On the use of Measures of Separability of Classes to characterise the Domains of Competence of a Fuzzy Rule Based Classification System

Julián Luengo, Francisco Herrera[1]

1. Dept. of Computer Science and Artificial Intelligence,
CITIC-UGR (Research Center on Information and Communications Technology)
University of Granada, Granada, 18071, Spain
Email: {julianlm,herrera}@decsai.ugr.es

***Abstract*—**

In this work we study the behaviour of a Fuzzy Rule Based Classification System, and its relationship to a certain data complexity measures family. As Fuzzy Rule Based Classification System we have selected a recent proposal called Positive Definite Fuzzy Classifier, which is a Fuzzy System that uses Support Vector Machines for its training, obtaining accurate results and a low number of rules.

We have examined several data complexity metrics of separability of classes *over a wide range of data sets built from real data, and try to extract behaviour patterns from the results for this learning method. Using these data complexity measures and the accuracy results of the Positive Definite Fuzzy Classifier, we have built a rule set which describes both good or bad behaviours of this Fuzzy Rule Based Classification System.*

These rules use different values of such data complexity measures as antecedents, so we aim to predict the behaviour of the method from the data set complexity metrics prior to its application. Therefore, the rule set could characterise the domains of competence of this particular Fuzzy Rule Based Classification System.

***Keywords*—** Classification, Data complexity, Fuzzy Rule Based Systems, Support Vector Machines

1 Introduction

Fuzzy Rule Based Classification Systems (FRBCSs) [12, 14] are a very useful tool in the ambit of Data Mining, since they are capable of building a linguistic model clearly interpretable by human beings. There is a vast literature in the field of FRBCSs [14], which is very active at this time [9, 1, 16, 13].

The prediction capabilities of classifiers are strongly dependent on the problem's characteristics. An emergent field, that uses a set of complexity measures applied to the problem to describe its difficulty, has recently arisen. These measures quantify particular aspects of the problem which are considered complicated to the classification task [11]. Studies of data complexity metrics applied to particular classification's algorithms can be found in [11, 4, 3, 19].

In this work we are interested in analysing the relationship between FRBCSs and the complexity measures, considering a case of study using the Positive Definite Fuzzy Classifier (PDFC) proposed by Chen and Wang [5]. In particular we consider one type of data complexity measures based on the separability of classes.

To perform this study, we have created several binary classification data sets from real world problems, 438 ones, and computed the value of 3 metrics proposed by Ho and Basu [10]. We have analysed the intervals of the complexity measures values related to the created data sets, in which PDFC method performs well or badly, and then formulated a rule for such intervals. The rules try to describe the ranges where some information and conclusions about the behaviour of PDFC method can be stated.

This contribution is organised as follows. In Section 2 we describe the FRBCS we have used. In Section 3 the considered complexity measures are described. In Section 4 we include the experimental set-up, the results obtained and the rules extracted, along with their analysis. Finally, in Section 5 some concluding remarks are made.

2 Preliminaries: Fuzzy Rule Based Classification System

Any classification problem consists of l training patterns $x_p = (x_{p1}, \ldots, x_{pn})$, $p = 1, 2, \ldots, l$ from M classes where x_{pi} is the ith attribute value ($i = 1, 2, \ldots, n$) of the p-th training pattern.

Let us consider a two-class classification problem of assigning class label $y \in \{+1, -1\}$ to input feature vector x_p. In this work we consider the additive FRBCSs with constant `THEN`-parts.

As learning method we use the PDFC method [5], which uses a Support Vector Machine (SVM) approach to build up the model. In the following two Subsections, first we include the fuzzy inference model and a complete description of the PDFC algorithm.

2.1 Fuzzy Inference Model: PDFC Method

Consider a fuzzy model with m fuzzy rules of the form:

$$\text{Rule } j: \text{ If } A_j^1 \text{ AND } A_j^2 \text{ AND } \ldots \text{ AND } A_j^n \text{ THEN } b_j \qquad (1)$$

where A_j^k is a fuzzy set with membership function $a_j^k : \mathbb{R} \to [0, 1]$, $j = 1, \ldots, m$, $k = 1, \ldots, n$, $b_j \in \mathbb{R}$. If we choose product as the fuzzy conjunction operator, addition for fuzzy rule aggregation, and center of area defuzzification, then the model becomes a special form of the Takagi-Sugeno fuzzy model, and the input output mapping, $\mathcal{F} : \mathbb{R}^n \to \mathbb{R}$, of the model is defined as

$$\mathcal{F}(x_p) = \frac{\sum_{j=1}^{m} b_j \prod_{k=1}^{n} a_j^k(x_k)}{\sum_{j=1}^{m} \prod_{k=1}^{n} a_j^k(x_k)}. \qquad (2)$$

Equation (2) could not be well-defined on $\mathbb{R}$ if $\sum_{j=1}^{m} \prod_{k=1}^{n} a_j^k(x_k) = 0$ for some $x_p \in \mathbb{R}^n$, which could happen if the input space is not wholly covered by fuzzy rule "patches". To fix this problem, we add a fuzzy rule so that the

ISBN: 978-989-95079-6-8

denominator $\sum_{j=1}^{m}\prod_{k=1}^{n}a_j^k(x_k) > 0$ for all $x_p \in \mathbb{R}$. Thus the following rule is added:

$$\text{Rule } 0: \text{ If } A_j^1 \text{ AND } A_j^2 \text{ AND } \ldots \text{ AND } A_j^n \text{ THEN } b_0 \quad (3)$$

where $b_0 \in \mathbb{R}$, the membership functions $a_0^k(x_k) \equiv 1$ for $k = 1, \ldots, n$ and any $x_p \in \mathbb{R}^n$. Consequently, the input output mapping becomes

$$\mathcal{F}(x_p) = \frac{b_0 + \sum_{j=1}^{m} b_j \prod_{k=1}^{n} a_j^k(x_k)}{1 + \sum_{j=1}^{m}\prod_{k=1}^{n} a_j^k(x_k)}. \quad (4)$$

A classifier associates class labels with input features, i.e., it is essentially a mapping from the input space to the set of class labels. In binary case, thresholding is one of the simplest ways to transform $\mathcal{F}(x_p)$ to class labels $+1$ or -1.

Considering the FRBCS with $m+1$ fuzzy rules where Rule 0 is given by (3), then the system induces a binary fuzzy classifier, f , with decision rule

$$f(x_p) = \text{sign}(\mathcal{F}(x_p) + t) \quad (5)$$

where $t \in \mathbb{R}$ is a threshold. We can assume $t = 0$ without loss of generality.

The membership functions for a binary fuzzy classifier defined above could be any function from $\mathbb{R}$ to $[0, 1]$. We narrow our interests to a class of membership functions, $a_j^k : \mathbb{R} \to [0, 1]$, $j = 1, \ldots, m$, which are generated from a reference function a^k through location transformation [8], and the classifiers defined on them. In [5] well-known types of reference functions can be found, like the symmetric triangle and the gaussian function.

As consequence of the presented formulation, the decision rule of our binary fuzzy classifier can be written as:

$$f(x_p) = \text{sign}\left(\sum_{j=1}^{m} b_j K(x_p, z_j) + b_0\right) \quad (6)$$

where $z_j = [z_j^1, z_j^2, \ldots, z_j^n]^T \in \mathbb{R}$ contains the location parameters of a_j^k. $K : \mathbb{R}^n \times \mathbb{R}^n \to [0, 1]$ is a translation invariant kernel defined as

$$K(x_p, z_j) = \prod_{k=1}^{n} a^k(x_p^k - z_j^k) \quad (7)$$

which is actually a Mercer Kernel [6], if it has nonnegative Fourier transform. Again, from [5] some Mercer kernels can be built using the reference functions mentioned above.

Thus, the decision rule of a binary fuzzy classifier is

$$f(x_p) = \text{sign}\left(b_0 + \sum_{j=1}^{m} b_j \prod_{k=1}^{n} a_j^k(x_p^k)\right) \quad (8)$$

2.2 Learning method: SVM approach to build PDFC

Here, we assume that the reference functions are predetermined. So the remaining question is how to find a set of fuzzy rules ($\{z_1, \ldots, z_m\}$ and $\{b_0, \ldots, b_m\}$) from the given training so that the PDFC method has good generalization.

As given in (7), for a PDFC, a Mercer kernel can be constructed from the positive definite reference functions. The kernel implicitly defines a nonlinear mapping Φ that maps $\mathbb{X}$ into a kernel-induced feature space $\mathbb{F}$. Theorem 3.12 in [5] states that the decision rule of a PDFC can be viewed as a hyperplane in $\mathbb{F}$. It is well-known that the SVM algorithm finds a separating hyperplane with good generalization by reducing the empirical risk and, at the same time, controlling the hyperplane margin [21]. Thus we can use the SVM algorithm to find an optimal hyperplane in $\mathbb{F}$. Once we get such a hyperplane, fuzzy rules can be easily extracted. The whole procedure is described by the following algorithm 1.

Algorithm 1 SVM learning for PDFC

INPUTS: Positive definite reference functions $a^k(x_p)$ associated with n input variables, and a set of training samples
OUTPUTS: A set of fuzzy rules parameterized by z_j, b_j, and m. z_j contains the location parameters of the `IF`-part membership functions of the jth fuzzy rule, b_j is the `THEN`-part constant of the jth fuzzy rule, and $m+1$ is the number of fuzzy rules.
Steps:
1 Construct a Mercer kernel, K, from the given positivedefinite reference functions according to (7).
2 Construct an SVM to get a decision rule of the form

$$f(x) = sign\left(\sum_{i \in S} y_i \alpha_i K(x, x_i) + b\right),$$

being S the index set of the support vectors:
2.1 Assign some positive number to the cost C, and solve the quadratic program defined by the proper SVM to get the Lagrange multipliers α_i.
2.2 Find b (details can be found in, for example, [18]).
2.3 Extracting fuzzy rules from the decision rule of the SVM:
$b0 \leftarrow b$
$j \leftarrow 1$
for $i = 1$ **to** l **do**
  if $\alpha_i > 0$ **then**
    $z_j \leftarrow x_i$
    $b_j \leftarrow y_i \alpha_i$
    $j \leftarrow j + 1$
  end if
end for
$m \leftarrow j - 1$

In our study, we have considered the parameters values recommended by the authors. They are summarized as follows:

- $C = 100$ (weight of the classification error)
- $d = 0.25$ (parameter used by the reference functions)
- Type of reference functions: Gaussian

3 Data Complexity Measures Based on the Separability of Classes

In this section we describe the three metrics we have used in this contribution, with their correspondent acronym.

For our study, we will examine three measures of separability of classes from [10] which offer information for the PDFC method. They are described next.

- **N1**: fraction of points on class boundary. This method constructs a class-blind minimum spanning tree over the entire data set, and counts the number of points incident to an edge going across the two classes. The fraction of

ISBN: 978-989-95079-6-8

such points over all points in the data set is used as a measure. For two heavily interleaved classes, a majority of points are located next to the class boundary. However, the same can be true for a sparsely sampled linearly separable problem with margins narrower than the distances between points of the same class.

- **N2**: ratio of average intra/inter class Nearest Neighbour (NN) distance. For each input instance x_p, we calculate the distance to its nearest neighbour within the class (intraDist(x_p)) and the distance to nearest neighbour of any other class (interDist(x_p)). Then, the result is the ratio of the sum of the intra-class distances to the sum of the inter-class distances for each input example, i.e.,

$$N2 = \frac{\sum_{i=0}^{m} intraDist(x_i)}{\sum_{i=0}^{m} interDist(x_i)},$$

where m is the number of examples in the data set. This metric compares the within-class spread with the distances to the nearest neighbours of other classes. Low values of this metric suggest that the examples of the same class lay closely in the feature space. Large values indicate that the examples of the same class are disperse. It is sensitive to the classes of the closest neighbours to a point, and also to the difference in magnitude of the between-class distances and that of the within-class distances.

- **N3**: error rate of 1-NN classifier. This is simply the error rate of a nearest-neighbour classifier measured with the training set. The error rate is estimated by the leave-one-out method. The measure denotes how close the examples of different classes are. Low values of this metric indicate that there is a large gap in the class boundary.

4 Experimental Study: Analysis of the PDFC with Data Complexity Measures

In this Section we analyse the obtained results for the PDFC method. First, in Subsection 4.1 we present the experimental framework, with the data sets generation method, accuracy validation scheme, and the global average results of the PDFC method. Next we determine several rules based on PDFC's behaviour in Subsection 4.2. Finally we analyse the collective evaluation of the set of rules in Subsection 4.3.

4.1 Experimental Framework: Data Sets Generation

We evaluate the PDFC method on a set of 438 binary classification problems. These problems are generated from pairwise combinations of the classes of 20 problems from the University of California, Irvine (UCI) repository [2]. These are *iris, wine, new-thyroid, solar-flare, led7digit, zoo, yeast, tae, balanced, car, contraceptive, ecoli, hayes-roth, shuttle, australian, pima, monks, bupa, glass, haberman* and *vehicle*.

In order to do that, first we take each data set and extract the examples belonging to each class. Then we construct a new data set with the combination of the examples from two different classes. This will result in a new data set with only 2 classes and the examples which have two such classes as output. We perform this process for every possible pairwise combination of classes. However, if an obtained data set with this procedure proves to be linearly-separable, we discard it (since we could classify it with a linear classifier with no error). The complexity measure L1 from [10] indicates if a problem is linearly-separable if its value is zero, so every data set with a L1 value of zero will be discarded.

This method for generating binary data sets is limited by the proper combinatorics, and we can only obtain over 200 new data sets with the original 20 data sets with this first approach. In order to obtain more data sets, we group the classes two by two, that is, we create a new binary data set, and each of its two classes are the combination of two original classes each. For this second approach we have used *ecoli*, *glass* and *flare* data sets, since they have a high number of class labels. Again, those data sets with a L1 value of zero are discarded.

In order to measure the PDFC performance, we have applied a 10-fcv validation scheme. In Table 1 we show the global Training and Test accuracy obtained by the PDFC method.

Table 1: Global Average PDFC Training and Test Accuracy

PDFC Global % Accuracy Training	94.06%
PDFC Global % Accuracy Test	91.22%

4.2 Determination of Rules Based on the PDFC Behaviour

In the following we present the results of the execution over the 438 data sets summarized in Figures 1 to 3.

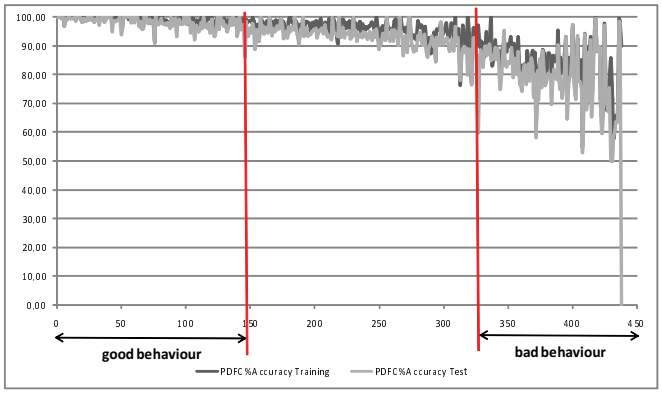

Figure 1: PDFC accuracy in Training/Test sorted by N1

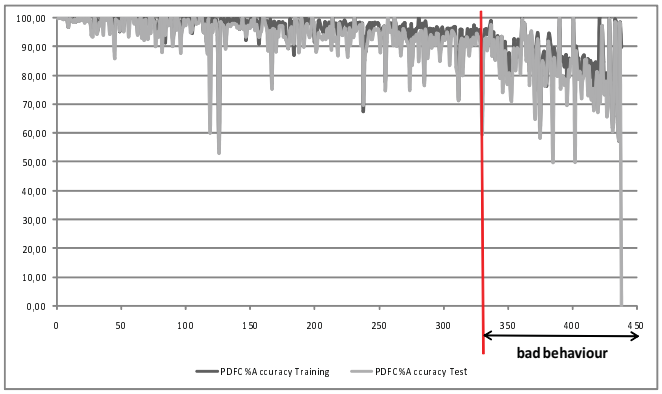

Figure 2: PDFC accuracy in Training/Test sorted by N2

For each complexity measure (N1, N2 and N3), the data sets are sorted by the ascending value of the corresponding complexity measure, and put altogether in a Figure. In the X

ISBN: 978-989-95079-6-8

Table 3: Rules with one metric obtained from the intervals

Id.	Rule	Support	%Training	Training Diff.	% Test	Test Diff.
R1+	If N1[X] $<$ 0.089 then *good behaviour*	33.11%	99.09%	4.91%	98.15%	6.55%
R2+	If 0 $<$ N3[X] $<$ 0.047 then *good behaviour*	28.31%	98.92%	4.86%	97.70%	6.48%
R1-	If N1[X] $\geq$ 0.25 then *bad behaviour*	25.57%	84.64%	-9.42%	79.01%	-12.21%
R2-	If N2[X] $>$ 0.5196 then *bad behaviour*	24.89%	86.14%	-7.92%	80.88%	-10.34%
R3-	If N3[X] $>$ 0.175 then *bad behaviour*	19.41%	82.64%	-11.42%	76.77%	-14.45%

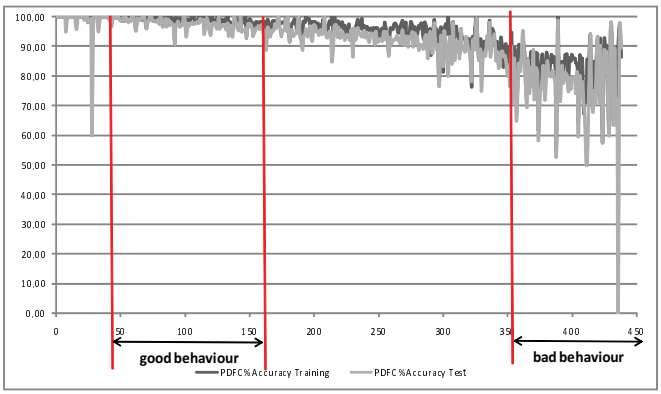

Figure 3: PDFC accuracy in Training/Test sorted by N3

axis we represent the data sets, not the complexity measure value, and the Y axis depicts the accuracy obtained both in training and test. The reason to do so is to give each data set the same space in the graphic representation. For those measures where we can find different *ad-hoc* intervals which present *good* or *bad behaviour* of the PDFC, we use a vertical line to delimit the interval of the region of interest.

- We understand for *good behaviour* an average high test accuracy in the interval, as well as the absence of over-fitting.
- By *bad behaviour* we refer to the presence of over-fitting and/or average low test accuracy in the interval.

In Table 2 we have summarized the intervals found ad-hoc from Figures 1 to 3.

Table 2: Significant intervals

Interval	PDFC Behaviour
$N1 < 0.089$	*good behaviour*
$0 < N3 < 0.047$	*good behaviour*
$N1 \geq 0.25$	*bad behaviour*
$N2 > 0.5196$	*bad behaviour*
$N3 > 0.175$	*bad behaviour*

From these ad-hoc intervals we construct several rules that model the performance of the FRBCS we have used. In Table 3 we have summarized the rules derived from Table 2. Given a particular data set X, we get the complexity measure of X with the notation $CM[X]$. Table 3 is organised with the following columns.

- The first column corresponds to the identifier of the rule for further references.
- The "Rule'" column presents the rule itself.
- The third column "Support" presents the percentage of data sets which verifies the antecedent of the rule.
- The column "% Training" shows the average accuracy in training of all the examples which are covered by the rule.
- The column "Training Diff." contains the difference between the training accuracy of the rule and the training accuracy across all 438 data sets.
- The column "% Test" shows the average accuracy in test of all the examples which are covered by the rule.
- The column "Test Diff." contains the difference between the test accuracy of the rule and the test accuracy across all 438 data sets.

As we can see in Table 3, the positive rules (denoted with a "+" symbol in their identifier) always show a positive difference with the global average, both in training and test accuracy. The negative ones (with a "-" symbol in their identifier) verify the opposite case. The support of the rules shows us that we can characterize a wide range of data sets and obtain significant differences in accuracy.

From this set of rules we can state that a low N1 value results in a good behaviour of the PDFC method. A low N3 value obtains the same results. In the other hand, a high value in the N1 metric produces a bad behaviour of the PDFC considered in our analysis. A high N3 value will also produce a bad behaviour of the PDFC method. With similar outcome, if N2 presents a high value, the PDFC method will obtain bad behaviour.

Although we have obtained some interesting rules, we can extend our study by considering the combination of these complexity metrics in order to obtain more precise and descriptive rules.

4.3 Collective Evaluation of the Set of Rules

The objective of this section is to analyse the good rules jointly, and the bad rules together as well. Thus we can arrive at a more general description, with wider support, of the behaviour of the PDFC with these joint rules. We perform the disjunctive combination of all the positive rules to obtain a single rule, and all the negative ones, so we obtain another rule.

ISBN: 978-989-95079-6-8

Table 4: Disjunction Rules from all simple rules

Id.	Rule	Support	%Training	Training Diff.	% Test	Test Diff.
PRD	If R1+ or R2+ then *good behaviour*	37.44%	98.99%	4.93%	97.90%	6.68%
NRD	If R1- or R2- or R3- then *bad behaviour*	31.51%	86.23%	-7.83%	81.24%	-9.98%
not characterised	If not PRD and not NRD then *good behaviour*	31.05%	96.06%	2.00%	93.27%	2.05%

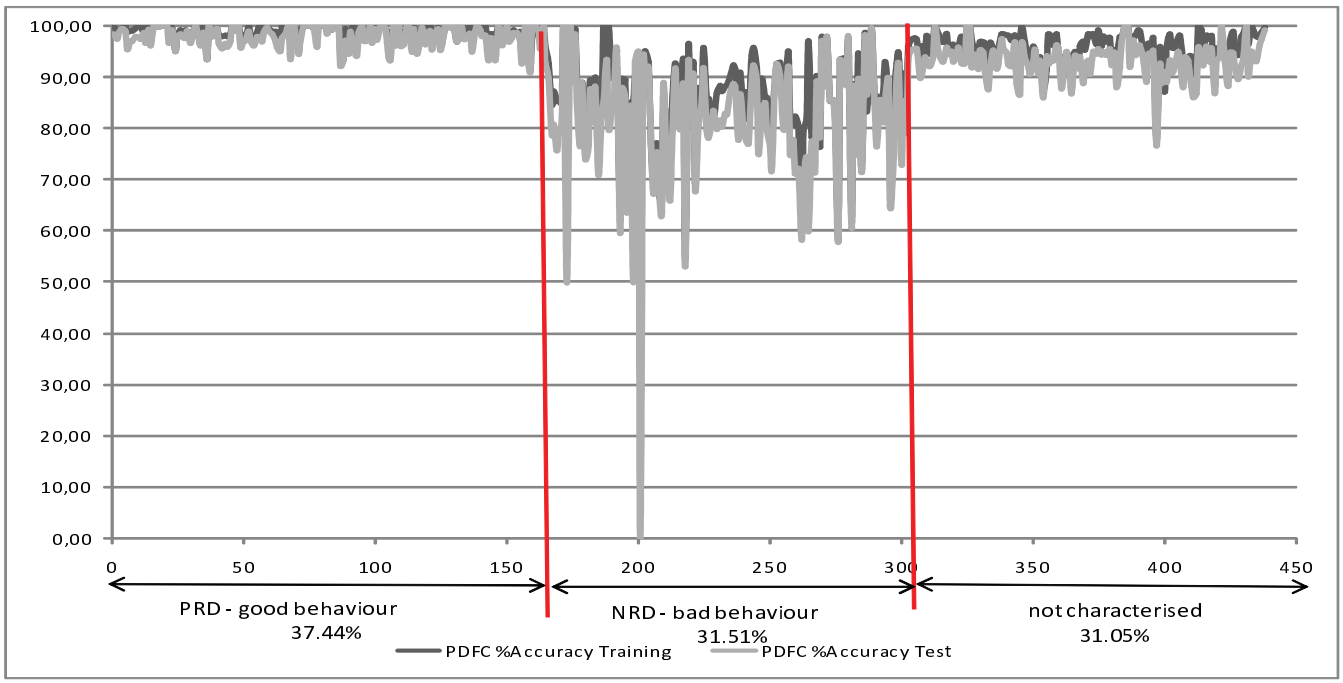

Figure 4: Three blocks representation for PRD, NRD and not covered data sets for PDFC

The new disjunctive rule will be activated if any of the component rules' antecedents are verified. In Table 4 we summarize both disjunctions, and a third rule representing those data sets which are not charaterised by either disjunction rules.

From the collective rules we can observe that the support has been increased from the single rules both for the Positive Rule Disjunction (PRD) and Negative Rule Disjunction (NRD). In the other hand, the Test and Training Accuracy Differences are similar to the single rules from Table 3. Since there are no data sets in PRD and NRD simultaneously, we can consider three blocks of data sets with their respective support, as depicted in Figure 4 (with no particular data set order within each block):

- The first block (the left-side one) represents the data sets covered by the PRD rule. They are the data sets recognized as being those in which the PDFC has good accuracy.
- The second block (the middle one) plots the data sets for the rule NRD, which are bad data sets for the PDFC method considered.
- The third and last block (the right-side one) contains the unclassified data sets by the previous two rules.

We can see that almost the 70% of the analysed data sets are covered by these two rules, and hence the *good behaviour* and *bad behaviour* consequents represent well the accuracy of PDFC methods.

5 Concluding Remarks

We have performed a study over a set of binary data sets with the PDFC method. We have computed some data complexity measures for the data sets in order to obtain intervals of such metrics in which the method's performance is significantly good or bad. We have constructed descriptive rules, and studied the interaction between the intervals and the proper rules.

We have obtained two rules which are simple and precise to describe both good and bad performance of the PDFC. Furthermore, we present the possibility of determining which data sets PDFC would performs well or badly prior to their execution, using the Data Complexity measures.

We must point out that this is a particular study for one specific method, the PDFC. On the other hand, this work presents a new challenge that could be extended to other FRBCSs, to analyse their domains of competence, and to develop new measures which could give more information on the behaviours of FRBCSs for pattern recognition.

Acknowledgment

This work has been supported by the Spanish Ministry of Science and Technology under Project TIN2008-06681-C06-01.

ISBN: 978-989-95079-6-8

J. Luengo holds a FPU scholarship from Spanish Ministry of Innovation and Science.

References

[1] P. Angelov, E. Lughofer and X. Zhou. Evolving fuzzy classifiers using different model architectures. *Fuzzy Sets and Systems*, 159(23):3160–3182, 2008.

[2] A. Asuncion and D.J. Newman. UCI Machine Learning Repository [http://www.ics.uci.edu/ mlearn/MLRepository.html]. Irvine, CA: University of California, School of Information and Computer Science, 2007.

[3] R. Baumgartner, R.L. Somorjai. Data complexity assessment in undersampled classification. *Pattern Recognition Letters*, 27:1383–1389, 2006.

[4] E. Bernadó-Mansilla, T.K. Ho. Domain of Competence of XCS Classifier System in Complexity Measurement Space. *IEEE Transactions on Evolutionary Computation*, 9(1):82–104, 2005.

[5] Y. Chen and J.Z. Wang. Support Vector Learning for Fuzzy Rule-Based Classification Systems. *IEEE Transactions on Fuzzy Systems*, 11(6):716–728,2003.

[6] N. Cristianini and J. Shawe-Taylor. *An introduction to Support Vector Machines and Other Kernel-Based Learning Methods*. Cambridge, U.K.: Cambridge Univ. Press, 2000.

[7] M. Dong, R. Kothari. Feature subset selection using a new definition of classificabilty. *Pattern Recognition Letters*, 24:1215–1225, 2004.

[8] D. Dubois and H. Prade. Operations on fuzzy numbers. *Int. J. Syst. Sci.*, 9(6):613–626, 1978.

[9] A. Fernández, S. García, M.J. del Jesus and F. Herrera. A study of the behaviour of linguistic fuzzy rule based classification systems in the framework of imbalanced data-sets. *Fuzzy Sets and Systems*, 159(18):2378–2398, 2008.

[10] T.K. Ho, M. Basu. Complexity Measures of Supervised Classification Problems. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(3):289–300, 2002.

[11] M. Basu, T.K. Ho. *Data Complexity in Pattern Recognition*. Springer, 2006.

[12] H. Ishibuchi, T. Nakashima, M. Nii. *Classification and modeling with linguistic information granules: advanced approaches to linguistic data mining*. Springer, Berlin, 2004.

[13] M.Z. Jahromi and M. Taheri. A proposed method for learning rule weights in fuzzy rule-based classification systems. *Fuzzy Sets and Systems*, 159(4):449-459, 2008

[14] L. Kuncheva L. *Fuzzy classifier design*. Springer, Berlin, 2000.

[15] Y. Li, M. Dong, R. Kothari. Classifiability-Based Omnivariate Decision Trees. *IEEE Transactions on Neural Networks*, 16(6):1547–1560, 2005.

[16] E.G. Mansoori, M.J. Zolghadri and S.D. Katebi. SGERD: A steady-state genetic algorithm for extracting fuzzy classification rules from data. *IEEE Transactions on Fuzzy Systems*, 16(4):1061–1071, 2008.

[17] R.A. Mollineda, J.S. Sánchez, J.M. Sotoca. Data Characterization for Effective Prototype Selection. *First edition of the Iberian Conference on Pattern Recognition and Image Analysis (IbPRIA 2005). Lecture Notes in Computer Science 3523, Springer-Verlag*, 27–34, 2005.

[18] J. Platt. Machines using Sequential Minimal Optimization. In *B. Schoelkopf, C. Burges and A. Smola (Eds.), Advances in Kernel Methods - Support Vector Learning*, 1998.

[19] J.S. Sánchez, R.A. Mollineda, J.M. Sotoca. An analysis of how training data complexity affects the nearest neighbor classifiers. *Pattern Analysis and Applications*, 10(3):189–201, 2007.

[20] S. Singh. Multiresolution Estimates of Classification Complexity. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(12):1534–1539, 2003.

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

ISBN: 978-989-95079-6-8