Unification of Fuzzy SVMs and Rule Extraction Methods through imprecise Domain Knowledge

Christian Moewes
Faculty of Computer Science,
University of Magdeburg, Germany
cmoewes@cs.uni-magdeburg.de

Rudolf Kruse
Faculty of Computer Science,
University of Magdeburg, Germany
kruse@iws.cs.uni-magdeburg.de

Abstract

In this paper, we want to motivate the combination of kernel-based methods with fuzzy rule extraction methods to describe uncertain domains by fuzzy models. We thus introduce and motivate the concept of a fuzzy support vector machine (FSVM) to incorporate impreciseness into kernel machines. Furthermore, we present the idea of a positive definite fuzzy classifier (PDFC), the rules of which are obtained by kernel-based models. We conclude with two vague conceptions to associate FSVM with PDFC to finally obtain understandable and meaningful fuzzy rules.

Keywords: Binary Classification, Fuzzy Rule-Based Classifier, Fuzzy Support Vector Machine.

1 Introduction

Kernel-based methods, and support vector machines (SVMs) in particular [20], play one of the most important roles in machine learning today. They are announced to both generalize considerably on unseen data and perform well on high-dimensional input spaces. Nonetheless, the application of these methods is not popular compared to intuitive learning machines.

Especially in automation and control, the application of models based on fuzzy set theory (FST) [21] became substantive. The vague and imprecise expressions that are used by human beings to describe processes can be modeled gracefully by FST. Fuzzy classifiers (FCs) based on linguistic rules provide a comprehensive way to illustrate underlying concepts of complicated systems. Nowadays, they can be found in many real-world applications [14].

Several attempts have been made to find connections between fuzzy models and kernel-based methods. Essentially, two directions can be distinguished in the research community. First of all, we find approaches that try to incorporate FST directly into SVMs, i.e. *Fuzzy Support Vector Machines (FSVMs)*, for both classification [7, 12] and regression problems [18]. The main motivation is the fact that SVMs are quite sensitive to outliers and noise. FST provides an appropriate toolbox of methods to tackle those problems, i.e., uncertainty, impreciseness, and noisy data [8].

The second direction focuses on the generation of fuzzy classifiers based on the output of kernel machines. In essence, we encounter methods to extract fuzzy-rule based classifiers from SV machines for both settings, i.e., classification [4, 5] and regression [6]. The objectives are different compared to the first direction. Fuzzy models become cumbersome in complex systems with dozens of input variables since they suffer from "the curse of dimensionality". Thus combining the generalization of SVMs with the interpretability of FCs might be a striking idea to overcome these difficulties.

We claim that these two directions can be uni-

L. Magdalena, M. Ojeda-Aciego, J.L. Verdegay (eds): Proceedings of IPMU'08, pp. 1527–1534
Torremolinos (Málaga), June 22–27, 2008

fied with the goal to overcome the disadvantages of these approaches. Therefore we will briefly introduce the concept of both support vector machines and its extension to FST in Sect. 2. Afterwards we will give an overview of rule-based fuzzy classifiers in Sect. 3. Moreover, we will present *Positive Definite Fuzzy Classifiers (PDFCs)* since their rule bases can be extracted from special kernel machines. Applying model reduction methods to PDFCs will enable us to obtain comprehensible and interpretable fuzzy rules. Section 4 will raise the question if it is feasible to unify FSVMs and PDFCs since both are based on kernel machines. Finally, we will conclude our report by summarizing the main thoughts in Sect. 5.

2 Support Vector Machines

Let us start to formally introduce the basic concepts that we are going to talk about. Therefore we must define the prerequisites of the problems we are looking at. Suppose we are given the input space $\mathcal{X}$ (not necessarily a vector space) and the output space $\mathcal{Y}$. Since we deal with a binary classification problem, $\mathcal{Y} = \{\pm 1\}$. We observe l training patterns $(x_i, y_i) \in \mathcal{S} \subseteq \mathcal{X} \times \mathcal{Y}$ where $i = 1, \ldots, l$. They have been drawn i.i.d. from an unknown distribution.

Given numerical input variables, we can write $\mathcal{X} \subset \mathbb{R}^n$ and hence $x_i \mapsto \mathbf{x}_i$. Our goal is to separate the data with a linear hyperplane $\{\mathbf{x} : \langle \mathbf{w}, \mathbf{x} \rangle + b = 0\}$ where $\mathbf{w} \in \mathbb{R}^n$ and $b \in \mathbb{R}$ are the norm vector and the bias of the hyperplane, respectively. The decision function of a hyperplane classifier which shall predict y' for any $\mathbf{x}$ corresponds to

$$f(\mathbf{x}) = \operatorname{sgn}\left(\langle \mathbf{w}, \mathbf{x} \rangle + b\right). \tag{1}$$

There is an infinite number of possible hyperplanes, however, we are looking for the one that maximizes the margin between every training pattern and the hyperplane. Such a hyperplane is called optimal since it is unique and has the best generalization performance on unseen data. If all points $(x_i, y_i) \in \mathcal{S}$ can be separated linearly by a hyperplane, we can obtain the optimal hyperplane by solving the following quadratic optimization problem with linear inequality constraints:

$$\begin{aligned} \underset{\mathbf{w},b}{\text{minimize}} \quad & \tau(\mathbf{w}) = \frac{1}{2}\|\mathbf{w}\| \\ \text{subject to} \quad & y_i(\langle \mathbf{w}, \mathbf{x}_i \rangle + b) \geq 1, \ \forall i = 1, \ldots, l \end{aligned}$$

Usually not all training patterns can be separated perfectly. Therefore we introduce slack variables ξ_i with $i = 1, \ldots, l$ in order to relax the optimization problem to

$$\underset{\mathbf{w},b,\boldsymbol{\xi}}{\text{minimize}} \quad \tau(\mathbf{w}, \boldsymbol{\xi}) = \frac{1}{2}\|\mathbf{w}\| + C\sum_{i=1}^{l} \xi_i \tag{2}$$

$$\text{subject to} \quad y_i(\langle \mathbf{w}, \mathbf{x}_i \rangle + b) \geq 1 - \xi_i \tag{3}$$

$$\text{and} \quad \xi_i \geq 0, \ \forall i = 1, \ldots, l. \tag{4}$$

Here, $\boldsymbol{\xi} = (\xi_1, \ldots, \xi_l)$ corresponds to the slack variables ξ_i and C is a global parameter that has to be determined by the user. The bigger C, the easier training patterns may violate the constraint (3). By introducing the Lagrangian of the primal problem (2), we end up solving the dual

$$\underset{\alpha_1,\ldots,\alpha_l}{\text{maximize}} \quad \sum_{i=1}^{l} \alpha_i - \frac{1}{2}\sum_{i,i'=1}^{l} y_i y_{i'} \alpha_i \alpha_{i'} \langle \mathbf{x}_i, \mathbf{x}_{i'} \rangle \tag{5}$$

$$\text{subject to} \quad \sum_{i=1}^{l} y_i \alpha_i = 0 \tag{6}$$

$$\text{and} \quad 0 \leq \alpha_i \leq C, \ \forall i = 1, \ldots, l. \tag{7}$$

In practice, only few problems can be solved by a linear classifier. Hence the problem has to be reformulated in a nonlinear way. This is done by mapping the input space $\mathcal{X}$ to some high-dimensional feature space $\mathcal{H}$ by $\Phi : \mathcal{X} \mapsto \mathcal{H}$ where Φ satisfies Mercer's condition [16]. We can thus solve our nonlinear optimization problem linearly in $\mathcal{H}$ by computing the scalar product $K(x, x') = \langle \Phi(x), \Phi(x') \rangle$ which is called kernel. We must simply replace the occurrence of the scalar product in Eq. (5) with a chosen kernel function. Finally, the discrimination function (1) becomes

$$f(x) = \operatorname{sgn}\left(\sum_{i=1}^{l} y_i \alpha_i K(x, x_i) + b\right).$$

See [16] for a collection of kernel functions and further details on SVMs.

2.1 Fuzzy Support Vector Machines

The acquisition of data in most real-world applications is usually imprecise, uncertain and not complete. Therefore is is suitable to embody the abstracted information by fuzzy sets. Especially SV machines seem to be quite sensitive to noise and points that were rather improbably drawn from the underlying data generating distribution.

The only free parameter of a SVM is C which regularizes the penalty term in Eq. (2) and hence the classification error. This parameter is usually fixed for every input pattern during the training process. A priori, all training patterns are treated the same which might be crucial for the SVM regarding outliers and noisy data points. So, the learning machine may suffer from overfitting.

As a consequence, the concept of a fuzzy support vector machine (FSVM) has been introduced [7, 12]. In particular, a membership value μ_i is assigned to every training pattern x_i. Thus the training sample $\mathcal{S}$ is mapped to a fuzzy training sample

$$\mathcal{S}_f = \{(x_1, y_1, \mu_1), \ldots, (x_l, y_l, \mu_l)\}$$

where the membership values for positive and negative class are denoted as μ_i^+ and μ_i^-, respectively. Both values are assigned independently.

Similar to the constrained optimization problem of Eq. (2), FSVM tries to optimize the same variables. However, it fuzzifies the penalty term containing the regularizer C. The optimal hyperplane using FSVM can obtained solving

$$\underset{\mathbf{w},b,\boldsymbol{\xi},\boldsymbol{\mu}}{\text{minimize}} \quad \tau(\mathbf{w}, \boldsymbol{\xi}, \boldsymbol{\mu}) = \frac{1}{2}\|\mathbf{w}\| + C\sum_{i=1}^{l} \mu_i^m \xi_i \quad (8)$$

subject to constraints (3) and (4) where $\boldsymbol{\mu} = (\mu_1, \ldots, \mu_l)$ and m regularizes the fuzziness of the fuzzified penalty term. The dual problem for FSVM can be obtained by deriving the Lagrangian of Eq. (8) and hence only differs in constraining the α_i's: Maximize (5) subject to (6) and

$$0 \leq \alpha_i \leq C\mu_i^m, \quad \forall i = 1, \ldots, l.$$

In order to apply FSVM, the membership values $\boldsymbol{\mu}$ have to be defined. In [7], the authors suggested to learn these values as follows. First they removed outliers and then fuzzified the remaining positive and negative instances independently by some membership functions. Finally, both sets were combined to $\mathcal{S}_f$.

3 Additive Rule-Based FCs

Whereas the last section gave an overview how to incorporate uncertainty into a kernel machine, this section explaines how fuzzy rules can be obtained from such learning machines. Therefore we have to define a special type of FC and special kernel functions (see Sect. 3.1). Before we come to this point, let us introduce fuzzy rules and classifiers that are based on them.

Assume that every input dimension x_j with $j = 1, \ldots, n$ is associated with K_j linguistic labels [11], e.g., *warm*, *large*, and *low*. All those labels are represented by fuzzy sets A_j^k for $k = 1, \ldots, K_j$ on the input axes, e.g., temperature, size, and pressure. Furthermore, let M zero-order Takagi-Sugeno fuzzy rules [19] of the following form be given.

$$R_m: \text{ IF } \bigwedge_{j=1}^{n} x_j \text{ is } A_j^{k(j,m)} \text{ THEN } b_m \quad (9)$$

Here, $m = 1, \ldots, M$, the consequent $b_m \in \mathbb{R}$, $\bigwedge_{j=1}^n$ is the fuzzy conjunction over all n input variables, i.e., product. The subscript $k(j, m)$ is an index function that determines which of the fuzzy sets A_j^k applies in the mth rule. The membership function of A_j^k is denoted as $\mu_j^k : \mathbb{R} \mapsto [0, 1]$.

Since we deal with additive fuzzy classifiers, addition is selected as fuzzy aggregation in order to compute the output. The well known center-of-area (COA) method is used for defuzzification s.t. the output $F : \mathbb{R}^n \mapsto \mathbb{R}$ can be computed as

$$F(\mathbf{x}) = \frac{\sum_{m=1}^{M} b_m \prod_{j=1}^{n} \mu_j^{k(j,m)}(x_j)}{\sum_{m=1}^{M} \prod_{j=1}^{n} \mu_j^{k(j,m)}(x_j)}. \quad (10)$$

Note that (10) is not defined if the denominator is zero. The authors of [4] elegantly add

$$R_0: \text{ IF } \bigwedge_{j=1}^{n} x_j \text{ is } A_j^{k(j,0)} \text{ THEN } b_0 \qquad (11)$$

whereas $b_0 \in \mathbb{R}$ and $\mu_0^k(x_k) \equiv \mu_0^k \equiv 1$ for any $k \in \mathbb{N}$. This rule's consequent b_0 will represent the bias of Eq. (1) as we will see later. Adding (11) to (10), we eventually get

$$F(\mathbf{x}) = \frac{b_0 + \sum_{m=1}^{M} b_m \prod_{j=1}^{n} \mu_j^{k(j,m)}(x_j)}{1 + \sum_{m=1}^{M} \prod_{j=1}^{n} \mu_j^{k(j,m)}(x_j)}$$

which can be directly used for binary classification by thresholding. We thus obtain the binary decision rule

$$f(\mathbf{x}) = \operatorname{sgn}(F(\mathbf{x})). \qquad (12)$$

3.1 Positive Definite Fuzzy Classifiers

Note that (12) already resembles (1) to a high degree. We will arrive at a decision function based on kernels if all μ_j^k have been generated by transformation of positive definite reference functions[1] μ_j, i.e., $\mu_j^k(x_j) = \mu_j(x_j - z_j^k)$ where $z_j^k \in \mathbb{R}$. Fuzzy classifiers that fulfill this criterion are called *Positive Definite FCs (PDFCs)*. The decision function of a PDFC is comprised by

$$f(\mathbf{x}) = \operatorname{sgn}\left(\sum_{m=1}^{M} b_m K(\mathbf{x}, \mathbf{z}_m) + b_0\right)$$

where $z_m = (z_1^m, \ldots, z_n^m)^T$ and $K(\mathbf{x}, \mathbf{z}_m) = \prod_{j=1}^{n} \mu_j(x_j - z_j^m)$ represents a Mercer kernel[2] as introduced in Sect. 2. Many reference functions but, e.g., the asymmetric triangle and the trapezoid, are positive definite s.t. numerous different PDFCs can be constructed. Here, we only want to name the Gaussian membership (reference) function

$$\mu_j^k = \exp\left(\frac{(x_j - z_j^k)^2}{2(\sigma_j^k)^2}\right) \qquad (13)$$

where z_j^k and σ_j^k represent the center and the width of the kth fuzzy set A_j^k.

[1] See [4] for definition of PD reference function.

[2] Note that $K_j \equiv 1 \; \forall j = 1, \ldots, n$ since $z_j^m \equiv z_j^k$.

3.2 FCs from Kernel Machines

Having established the link between a PDFC and a SVM, we are able to extract fuzzy rules directly from a kernel machine. In [4], the algorithm that constructs a PDFC from a SVM is rather naïve. However, we must note that it seems to be the only procedure in the literature so far.

First, a positive definite reference function has to be chosen to form a Mercer kernel. Applying SVM with this kernel in the next step, a set of support vectors $\mathcal{X}_{SV} = \{\mathbf{x}_{(m)} : m = 1, \ldots, M\} \subseteq \mathcal{X}$, their corresponding weights $\{y_{(m)}\alpha_{(m)} : y_{(m)} \in \mathcal{Y},\ 0 < \alpha_{(m)} \leq C,\ m = 1, \ldots, M\}$ and the bias b (see Eq. (1)) are obtained.

Ultimately the PDFC is found by mapping every weighted SV to a fuzzy rule R_m, i.e., $\mathbf{z}_m \leftarrow \mathbf{x}_{(m)}$ and $b_m \leftarrow y_{(m)}\alpha_{(m)}$, where $m = 1, \ldots, M$. The hyperplane bias $b_0 \leftarrow b$.

There are some advantages applying SVM for fuzzy rule generation. Whereas FCs very often fail in high-dimensional input spaces $\mathcal{X}$, SVMs generalize very well in those spaces provided that the kernel has been selected in a reasonable way. In particular, m is only bounded by l. However, $m \approx l$ can be still very large and thus the rule base will be everything but intuitive.

Still, we have to ask the question if a SVM can generate a fuzzy rule base. Clearly, a SVM with Gaussian kernel is functional equivalent to a Gaussian RBF network [17]. Such a network is then again comparable to fuzzy inference systems in terms of the decision function [9]. This might be a theoretical link between a PDFC and a kernel machine, i.e, SVM.

But what if we choose another reference function than the Gaussian one (13)? In this case there is no theoretical justification for renaming a basis function a "fuzzy set" in order to obtain a comprehensible model [8]. Contrary to this statement, we found the following claim [4]. Using an arbitrary reference function for a kernel, the obtained FC will be still related to a generalized SVM [13].

3.3 Model Reduction Methods

The extracted rules by SV learning, as proposed in the last section, will generalize quite well if all constraints are met. Nevertheless the interpretation of the rule base is cumbersome and hard since SV learning is not intuitive to human beings. Therefore several model reduction methods have been proposed. We will discuss the most promising ideas in the following.

In the year 2005, the incorporation of a fuzzy partition given by domain experts was suggested [15] to approximate the fuzzy rule base received by SV learning. Not surprisingly, this algorithm is called *Interpretable Fuzzy Set Approximation (IFSA)*. Only fuzzy rules that have a large contribution are considered as output rules. In fact, the membership degree of every component $x_j \in \mathbb{R}$ of $\mathbf{x}_m \in \mathcal{X}_{SV}$ is evaluated at every component of the given fuzzy partition. If this membership values is below a user defined threshold, then the input feature x_j will be removed from the rule R_m.

Using standard benchmark data sets and one real-world data set as well, two heavy improvements could be shown compared to [4]. First of all, the number of rules became smaller. Second, the antecedents of the rules only needed a subset of input features. Although the performance of the approximation heavily depends on the fuzzy partition, even this enables the user to apply domain knowledge to the learning step.

In 2007, a combination of three reduction steps was proposed [10]. Initially, the SVM is learned by the so-called reduced set (RS) method that identifies only significant SVs. The reduced SVs are then transformed to fuzzy rules before the second step is applied. Here, alike fuzzy sets are merged together based on a similarity measure. The remaining fuzzy rules are finally approximated by an orthogonal least-square fit.

In the same year, another rule reduction method called was proposed [3]. This algorithm named *Fuzzy Rule Extraction from SVM (FREx-SVM)* also needs a user defined fuzzy partition of all input variables. Compared to IFSA [15], however, two different evaluation criteria, i.e., fuzzy accuracy, and fuzzy coverage, are utilized to reduce the rule base.

Table 1: Comparison of model reduction methods based on the Iris data set.

METHOD	M	$\#\mu_j$	**ACC.**
IFSA [15]	5	10	87.3%
combination [10]	27	81	-
FREx-SVM[3]	33.5	134	94%

In order to evaluate the performance, every approach was tested on several data sets. Taking the intersection of all tested data sets, the famous Iris data set [2] is the only application where results can be compared for all three methods. The Iris data contain 3 classes, 50 instances per class and 4 numeric attributes.

Based on the published results, Table 1 shows three important evaluation criteria, i.e, the number of generated rules M, the number of membership functions $\#\mu_j$ used in the IF part of the rules, and the accuracy on test data computed by the leave-one-out method. The SV machines were trained using the one-against-one approach in order to handle this multi-class problem.

Not surprisingly, IFSA results in a small rule base with $M = 5$ due to the well chosen fuzzy partition. Nevertheless it is astonishing that only 50% of all possible membership functions and thus input features have been used to construct rules. The other algorithms come up with bigger rule bases since they either do not have any user input (combination of tools) or the given fuzzy partition is rather simple (FREx-SVM).

Of course there do exist several fuzzy classifier, e.g. *eClass* [1], that perform better on the Iris data. They even result in less fuzzy rules. Here we only concentrate on model reduction approaches, however, which are based on SV learning. Exhaustive tests of the proposed methods on more real-world data sets have to be performed in future.

4 Fuzzy Rule Extraction from FSVM

So far, we presented two approaches to integrate uncertainty into kernel machines, i.e, the fuzzy support vector machine, and the PDFC that is based on SV learning. Whereas the former model is trained on fuzzified input data $\mathcal{S}_f$, the latter one fuzzifies the model output (the support vectors). FSVM elegantly incorporates domain experts' knowledge by fuzzy partition in order to obtain the relevant SVs. Using fuzzy models based on (9), we are strongly interested in obtaining a few comprehensible rules that still generalize well.

We could not find any approach in literature that tries to interlink FSVM and rule-based FC with each other. Suppose the following procedure. Experts initially specify a fuzzy partition for every input feature. Then the given training data is fuzzified based on the domain knowledge. Consequently, a FSVM can be modeled out of $\mathcal{S}_f$. Eventually its support vectors are used to create fuzzy rules.

In Sect. 3.3, we suggested different ideas that can be used to reduce the resulting fuzzy rule base. Linking FSVM and PDFCs, we strongly recommend to apply IFSA [15] since it performed well in practice. Furthermore, IFSA can simply reprocess the domain experts' fuzzy partition with the objective to obtain interpretable rules based on relevant support vectors.

The relation between FSVM and the rule extraction process, however, heavily depends on the used kernel. In fact, the only kernel that allows this combination using FSVM is the Gaussian RBF kernel generated by Gaussian reference functions (13). Yet there are several alternatives to preserve the advantages of both kernel machines and fuzzy classifiers.

Foremost, one could possibly extend FSVM to generalized FSVM following the ideas of [13]. Thus arbitrary reference functions μ_j could be used to construct a kernel machine. Another potential idea is to define a optimization problem that directly outputs fuzzy points instead of crisp support vectors. Reviewing relevant literature on FSVM and fuzzy classifiers based on SV learning, we claim that the unification of both directions has neither been analyzed nor implemented yet.

5 Conclusions

In this paper, we tried to close the gap between comprehensible fuzzy models that are cumbersome in high-dimensional spaces and kernel machines which perform well even in infinite spaces. However, kernel-based classifiers are sensitive to outliers and not easy to understand. To overcome these adversities, we basically suggested to combine both methods by incorporating uncertainty, e.g., a fuzzy partition of all input features, into the kernel machine.

Therefore we introduced the concept of fuzzy support vector machines that assign membership values to every training instance. Solving a quadratic optimization problem containing fuzzy numbers, the resulting model finally outputs a smaller set of support vectors compared to standard SV machines.

The second approach constructs fuzzy rules based on support vector learning by one-to-one mapping of a SV to a new fuzzy rule. The initial set of rules usually has to be reduced to guarantee comprehensibility and interpretability of the fuzzy classifier. We thence showed some model reduction methods and tried to compare them in terms of their fuzzy output.

In order to take advantages of both ideas, we finally suggested to link FSVM and fuzzy rule extraction based on SV learning. We will focus our research on proving theoretical justifications and performing on potential real-world problems to validate the proposed unification.

Acknowledgements

We are very grateful to two anonymous reviewers who offered several suggestions and critical comments that we partly used to revise and improve this article.

References

[1] P. Angelov, X. Zhou, and F. Klawonn. Evolving fuzzy rule-based classifiers. In *Proceedings of the 2007 IEEE Symposium on Computational Intelligence in Image and Signal Processing. (CIISP 2007)*, pages 220–225, Honolulu, HI, Apr. 2007.

[2] A. Asuncion and D. J. Newman. *UCI Machine Learning Repository*, 2007. University of California, Irvine, School of Information and Computer Sciences. `http://archive.ics.uci.edu/ml/`.

[3] A. Chaves, M. Vellasco, and R. Tanscheit. Fuzzy rule extraction from support vector machines. In N. Nedjah, L. Mourelle, A. Abraham, and M. Köppen, editors, *HIS*, pages 335–340. IEEE Computer Society, 2005.

[4] Y. Chen and J. Z. Wang. Support vector learning for fuzzy rule-based classification systems. *IEEE Transactions on Fuzzy Systems*, 11(6):716–728, Dec. 2003.

[5] J.-H. Chiang and P.-Y. Hao. Support vector learning mechanism for fuzzy rule-based modeling: a new approach. *IEEE Transaction on Fuzzy Systems*, 12(1):1–12, 2004.

[6] P.-Y. Hao and J.-H. Chiang. A fuzzy model of support vector regression machine. *International Journal of Fuzzy Systems*, 9(1):45–50, Mar. 2007.

[7] H.-P. Huang and Y.-H. Liu. Fuzzy support vector machines for pattern recognition and data mining. *International Journal of Fuzzy Systems*, 4(3):826–835, Sept. 2002.

[8] E. Hüllermeier. Fuzzy methods for data mining and machine learning: State of the art and prospects. In H. Bustince, F. Herrera, and J. Montero, editors, *Fuzzy Sets and Their Extensions: Representation, Aggregation*, volume 220 of *Studies in Fuzziness and Soft Computing*, pages 357–475. Springer, 2008.

[9] J. Jang and C.-T. Sun. Functional equivalence between radial basis function networks and fuzzy inference systems. *IEEE Transactions on Neural Networks*, 4(1):156–159, 1993.

[10] T. Kenesei, J. A. Roubos, and J. Abonyi. A combination-of-tools method for learning interpretable fuzzy rule-based classifiers from support vector machines. In H. Yin, P. Tiňo, E. Corchado, W. Byrne, and X. Yao, editors, *IDEAL*, volume 4881 of *Lecture Notes in Computer Science*, pages 477–486. Springer, 2007.

[11] L. I. Kuncheva. How good are fuzzy if-then classifiers? *IEEE Transactions on Systems, Man, and Cybernetics, Part B*, 30(4):501–509, 2000.

[12] C.-F. Lin and S.-D. Wang. Fuzzy support vector machines. *IEEE Transactions on Neural Networks*, 13(2):464–471, Mar. 2002.

[13] O. L. Mangasarian. Generalized support vector machines. In A. J. Smola, P. L. Bartlett, B. Schölkopf, and D. Schuurmans, editors, *Advances in Large Margin Classifiers*, pages 135–146. MIT Press, Cambridge, MA, USA, 2000.

[14] K. Michels, F. Klawonn, A. Nürnberger, and R. Kruse. *Fuzzy Control: Fundamentals, Stability and Design of Fuzzy Controllers*. Springer, Berlin/Heidelberg, Germany, 2006.

[15] S. Papadimitriou and C. Terzidis. Efficient and interpretable fuzzy classifiers from data with support vector learning. *Intell. Data Anal.*, 9(6):527–550, 2005.

[16] B. Schölkopf and A. J. Smola. *Learning with Kernels*. MIT Press, Cambridge, MA, 2002.

[17] B. Schölkopf, K.-K. Sung, C. Burges, F. Girosi, P. Niyogi, T. Poggio, and V. Vapnik. Comparing support vector machines with Gaussian kernels to radial basis function classifiers. *IEEE Transactions on Signal Processing*, 45(11):2758–2765, 1997.

[18] Z. Sun and Y. Sun. Fuzzy support vector machine for regression estimation. *IEEE International Conference on Systems, Man and Cybernetics*, 4:3336–3341, Oct. 2003.

[19] T. Takagi and M. Sugeno. Fuzzy identification of systems and its applications to modeling and control. *IEEE Transactions on Systems, Man, and Cybernetics*, 15(1):116–132, Feb. 1985.

[20] V. Vapnik. *Statistical Learning Theory*. Wiley, New York, 1998.

[21] L. A. Zadeh. Fuzzy sets. *Information and Control*, 8(3):338–353, 1965.