sciendo

JAISCR, 2021, Vol. 11, No. 4, pp. 307 – 318
10.2478/jaiscr-2021-0018

PERFORMANCE ANALYSIS OF ROUGH SET–BASED HYBRID CLASSIFICATION SYSTEMS IN THE CASE OF MISSING VALUES

Robert K. Nowicki[1,*], Robert Seliga[2,3], Dariusz Żelasko[4], Yoichi Hayashi[5]

[1]*Department of Intelligent Computer Systems, Czestochowa University of Technology, Czestochowa, Poland*

[2]*Management Department, University of Social Science, 90–113 Lodz, Poland*

[3]*Clark University, Worcester, MA 01610, USA*

[4]*Faculty of Computer Science and Telecommunications, Cracow University of Technology Warszawska 24, 31-155 Krakow, Poland*

[5]*Department of Computer Science, Meiji University, Kawasaki 214-8571 Japan*

[*]*E-mail: robert.nowicki@pcz.pl*

Submitted: 7th January 2021; Accepted: 23rd July 2021

Abstract

The paper presents a performance analysis of a selected few rough set–based classification systems. They are hybrid solutions designed to process information with missing values. Rough set-–based classification systems combine various classification methods, such as support vector machines, k–nearest neighbour, fuzzy systems, and neural networks with the rough set theory. When all input values take the form of real numbers, and they are available, the structure of the classifier returns to a non–rough set version. The performance of the four systems has been analysed based on the classification results obtained for benchmark databases downloaded from the machine learning repository of the University of California at Irvine.

Keywords: rough sets, support vector machines, fuzzy systems, neural networks

1 Introduction

The Pawlak rough set theory [24, 25] is a tool allowing to approximate sets in the space in which particular objects have a limited description. Thus, groups of various objects are indistinguishable. They form equivalence classes. The theory operates only on the whole equivalence classes and defines the lower and upper sets approximations. In practice, any description of any object is limited to certain sets of features. The theory allows us to evaluate sets of features using certain indicators, e.g. the quality of approximation and, consequently, to choose the most effective set of features in the description for a given purpose. Formally, in space U with elements described by features $c_i \in Q$, the equivalence class is defined as follows

$$[\hat{x}]_{\tilde{P}} = \left\{ x \in U : \hat{x} \tilde{P} x \right\}. \quad (1)$$

where $\tilde{P}$ is a relation dependent on set $P \subseteq Q$ — a set of selected or available features. Two elements in U are in relation $\tilde{P}$ when they have the same values of all features belonging to P. The set of the omitted features or the features with unavailable (missing) values is defined as G, thus $P \cup G = Q$. In consequence, the lower and upper approximations of any set X have the following forms

$$\underline{\tilde{P}}X = \{x \in U : [x]_{\tilde{P}} \subseteq X\}, \quad (2)$$

$$\overline{\tilde{P}}X = \{x \in U : [x]_{\tilde{P}} \cap X \neq \emptyset\}. \quad (3)$$

In such approximation space, various classification systems can be defined. They make the decision, not for a single object but the whole equivalence class. Thus, they do not give a single answer but a scope of answers for all possible objects in a given equivalence class. It is a direct consequence of the assumption that the objects in an equivalence class are indistinguishable, although they vary.

In this paper, we investigate the performance of the rough sets–based hybrid classification systems developed recently in [21]. It is worth noting that the concept of combining the rough sets theory with various machine learning algorithms has been explored in the literature not only in [21] , see e.g. [1, 7, 8, 11, 14, 16, 17, 26, 29]. However, prior the work [21] the proposed solutions most often boiled down to the use of the theory of approximate sets at particular stages of design (e.g. learning, designing rules, initial processing of pattern data), of the decision system or used in independent elements of data processing (e.g. preprocessing, post processing).

Rough set–based classification systems analysed in [21], and in this article, are characterized by the use of elements of the rough set theory concept during the classification process. The developed Rough set–based classification Systems use directly the abstraction class (e.g. input data), approximate set (e.g. the neighbour in the rough k–nearest neighbour classifier and the result of classification in each of the systems), or approximate set in the sense of Dubois and Prade [3, 4]. The works published so far [21, 22] presented the concept of rough set–based classification Systems, together with specific examples of such rough support vector machines, a rough k–nearest neighbour classifier, a rough neural network, and a rough fuzzy system. Individual The publications described their operation as independent classifiers [18, 19, 22], elements of an iterative system [21] or elements of assemblies [9, 10, 21, 23]. The research results presented classification tasks in which the number of conditional attributes usually did not exceed 10. An analysis of the time consumption of individual systems has never been presented. This paper fills in these gaps. It presents new and unpublished results of the performance of all the above–mentioned rough set–based classification systems for two classification tasks in the field of banking, in which the number of conditional attributes is 20 and 23. The new results are presented in the form of tables and then compared with the previous results in joint charts. The study also compared the results of measurements of the decision–making time of individual systems.

The summarized results show that a specific number of conditional attributes does not noticeably affect the ability to classify in the absence of values of these attributes. Other, as yet unexplored factors play a role here. Time measurements indicate, however, that a slight advantage of the effectiveness of the rough k–nearest neighbour classifier is due to a very high processing time by this algorithm. The time is 100 times longer than in the case of other systems, which essentially eliminates the rough k–nearest neighbour classifier from many applications. From this perspective, rough support vector machines and a rough neural network, deserve attention. Especially the latter, because simple two– and three–layer feed–forward networks were used in the experiments.

The main contributions of this paper can be summarized as follows:

- The results of the operation of four systems and for databases larger than before have been presented. These systems are
  - a rough support vector machine,
  - a rough k–nearest neighbour classifier,
  - a rough neural network,
  - a rough fuzzy system.
- The classification time for above four rough set–based classification systems was compared.

– Through intensive simulations, which results are given in Tables 1–4, the classification performance (as a function of the number of missing value) has been analysed for a different number of conditional attributes taken into account.

The paper is organized as follows. Section 2 contains a brief description of the basic version of selected rough set–based classification systems, i.e. rough support vector machines, a rough k–nearest neighbour classifier, a rough neural network, and a rough fuzzy system. More details have been presented in the works already published [18, 19, 20, 21, 22].

The classification systems described in Section 3 have been tested using widely available selected benchmarks. The details of the investigation procedure are presented in Section 3. The results are shown and discussed in Section 4. The new results have been compiled with the results from the previous investigations. The paper ends with Conclusions, which also contain a few remarks and plans for further research.

2 Preliminaries

The rough set–based classification systems considered in the paper have been described in detail in [21] and [22]. All systems have been defined using a common form of input and output data as well as notation. Thus, they are compatible and can substitute each other if needed. The individual properties of particular systems can be combined in homogeneous or hybrid ensembles and the iterative model.

The input data represent state or object x under classification which is described by vector $\mathbf{v} = [v_1, v_2, \ldots, v_n] \in V$. Individual values v_i remain real values or can be substituted by intervals $\overline{v}_i = [\underline{v}_i, \overline{v}_i]$ or even $\overline{v}_i = [v_{\min}, v_{\max}]$ in the case of a missing value. The output is expressed by intervals $\left[\underline{z}_j, \overline{z}_j,\right]$ for each j–th class or hypothesis, where $j = 1, \ldots, N$. Intervals $\left[\underline{z}_j, \overline{z}_j\right]$ are reduced to the final decision. There are many methods of type reduction [5, 21, 28]. Here, referring to the rough set theory, the following is suggested

$$x \in \begin{cases} \mathrm{Pos}(\omega_j) & \text{if } \underline{z}_j \geq \frac{1}{2} \text{ and } \overline{z}_j > \frac{1}{2} \\ \mathrm{Neg}(\omega_j) & \text{if } \underline{z}_j < \frac{1}{2} \text{ and } \overline{z}_j \leq \frac{1}{2} \\ \mathrm{Bn}(\omega_j) & \text{otherwise.} \end{cases} \quad (4)$$

Assigning element x to the positive region of set ω_j is the same as classifying the element to this set, i.e. $x \in \omega_j$. Assigning the object to the negative region ($\mathrm{Neg}(\omega_j)$) means that the object does not belong to the set, i.e. $x \notin \omega_j$. The last case ($x \in \mathrm{Bn}(\omega_j)$) occurs when an unequivocal classification is not possible.

2.1 Rough support vector machines

Construction of the classification algorithm under investigation is an implementation of the original Support Vector Machine proposed by Vladimir Vapnik [30]. The original version processes input data in the form of a vector of values, whereas the rough version of support vector machines processes equivalence classes, instead. As it has been derived in [22], the answer of the rough support vector machines is given by two bounds of interval $[\underline{z}(\tau), \overline{z}(\tau)]$, i.e.

$$\underline{z}(\tau) = sgn\left(\sum_{r:\, x^r_{\mathrm{ref}}} \alpha^r d^r K\left(\mathbf{v}^r_{\mathrm{ref}}, \mathbf{v}_*(\tau)\right) + w_0\right) \quad (5)$$

and

$$\overline{z}(\tau) = sgn\left(\sum_{r:\, x^r_{\mathrm{ref}}} \alpha^r d^r K\left(\mathbf{v}^r_{\mathrm{ref}}, \mathbf{v}^*(\tau)\right) + w_0\right), \quad (6)$$

where K is a non–linear kernel function, values $\mathbf{v}_*(\tau) = [v_{*1}(\tau), \ldots, v_{*n}(\tau)]$ and $\mathbf{v}^*(\tau) = [v^*_1(\tau), \ldots, v^*_n(\tau)]$ meet the following conditions

$$K\left(\mathbf{v}^{rT}_{\mathrm{ref}}, \mathbf{v}_*(\tau)\right) = \inf_{\substack{\hat{v}_i \in \left[\underline{v}_i(\tau), \overline{v}_i(\tau)\right] \\ i:\, c_i \in G}} K\left(\mathbf{v}^{rT}_{\mathrm{ref}}, \hat{\mathbf{v}}(\tau)\right), \quad (7)$$

$$K\left(\mathbf{v}^{rT}_{\mathrm{ref}}, \mathbf{v}^*(\tau)\right) = \sup_{\substack{\hat{v}_i \in \left[\underline{v}_i(\tau), \overline{v}_i(\tau)\right] \\ i:\, c_i \in G}} K\left(\mathbf{v}^{rT}_{\mathrm{ref}}, \hat{\mathbf{v}}(\tau)\right), \quad (8)$$

and $\hat{\mathbf{v}}(\tau) = [\hat{v}_1(\tau), \ldots, \hat{v}_n(\tau)]$, but for features $c_i \in P$ value $\hat{v}_i$ is just equal to $v_i(\tau)$. In general, an analytical determination of values $\mathbf{v}_*(\tau)$ and $\mathbf{v}^*(\tau)$ may not be possible. Then, it is necessary to use alternative methods, e.g. the Monte Carlo sampling. In the case of the linear kernel, we have a rather simple solution

$$v_{*i}(\tau) = \begin{cases} \underline{v}_i(\tau) & \text{if } w_i > 0 \\ \overline{v}_i(\tau) & \text{if } w_i < 0 \end{cases} \quad (9)$$

$$v^*_i(\tau) = \begin{cases} \underline{v}_i(\tau) & \text{if } w_i < 0 \\ \overline{v}_i(\tau) & \text{if } w_i > 0. \end{cases} \quad (10)$$

2.2 Rough k–nearest neighbour classifier

The original k–nearest neighbour algorithm [2] consists of two main steps. The first one involves selecting the k size neighbourhood of the sample under classification out of reference samples $\Upsilon_k(x(\tau))$. The second one consists in determining the most numerous class from this neighbourhood. In [21] the concept of the k–nearest neighbour classifier has been extended by certain elements of the rough set theory in both steps. In the case of missing values or, in general, interval values, object $x(\tau)$which needs to be classified is described by the vector of intervals $\overline{\underline{\mathbf{v}}}(\tau) = [\overline{\underline{v}}_1(\tau), \ldots, \overline{\underline{v}}_n(\tau)]$ as well as reference samples x^r_{ref} which are described by vector $\overline{\underline{\mathbf{v}}}^r_{\text{ref}} = [\overline{\underline{v}}^r_{\text{ref}1}, \ldots, \overline{\underline{v}}^r_{\text{ref}n}]$. Thus, the distance between them becomes also an interval, whose left and right bounds are calculated as follows

$$\underline{\rho}\left(x(\tau), x^r_{\text{ref}}\right) = \inf_{v_i(\tau)\in\overline{\underline{v}}_i(\tau), v^r_{\text{ref}i}\in\overline{\underline{v}}_i(\tau)} \left\|\mathbf{v}(\tau), \mathbf{v}^r_{\text{ref}}\right\|, \quad (11)$$

$$\overline{\rho}\left(x(\tau), x^r_{\text{ref}}\right) = \sup_{v_i(\tau)\in\overline{\underline{v}}_i(\tau), v^r_{\text{ref}i}\in\overline{\underline{v}}_i(\tau)} \left\|\mathbf{v}(\tau), \mathbf{v}^r_{\text{ref}}\right\|. \quad (12)$$

The obtained intervals are approximations of real distances $\rho\left(x(\tau), x^r_{\text{ref}}\right)$. Thus, the order of distances and, consequently, belonging to neighbourhood $\Upsilon_k(x(\tau))$ are not unambiguous. The answer is the approximation of neighbourhood $\Upsilon_k(x(\tau))$ by a rough set consisting of lower $\underline{\Upsilon}_k(x(\tau))$ and upper $\overline{\Upsilon}_k(x(\tau))$ approximation of set $\Upsilon_k(x(\tau))$. The approximations are defined as follows [21]

$$\frac{x^r_{\text{ref}} \in \underline{\Upsilon}_k(x(\tau)) \Leftrightarrow}{\left\{x^{r'}_{\text{ref}} \in X_{\text{ref}} : \underline{\rho}\left(x(\tau), x^{r'}_{\text{ref}}\right) \le \overline{\rho}\left(x(\tau), x^r_{\text{ref}}\right)\right\}} \le k \quad (13)$$

$$\frac{x^r_{\text{ref}} \in \overline{\Upsilon}_k(x(\tau)) \Leftrightarrow}{\left\{x^{r'}_{\text{ref}} \in X_{\text{ref}} : \overline{\rho}\left(x(\tau), x^{r'}_{\text{ref}}\right) \le \underline{\rho}\left(x(\tau), x^r_{\text{ref}}\right)\right\}} \le k \quad (14)$$

where $\overline{\overline{\Omega}}$ is the cardinality of set Ω. Finally, the algorithm checks the most numerous class from all possible neighbours $\Upsilon_k(x(\tau))$ which meet

$$\underline{\Upsilon}_k(x(\tau)) \subseteq \Upsilon_k(x(\tau)) \subseteq \overline{\Upsilon}_k(x(\tau)). \quad (15)$$

The formal description of the final result is presented in detail in [21]. However, the algorithm is realised in the following two steps

1. sorting the left and right bounds of distances $\underline{\rho}\left(x(\tau), x^r_{\text{ref}}\right)$ and $\overline{\rho}\left(x(\tau), x^r_{\text{ref}}\right)$ for all $x^r_{\text{ref}} \in \overline{\Upsilon}_k(x(\tau))$, and denoting them as sequence $\rho_1, \rho_2, \ldots$,
2. determining the most numerous class for each interval $[\rho_k, \rho_{k+1}]$.

The final result is also in the form of rough sets. Thus, when class ω_j is most numerous in at least one interval $[\rho_k, \rho_{k+1}]$, object $x(\tau)$ which is being classified belongs to the upper approximation of the class. When the class is most numerous in all intervals $[\rho_k, \rho_{k+1}]$, object $x(\tau)$ belongs to the lower approximation of class ω_j, i.e. positive region $Pos(\omega_j)$. When the class is not most numerous in any interval $[\rho_k, \rho_{k+1}]$, object $x(\tau)$ belongs to the negative region of class — $Neg(\omega_j)$.

2.3 Rough neural network

The rough feedforward neural network described in [21] is based on a variation of the Lingras rough neuron [12, 13]. In general, the neural network is non–linear and non–monotonic. However, a single neuron with a monotonic activation function can be treated as a monotonic block. Thanks to this, certain assumptions have been met. Then, for the input information given in the form of interval $\left[\underline{v}^{(l)}_i, \overline{v}^{(l)}_i\right]$, where $i = 1, \ldots, n$, the j–th neuron in l–th layer obtains the activation level also in the form of interval $\left[\underline{s}^{(l)}_j, \overline{s}^{(l)}_j\right]$. The left and right bounds are calculated as follows

$$\underline{s}^{(l)}_j = \sum_{\substack{i=0 \\ i:\, w^{(l)}_{ji}>0}}^{N^{(l-1)}} w^{(l)}_{ji}\underline{v}^{(l)}_i + \sum_{\substack{i=0 \\ i:\, w^{(l)}_{ji}<0}}^{N^{(l-1)}} w^{(l)}_{ji}\overline{v}^{(l)}_i, \quad (16)$$

$$\overline{s}^{(l)}_j = \sum_{\substack{i=0 \\ i:\, w^{(l)}_{ji}>0}}^{N^{(l-1)}} w^{(l)}_{ji}\overline{v}^{(l)}_i + \sum_{\substack{i=0 \\ i:\, w^{(l)}_{ji}<0}}^{N^{(l-1)}} w^{(l)}_{ji}\underline{v}^{(l)}_i. \quad (17)$$

For non–decreasing activation function f, the output of the neuron is given by interval $\left[\underline{y}^{(l)}_j, \overline{y}^{(l)}_j\right]$ where

$$\underline{y}^{(l)}_j = f\left(\underline{s}^{(l)}_j\right), \quad (18)$$

$$\overline{y}^{(l)}_j = f\left(\overline{s}^{(l)}_j\right). \quad (19)$$

The output of the last layer is an output of the network, thus $\left[\underline{z}_j, \overline{z}_j\right] = \left[\underline{y}^{(L)}_j, \overline{y}^{(L)}_j\right]$.

2.4 Rough fuzzy system

An in–depth presentation of the rough fuzzy systems in the form analysed in the paper can be found in [21]. This book offers also different versions of such systems. The simplest one applies Mamdani reasoning and DCOG defuzzification. Thus, the knowledge of the system is reduced to the rules in the following form

$$R^r\text{: IF } \mathbf{v} \text{ is } A^r \text{ THEN } \bigwedge_{j=1}^{m} z_j = \overline{z}_j^r \tag{20}$$

The output in the form of interval $\left[\underline{z}_j, \overline{z}_j\right]$ is calculated according to the following two formulas

$$\underline{\overline{z}_j} = \frac{\sum\limits_{r=1}^{N} \underset{r:\,\overline{z}_j^r=1}{\underset{k=1}{\overset{N}{S}}} \mu_{A_{\mathrm{L}}^k}(\mathbf{v})}{\sum\limits_{r=1}^{N} \underset{k:\,\overline{z}_j^k=\overline{z}_j^r}{\underset{k=1}{\overset{N}{S}}} \mu_{A_{\mathrm{L}}^k}(\mathbf{v})} \tag{21}$$

and

$$\overline{\overline{z}_j} = \frac{\sum\limits_{r=1}^{N} \underset{r:\,\overline{z}_j^r=1}{\underset{k=1}{\overset{N}{S}}} \mu_{A_{\mathrm{U}}^k}(\mathbf{v})}{\sum\limits_{r=1}^{N} \underset{k:\,\overline{z}_j^k=\overline{z}_j^r}{\underset{k=1}{\overset{N}{S}}} \mu_{A_{\mathrm{U}}^k}(\mathbf{v})}, \tag{22}$$

where

$$A_{\mathrm{L}}^r = \begin{cases} \underline{\widetilde{P}} A^r & \text{gdy } \overline{z}_j^r = 1 \\ \overline{\widetilde{P}} A^r & \text{gdy } \overline{z}_j^r = 0 \end{cases} \tag{23}$$

and

$$A_{\mathrm{U}}^r = \begin{cases} \overline{\widetilde{P}} A^r & \text{gdy } \overline{z}_j^r = 1 \\ \underline{\widetilde{P}} A^r & \text{gdy } \overline{z}_j^r = 0. \end{cases} \tag{24}$$

3 Investigation procedure

All rough set–based classification systems reminded above have been examined using the two benchmark databases freely available on the machine learning repository of the University of California at Irvine [15]. They are "default of credit card clients Data Set" and "South German Credit (UPDATE) Data Set". The first one contains 30,000 instances described by 24 numerical attributes (including $n = 23$ conditional attributes and $m = 1$ decision attribute). The second one contains 1,000 instances described by 21 attributes ($n = 20$ conditional and $m = 1$ decision). In both original sets, there is no missing value. During the investigation, the missing values were simulated by eliminating the data using previously defined patterns that take into account all possible numbers of missing values. In the case of 1 and $n-1$ missing values, all n possible combinations have been tested. For the other numbers of the missing values, random yet varied patterns have been prepared. Testing all possible combinations of the missing values for 20 and more conditional attributes would, as for the time being, take more than a lifetime if such testing were to be conducted with the use of computer systems of today. The separate patterns of the missing values have been prepared for particular databases and remained without any changes for all classification systems under investigation.

Each dataset has been divided into 10 subsets in order to perform a 10–fold cross–validation procedure. In each stage, 9 various subsets were merged and treated as a set of reference samples. They were used to select support vectors in the SVM algorithm, just as reference samples in k–nearest neighbour classifiers, to learn neural networks, and to extract the knowledge in the form of rules for fuzzy systems.

In every cross–validation stage, a set consisting of 9 merged subsets containing the reference samples as well as a single 10^{th} subset used as a testing set were applied to test classification system.

4 Results of the investigations

4.1 Effectiveness of classification

The results obtained in the investigations have been presented in two ways. In the first one, Figures 1–4 contain the results for a single classification system and all benchmark datasets (DCCC — Default of Credit Card Clients' Classification, SGC — South German Credit Classification, GI — Glass Identification, BCW — Breast Cancer Wisconsin, WC — Wine, PID — Pima Indian Diabetes, Iris — Fisher's Iris), the learning (lrn) and the testing (tst) sets. This allows us to compare the consequences of the missing data for the problems described by various numbers of features (conditional) classified by particular rough set–based classification systems. In the other one, Figures 5 and 6 show the results for a single benchmark (data set) and all classifi-

cation systems (RSVM – rough support vector machine, RkNN — rough k–nearest neighbour, RNN — rough neural network, RFuzz — rough fuzzy system). This allows us to compare the performance of particular rough set–based classification systems. The results are also presented in the form of Tables 1–10.

Table 1. The results of the default of credit card clients' classification test using the rough support vector classifier

Number of missing values	Decisions [%]		
	correct	incorrect	no decision
0	71.1/70.7	28.9/29.3	0.0/0.0
1	11.2/11.4	2.6/2.6	86.2/86.0
2	2.5/2.6	0.7/0.7	96.7/96.7
3	0.8/0.8	0.4/0.4	98.8/98.8
4	0.8/0.8	0.3/0.3	98.9/98.9
5	0.8/0.7	0.2/0.2	99.0/99.1
6-9	0.0/0.0	0.1/0.1	99.9/99.9
10-23	0.0/0.0	0.0/0.0	100.0/100.0

Table 2. The results of the South German Credit classification test using the rough support vector classifier

Number of missing values	Decisions [%]		
	correct	incorrect	no decision
0	100.0/72.8	0.0/27.2	0.0/0.0
1	31.0/23.2	0.0/3.5	69.0/73.3
2	16.6/13.6	0.0/1.8	83.4/84.6
3	10.1/8.8	0.0/1.3	89.9/89.9
4	7.8/6.5	0.0/1.0	92.2/92.4
5	5.7/4.5	0.0/0.7	94.3/94.8
6	3.7/2.9	0.0/0.5	96.4/96.6
7	2.9/2.6	0.0/0.5	97.1/96.9
8	2.6/2.3	0.0/0.5	97.4/97.3
9	1.9/1.7	0.0/0.4	98.1/97.9
10	1.5/1.3	0.0/0.4	98.5/98.3
11-12	1.2/1.1	0.0/0.4	98.8/98.5
13	1.1/1.0	0.0/0.4	98.9/98.6
14	1.1/1.0	0.0/0.4	98.9/98.7
15-19	0.7/0.7	0.0/0.2	99.3/99.1
20	0.0/0.0	0.0/0.0	100.0/100.0

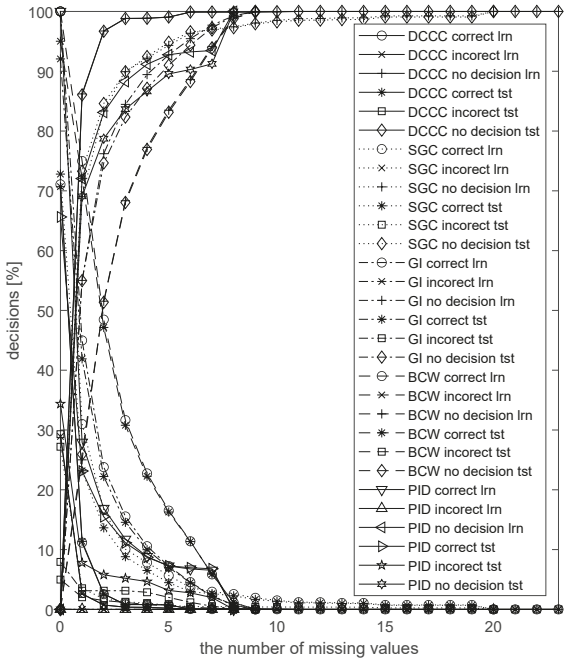

Figure 1. The results of the classification using the rough support vector machines

Table 3. The results of the default of credit card clients' classification test using the rough k-nearest neighbour algorithm

Number of missing values	Decisions [%]		
	correct	incorrect	no decision
0	71.8/61.5	28.2/38.5	0.0/0.0
1	59.4/58.1	27.7/30.6	12.9/11.2
2	53.0/57.1	26.0/28.6	21.0/14.3
3	44.0/52.2	18.6/21.8	37.4/26.0
4	43.2/47.5	17.9/15.6	39.0/36.9
5	40.7/44.9	17.7/9.7	41.5/45.4
6	35.5/44.7	16.4/7.1	48.1/48.1
7	32.6/38.6	7.3/0.5	60.1/60.9
8	26.8/35.2	2.7/0.0	70.5/64.8
9	26.3/28.5	0.0/0.0	73.7/71.5
10	23.2/19.6	0.0/0.0	76.8/80.4
11	20.3/10.9	0.0/0.0	79.7/89.1
12	19.2/4.4	0.0/0.0	80.8/95.6
13	14.1/0.0	0.0/0.0	85.9/100.0
14	6.6/0.0	0.0/0.0	93.4/100.0
15	1.8/0.0	0.0/0.0	98.2/100.0
16-23	0.0/0.0	0.0/0.0	100.0/100.0

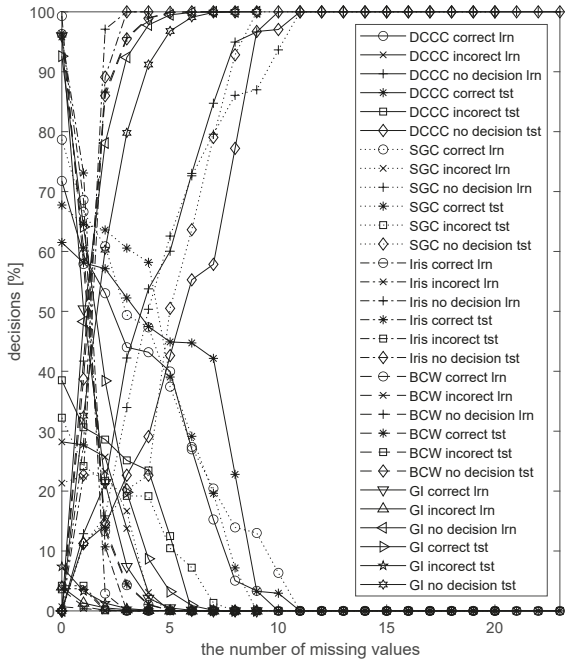

Figure 2. The results of the classification using the rough k-nearest neighbour algorithm

Table 4. The results of the South German Credit classification test using the rough k–nearest neighbour algorithm

Number of missing values	Decisions [%]		
	correct	incorrect	no decision
0	78.7/67.8	21.3/32.3	0.0/0.0
1	66.6/64.5	22.5/24.2	10.9/11.3
2	60.9/63.6	23.3/21.7	15.9/14.7
3	49.4/60.6	17.3/19.2	33.3/20.2
4	47.3/56.9	11.3/18.3	41.4/24.8
5	42.3/49.4	4.0/12.2	53.7/38.4
6	35.9/45.0	0.0/7.7	64.1/47.4
7	32.0/37.1	0.0/3.5	68.0/59.4
8	29.7/34.5	0.0/0.0	70.3/65.5
9	26.1/26.6	0.0/0.0	73.9/73.4
10	22.0/25.2	0.0/0.0	78.0/74.8
11	17.3/23.8	0.0/0.0	82.7/76.2
12	13.8/16.5	0.0/0.0	86.2/83.5
13	10.9/10.8	0.0/0.0	89.1/89.2
14	5.0/7.6	0.0/0.0	95.0/92.4
15	3.1/0.3	0.0/0.0	96.9/99.7
16-20	0.0/0.0	0.0/0.0	100.0/100.0

Table 5. The results of the default of credit card clients' classification test using the rough neural network no 1

Number of missing values	Decisions [%]		
	correct	incorrect	no decision
0	70.8/69.8	29.2/30.2	0.0/0.0
1	23.5/27.2	5.4/8.1	71.1/64.7
2	6.2/7.2	0.8/2.2	93.0/90.6
3	1.8/2.3	0.2/0.8	98.0/96.9
4	0.4/0.6	0.0/0.2	99.6/99.2
5	0.1/0.1	0.0/0.0	99.9/99.8
6–23	0.0/0.0	0.0/0.0	100.0/100.0

Table 6. The results of the default of credit card clients' classification test using the rough neural network no 2

Number of missing values	Decisions [%]		
	correct	incorrect	no decision
0	75.1/71.1	24.9/28.9	0.0/0.0
1	1.8/6.1	0.3/1.4	97.9/92.5
2	0.0/0.1	0.0/0.0	100.0/99.8
3-23	0.0/0.0	0.0/0.0	100.0/100.0

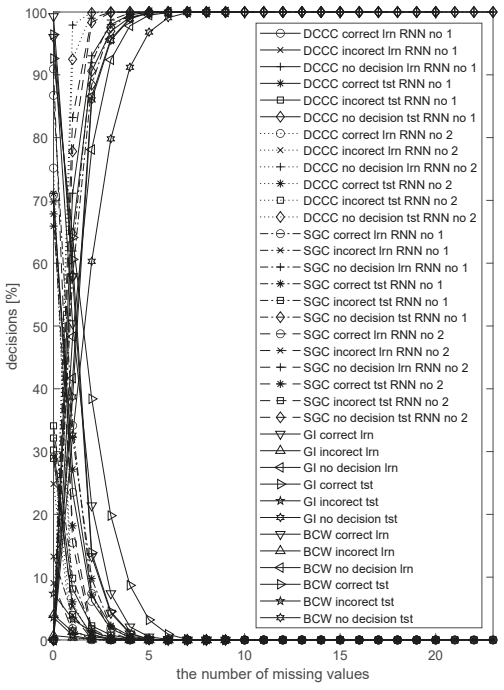

Figure 3. The results of classification using the rough neural networks

Table 7. The results of the South German Credit classification test using the rough neural network no 1

Number of missing values	Decisions [%]		
	correct	incorrect	no decision
0	86.7/65.9	13.3/34.1	0.0/0.0
1	34.2/32.3	3.9/9.8	61.9/57.9
2	7.4/9.8	0.7/1.8	91.9/88.4
3	1.2/2.1	0.1/0.3	98.7/97.6
4	0.1/0.3	0.0/0.0	99.9/99.6
5-20	0.0/0.0	0.0/0.0	100.0/100.0

Table 8. The results of the South German Credit classification test using the rough neural network no 2

Number of missing values	Decisions [%]		
	correct	incorrect	no decision
0	90.9/67.9	9.1/32.2	0.0/0.0
1	15.5/18.2	1.3/4.0	83.2/77.8
2	0.9/1.5	0.1/0.2	99.0/98.3
3	0.0/0.1	0.0/0.0	100.0/99.9
4-20	0.0/0.0	0.0/0.0	100.0/100.0

Table 9. The results of the default of credit card clients' classification test using the rough fuzzy system

Number of missing values	Decisions [%]		
	correct	incorrect	no decision
0	82.4/81.1	17.6/18.9	0.0/0.0
1	58.4/57.5	10.5/11.4	31.1/31.1
2	34.8/34.3	5.5/6.2	59.6/59.5
3	19.6/19.4	2.9/3.3	77.5/77.4
4	8.5/8.3	1.1/1.3	90.4/90.4
5	3.3/3.2	0.3/0.4	96.4/96.4
6	0.8/0.8	0.1/0.1	99.1/99.1
7	0.1/0.1	0.0/0.0	99.9/99.9
8-23	0.0/0.0	0.0/0.0	100.0/100.0

Table 10. The results of the South German Credit classification test using the rough fuzzy system

Number of missing values	Decisions [%]		
	correct	incorrect	no decision
0	68.4/66.6	31.6/33.4	0.0/0.0
1	52.6/51.3	21.8/22.9	25.6/25.9
2	34.5/33.9	12.3/14.1	53.2/52.0
3	22.5/22.6	7.4/8.5	70.2/68.9
4	14.7/14.8	4.6/5.5	80.7/79.7
5	8.6/8.8	2.4/3.0	89.0/88.2
6	4.8/4.8	1.3/1.7	93.8/93.4
7	2.3/2.3	0.6/0.8	97.1/96.9
8	1.2/1.2	0.3/0.4	98.5/98.4
9	0.4/0.3	0.1/0.1	99.5/99.6
10	0.2/0.2	0.0/0.0	99.7/99.8
11	0.1/0.1	0.0/0.0	99.9/99.9
12-20	0.0/0.0	0.0/0.0	100.0/100.0

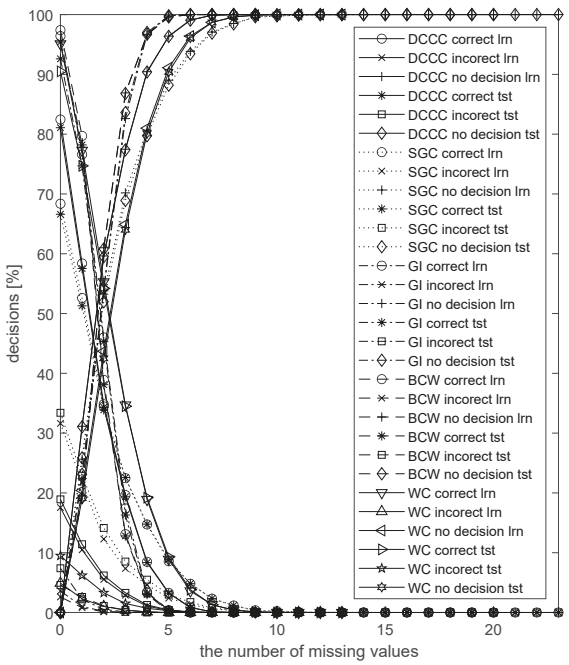

Figure 4. The results of the classification using the rough fuzzy systems

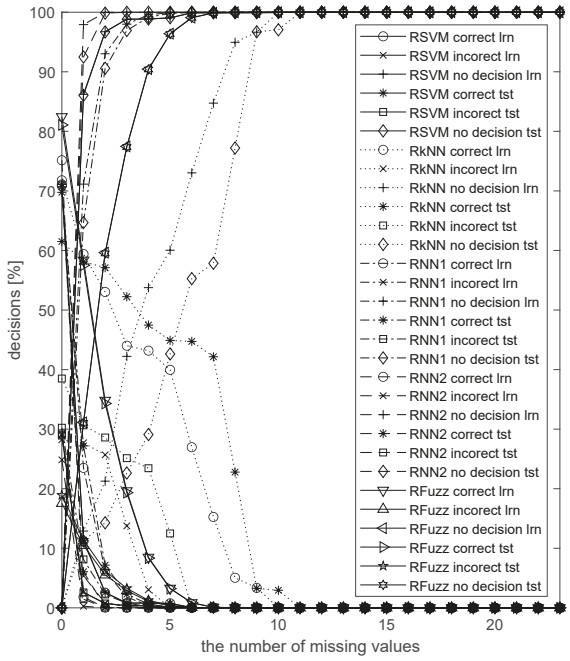

Figure 5. The results of the default of credit card clients' classification

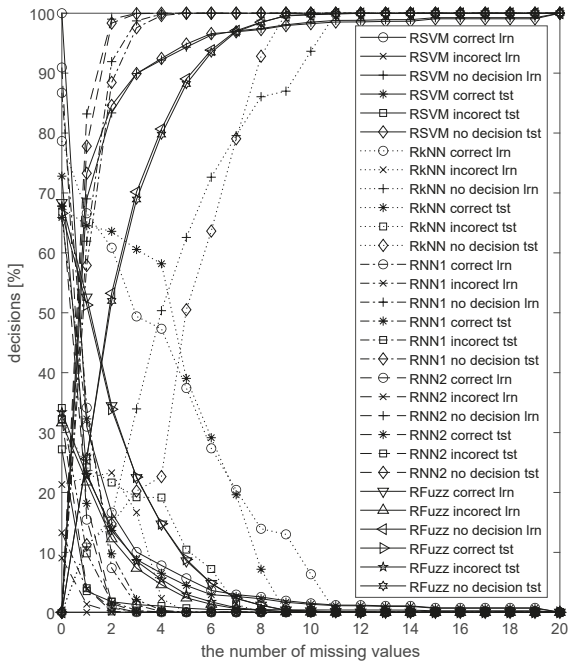

Figure 6. The results of the South German Credit classification

4.2 Classification run–time

Computational complexity of particular types of classification systems varies. An extremely high complexity of the rough k–nearest neighbour classifier was already noticed in [21]. The complexity of neural networks and fuzzy systems, as well as rough set–based versions, strictly depends on the number of neurons and the number of rules, respectively. The number of desired support vectors in the SVM algorithm depends on the number of conditional features. However, this data parameter affects the time which a system needs to process the data.

To compare the time of a classification process realised by the particular classification systems, the systems have been implemented in a common environment with a common data structure and each of them did multiple runs on a single CPU core. The time needed for preparing particular classification systems is not analysed as in the authors' opinion cannot be compared. As is known, the SVM preparation comes down to selecting the support vectors [30]. In the case of k–nearest neighbour algorithm all reference samples must be just located in the memory [2]. The learning of neural network is a stochastic process [27, 31], but creating the rules from data was realised using deterministic algorithm [6, 7].

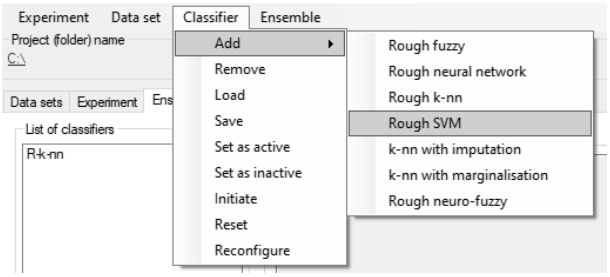

Figure 7. Fragment of the application window that implements the examined classification systems

Table 11. Classification process time

Rough set–based classification system	Time relative to the slowest system [%]
Rough support vector machines	0.011
Rough k–nearest neighbour	100.000
Rough neural network	0.127
Rough fuzzy system	1.051

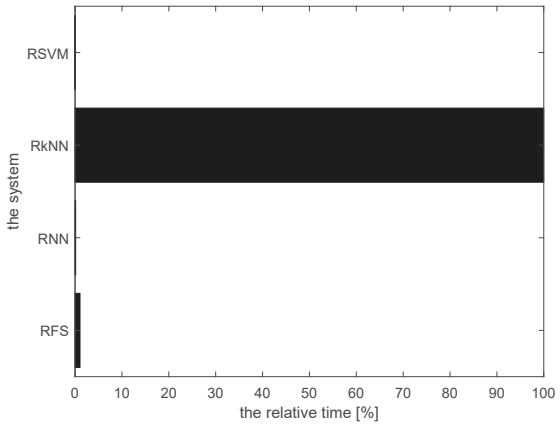

Figure 8. Classification process time

Figure 7 shows a fragment of the corresponding author's application window that implements the examined classification systems. The application has been developed to test the flexibly configured ensembles of various rough set–based classification systems and the iterative mode of a single system [21]. This is the reason why the illustration shows some additional elements. However, it can also work with a single classification system. In order not to make the results dependent on specific implementation and a hardware platform, the measured times have been presented according to the slowest system, i.e. the rough k–nearest neighbour with the value of 100%. The values obtained by all systems under investigation have been presented in Table 11 and Figure 8.

Conclusions

The paper contains the results of extended investigations of the rough set–based classification systems proposed in previous works [21, 22]. The investigations have been processed for benchmarks with more than 20 conditional attributes. Such a number allows for the testing of a representative number of missing values patterns in an acceptable time and, together with the earlier results, it allows us to examine the effect of missing the value for various numbers of the considered features.

The consequence of inaccessibility of input values in the classification process strictly depends on the redundancy level in the data. An important step in designing any decision system is selecting conditional features in order to limit the number of features and redundancy in input data. The rule generation processes also minimize the redundancy in the resultant rule set and the input vector. This significantly reduces the ability of make a decision in the case of missing values. Despite this, the results indicate that redundancy still exists in both the data and rules. Obviously, the redundancy is not visible in all features to the same extent and, as the result, the relevance of particular parts of input data varies. It can be observed in the detailed results, which are not presented in the paper. The main results, presented in Section 4.1, show that the loss of a single value is felt more in the case of a small number of available values, and felt less in the case of a higher number of available values. However, when we compare the percentage loss of input data concerning all considered features, we can see that the number of unclassified samples is higher in the case of the benchmark with a higher number of considered conditional features. It can be concluded that in such cases, a single feature carries less information and the level of redundancy is lower than in the cases with a small number of features. It is especially visible for rough fuzzy classification systems, where the redundancy has been capped also in the rule set. It would be interesting to check the correlation between the above observations and the significance of attributes calculated following Pawlak's rough sets theory.

The results indicate a weakness of the k–nearest neighbour algorithm. The algorithm is extremely simple, easy to implement, and very efficient. However, it needs a huge amount of resources, i.e. memory space and processor time. It is specifically evident when the double rough functionality has been added to the algorithm. For this reason, the other methods are more appropriate in real applications. The rough k–nearest neighbour method must be redefined and optimized in future work.

References

[1] Amiri, M., Jensen, R., Eftekhari, M., Parthaláin, N.M.: Dataset condensation using owa fuzzy-rough set-based nearest neighbor classifier. In: 2016 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE), pp. 1934–1941 (2016). DOI 10.1109/FUZZ-IEEE.2016.7737928

[2] Cover, T., Hart, P.: Nearest neighbor pattern classification. IEEE Transactions on Information Theory

13 (1), 21–27 (1967)

[3] Dubois, D., Prade, H.: Rough fuzzy sets and fuzzy rough sets. International Journal of General Systems 17 (2–3), 191–209 (1990)

[4] Dubois, D., Prade, H.: Putting rough sets and fuzzy sets together. In: R. Słowiński (ed.) Intelligent Decision Support: Handbook of Applications and Advances of the Rough Sets Theory, pp. 203–232. Kluwer, Dordrecht (1992)

[5] Dziwiński, P., Starczewski, J.T., Bartczuk, Ł.: New linguistic hedges in construction of interval type-2 fls. In: L. Rutkowski, R. Scherer, R. Tadeusiewicz, L.A. Zadeh, J.M. Zurada (eds.) Artifical Intelligence and Soft Computing, pp. 445–450. Springer Berlin Heidelberg, Berlin, Heidelberg (2010)

[6] Grzymala-Busse, J.W.: An overview of the LERS1 learning systems. In: Proceedings of the 2nd International Conference on Industrial and Engineering Applications of Artificial Intelligence and Expert Systems, pp. 838–844 (1989)

[7] Grzymala-Busse, J.W.: LERS — a system for learning from examples based on rough sets. In: R. Słowiński (ed.) Intelligent Decision Support: Handbook of Applications and Advences of the Rough Sets Theory, pp. 3–18. Kluwer, Dordrecht (1992)

[8] Guo, Q., Qu, Y., Deng, A.: Invasive weed optimisation inspired fuzzy-rough feature selection. In: 2016 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE), pp. 1942–1947 (2016). DOI 10.1109/FUZZ-IEEE.2016.7737929

[9] Korytkowski, M., Nowicki, R., Scherer, R., Rutkowski, L.: Ensemble of rough–neuro–fuzzy systems for classification with missing features. In: Proceedings of World Congress on Computational Intelligence 2008, pp. 1745–1750 (2008)

[10] Korytkowski, M., Nowicki, R.K., Rutkowski, L., Scherer, R.: MICOG defuzzification rough–neuro–fuzzy system ensemble. In: 2010 IEEE International Conference on Fuzzy Systems, IEEE World Congress on Computational Intelligence, pp. 2268–2273. Barcelona, Spain (2010)

[11] Li, D., Zhang, H., Li, T., Bouras, A., Yu, X., Wang, T.: Hybrid missing value imputation algorithms using fuzzy c-means and vaguely quantified rough set. IEEE Transactions on Fuzzy Systems pp. 1–1 (2021). DOI 10.1109/TFUZZ.2021.3058643

[12] Lingras, P.: Comparison of neofuzzy and rough neural networks. Information Sciences 110 (3–4), 207–215 (1998)

[13] Lingras, P.: Fuzzy–rough and rough–fuzzy serial combinations in neurocomputing. Neurocomput. 36 (1–4), 29–44 (2001)

[14] Liu, H., Tuo, H., Liu, Y.: Rough neural network of variable precision. Neural Processing Letters 19 (1), 73–87 (2004). DOI 10.1023/B:NEPL.0000016851.47914.40. URL https://doi.org/10.1023/B:NEPL.0000016851.47914.40

[15] Mertz, C.J., Murphy, P.M.: UCI respository of machine learning databases. Available online: http://www.ics.uci.edu/pub/machine-learning-databases

[16] Mitra, S., Hayashi, Y.: Neuro-fuzzy rule generation: survey in soft computing framework. IEEE Transactions on Neural Networks 11 (3), 748–768 (2000). DOI 10.1109/72.846746

[17] Nguyen, H.S.: On Exploring Soft Discretization of Continuous Attributes, pp. 333–350. Springer Berlin Heidelberg, Berlin, Heidelberg (2004)

[18] Nowicki, R.: On combining neuro–fuzzy architectures with the rough set theory to solve classification problems with incomplete data. IEEE Trans. on Knowledge and Data Engineering 20 (9), 1239–1253 (2008). DOI 10.1109/TKDE.2008.64

[19] Nowicki, R.: Rough–neuro–fuzzy structures for classification with missing data. IEEE Trans. on Systems, Man, and Cybernetics—Part B: Cybernetics 39 (6), 1334–1347 (2009). DOI 10.1109/TSMCB.2009.2012504

[20] Nowicki, R.: On classification with missing data using rough-neuro-fuzzy systems. International Journal of Applied Mathematics and Computer Science 20 (1), 55–67 (2010)

[21] Nowicki, R.K.: Rough Set–Based Classification Systems. Springer International Publishing, Cham (2019). DOI 10.1007/978-3-030-03895-3

[22] Nowicki, R.K., Grzanek, K., Hayashi, Y.: Rough support vector machine for classification with interval and incomplete data. Journal of Artificial Intelligence and Soft Computing Research 10 (1), 47–56 (2020). DOI 10.2478/jaiscr-2020-0004

[23] Nowicki, R.K., Korytkowski, M., Scherer, R.: Rough neural network ensemble for interval data classification. In: 2018 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE), pp. 1–7 (2018). DOI 10.1109/FUZZ-IEEE.2018.8491609

[24] Pawlak, Z.: Information systems — theoretical foundations. Information Systems 6, 205–218 (1981)

[25] Pawlak, Z.: Rough sets. International Journal of Computer and Information Sciences 11 (5), 341–356 (1982)

[26] Qiu, W., Hu, Z.: Composed fuzzy rough set and its applications in fuzzy rsar. In: M. Xu, Y. Zhan, J. Cao, Y. Liu (eds.) Advanced Parallel Processing Technologies, pp. 753–763. Springer Berlin Heidelberg, Berlin, Heidelberg (2007)

[27] Rumelhart, D.E., Hinton, G.E., Williams, R.J.: Learning representations by back-propagating errors. Nature 323, 533–536 (1986)

[28] Starczewski, J.T.: Advanced Concepts in Fuzzy Logic and Systems with Membership Uncertainty, Studies in Fuzziness and Soft Computing, vol. 284. Springer-Verlag, Berlin Heidelberg (2013)

[29] Tsang, E.C.C., Zhao, S.: A fast algorithm to building a fuzzy rough classifier. In: X. Wang, W. Pedrycz, P. Chan, Q. He (eds.) Machine Learning and Cybernetics, pp. 409–417. Springer Berlin Heidelberg, Berlin, Heidelberg (2014)

[30] Vapnik, V.: Estimation of Dependences Based on Empirical Data. Springer-Verlag (1982)

[31] Żurada, J.M.: Introduction to Artificial Neural Systems. West Publishing Company (1992)

Robert K. Nowicki received the M.Sc. degree in electrical engineering from the Czestochowa University of Technology, in 1997, the Ph.D. degree in computer engineering from AGH University of Science and Technology, Krakow, Poland, in 2000 and the D.Sc. degree in computer engineering from the System Research Institute of the Polish Academy of Sciences, in 2010. Since 2020, he is the full professor. He works in the Department of Computational Intelligence at the Czestochowa University of Technology. He has published over 100 technical papers in journals, conference proceedings, and book chapters as well as 2 books. In 2010, he received the scientific prize of the Engineering Sciences Division of the Polish Academy of Sciences. His research interests include fuzzy systems, artificial neural networks, genetic algorithms, rough sets, and hybrid methods.

Robert Seliga is a doctor of economic sciences. His research interest include organization and management, and in particular: public marketing, relationship marketing and applications of artificial intelligence techniques in management. Currently, he is an assistant professor of the University of Social Sciences, Rector's Plenipotentiary for teaching and scientific reporting, Vice-Dean of the Department at the University of Social Sciences in Lodz. He serves as the associate editor of the Journal of Intercultural Management, editor of the Entrepreneurship and Management Journal.

Dariusz Zelasko received his B.Eng. degree in information technology at the College of Information Technology, Katowice, Poland, in 2010. In 2012 he received his M.Sc. degree in computer science at the Cracow University of Technology, Poland. His research interests include computer networks, QoS, cybersecurity, IoT, mobile networks, machine learning, and ensemble learning.

Yoichi Hayashi received the Dr. Eng. degree in Systems Engineering from the Tokyo University of Science, Tokyo in 1984. In 1986, he joined the Computer Science department of Ibaraki University, Japan, as an Assistant Professor and was a Visiting Professor of Computer Science Department of the University of Alabama at Birmingham and the University of Canterbury, respectively. Since 1996, he is a full Professor of Computer Science department at Meiji University, Tokyo. He is the author of over 230 papers published in computer science. His current research interests include Artificial Intelligence, high performance classifier, Deep Learning, high accuracy rule extraction and medical informatics. He is the editorial board member of Artificial Intelligence in Medicine and was the Associate Editor of the IEEE Transactions on Fuzzy Systems, the Action Editor of Neural Networks. He is a senior member of the IEEE since 2000.