SD-SM-QP		SD-KM-QP	
	Acc	$\|\|\boldsymbol{\theta}\|\|$	Acc	$\|\|\boldsymbol{\theta}\|\|$	Acc	$\|\|\boldsymbol{\theta}\|\|$	Acc	$\|\|\boldsymbol{\theta}\|\|$
Iris	98.00	13.95	96.67	13.56	95.33	24.19	96.67	18.69
Balance	91.20	149.50	91.04	101.70	90.72	127.75	90.56	65.65
Diabetes	77.34	46.33	77.34	67.01	77.86	9.90	77.73	23.30
Cancer	97.07	12.39	97.07	2.82	97.36	2.89	97.07	43.10
Glass	71.50	105.89	69.63	18.41	69.16	31.55	68.69	26.58
Wine	98.88	8.26	98.88	8.67	98.88	20.60	100.00	16.23
Heart	84.44	3.16	84.44	3.19	85.19	6.26	85.56	6.96
Image	87.14	7.8E + 08	86.67	13.17	86.19	9.58	86.19	16.57
Ionosphere	95.73	1.1E + 05	95.73	42.76	95.16	27.07	95.16	30.66
Sonar	87.50	2.9E + 09	87.50	17.89	86.06	16.06	87.02	20.96

Table 3
Number of rules and processing time for the classification datasets.

	Reference design				Spectral design			
	RD-KM-RR		RD-KM-QP		SD-SM-QP		SD-KM-QP	
	#Rules	Time (s)	#Rules	Time (s)	#Rules	Time (s)	#Rules	Time (s)
Iris	11	0.10	23	0.01	43	0.64	33	0.29
Balance	73	1.80	165	0.30	203	43.02	90	2.40
Diabetes	89	0.91	110	0.47	20	2.28	43	2.41
Cancer	33	0.08	6	0.15	6	1.62	95	2.33
Glass	42	0.26	33	0.14	52	0.82	45	0.66
Wine	22	0.08	22	0.03	51	0.41	41	0.32
Heart	5	0.04	5	0.01	12	0.28	15	0.29
Image	78	0.29	33	0.46	24	0.32	43	0.70
Ionosphere	78	1.37	125	0.71	66	0.94	70	1.14
Sonar	52	0.40	52	0.37	46	0.47	55	0.81

Table 4
Accuracy and model size results for the regression datasets.

	Reference design				Spectral design			
	RD-KM-RR		RD-KM-QP		SD-SM-QP		SD-KM-QP	
	RMS	$\|\|\boldsymbol{\theta}\|\|$	RMS	$\|\|\boldsymbol{\theta}\|\|$	RMS	$\|\|\boldsymbol{\theta}\|\|$	RMS	$\|\|\boldsymbol{\theta}\|\|$
Voltage	0.0135	6.5E+04	0.0146	89.51	0.0310	9.52	0.0155	55.71
Electric	0.1361	11.80	0.1362	16.39	0.1357	10.65	0.1347	10.99
Auto mpg	0.1210	9.33	0.1228	18.63	0.1177	21.13	0.1230	33.60
Concrete	0.2214	100.36	0.2227	84.40	0.2295	60.15	0.2236	71.15
Stock	0.1186	12.61	0.1171	17.26	0.1352	4.14	0.1303	13.55
Housing	0.1629	315.08	0.1643	23.63	0.1663	11.99	0.1566	34.05
Mortgage	0.0125	17.48	0.0126	18.39	0.0114	10.31	0.0125	14.36

Table 5
Number of rules and processing time for the regression datasets.

	Reference design				Spectral design			
	RD-KM-RR		RD-KM-QP		SD-SM-QP		SD-KM-QP	
	#Rules	Time (s)	#Rules	Time (s)	#Rules	Time (s)	#Rules	Time (s)
Voltage	223.0	2.23	308.0	1.53	33.3	4.89	176.7	5.93
Electric	71.0	0.18	51.0	0.35	39.2	0.54	39.2	0.51
Auto mpg	50.0	0.17	55.0	0.13	60.9	0.89	88.3	1.22
Concrete	204.0	2.16	217.0	1.82	160.0	18.26	174.2	6.54
Stock	98.0	0.69	98.0	0.67	22.4	4.25	81.1	4.54
Housing	93.0	0.64	93.0	0.87	59.1	1.26	117.2	2.35
Mortgage	306.0	2.66	306.0	2.63	110.4	14.30	175.2	8.89

Table 6
Default design evaluation for the classification datasets.

	Default design				LibSVM (Weka)			
	DD–SM–QP		DD–KM–QP		C-SVM		ν-SVM	
	Acc	#Rules	Acc	#Rules	Acc	#SV	Acc	#SV
Iris	93.33	14.0	88.67	14.0	96.00	80	95.33	28
Balance	90.08	58.0	90.08	58.0	89.44	218	97.92	96
Diabetes	76.82	147.3	76.82	147.4	77.47	400	61.97	273
Cancer	96.93	89.6	96.78	82.7	97.07	76	97.36	66
Glass	67.29	34.3	68.69	34.3	52.80	183	59.81	119
Wine	98.88	48.3	98.31	44.7	98.31	84	98.87	41
Heart	81.48	77.9	83.70	71.0	84.44	135	74.07	95
Image	82.38	43.8	86.19	40.9	84.76	170	88.57	98
Ionosphere	95.16	65.7	95.16	64.5	90.02	170	93.73	62
Sonar	81.73	59.9	85.58	54.7	78.36	152	85.57	75

Table 7
Default design evaluation for the regression benchmark datasets.

	Default design				LibSVM (Weka)			
	DD–SM–QP		DD–KM–QP		ε-SVM		ν-SVM	
	RMS	#Rules	RMS	#Rules	RMS	#SV	RMS	#SV
Voltage	0.0310	33.3	0.0356	33.3	0.0598	14	0.0303	105
Electric	0.1376	51.0	0.1387	51.5	0.1380	138	0.1420	38
Auto mpg	0.1246	47.5	0.1271	48.2	0.1210	112	0.1254	40
Concrete	0.2324	128.8	0.2389	140.3	0.1560	449	0.1733	103
Stock	0.1529	61.6	0.1361	61.9	0.0739	144	0.0812	94
Housing	0.2440	93.7	0.1759	92.3	0.1442	142	0.1595	57
Mortgage	0.0174	46.8	0.0209	47.0	0.0472	11	0.0161	106

Table 8
Average ranking of Friedman Aligned test for the classification datasets.

Algorithm	Acc.	#Rules	$\lVert\boldsymbol{\theta}\rVert$	Time (s)
SD-SM-QP	24.20	19.90	17.60	30.00
SD-KM-QP	25.65	21.90	17.20	27.50
RD-KM-RR	13.40	18.95	31.00	14.10
RD-KM-QP	18.75	21.25	16.20	10.40
p-Value	0.2217	0.9586	0.0164	0.0007

Table 9
Average ranking of Friedman Aligned test for the regression datasets.

Algorithm	RMS	#Rules	$\lVert\boldsymbol{\theta}\rVert$	Time (s)
SD–SM–QP	19.43	5.64	9.00	21.86
SD–KM–QP	13.64	13.43	12.86	20.29
RD–KM–RR	11.36	18.93	20.43	8.14
RD–KM–QP	13.57	20.00	15.71	7.71
p-Value	0.3884	0.0137	0.0828	0.0020

Table 10
Summary of hypothesis tests evaluation.

Measure	Classification	Regression
Acc./ RMS	Accept*	Accept
# Rules	Accept	Reject
$\lVert\boldsymbol{\theta}\rVert$	Reject	Reject*
Time (s)	Reject	Reject

considering that the null hypothesis is true. Typically, a confidence value of 95% is adopted; thus, the null hypothesis is rejected if the *p*-value is less than 0.05, which corresponds to a probability of 5% of incurring a Type I error (reject a true null hypothesis).

The Friedman Aligned Ranks test was used to compare the four best design algorithms according to the suggestions given by Garcia et al. [52]. The Friedman Aligned Ranks test is recommended when a small number of algorithms is being compared (not more than 4 or 5 algorithms) because it provides comparisons among different datasets besides the comparisons among algorithms for the same dataset [52]. The ranking computed by the Friedman Aligned Ranks test takes into account not only the intra-dataset differences in the performance of the algorithms but also the inter-dataset differences.

The results of the average ranking and *p*-values of the Friedman Aligned Ranks test for the classification datasets are shown in Table 8. The Friedman Aligned test results suggest that the processing times of all spectral design algorithms (SD) are greater than those of the reference design (RD) algorithms, thus rejecting the null hypothesis, as expected, because of the cost of the eigenstructure analysis of the similarity matrix. For the model size $\|\boldsymbol{\theta}\|$, the null hypothesis is also rejected,

Table 11
Average ranking of Friedman Aligned test for the external tests.

Algorithm	Classification		Regression	
	Acc.	#Rules/#SV	RMS	#Rules/#SV
DD–SM–QP	19.95	12.75	18.85	12.35
DD–KM–QP	17.50	11.65	17.57	12.92
C-SVM/ ε-SVM	20.30	34.80	11.57	19.28
ν-SVM	24.25	22.80	10.00	13.42
p-Value	0.7050	0.0001	0.1779	0.4273

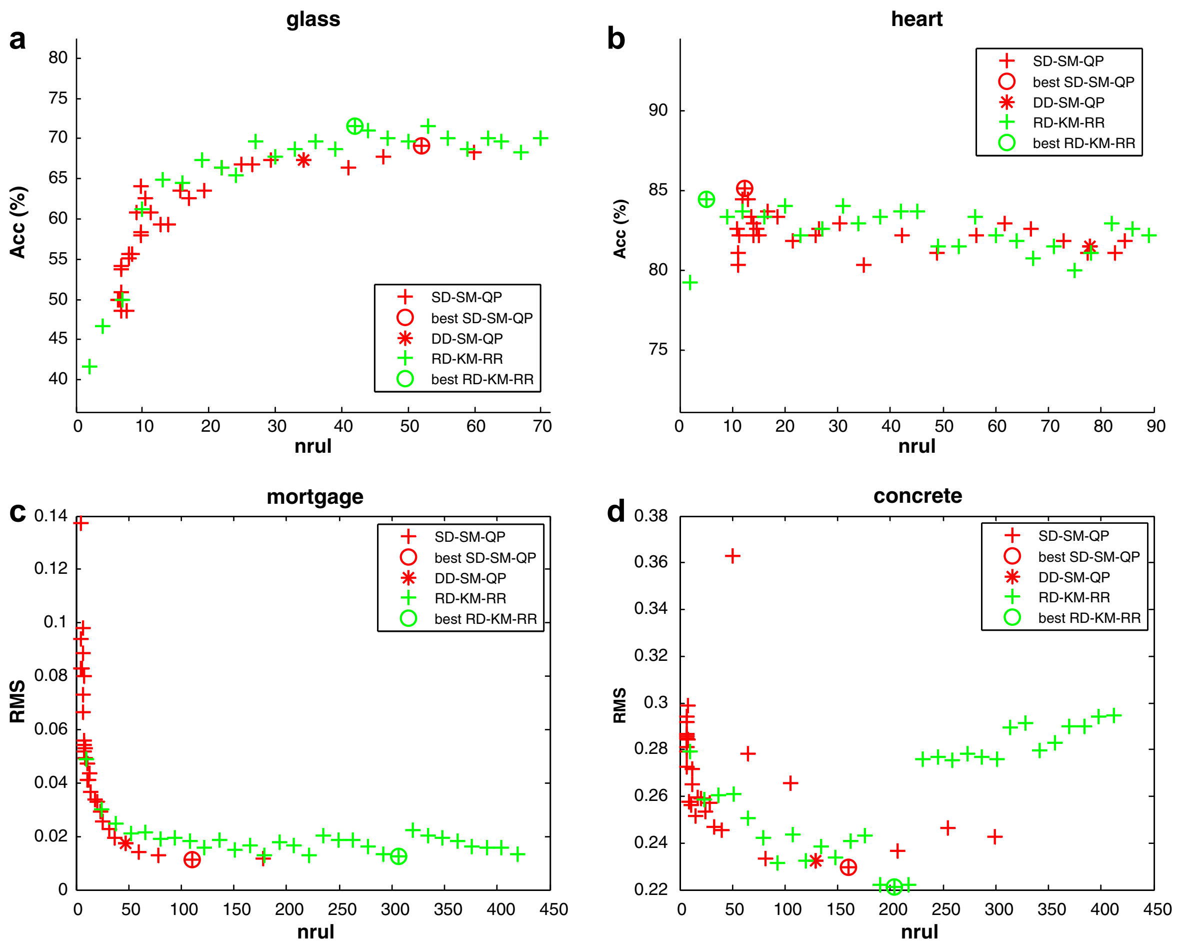

Fig. 2. Performance evaluation: (a) glass, (b) heart, (c) mortgage and (d) concrete.

indicating that the quadratic programming (QP) algorithm for parameter estimation is more efficient to constrain the model size than the ridge regression (RR) algorithm. According to the average rankings and the corresponding *p*-values shown in Table 8, no algorithm performs better than the others with respect to the number of rules, accepting the null hypothesis at a high level of statistical significance. The differences among the average rankings for the accuracy results show that the reference design (RD) algorithms perform slightly better than the spectral design (SD), but the *p*-value equal to 0.2217 suggests that the null hypothesis cannot be rejected in this case.

The differences observed among the average rankings can be further analyzed by using a post hoc test to confirm whether the differences are statistically significant [52]. When comparing multiple algorithms, the *p*-value reflects the probability of error in a certain comparison, but it does not take the other comparisons into account; thus, when considering all comparisons, the probability of making one or more Type I errors is relatively high. To avoid this problem, the adjusted *p*-values (APVs) must be computed. The APVs take into account multiple pairwise comparisons when comparing multiple algorithms. According to Garcia et al. [52], the Finner post hoc test is indicated to be used in conjunction with the Friedman Aligned Ranks due to its low probability of incurring a Type II error (accept a false null hypothesis).

For the accuracy results of the classification experiments, the Finner test between the first and the last algorithms according to the average ranking of the Friedman Aligned test (RD–KM–RR $\times$ SD–KM–QP) resulted in an APV = 0.0562; thus, the null hypothesis can be accepted at a level of significance very close to the acceptance threshold (5%). This result suggests that the differences in the accuracy among the four algorithms may not be statistically significant, as previously indicated by the *p*-value of the Friedman Aligned test. The APV for $\|\boldsymbol{\theta}\|$ considering the comparison between the algorithms RD–KM–RR x RD–KM–QP is 0.0138, which confirms the conclusions drawn for the QP algorithm. The Friedman Aligned tests for the processing time and the number of rules resulted in *p*-values with high statistical significance, and the APVs for these measures confirmed the previous results.

The evaluation of the experimental results for the regression datasets follows the same idea presented above for the classification datasets. The average rankings and *p*-values of Friedman Aligned Ranks test are shown in Table 9. In this case, the average rankings show different performance levels in all evaluated measures, although the *p*-values for precision and $\|\boldsymbol{\theta}\|$

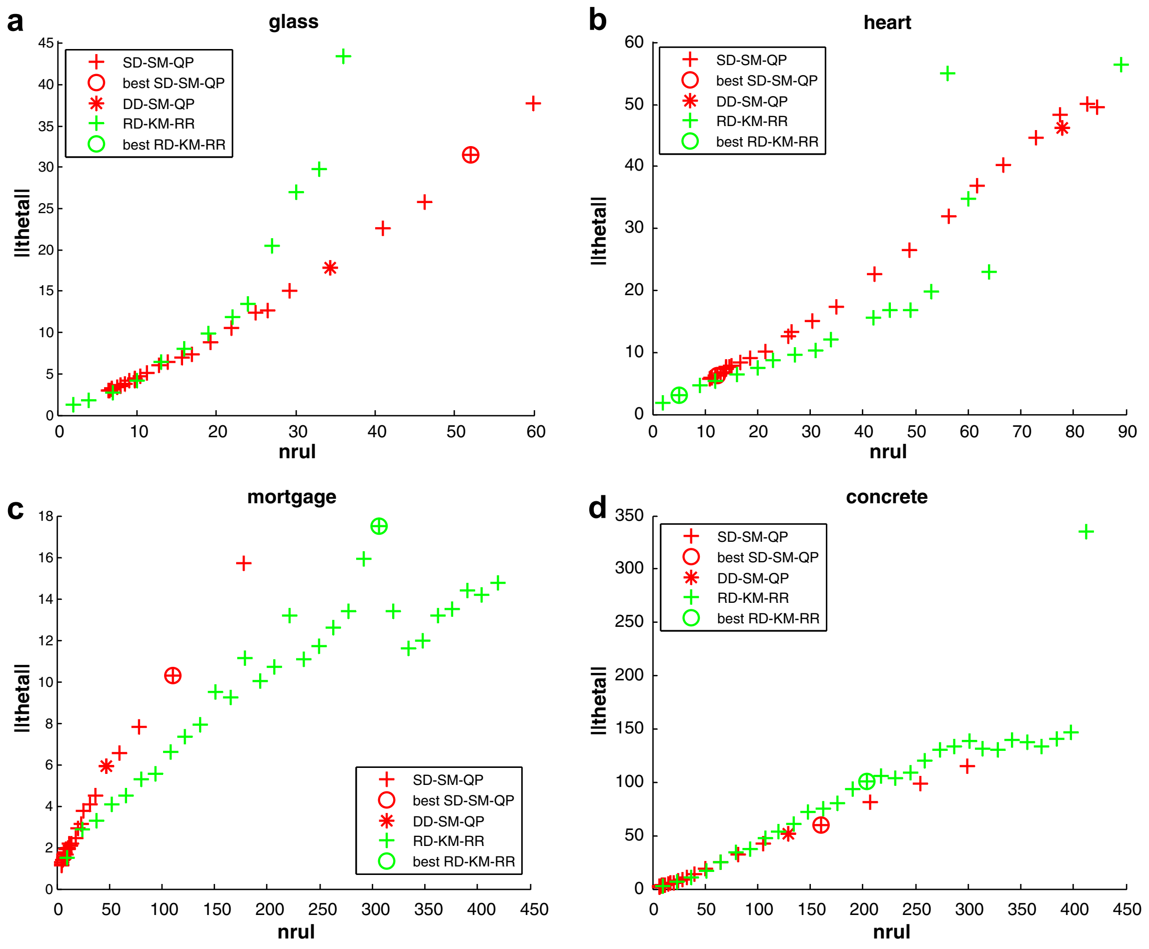

Fig. 3. Model size evaluation: (a) glass, (b) heart, (c) mortgage and (d) concrete.

suggest that the null hypothesis cannot be rejected in these cases. The differences among the average rankings were thus analyzed using the Finner post hoc test to confirm whether the differences are statistically significant.

For the precision results, the Finner test for the comparison between the best and the worse algorithms (RD–KM–RR × SD–SM–QP) resulted in an APV = 0.1862, which confirms the Friedman Aligned test result and indicates that the null hypothesis cannot be rejected in this case. The algorithms can thus be considered to perform similarly, and the observed differences are not statistically significant. In the case of the number of rules, the comparison between the first and the last algorithms (SD–SM–QP × RD-KM-QP) resulted in an APV = 0.0033, suggesting that the spectral design (SD) algorithms result in a more compact model than the reference design (RD) algorithms. In the case of $\|\boldsymbol{\theta}\|$, the comparison SD–SM–QP × RD–KM–RR resulted an APV = 0.0278. This result shows that the pairwise comparisons in the Finner test reveal differences that were not detected by the Friedman Aligned test and suggests that the quadratic programming (QP) algorithm is more effective than the ridge regression (RR) algorithm to constrain the model size. For the processing time, the Finner test confirmed the conclusion obtained by the Friedman Aligned test as expected because of the cost of the eigenstructure analysis of the similarity matrix.

The summary of the nonparametric statistical tests evaluation is shown in Table 10 for the hypothesis testing results. The asterisks indicate the cases where statistical tests do not provide the elements for a categorical conclusion. In the case of the accuracy evaluation of the classification dataset results, the Finner post hoc test resulted in a *p*-value very close to the threshold, although the Friedman Aligned test accepted the null hypothesis. In the case of $\|\boldsymbol{\theta}\|$, in the evaluation of regression datasets, the null hypothesis was accepted according to the Friedman Aligned test, but the Finner post hoc test did not confirm that result. For the other results, the null hypothesis was accepted or rejected with a high level of statistical significance, and the Finner post hoc test corroborated the results of the Friedman Aligned test.

The results of the average ranking and the *p*-values computed by the Friedman Aligned test for the evaluation of the default design in the classification and regression experiments are shown in Table 11. According to these results, the null hypothesis was accepted for all evaluation measures, except for the number of interpolation functions (rules or support vectors) in classification datasets. The Finner post hoc test confirmed all these results. The only caveat concerns the comparison

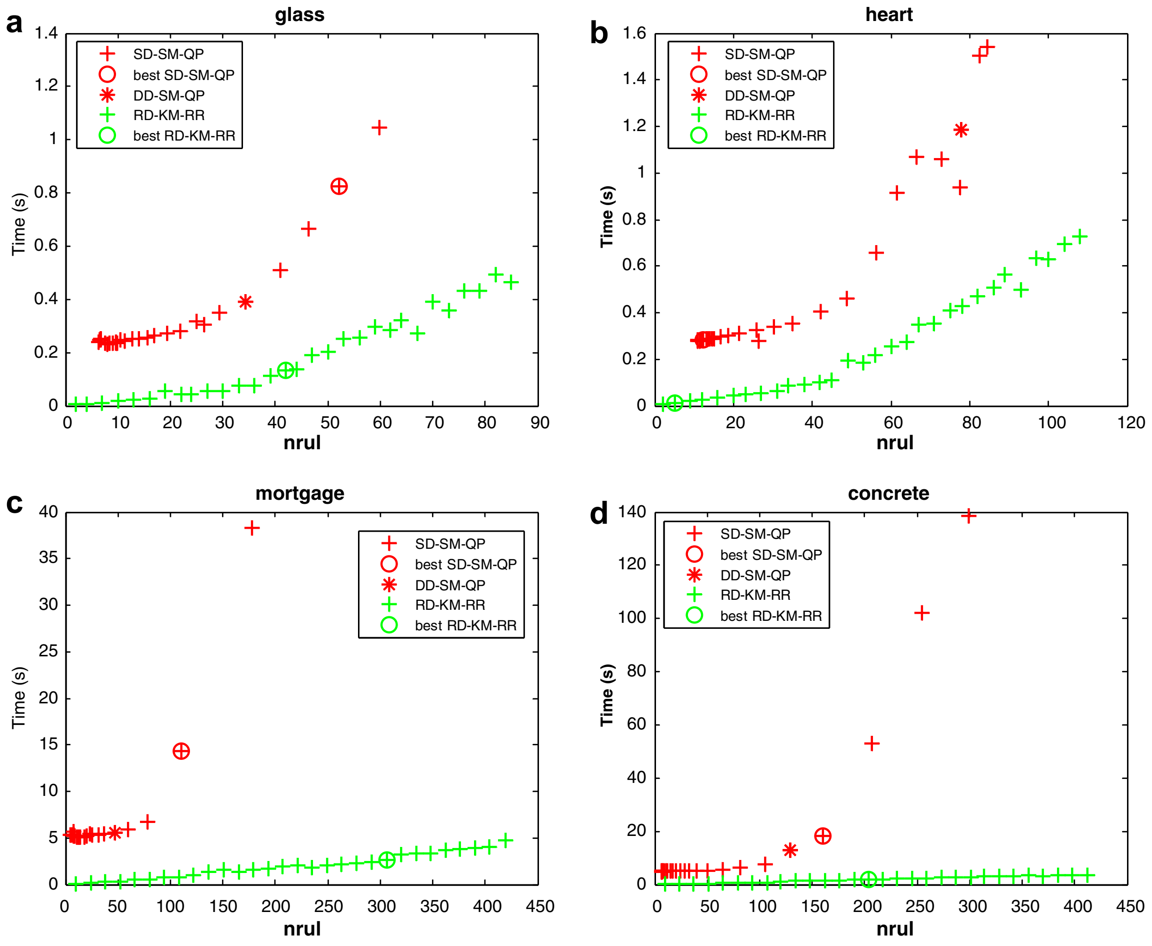

Fig. 4. Processing time evaluation: (a) glass, (b) heart, (c) mortgage and (d) concrete.

between DD–KM–QP and the ν-SVM algorithm, considering the number of rules and the support vectors in classification problems. In this case, the Finner test obtained APV = 0.0490, which confirms the Friedman Aligned test, but at a level of significance very close to the 95% threshold.

6.3. Model exploitation

The numerical results present a global view of performance of the best models, but do not show how the performance of the models behaves according to the selected number of rules. The experimental study used different algorithms and benchmark datasets. However, due to space constraints, four selected benchmark datasets are used to discuss the common patterns found in the results. The behavior of two of the most representative algorithms are discussed; one for the reference design (RD–KM–RR) and the other for spectral design (SD–SM–QP). In all figures below, the cross sign (+) represents the values obtained in the cross validation test, the circle (o) represents the best model and the asterisk (*) stands for the default design.

There are two very distinctive patterns that occur in both the classification and regression problems, as shown by the selected results in Fig. 2. For the glass and mortgage datasets, the performance of the algorithms varies strongly with the number of rules in the expected way, i.e., the greater the number of rules, the better the performance. For the heart and concrete datasets, the performance does not vary significantly with the number of rules and remains roughly stable with respect to the number of rules, or it can even be worse for larger models. A closer look at these results shows that the performance varies slightly but apparently randomly for small differences in the number of rules.

The evaluation of the model size $\|\boldsymbol{\theta}\|$ against the number of rules is shown in Fig. 3. The value of $\|\boldsymbol{\theta}\|$ obtained using ridge regression parameter estimation grows exponentially with the number of rules, whereas the quadratic programming algorithm remains stable and proportional to the number of rules. For high dimensional problems, the ridge regression parameter estimation algorithms usually result in models with $\|\boldsymbol{\theta}\| > 10^5$ for large values of the number of rules. This behavior has also been observed for most of the regression datasets, except for the mortgage and stock benchmarks where the ridge regression algorithm results in smaller models. In Fig. 3(a) and (b), the values of $\|\boldsymbol{\theta}\| > 10^3$ have been omitted for better visualization.

The evaluation of the processing time of the algorithms as a function of the number of rules is shown in Fig. 4, where the time in seconds is the average time for the ten runs in the cross validation. The results show somewhat similar behavior for all datasets, with a slight difference between the problem types. For classification problems, the increase in processing time with the number of rules is not too severe, as it is for the regression problems. The behavior of the reference design resulted

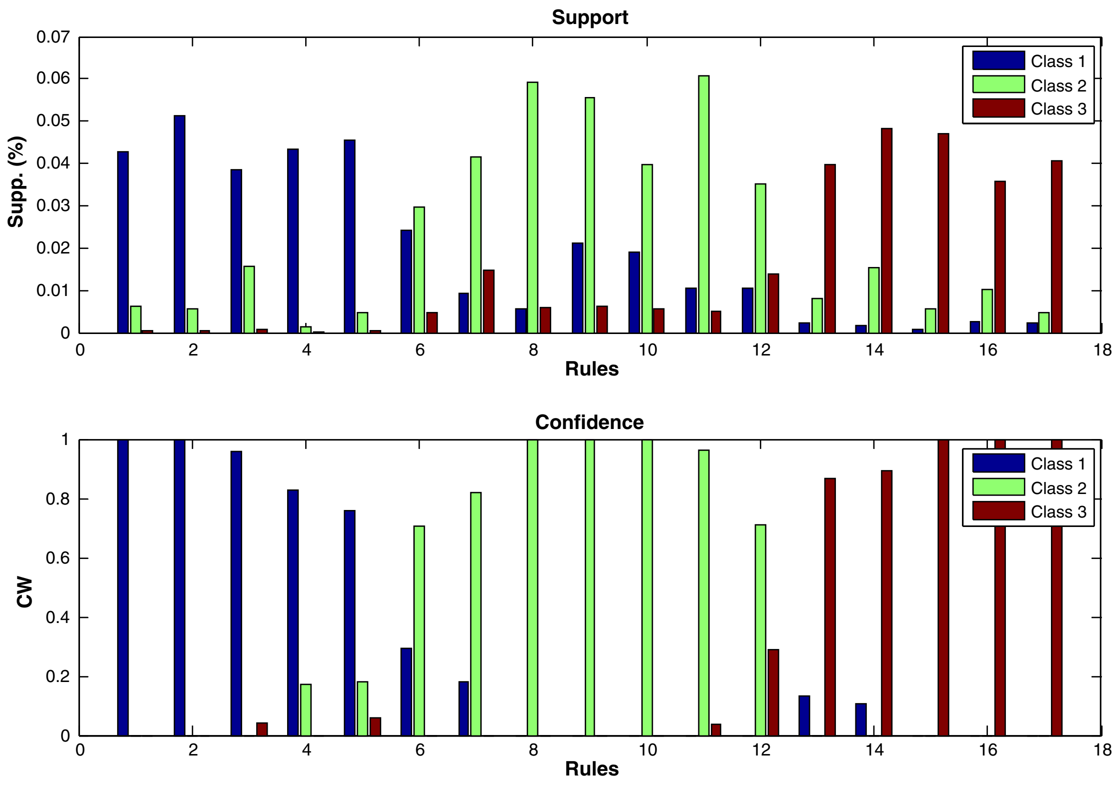

Fig. 5. Support and confidence values for the symbolic interpretation of the rule base.

in similar times taken in processing the results for both parameter estimation algorithms. Spectral design also resulted in similar processing times for the two clustering algorithms. The difference between the reference and spectral design is essentially due to the computation of the eigenvalues in spectral design. Hence, for values of the parameter ρ that result in large numbers of rules, the computation of eigenvalues is more time consuming.

The performance evaluations presented above show that, as expected, no algorithm performs better than all others in all cases and for all evaluation criteria. However, the results show clearly that spectral clustering using the quadratic programming parameter estimation algorithm may produce equal or smaller models than k-means with ridge regression, although it requires more processing time.

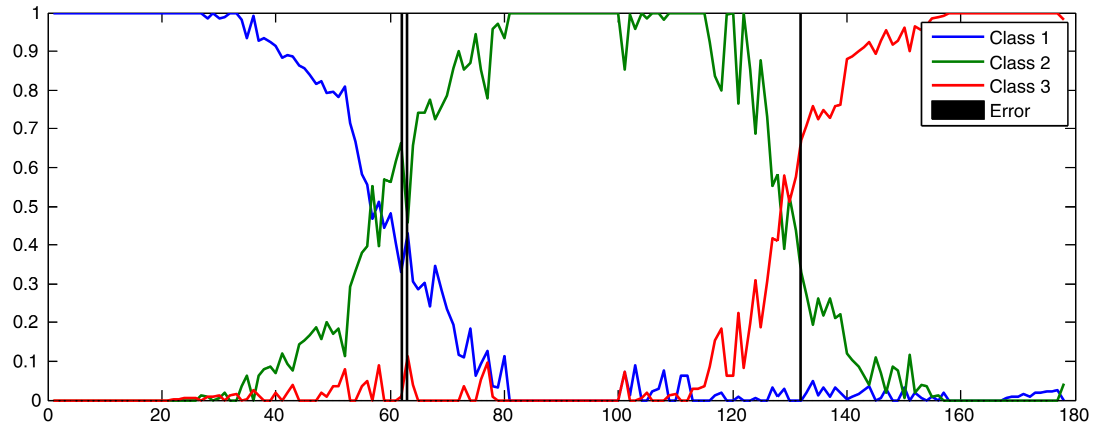

Fig. 6. Output membership degrees and error location for the training data records.

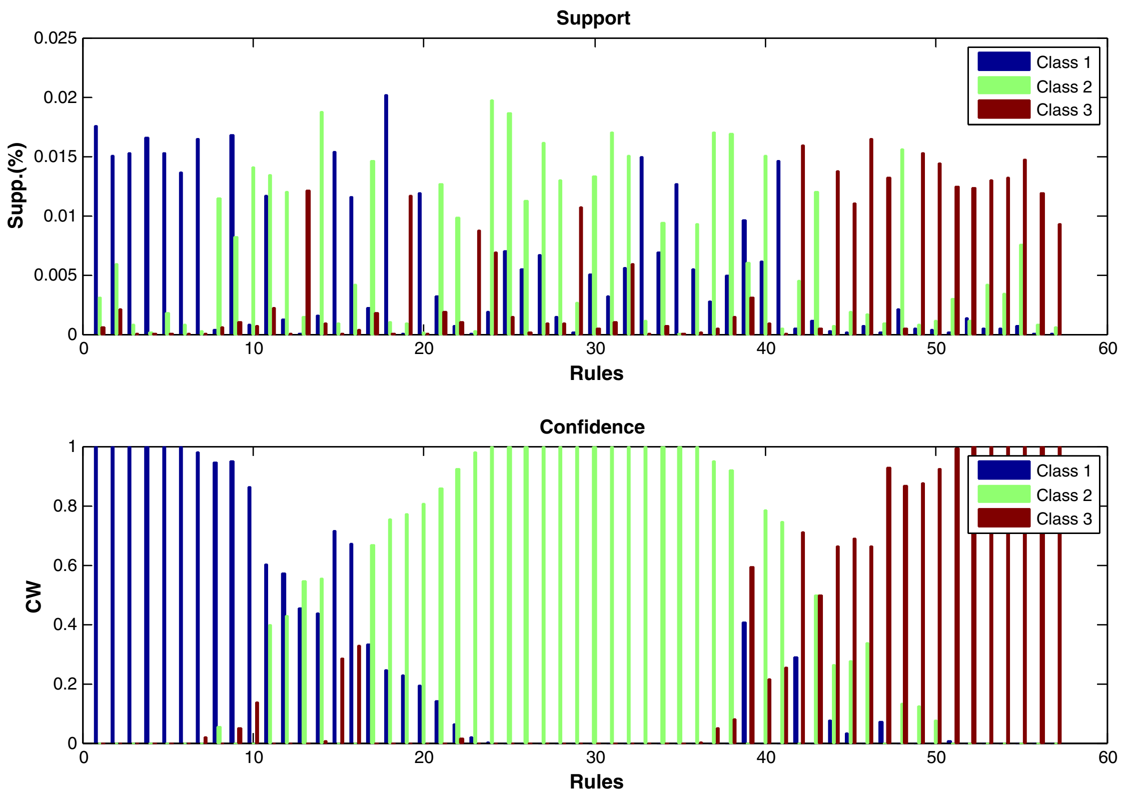

Fig. 7. Support and confidence values for the symbolic interpretation of the rule base.

6.4. Model interpretation

For one specific model, whether it is the best model selected in cross validation testing or the default model, symbolic interpretation of the parameters and graphic visualization allow the understanding of the model. The model interpretation and data visualization will be discussed for the spectral clustering algorithm for two well known benchmark datasets: the wine dataset (classification) and the housing dataset (regression).

Recall that the parameter ρ in (14) determines a neighborhood relation that results in the number of the rules of the model, computed by the spectral analysis. A large value of the parameter ρ results in a more general model, with a small number of rules, which is better for interpretation but less accurate than a more complex model. The analyst can use this parameter to adjust the model structure and obtain a more accurate or a more easily interpretable model depending on the objective of the analysis. Considering that the data have been standardized as the estimated z-scores, the default design is obtained by setting the parameter $\rho = 1.0$.

A simple and interpretable model for the wine dataset can be obtained by setting the parameter $\rho = 3.0$, resulting in a model with 17 rules. The support values, computed by (28), and the confidence values, computed from the rule outputs by (27), are shown in Fig. 5. The rules have been ordered to enhance the visualization. The support values show the percentage of records of each class for each rule. The confidence values represent the confidence of the model prediction from the corresponding rule.

The interpretation of the confidence values is related to the certainty of the prediction. Rules with unity confidence values result in a more reliable predictions than rules with partial confidence values. The output membership degrees, computed from the confidence rules in (7) for the training dataset records, are shown in Fig. 6, along with the misclassified examples. The records were ordered according to the class index and the membership values to enhance visualization. As shown in

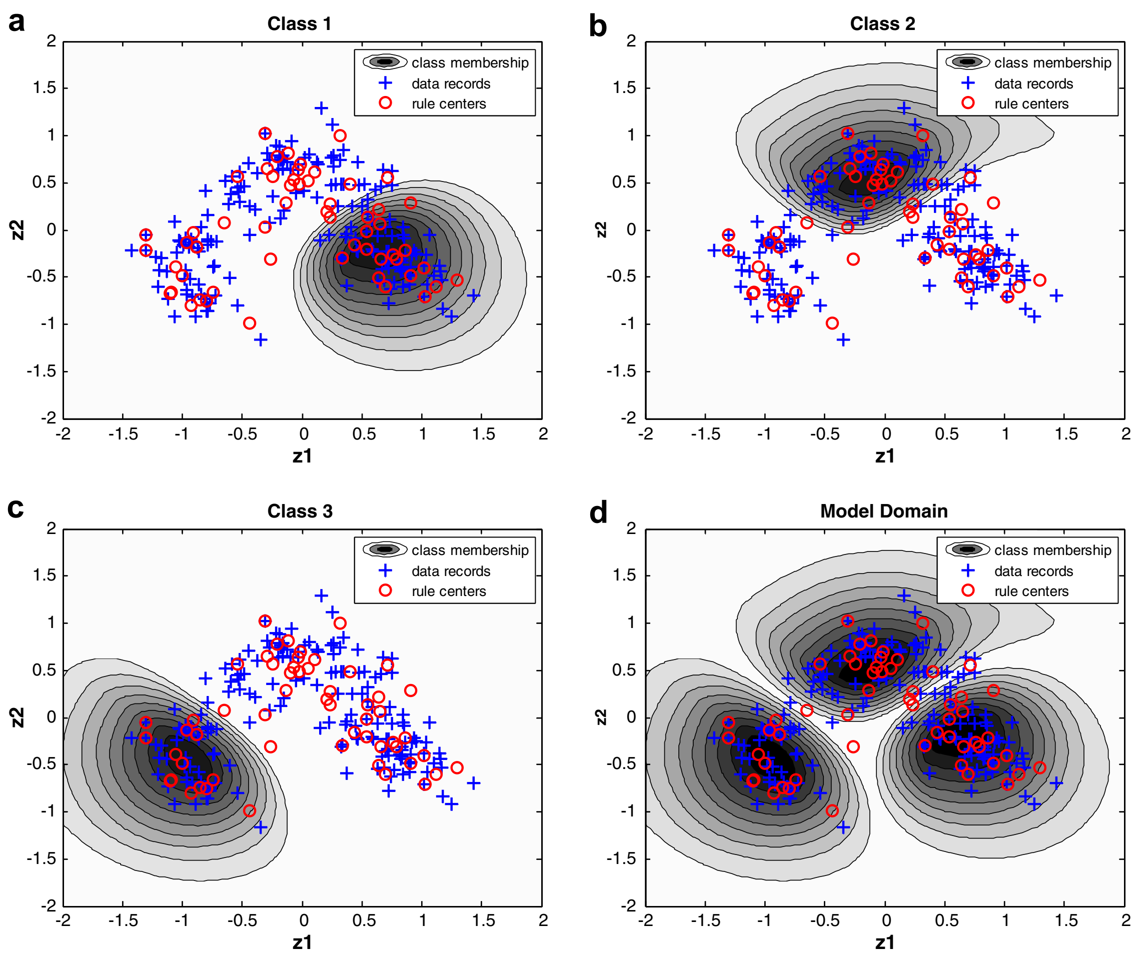

Fig. 8. Regions on the model domain in the principal components plane: (a) class 1: “small”, (b) class 2: “medium”, (c) class 3: “large”, and (d) the entire model domain.

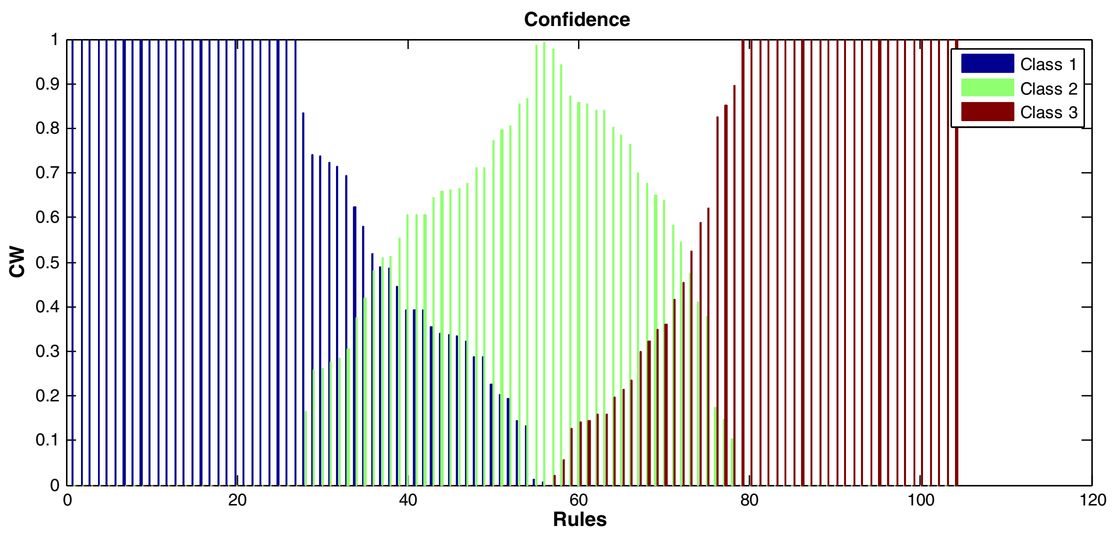

Fig. 9. Symbolic interpretation of the rule base.

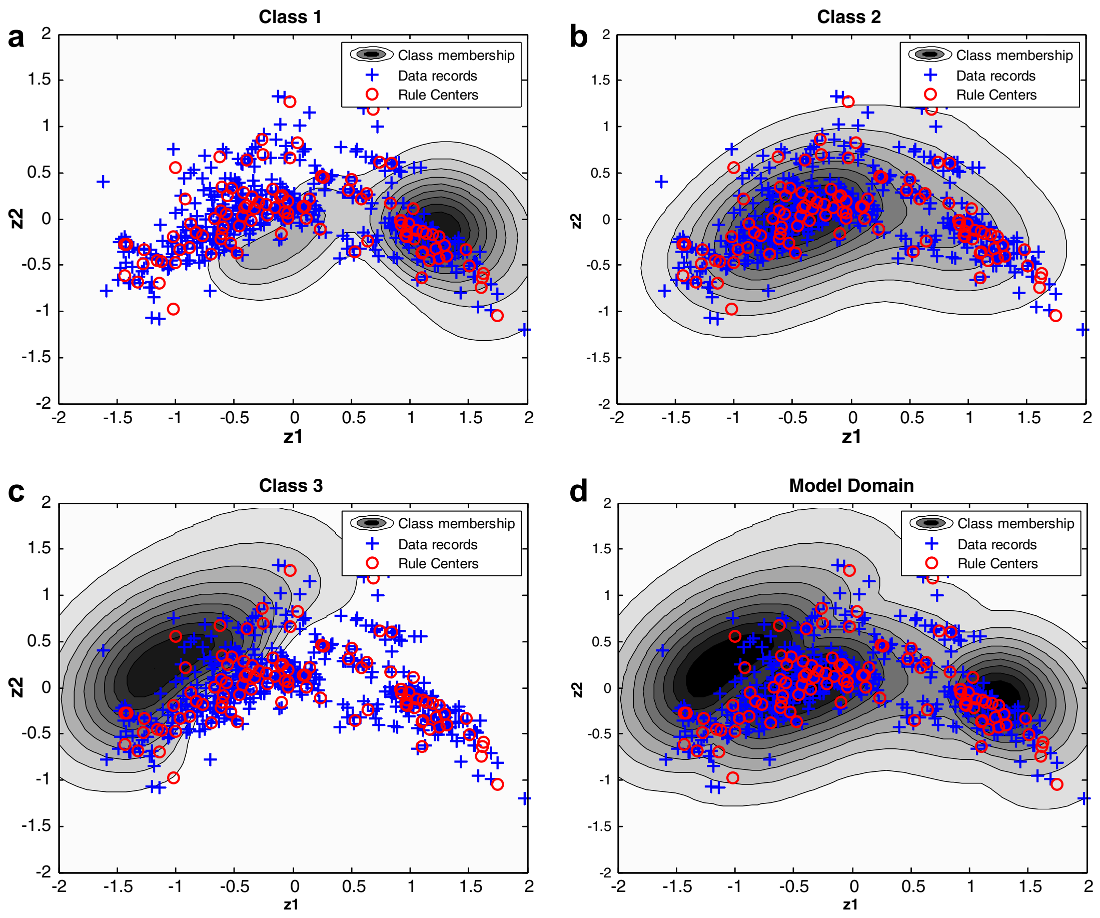

Fig. 10. Regions on the model domain in the principal components plane: (a) class 1: "small", (b) class 2: "medium", (c) class 3: "large", and (d) the entire model domain.

Fig. 6, the misclassification errors are likely to occur for small differences of the output membership values, which are produced by rules with partial confidence.

In this simple example, it is shown that a linguistic interpretation of the rules is not necessary to understand the model and use the model weights as a support for decision making. Nevertheless, it is possible for the analyst to associate the rules to symbols or names, which recall past experiences with the same problem. This type of reasoning can be extended straightforwardly for more complex models.

The support values and the rule confidence for the model computed based on the default design ($\rho = 1.0$) for the wine dataset are shown in Fig. 7. Despite of the great number of rules for this relatively simple problem, it is possible to identify the most representative rules for each class as so as the rules of which the centers are near to the border of each class.

The regions associated with each class, the positions of the rule centers and the model domain are visualized along with the data records in the plane generated by the two principal components in Fig. 8. Despite the distortion caused by the dimension reduction, all of the training data are enclosed by the region associated with each class. The model domain is the union of the regions associated with each class. If a new data record falls outside the model domain, the result provided by the model should be taken (with care) as an extrapolation.

The results of the model computed by the default design for the housing dataset are shown in Fig. 9. Because it is a regression problem, artificial classes were created to help the analyst with the model interpretation, whose confidence values, shown in Fig. 9, were computed by Eq. (26). These classes may represent ranges of the output variable values and may be interpreted as "small", "medium" and "large", for instance. This labeling may help the decision maker in the analysis of new cases by determining which rules are activated when a new datum is to be predicted. In regression problems, many rules have partial confidences, such that there is a great overlapping among the "classes". Although there are a large number of rules for this problem (104 in total), it is possible to identify the most prototypical rules for each range of values as the ones with the highest confidence values.

In Fig. 10, the regions for each class along with the data records and the rule centers are shown in the plane generated by the two principal components. The model domain is the union of the regions associated to each class. The interpretation of the data visualization is somewhat similar to that for classification problems.

7. Conclusions

This work has presented a semantic, coherent and organized set of known methods to obtain a conceptual framework for data understanding and approximate reasoning in complex systems modeling. In the proposed framework (which can be employed in classification or regression learning tasks), the number of interpolation functions of the model (Eqs. (1) and (2)) is computed using spectral analysis. Each interpolation function is associated with a symbol to generate a fuzzy rule. Each symbol is related to a center or prototype that is computed using a clustering algorithm and that can be used to interpret the model in the context of an application. The model parameters are optimized by solving a bounded quadratic programming problem and then interpreted as rule confidences. Graphic visualization of the rule confidences and projecting the data into a two-dimensional space spawned by the principal components have also been proposed, and these may help the user to understand the data and the model. The user can then use it a tool for decision support in the prediction of new data.

Numerical results obtained for the classification and regression benchmark datasets show that the selection of the model through cross-validation allows an accurate model to be found. The statistical tests have shown that the performance of the proposed approach is similar to that of the standard methods used to design the model (1). The solution of the bounded quadratic problem has shown to be effective at controlling the model capacity while preserving accuracy in classification and regression problems. Moreover, the default design with no free parameters to be set has obtained similar performance results to those of the LibSVM models in classification and regression problems. Nevertheless, the spectral analysis in the default design has resulted in a smaller number of interpolating functions to represent the model and is consequently more interesting for model interpretation and exploitation than SVM models.

The main focus of the proposed method is on the interpretability rather than the accuracy of the resulting model. This interpretability is achieved by the compact model obtained by the spectral analysis and quadratic programming parameter optimization. The graphical visualization of the rules' weights and the 2D linear projection of the data and the rules' centers enable the model interpretation for a better understanding of the problem. The main drawback of the method is the high computational cost for the solution of the spectral analysis, which can be a hurdle for problems with a huge number of records. Nevertheless, advances in multi-core hardware platforms and parallel implementation of numerical algorithms can deal with this handicap. Moreover, the more difficult problems are usually those with a low number of records, while those with a huge number of records can always be treated using sampling methods to obtain a more compact dataset.

The future direction of this research is to deeply explore the number of rules generated by the variation of the dispersion parameter in the affinity matrix in spectral analysis, so that it can be used as a tool for understanding data. Model interpretation using this set of models could also be improved in a hierarchical structure, thereby allowing a multi-modeling approach with multiple levels of representation.

Acknowledgments

The authors are grateful to the Brazilian Research Agencies, CNPq, FINEP and to CAPES-COFECUB Collaboration Agreement for their financial support for this research.

References

[1] D.S. Huang, Radial basis probabilistic neural networks: model and application, Int. J. Pattern Recognition Artif. Intell. 13 (1999) 1083–1101.
[2] P.V. Yee, S. Haykin, Regularized Radial Basis Function Networks: Theory and Applications, Wiley-Interscience, 2001.
[3] C.J.C. Burges, A tutorial on support vector machines for pattern recognition, Data Min. Knowl. Disc. 2 (1998) 121–167.
[4] B. Schölkopf, A.J. Smola, Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond, MIT Press, 2001.
[5] J.-S.R. Jang, C.-T. Sun, E. Mizutani, Neuro-fuzzy and Soft Computing: A Computational Approach to Learning and Machine Intelligence, Prentice-Hall, 1997.
[6] R. Babuka, H. Verbruggen, Neuro-fuzzy methods for nonlinear system identification, Ann. Rev. Contr. 27 (2003) 73–85.
[7] Y. Chen, J.Z. Wang, Support vector learning for fuzzy rule-based classification systems, IEEE Trans. Fuzzy Syst. 11 (6) (2003) 716–728.
[8] J. Chiang, P. Hao, Support vector learning mechanism for fuzzy rule-based modeling: a new approach, IEEE Trans. Fuzzy Syst. 12 (1) (2004) 1–12.
[9] J.L. Castro, L.D. Flores-Hidalgo, C.J. Mantas, J.M. Puche, Extraction of fuzzy rules from support vector machines, Fuzzy Sets Syst. 158 (2007) 2057–2077.
[10] E. Hüllermeier, Fuzzy methods in machine learning and data mining: status and prospects, Fuzzy Sets Syst. 156 (2005) 387–406.
[11] E. Hüllermeier, Fuzzy methods in machine learning and data mining: status and prospects, Appl. Soft Comput., in press, doi:10.1016/j.asoc.2008.01.004.
[12] S.-M. Zhou, J.Q. Gan, Low-level interpretability and high-level interpretability: a unified view of data-driven interpretable fuzzy system modeling, Fuzzy Sets Syst. 159 (2008) 3001–3131.
[13] J.M. Alonso, L. Magdalena, G. González-Rodríguez, Looking for a good fuzzy system interpretability index: an experimental approach, Int. J. Approx. Reason. 51 (1) (2009) 115–134.
[14] S. Guillaume, Designing fuzzy inference systems from data: an interpretability-oriented review, IEEE Trans. Fuzzy Syst. 9 (3) (2001) 423–443.
[15] R. Mikut, J. Jäkel, L. Gröll, Interpretability issues in data-based learning of fuzzy systems, Fuzzy Sets Syst. 150 (2005) 179–197.
[16] S. Chen, S.A. Billings, W. Luo, Orthogonal least squares methods and their application to non-linear system identification, Int. J. Contr. 50 (5) (1989) 1873–1896.
[17] S. Chen, C.F.N. Cowan, P.M. Grant, Orthogonal least squares learning algorithm for radial basis function networks, IEEE Trans. Neural Netw. 2 (2) (1991) 302–309.
[18] S.A. Billings, G.L. Zheng, Radial basis function network configuration using genetic algorithms, Neural Netw. 8 (6) (1995) 877–890.
[19] G.P. Liu, V. Kadirkamanathan, Multiobjective criteria for neural network structure selection and identification of nonlinear systems using genetic algorithms, IEE Proc. Part P: Contr. Theory Appl. 146 (5) (1999) 373–382.
[20] J. Gonzalez, I. Rojas, J. Ortega, H. Pomares, J. Fernandez, A.F. Diaz, Multiobjective evolutionary optimization of the size, shape, and position parameters of radial basis function networks for function approximation, IEEE Trans. Neural Netw. 14 (6) (2003) 1478–1495.
[21] S. Chen, Y. Wu, B.L. Luk, Combined genetic algorithm optimization and regularized orthogonal least squares learning for radial basis function networks, IEEE Neural Netw. 10 (5) (1999) 1239–1243.
[22] A.M.S. Barreto, H.J.C. Barbosa, N.F.F. Ebecken, GOLS—genetic orthogonal least squares algorithm for training RBF networks, Neurocomputing 69 (2006) 2041–2064.
[23] Y. Yam, P. Baranyi, C.-T. Yang, Reduction of fuzzy rule base via singular value decomposition, IEEE Trans. Fuzzy Syst. 7 (2) (1999) 120–132.
[24] M. Setnes, R. Babuška, Rule base reduction: some comments on the use of orthogonal transforms, IEEE Trans. Syst. Man Cybernet., Part B 31 (2) (2001) 199–206.
[25] O. Cordon, F. Gomide, F. Herrera, F. Hoffmann, L. Magdalena, Ten years of genetic fuzzy systems: current framework and new trends, Fuzzy Sets Syst. 141 (1) (2004) 5–31.
[26] O. Cordon, R. Alcala, J. Alcala-Fernandez, I. Rojas, Guest editorial genetic fuzzy systems: What's next? An introduction to the special section, IEEE Trans. Fuzzy Syst. 15 (4) (2007) 533–535.
[27] P. Espejo, S. Ventura, F. Herrera, A survey on the application of genetic programming to classification, IEEE Trans. Systems Man Cybernet. – Part C: Appl. Rev. 40 (2) (2010) 121–144.
[28] O. Cordon, F. Herrera, P. Villar, Generating the knowledge base of a fuzzy rule-based system by the genetic learning of the data base, IEEE Trans. Fuzzy Syst. 9 (4) (2001) 667–674.
[29] H. Ishibuchi, T. Yamamoto, Fuzzy rule selection by multi-objective genetic local search algorithms and rule evaluation measures in data mining, Fuzzy Sets Syst. 141 (1) (2004) 59–88.
[30] J. Casillas, O. Cordón, M. del Jesus, F. Herrera, Genetic tuning of fuzzy rule deep structures preserving interpretability and its interaction with fuzzy rule set reduction, IEEE Trans. Fuzzy Syst. 13 (1) (2005) 13–29.
[31] J. González, I. Rojas, H. Pomares, L.J. Herrera, A. Guillén, J.M. Palomares, F. Rojas, Improving the accuracy while preserving the interpretability of fuzzy function approximators by means of multi-objective evolutionary algorithms, Int. J. Approx. Reason. 44 (2007) 32–44.
[32] A.G. Evsukoff, S. Galichet, B.S.L.P. de Lima, N.F.F. Ebecken, Design of interpretable fuzzy rule-based classifiers using spectral analysis with structure and parameters optimization, Fuzzy Sets Syst. 160 (2009) 857–881.
[33] F.J. Berlanga, A.J. Rivera, M.J. del Jesus, F. Herrera, GP-COACH: genetic programming based learning of COmpact and ACcurate fuzzy rule based classification systems for high dimensional problems, Inform. Sci. 180 (8) (2010) 1183–1200.
[34] M.J. Gacto, R. Alcalá, F. Herrera, Integration of an index to preserve the semantic interpretability in the multi-objective evolutionary rule selection and tuning of linguistic fuzzy systems, IEEE Trans. Fuzzy Syst. 18 (3) (2010) 515–531.
[35] H. Ishibuchi, T. Nakashima, Effect of rule weights in fuzzy rule-based classification systems, IEEE Trans. Fuzzy Syst. 9 (4) (2001) 506–515.
[36] H. Ishibuchi, T. Yamamoto, Rule weight specification in fuzzy rule-based classification systems, IEEE Trans. Fuzzy Syst. 13 (4) (2005) 428–435.
[37] M.J. Zolghadri, E.G. Mansoori, Weighting fuzzy classification rules using receiver operating characteristics (ROC) analysis, Inform. Sci. 177 (2007) 2296–2307.
[38] L. Rondeau, R. Ruelas, L. Levrat, M. Lamotte, A defuzzification method respecting the fuzzification, Fuzzy Sets Syst. 86 (1997) 311–320.
[39] F.R.K. Chung, Spectral Graph Theory, CBMS Regional Confernece Series in Mathematics, vol. 92, American Mathematic Society, 1997.
[40] U. Luxburg, A tutorial on spectral clustering, Statist. Comput. 17 (4) (2007) 395–416.
[41] W. Li, W.-K. Ng, Y. Liu, K.-L. Ong, Enhancing the effectiveness of clustering with spectra analysis, IEEE Trans. Knowl. Data Eng. 19 (7) (2007) 887–902.
[42] M. Filippone, F. Camastra, F. Masulli, S. Rovetta, A survey of kernel and spectral methods for clustering, Pattern Recognition 41 (2008) 176–190.
[43] T. Xiang, S. Gong, Spectral clustering with eigenvector selection, Pattern Recognition 41 (2008) 1012–1029.
[44] A.Y. Ng, M.I. Jordan, Y. Weiss, On spectral clustering: analysis and an algorithm, in: T.G. Dietterich, S. Becker, Z. Ghahramani (Eds.), Advances in Neural Information Processing Systems, vol. 14, MIT Press, Cambridge, MA, 2002.
[45] D. Dubois, E. Hüllermeier, H. Prade, A systematic approach to the assessment of fuzzy association rules, Data Min. Knowl. Disc. 13 (2006) 167–192.
[46] M. Holena, Measures of ruleset quality for general rules extraction methods, Int. J. Approx. Reason. 50 (2009) 867–879.
[47] D.C. Montgomery, E.A. Peck, G.G. Vining, Introduction to Linear Regression Analysis, fourth ed., John Wiley and Sons, 2006.
[48] V.F. Vieira, A.G. Evsukoff, B.S.L.P. de Lima, S. Galichet, Learning fuzzy rule based classifier in high performance computing environment, in: Proceedings of the IFSA-EUSFLAT 2009, Lisbon, 20–24 July, 2009.
[49] A. Asuncion, D.J. Newman, UCI Machine Learning Repository. University of California, School of Information and Computer Science, Irvine, CA, 2007. <http://www.ics.uci.edu/~mlearn/MLRepository.html>.
[50] S. Garcia, F. Herrera, An extension on statistical comparisons of classifiers over multiple data sets for all pairwise comparisons, J. Mach. Learn. Res. 9 (2008) 2677–2694.

[51] S. Garcia, A. Fernandez, J. Luengo, F. Herrera, A study of statistical techniques and performance measures for genetics-based machine learning: accuracy and interpretability, Soft Comput. 13 (2009) 959–977.
[52] S. Garcia, A. Fernandez, J. Luengo, F. Herrera, Advanced nonparametric tests for multiple comparisons in the design of experiments in computational intelligence and data mining: experimental analysis of power, Inform. Sci. 180 (10) (2010) 2044–2064.
[53] J. Alcalá-Fdez, L. Sánchez, S. García, M.J. del Jesus, S. Ventura, J.M. Garrell, J. Otero, C. Romero, J. Bacardit, V.M. Rivas, J.C. Fernández, F. Herrera, KEEL: a software tool to assess evolutionary algorithms to data mining problems, Soft Comput. 13 (3) (2009) 307–318.
[54] J. Alcalá-Fdez, A. Fernandez, J. Luengo, J. Derrac, S. García, L. Sánchez, F. Herrera, KEEL data-mining software tool: data set repository, integration of algorithms and experimental analysis framework, J. Multiple-Valued Logic Soft Comput. 17 (2–3) (2011) 255–287.