DOI: 10.15439/2023F4139
ISSN 2300-5963 ACSIS, Vol. 35

Explainability in RIONA Algorithm Combining Rule Induction and Instance-Based Learning

Grzegorz Góra
University of Warsaw
Banacha 2, 02-097 Warszawa
Poland
Email: ggora@mimuw.edu.pl

Andrzej Skowron
Systems Research Institute PAS
Newelska 6, 01-447 Warszawa
Poland
Email: skowron@mimuw.edu.pl

Arkadiusz Wojna
DeepSeas
12121 Scripps Summit Court
San Diego, CA 92131, USA
Email: wojna@mimuw.edu.pl

***Abstract*—The article concerns the well-known RIONA algorithm. We focus on the explainability property of this algorithm. The theoretical results, formulated and proved in the paper, show the relationships of the RIONA classifiers to both instance- and rule-based classifiers. In particular, we show the equivalence (relative to the classification) of the RIONA algorithm with the rule-based algorithm generating all consistent and maximally general rules from the neighbourhood of the test case.**

I. Introduction

IN THE paper, we focus on the learning algorithm for supervised learning [1], [2], [3]. Specifically, we focus on the well-known RIONA algorithm [4], [5], [6]. This algorithm combines two widely-used empirical approaches: rule induction and instance-based learning [7], [8], [9], [10]. Both these approaches use reasoning schemes comprehensible to a human. It is essential since, Explainable AI [11], [12], [13] is becoming more and more useful in real-life applications. The classifying system should provide for the given test object not only decision but also its user-understandable explanation. In the paper, we present theoretical results of considered algorithm that allow to meet these requirement.

A few concepts form the framework of the RIONA algorithm. First, instead of inducing an excessive number of decision rules in advance to use them during testing, it induces decision rules relevant only for the test example. This is a strategy of so-called lazy learning [14]. Second, only rules from the neighbourhood of the given test example are considered. Third, it automatically groups numerical and symbolic values of attributes by using more general than commonly-used conditions. Fourth, RIONA computes optimal size for the neighbourhoods of objects.

The properties of RIONA algorithm are worth studying as it was reported in the literature as one of the most accurate classification methods in many experimental comparisons done by various researchers (the most commonly used RIONA implementation is a classifier in the WEKA platform named RseslibKnn [15]), to name a few: Facebook content recognition [16, Chapter 1] (RIONA was the best one of 21 tested algorithms), environmental sound recognition [17] (best of 9 algorithms), metabolic pathway prediction of plant enzymes [18] (2nd of 47 algorithms), acoustic-based environment monitoring [19] (2nd of 8 algorithms), context awareness of a service robot [20] (2nd of 8 algorithms), student performance prediction [21] (5th of 47 algorithms).

The novelty of the paper is in theoretical results creating the basis for explainability of classifications returned by classifiers. The results concern the relationships of the classifiers generated by the RIONA algorithm to classifiers obtained by applying instance-based as well as rule-based approaches. In particular, it occurs that RIONA classifiers are equivalent (relative to the classification property) to classifiers produced by the rule-based algorithm based on all consistent and maximally general rules generated from the neighbourhood of the test case. Such rules are easily interpretable by humans.

The paper relates to the PhD thesis [6].

II. Basic notions

$|X|$ denotes cardinality (size) of the set X set. If $\mathbf{A}$ and $\mathbf{B}$ are algorithms then the equality $\mathbf{A}(v) = \mathbf{B}(w)$ means that the values returned by $\mathbf{A}$ on input v and by $\mathbf{B}$ on input w are the same.

By $\mathbf{X}$ we denote a set of objects, called the domain of learning and by $Atr = A \cup \{d\}$ a finite set of attributes atr, where $atr : \mathbf{X} \longrightarrow V_{atr}$. V_{atr} is called the value set of atr. Attributes from A are called *conditional attributes* and the attribute $d \notin A$ is called the *decision*. V_d is called the decision set. We assume for simplicity of notation that $V_d = \{1, \ldots, n_d\}$. Any object $x \in \mathbf{X}$ is represented relative to its signature $Inf_{A(x)}$, i.e. a set of pairs $(a, a(x))$ for each $a \in A$. We use the symbols A_{sym} and A_{num} for denoting the sets of symbolic and numerical attributes, respectively. If $a \in A_{sym}$ then V_a is a finite set. If $a \in A_{num}$ then without loss of generality, we assume that V_a is equal to an interval (l_a, u_a), where $l_a, u_a \in \mathbb{R}$ (possibly not all of the values from the interval are used).

A *decision system* is a triplet $(\mathbf{X}, A, d)$ if $\mathbf{X}$ is the set of objects, A is a set of attributes, and d is a decision function. A *pseudometric decision system* is a 4-tuple $(\mathbf{X}, A, d, \{\varrho_a\}_{a\in A})$ if $(\mathbf{X}, A, d)$ is a decision system and for any attribute $a \in A$, ϱ_a is a pseudometric on the respective value set V_a, i.e. for any $a \in A$, (V_a, ϱ_a) is a pseudometric space [22].

In the sequel by $\mathbb{D}$ is denoted a given (pseudometric) decision system of the above form.

In the paper we study a combination of two methods (learning algorithms) inducing a *classifier* from a given subdecision system $(trnSet, A, d)$, where $trnSet \subseteq \mathbf{X}$ and attributes from $A \cup \{d\}$ are restricted to $trnSet$, which for any $x \in \mathbf{X}$ computes the decision $\hat{d}(x)$ in such a way that $\hat{d}$ is *close* to d [2]. It should be noted that in practical experiments, the normalized Euclidean metric was used for numerical attributes, and the SVDM [23] pseudometric for symbolic attributes. If $a \in A_{num}$, the normalisation is obtained by using $a^{\max}$ and $a^{\min}$, which denote the maximal and the minimal values for an attribute a among training examples $trnSet$.

A. Rule-based methods

The induction of rule sets is one of the fundamental Machine Learning (ML) techniques (see e.g. [7]). Its significance stems from the fact that a human may easily comprehend knowledge representation in the form of rules. Decision rules specify the appropriate course of action in a given circumstance. Decision rules frequently have the form 'if φ then ψ', where φ denotes the premise of the rule and ψ denotes its consequence; ψ is a formula defined by the decision attribute d.

Decision rules are generated using rule induction algorithms from a training set. The premises of the rules are represented by a conjunction of *elementary conditions* and the consequences are describing the particular decision. Each elementary condition describes a collection of the attribute's values. Roughly speaking, it has the form $a \in V$, where $V \subseteq V_a$, and a is an attribute. We first determine how such sets V of values can be expressed in a formal language. Next, we define the semantics (meaning) of specified expressions from this language in the powerset of the attribute value set V_a. For simplicity of notation we do not distinguish between symbols denoting values (or intervals) and values (or intervals) themselves.

Definition II.1. *Let $\mathbb{D}$ be a (pseudometric) decision system. For symbolic attributes $a \in A_{sym}$, the* description of any elementary set *has one of the following forms:*

$$\emptyset \tag{1}$$

$$\{v\}, \text{ where } v \in V_a, \tag{2}$$

$$V_a, \tag{3}$$

$$B(c,r), \text{ when } \mathbb{D} \text{ is pseudometric decision system and where } c \in V_a,\ r \in \mathbb{R}, r \geq 0. \tag{4}$$

The description of elementary set *for the decision attribute d is of the form $\{v\}$, for $v \in V_d$.*

For numerical attributes $a \in A_{num}$, the description of elementary set *has one of the following forms:*

$$\emptyset \tag{5}$$

$$[b,e], (b,e], [b,e), (b,e), \text{ where } b,\ e \in \mathbb{R} \text{ are such that the corresponding interval between points } b \text{ and } e \text{ is included in } V_a. \tag{6}$$

The semantics $||des||_{\mathbb{D}} \subseteq V_a$ of any description des of the elementary set for attribute $a \in A \cup \{d\}$ is defined as follows:

$$\begin{aligned} ||\emptyset||_{\mathbb{D}} = \emptyset,\ \ ||\{v\}||_{\mathbb{D}} = \{v\},\ \ ||V_a||_{\mathbb{D}} = V_a, \\ ||[b,e]||_{\mathbb{D}} = [b,e], ||(b,e]||_{\mathbb{D}} = (b,e], \\ ||[b,e)||_{\mathbb{D}} = [b,e), ||(b,e)||_{\mathbb{D}} = (b,e), \\ ||B(c,r)||_{\mathbb{D}} = \{w \in V_a : w \in B(c,r)\} \\ = \{w \in V_a : \varrho_a(c,w) \leq r\} \text{ (called the ball set).} \end{aligned}$$

Now, the elementary conditions expressed in a language and their semantics can be defined.

Definition II.2. *Let $\mathbb{D}$ be a (pseudometric) decision system.*

Any expression $a \in V$, where $a \in A$ and V is a description of elementary set for attribute a is called an elementary condition*. The semantics of $a \in V$ is defined by $[[a \in V]]_{\mathbb{D}} = \{x \in \mathbf{X} : a(x) \in ||V||_{\mathbb{D}}\}$.*

$[[a \in V]]_{\mathbb{D}}$ may be restricted to subsets of $\mathbf{X}$, e.g. $[[a \in V]]_{trnSet} = [[a \in V]]_{\mathbb{D}} \cap trnSet$.

An example (case) x satisfies the elementary condition $a \in V$(or, in short, $(a \in V)(x)$ is satisfied) if $x \in [[a \in V]]_{\mathbb{D}}$.

The set $||V||_{\mathbb{D}} \subseteq V_a$ for a given elementary condition $a \in V$ is equal to

- $\{v\}$ for some $v \in V_d$ for the decision attribute d (see set description 2 and its semantics),
- a proper interval for the numerical attribute (see set description 6 and its semantics), and
- $\{v\}$ for some $v \in V_a$, V_a or a ball set for the symbolic attribute a (see set descriptions 2, 3, 4, respectively and their semantics).

A given object x satisfies the elementary condition $a \in V$ if the value of a on x, i.e. $a(x)$ is in the set defined in $\mathbb{D}$ by V, i.e. $a(x) \in ||V||_{\mathbb{D}}$. Instead of $a \in \{v\}$ we write $a = v$ and instead of *trivial* elementary condition $a \in V_a$ (always true, i.e. the set of objects satisfying this condition is equal to the set $\mathbf{X}$) we write $a = *$.

Now, we introduce concepts related to syntax and semantics of decision rules.

Definition II.3. *Let $\mathbb{D}$ be a (pseudometric) decision system. A* decision rule *is an expression of the form*

$$if\ t_1 \wedge t_2 \wedge \ldots \wedge t_m\ then\ d = v,$$

where $m = |A|$, t_i is an elementary condition for an attribute a_i for $i = 1, 2..., m$, and $v \in V_d$.

The semantics of the premise $t_1 \wedge t_2 \wedge \ldots \wedge t_m$ of the rule r (in $\mathbb{D}$) is defined by

$$[[t_1 \wedge t_2 \wedge \ldots \wedge t_m]]_{\mathbb{D}} = [[t_1]]_{\mathbb{D}} \cap [[t_2]]_{\mathbb{D}} \ldots [[t_m]]_{\mathbb{D}}$$

If $x \in [[t_1 \wedge t_2 \wedge \ldots \wedge t_m]]_{\mathbb{D}}$ then we say that

- *the premise of the rule r is satisfied by example x (or x satisfies this premise),*
- *example x* matches *the rule r,*
- *r* covers *x.*

One can treat a single rule as a classifier assigning examples covered by that rule to the decision class from the rule's consequence. Ideally, we could search for rules $if\ \varphi\ then\ d = v$ such that $[[\varphi]]_{\mathbb{D}} \subseteq [[d = v]]_{\mathbb{D}}$. However, the restriction of this inclusion to $trnSet$, i.e. $[[\varphi]]_{trnSet} \subseteq [[d = v]]_{trnSet}$ is available only. Rules satisfying this last condition (for $trnSet$) are induced from $trnSet$ and an hypothesis on extension of the truth of this inclusion on $\mathbf{X}$ is made. Moreover, rules covering as many as possible examples are generated.

In description of decision rules, trivial conditions are usually omitted [1]. The typical conditions are equations $a = v$ in case of symbolic attributes and inclusions into intervals in case of numerical attributes, e.g.:

$$if\ a_2 = 3 \wedge a_4 \in [2,5] \wedge a_7 = 3\ then\ d = 2.$$

In this paper we use for symbolic attributes more general conditions, i.e. $a \in V$ (see Definition II.2), making it possible to extend singleton sets to the ball sets. If the data set under consideration has some numerical attributes, then by applying discretisation the relevant intervals can be constructed. By applying discretisation to a given decision system a new one is obtained with new attributes being characteristic functions of induced intervals including objects from $trnSet$ labeled by the same decision (see e.g. [24]).

By $t_i(r)$, where r is a given decision rule we denote the i-th condition t_i from Definition II.3. We write $t_a(r)$ instead of $t_i(r)$ if the condition t_i from Definition II.3 concerns the attribute a.

We define three kinds of decision rules by distinguishing elementary conditions (used in Definition II.3) occurring in them. In consequence, we obtain three sets of decision rules.

Definition II.4. *Let $\mathbb{D}$ be a decision system.*

The set $SimRules$ of simple rules *is the set of all rules from Definition II.3 with the elementary conditions of the form $a = v$ for $v \in V_a$ and $a = *$ only.*

The set $CombRules$ of combined rules *is the set of all rules from Definition II.3 with elementary conditions for symbolic attributes of the form as in $SimRules$ and for numerical attributes of the form $a \in I$ (where I is a proper interval description) only.*

Definition II.5. *Let $\mathbb{D}$ be a pseudometric decision system such that for any symbolic attribute $a \in A_{sym}$ there is a distinguished specific value $c_a \in V_a$.*

The set $GenRules\left(\{(\varrho_a, c_a)\}_{a \in A_{sym}}\right)$ (or simply $GenRules$ whenever pairs (ϱ_a, c_a) are clear from the context or irrelevant due to generality) of general rules *is the set of all rules from Definition II.3 where set descriptions in elementary conditions in the premises of the rules are*

(i) *as in the definition of $CombRules$ for numerical attributes and*
(ii) *of specific form of 4, i.e. $B(c, r)$, where $c = c_a$, $r = \varrho_a(c_a, v)$, $v \in V_a$ for symbolic attributes $a \in A_{sym}$.*

[1]In fact, in the description of rules only non-trivial conditions are used. We use trivial conditions only to make the notation simpler.

More details on *general rules* presented in Section III-A will help the reader to better understand our definition.

Definition II.6. *A rule if ϕ then $d = v$ is* consistent *with a set of objects $X \subseteq \mathbf{X}$ (or* consistent *if X is clear from the context) if $d(x) = v$ for any object $x \in X$ matching this rule.*

If the rule if ϕ then $d = v$ is not consistent than it is called inconsistent.

Typically, in the above definition $trnSet$ is used as X. Any rule if ϕ then $d = v$ consistent with $trnSet$ classifies correctly all the training examples covered by that rule i.e. $[[\phi]]_{trnSet} \subseteq [[d = v]]_{trnSet}$.

For further considerations the concept of maximality of rule will be useful.

Definition II.7. *Let $\mathbb{D}$ be a (pseudometric) decision system and let $a \in A$, and let V_1, V_2 be elementary conditions for a. The condition $a \in V_2$ is* more general than *(or is* implied by*) $a \in V_1$, in symbols $a \in V_1 \Rightarrow a \in V_2$ if $||V_1||_{\mathbb{D}} \subseteq ||V_2||_{\mathbb{D}}$.*

For any two rules r_1, r_2 (over $\mathbb{D}$) with the same consequence $d = v$, we say that a rule r_2 is more general than *(or is* implied by*), in symbols $r_1 \Rightarrow r_2$ if $t_i(r_1) \Rightarrow t_i(r_2)$ for $i = 1, \ldots, m$ and $m = |A|$.*

A consistent rule r with a training set $trnSet$ is maximally general *(relative to $trnSet$ and a given set of rules $Rules$) if there is no rule in $Rules$ more general than r which is different from r and consistent with $trnSet$.*

Definition II.8. *Let $\mathbb{D}$ be a (pseudometric) decision system and let $Rules$ be a given set of admissible rules. The* set of maximally general rules *(relative to $trnSet$) $MaxRules(Rules, trnSet)$ is equal to the set of all maximally general rules $r \in Rules$ consistent with $trnSet$.*

We use the following sets of rules: $SimRules$, $CombRules$, $GenRules$ from Definitions II.4 and II.5 as $Rules$ (see Definition II.8). We write $MaxRules$ instead of $MaxRules(Rules, trnSet)$, if from the context $Rules$ and $trnSet$ are known. Hence, $MaxRules$ may denote: $MaxRules(SimRules, trnSet)$, $MaxRules(CombRules, trnSet)$, or $MaxRules(GenRules, trnSet)$.

Computing from $trnSet$ the set of all consistent and maximally general matching at least one case from $trnSet$ is important for some learning algorithms.

In the case of $MaxRules(SimRules, trnSet)$, a consistent rule is maximally general relative to $trnSet$ if it becomes inconsistent (relative to $trnSet$) after substitution of the trivial condition instead of a non-trivial one. Hence, consistent rules from $MaxRules(SimRules, trnSet)$ can be characterised as the rules with minimal length (measured by the number of non-trivial conditions in predecessors). Hence, in the considered case, the problem is to generate the complete set of consistent and minimal decision rules (see e.g. [25]). One can observe that searching for the set of minimal rules can be motivated by the minimum description length principle (MDL) (see e.g. [26]). However, the computational time complexity of algorithms generating $MaxRules(SimRules, trnSet)$ is not feasible when the number of training objects or attributes are large. In fact, the size of the $MaxRules$ set can be exponential relative to $|trnSet|$ (see e.g. [27]). Hence, efficient heuristics are often used to overcome this drawback, especially when not necessarily complete sets of minimal rules are required (see e.g. [28]). There are also other approaches inducing a set of rules fully covering cases from $trnSet$ (see e.g. [29]). Here, we focus on the complete $MaxRules$ set.

In the case of $MaxRules(CombRules, trnSet)$, additionally we deal with numerical attributes. From $trnSet$ maximally general intervals of reals are induced. Searching for maximally general rules for numerical attributes is closely related to the problem of discretisation. A partition of reals in discretisation is consistent if each interval covers only objects with the same decision (see e.g. [24], [27]).

The discretisation problem is a complex task. For example, searching for a consistent partition with the minimal number of cuts is NP-hard (see e.g. [24]). In Subsection III-A we show how to overcome this drawback using lazy learning and focusing on a local part of $\mathbf{X}$ instead of on the whole universe. This is illustrated by the lazy rule induction algorithm Algorithm 3 or Algorithm 4.

In the case of $MaxRules(GenRules, trnSet)$, the additional search is performed for the relevant grouping of values for symbolic attributes into a partition of value sets of symbolic attributes. One can define a partition over an attribute a as any function $P_a : V_a \to \{1, \ldots, m_a\}$. It should be noted that the problem of searching for a consistent family of partitions with the minimal $\sum_{a \in A} |P_a(V_a)|$ is NP-hard (see e.g. [30]). We show in the paper how to overcome this drawback by limiting the number of possible groupings of values of any attribute (from 2^n to n^2, where n is the number of values for an attribute) and by using the lazy rule induction (see Section III-A).

The induced sets of rules from $trnSet$ are used to classify objects. First, for any test object tst there are selected all rules from the set matching this object. Next, the set of matched rules is checked. If all rules matched by tst have the same decision then this decision is assigned to tst else it should be resolved conflict between matching rules voting for different decisions (see e.g. [31]). Typically, it is selected the decision with the highest value of a selected measure used for conflict resolution. We use the commonly used measure for conflict resolution.

Definition II.9. *Let us assume that $\mathbb{D}$ – (pseudometric) decision system, $trnSet$ – training set $trnSet$, tst – test example (case), and $MaxRules$ – set of maximally general rules are given. By $supportSet(r) \subseteq trnSet$, where $r \in MaxRules$ we denote the set of all objects from $trnSet$ matching r, and by $MatchR(tst, v) \subseteq MaxRules$ (where $v \in \{1, \ldots, n_d\}$, i.e. v is a decision of d on some object from $trnSet$) the set of rules from $MaxRules$ with decision v matching the test object tst. Now, we define*

$$Strength(tst, v) = \left| \bigcup_{r \in MatchR(tst,v)} supportSet(r) \right| . \quad (7)$$

From definition it follows that by computing $Strength(tst, v)$ it is counted the number of objects from $trnSet$ covered by some maximally general rule from $MaxRules$ (i) with the decision v and (ii) covering the test example tst.

On the basis of $MaxRules$ and the defined conflict resolution strategy using $Strength$ we define the classifier assigning to a given test object tst the most frequent decision of such training examples from $trnSet$ which are covered by matched by tst rules from $MaxRules$, i.e.:

$$decision_{MaxRules}(tst) = \arg\max_{v \in V_d} Strength(tst, v). \quad (8)$$

As it was observed, the drawback of the presented approach comes from the high computational complexity of $MaxRules$ generation.

B. Lazy rule learning for symbolic attributes

The *lazy learning* (or *memory based learning*) algorithms do not require construction of sets of decision rules before classification of new objects.

kNN is the well known example of such algorithms (see Algorithm 1). For these algorithms, first for any test object tst it is defined its neighbourhood $N(tst, trnSet, k, \varrho) \subseteq trnSet$ (N, for short) with k the most similar to tst (relative to a given distance function ϱ) training examples (where k is a parameter). If more than one example has the same distance from tst as the one already added to the N under construction then all of them are added to $N(tst, trnSet, k, \varrho)$. Then the set $N(tst, trnSet, k, \varrho)$ may contain more than k examples[2].

An interesting example of lazy rule-based algorithm for $SimRules$ is presented in [32]. For a new tst it generates only the relevant for it decision rules and next tst is classified as before on the basis of such rules. The value of Eqn. 7 is computed for any tst object without computing the whole set $MaxRule$.

For given two objects tst, trn, we first define *simple local decision rule*, in symbols $s\text{-}rule(tst, trn)$. The relationship of the set of such rules with $SimRules$ will be presented in the following proposition.

Definition II.10. *Let $\mathbb{D}$ be a decision system, $trn \in trnSet$ and let tst be a test object. A* simple local decision rule *(for short* s-rule*) $s\text{-}rule(tst, trn)$ is the decision rule of the form* if $\bigwedge_{a \in A} t_a$ then $d = d(trn)$, *where conditions t_a for each symbolic attribute a are as follows*

$$t_a = \begin{cases} a = a(trn) & \text{if } a(tst) = a(trn) \\ a = * & \text{if } a(tst) \neq a(trn). \end{cases}$$

[2]This assumption is used in RIONA. Hence, we also adopt it for the kNN algorithm in the paper.

Algorithm 1: kNN(tst, $trnSet$, k, ϱ)

```
Input: a test example $tst$, training set $trnSet$, positive
       integer $k$, pseudometric $\varrho$
Output: predicted decision for $tst$
1  begin
2      $neighbourSet = N(tst, trnSet, k, \varrho)$
3      foreach decision $v \in V_d$ do
4          $supportSet(v) = \emptyset$
5      end
6      foreach $trn \in neighbourSet$ do
7          $v = d(trn)$
8          $supportSet(v) = supportSet(v) \cup \{trn\}$
9      end
10     return $\arg\max_{v \in V_d} |supportSet(v)|$
11 end
```

Algorithm 2: isCons(r, $verifySet$)

```
Input: a rule $r$ : if $\alpha$ then $d = v$,
       set of examples $verifySet$
Output: true if rule $r$ is consistent with $verifySet$, false
otherwise
for all $trn \in verifySet$ do
    if $d(trn) \neq v$ and $trn$ satisfies $\alpha$ then
        return $false$
    end if
end for
return $true$
```

Observe that both objects trn and tst are matching the rule $s\text{-}rule(tst, trn)$ which is maximally specific (the number of trivial conditions is minimal; or inversely, the number of non-trivial conditions is maximal). The following crucial relation between s-rule and maximally general consistent rules from $SimRules$ holds:

Theorem II.1. *[32][3] If $trn \in trnSet$ and tst is a test object than the rule $s\text{-}rule(tst, trn)$ is consistent with $trnSet$ if and only if $MaxRules(SimRules, trnSet)$ contains a rule covering both objects tst and trn.*

Hence, for any test object tst, decision $v \in V_d$ and $MaxRules(SimRules, trnSet)$ set the value $Strength(tst, v)$ from Eqn. 7 is equal to the number of $trn \in trnSet$ having decision $d(trn) = v$ and for which the rule $s\text{-}rule(tst, trn)$ is consistent with $trnSet$. The *simple lazy rule induction algorithm for symbolic attributes* (LAZY) presented in Algorithm 3 realises this idea.

$isCons(r, verifySet)$ in Algorithm 3 verifies if r is consistent with a $verifySet$. For a given object tst and any $trn \in trnSet$, the rule $s\text{-}rule(tst, trn)$ is constructed by

[3]The formulation of this proposition in [32] was different. However, in the case of $SimRules$ it is equivalent to the original proposition and makes it possible in a more direct way present the relationship between local rules as well as $MaxRules$ and algorithms based on these two types of rules.

Algorithm 3: LAZY(tst, $trnSet$)

```
Input: test example $tst$, training set $trnSet$
for all decision $v \in V_d$ do
    $supportSet(v) = \emptyset$
end for
for all $trn \in trnSet$ do
    $v = d(trn)$
    if $isCons(s\text{-}rule(tst, trn), trnSet)$ then
        $supportSet(v) = supportSet(v) \cup \{trn\}$
    end if
end for
return $\arg\max_{v \in V_d} |supportSet(v)|$
```

Algorithm 3. Next, Algorithm 3 is testing the consistency of the rule $s\text{-}rule(tst, trn)$ with the set $trnSet \setminus \{trn\}$, i.e. if all the training examples matching the left-hand side of $s\text{-}rule(tst, trn)$ have identical decision with trn. If the result of testing is positive than trn is added to the support set with the relevant decision. Finally, Algorithm 3 predicts the decision with the support set of the highest cardinality. From Theorem II.1 we obtain:

Corollary II.2. *The following equality holds: $LAZY(tst, trnSet) = decision_{MaxRules}(tst)$, where $trnSet$ is a training set, tst is a test object, and $decision_{MaxRules}(tst)$ is the classifier from Eqn. 8 with $MaxRules = MaxRules(SimRules, trnSet)$.*

From the above considerations it follows that LAZY takes into account only these decision rules that can be involved in the classification of a given test object.

III. RIONA DESCRIPTION

A. Extension and generalisation of lazy rule learning

We introduce an extension and generalization of the LAZY algorithm (see Algorithm 3) that was discussed in Subsection II-B. This novel algorithm permits the use of numerical attributes as well as more general conditions for symbolic attributes.

In Subsections III-A1, III-A2 we present a generalisation of rules introduced before.

For a given test object tst, training object $trn \in trnSet$ and pseudomietric decision system $\mathbb{D}$ with pseudometrics ϱ_a for $a \in A_{sym}$, in addition to simple local decision rule (in short s-rule) (see Subsection II-B) denoted by $s\text{-}rule(tst, trn)$, we consider two new types of local rules: *combined local decision rule* (in short c-rule) and *generalised local decision rule* (in short g-rule) denoted by $c\text{-}rule(tst, trn)$, $g\text{-}rule\left(tst, trn, \{\varrho_a\}_{a \in A_{sym}}\right)$ (or simply $g\text{-}rule\,(tst, trn)$), respectively. In this way we obtain sets composed out of simple rules, combined rules and general rules, denoted by $SimRules$ (see Subsection II-A), $CombRules$, $GenRules$, respectively. We already demonstrated (see Subsection II-B) an important relation between any s-rule and the set of maximally general consistent rules

$MaxRules(SimRules, trnSet)$. Here, we show analogous important relations between any c-rule or g-rule and sets of maximally general rules $MaxRules(CombRules, trnSet)$, $MaxRules(GenRules, trnSet)$ corresponding to sets of rules $CombRules$ and $GenRules$, respectively.

1) Extension of lazy rule learning for numerical attributes: In this section we assume that $\mathbb{D}$ is a given decision system and $trnSet \subseteq \mathbf{X}$.

Now, we extend the definition of the local decision rule to the case of symbolic and numerical attributes.

Definition III.1. *Let tst be a test object and $trn \in trnSet$. By t_a for $a \in A_{sym}$ we denote a condition as in Definition II.10 and $min_a = min(a(tst), a(trn))$, $max_a = max(a(tst), a(trn))$ for $a \in A_{num}$ We define the* combined local decision rule *(for short* c-rule*) if $\bigwedge_{a \in A} T_a$ then $d = d(trn)$, denoted by $c\text{-}rule(tst, trn)$, where conditions T_a for $a \in A$ are as follows*

$$T_a = \begin{cases} a \in [min_a, max_a] & \text{if } a \in A_{num} \\ t_a & \text{if } a \in A_{sym}. \end{cases}$$

Let us note that conditions for numerical attributes contain intervals with endpoints determined by attribute values of objects tst and trn.

The relationship of the set $MaxRules(CombRules, trnSet)$ and $c\text{-}rule(tst, trn)$ is analogous to the relation between $MaxRules(SimRules, trnSet)$ and $s\text{-}rule(tst, trn)$ as the following lemma states.

Lemma III.1. *Any rule $r \in MaxRules(CombRules, trnSet)$ covering the given test object tst and training object trn is implied by the rule $c\text{-}rule(tst, trn)$.*

Proof. From $r \in MaxRules(CombRules, trnSet)$ we have, in particular that r is consistent with $trnSet$. Hence, the postcondition of r is $d = d(trn)$, i.e. the decision of r is the same as $c\text{-}rule(tst, trn)$.

Since r covers tst and trn, $t_a(r)(trn)$ and $t_a(r)(tst)$ are satisfied for each $a \in A$, i.e. $trn \in [[t_a(r)]]_{\mathbb{D}}$ and $tst \in [[t_a(r)]]_{\mathbb{D}}$.

It is enough to show that for any $a \in A$ the implication $t_a(c\text{-}rule(tst, trn)) \Rightarrow t_a(r)$ holds, i.e. $V_a(c\text{-}rule(tst, trn)) \subseteq V_a(r)$, where for any rule r' if $t_a(r')$ is of the form $a \in V$ then $V_a(r')$ is defined as $V_a(r') = ||V||_{\mathbb{D}}$.

Let us first assume that $a \in A_{sym}$. If $t_a(r)$ is of the form $a \in V_a$, then the implication obviously holds (trivial condition is implied by any condition, because for any elementary condition $a \in V$ for attribute a, $||V||_{\mathbb{D}} \subseteq ||V_a||_{\mathbb{D}}$. If $t_a(r)$ is for some $v \in V_a$ of the form $a = v$ (i.e. $a \in \{v\}$) then $v = a(trn) = a(tst)$. The last equalities hold because we already concluded that $t_a(r)(trn)$ and $t_a(r)(tst)$ are both satisfied. Hence, we have $trn \in [[a = v]]_{\mathbb{D}}$ and $tst \in [[a = v]]_{\mathbb{D}}$, i.e. $a(trn) \in \{v\}$ and $a(tst) \in \{v\}$. It means that in the considered case the equality $t_a(r) = t_a(c\text{-}rule(tst, trn))$ holds (see Definition III.1 and Definition II.10).

If $a \in A_{num}$ is numerical then $t_a(r)$ is of the form $a \in I$, where I is the description of interval corresponding to the numerical attribute a of rule r. Because $t_a(r)(trn)$ and $t_a(r)(tst)$ are both satisfied then $a(trn) \in ||I||_{\mathbb{D}}$ and $a(tst) \in ||I||_{\mathbb{D}}$. Thus $\{a(trn), a(tst)\} \subseteq ||I||_{\mathbb{D}}$. Hence, all points between $a(trn)$ and $a(tst)$ are also in $||I||_{\mathbb{D}}$. In consequence, $[min_a, max_a] \subseteq ||I||_{\mathbb{D}}$, where $min_a = min(a(tst), a(trn))$, $max_a = max(a(tst), a(trn))$ what ends the proof of inclusion $V_a(c\text{-}rule(tst, trn)) \subseteq V_a(r)$ (see Definition III.1). □

Theorem III.2. *The rule $c\text{-}rule(tst, trn)$ for the test object tst and the training object trn is consistent with the training set $trnSet$ if and only if there exists a rule $r \in MaxRules(CombRules, trnSet)$ covering objects tst and trn.*

Proof. We start from a proof of the following fact: if $c\text{-}rule(tst, trn)$ is consistent with $trnSet$ then it can be extended to a rule from $MaxRules(CombRules, trnSet)$. Such a rule can be constructed inductively. From assumption we have that $r_0 = c\text{-}rule(tst, trn) \in CombRules$ is consistent with $trnSet$. In the induction step to define each next rule r_i, for $i = 1, 2, \ldots, m$, where $m = |A|$, we assume that r_{i-1} is consistent with $trnSet$ and conditions $t_j(r_{i-1})$ for all $j = 1, 2, \ldots, i-1$ are maximally general, i.e. if we replace any condition t_j with a more general t (i.e. $t_j \Rightarrow t$) preserving consistency, then $t_j = t$.

$t_i(r_i)$, in i-th induction step, is defined as the maximal generalisation of $t_i(r_{i-1}) = \ldots = t_i(r_0) = t_i(c\text{-}rule(tst, trn))$ preserving consistency with $trnSet$. All others conditions and the decision of the rule are not changed, i.e. $t_j(r_i) = t_j(r_{i-1})$ for $j \neq i$; $d(r_i) = d(r_{i-1})$. Hene, in i-th induction step we simply maximally generalise condition for attribute a_i.

If $a_i \in A_{sym}$ and $t_i(r_{i-1})$ is the trivial condition, then we put $r_i = r_{i-1}$. If $t_i(r_{i-1})$ is non-trivial, it is substituted by $a \in V_a$ if the consistency of the rule is preserved; otherwise, we put $r_i = r_{i-1}$.

If $a_i \in A_{num}$ then $t_i(r_{i-1})$ is of the form $a_i \in [min, max]$. Let us denote by $rule_i(r, t)$ the result of replacement in r of i-th condition by a condition t. Now, we define a set of values of attribute a by $a(Inc) = \{a(trn) : trn \in Inc\}$, where $Inc = \{trn \in trnSet : d(trn) \neq d(r_0) \wedge rule_i(r_{i-1}, a_i = *)$ covers $trn\}$, i.e. Inc contains objects which may violate the consistency of the rule under the maximal possible extension of the condition $t_i(r_{i-1})$. From the inductive assumption, r_{i-1} is consistent with Inc because $Inc \subseteq trnSet$. Hence, $a(Inc) \cap [min, max] = \emptyset$. Now we define $newmax = min\{v \in a(Inc) : v > max\}$. This minimum exists because Inc and also $a(Inc)$ are finite sets. If the set $\{v \in a(Inc) : v > max\}$ is empty we take $newmax = u_{a_i}$ (i.e. maximal possible extension of the right end of the interval). Analogously, we define $newmin = max\{v \in a(Inc) : v < min\}$. If $\{v \in a(Inc) : v < min\}$ is empty we put $newmin = l_{a_i}$ (i.e. maximal possible extension of the left end of the interval). Finally, we define $t_i(r_i)$ by $a \in (newmin, newmax)$. From the definition, r_i is consistent with $trnSet$ and is also maximal because other ends of the interval $(newmin, newmax)$ even if extended by one point to a closed interval will cause inconsistency (in case $newmin = l_{a_i}$ this end of the interval cannot be extended; analogously in case $newmax = u_{a_i}$).

It is easy to prove that all other conditions $t_j(r_i)$ for $j < i$ remain still maximal. To prove this, let us assume that for some $j < i$ $t_j(r_i)$ could be extended to t with preserving consistency, i.e. $rule_j(r_i, t)$ is consistent. We also have $rule_j(r_{i-1}, t) \Rightarrow rule_j(r_i, t)$. Therefore $rule_j(r_{i-1}, t)$ is consistent with $trnSet$. From the inductive assumption t is identical with $t_j(r_{i-1})$. Because $t_j(r_{i-1})$ is the same as $t_j(r_i)$ for $(j < i)$, then t is the same as $t_j(r_i)$. It means that $t_j(r_i)$ is maximally general.

Our inductive reasoning leads to a conclusion that the last rule r_m is consistent with $trnSet$ and maximally general.

We have $c\text{-}rule(tst, trn) \Rightarrow r$ for any rule $r \in MaxRules(CombRules, trnSet)$ covering objects tst and trn (see Lemma III.1). Hence, inconsistency of $c\text{-}rule(tst, trn)$ implies inconsistency of all rules $r \in MaxRules(CombRules, trnSet)$ covering tst and trn. $\square$

2) Generalisation of lazy rule learning for symbolic attributes:
In the section by $\mathbb{D}$ we denote a given pseudometric decision system and $trnSet \subseteq \mathbf{X}$ is a given training set.

In the previous Definitions II.10 and III.1, the trivial condition $a \in V_a$ for a symbolic attribute a is introduced. This condition represents the specific grouping of all possible values of an attribute and is satisfied by any object. However, a proper subset of V_a may be more relevant for the classification. Grouping of values can be obtained by applying a given pseudometric ρ_a for a. Here, now we formulate the following generalisation of Definition III.1, related to a grouping of values for symbolic attributes:

Definition III.2. *Let tst be a test object, $trn \in trnSet$, and $min_a = min(a(tst), a(trn))$, $max_a = max(a(tst), a(trn))$ for $a \in A_{num}$. We also use the following notation: (i) $r_a = \varrho_a(a(tst), a(trn))$ for radius, (ii) $B(c, R)$ for closed pseudometric ball of radius R centred at point c defined by the pseudometric ϱ_a. Now, we define the* generalised local decision rule *(for short* g-rule*)* if $\bigwedge_{a\in A} t_a$ then $d = d(trn)$, *denoted by $g\text{-}rule\left(tst, trn, \{\varrho_a\}_{a\in A_{sym}}\right)$ or simply $g\text{-}rule\,(tst, trn)$ (if parameters $\{\varrho_a\}_{a\in A_{sym}}$ are clear from the context or irrelevant due to generality of considerations), where:*

$$t_a = \begin{cases} a \in [min_a, max_a] & \text{if } a \text{ is numerical} \\ a \in B\left(a(tst), r_a\right) & \text{if } a \text{ is symbolic.} \end{cases}$$

Now, we prove that an analogous relationship of the set $MaxRules(GenRules, trnSet)$ and $g\text{-}rule\,(tst, trn)$ (g-rule) to the relation between $MaxRules(SimRules, trnSet)$ and $s\text{-}rule(tst, trn)$ holds.

Lemma III.3. *Let tst be any test object and $trn \in trnSet$. Let $GenRules$ be defined by parameters ϱ_a (from given pseudometric decision system) and $c_a = a(tst)$ for $a \in A_{sym}$ (see Definition II.5), i.e. $GenRules = GenRules\left(\{(\varrho_a, a(tst))\}_{a\in A_{sym}}\right)$. Then $g\text{-}rule\left(tst, trn, \{\varrho_a\}_{a\in A_{sym}}\right) \Rightarrow r$ for any $r \in MaxRules(GenRules, trnSet)$ covering objects tst and trn.*

Proof. The proof is an extension of proof of Lemma III.1. For numerical attributes, the proof is the same as before. For symbolic attributes, it is enough to change the part of the proof of Lemma III.1 by the following one. Let $a \in A_{sym}$. Then $t_a(r)$ is of the form $a \in B(a(tst), R_a)$, where $R_a = \varrho_a(a(tst), v)$, for some $v \in V_a$. Hence, $R_a \geq \varrho_a(a(tst), a(trn))$ because $t_a(r)(trn)$ is satisfied. So, we obtain $B\left(a(tst), r_a\right) \subseteq B(a(tst), R_a)$, where $r_a = \varrho_a(a(tst), a(trn))$. Hence, we have $t_a(g\text{-}rule\,(tst, trn)) \Rightarrow t_a(r)$. $\square$

Theorem III.4. *Under the assumptions of Lemma III.3, the rule $g\text{-}rule\left(tst, trn, \{\varrho_a\}_{a\in A_{sym}}\right)$ is consistent with $trnSet$ if and only if there exists a rule $r \in MaxRules(GenRules, trnSet)$ such that r covers tst and trn.*

Proof. The proof can be obtained by a modification of the proof of Theorem III.2.

It is enough to modify the inductive step of the proof of Theorem III.2 for $a_i \in A_{sym}$ as follows. If $a_i \in A_{sym}$ then $t_i(r_{i-1})$ is of the form $a \in B(a(tst), r_a)$, where $r_a = \varrho_a(a(tst), v)$, for some $v \in V_a$. Now, let us consider possible extensions of $a \in B(a(tst), r_a)$ by $a \in B(a(tst), R_a)$, where $R_a = \varrho_a(a(tst), w)$, for some $w \in V_a$ and $R_a \geq r_a$ preserving consistency (with $trnSet$) of the rule. In the finite set of such possible extensions (due to the fact that V_a is finite) we select the one with the maximal value of R_a. The selected extension is maximally general.

If $a_i \in A_{num}$ then we extend the formula as in Theorem III.2.

One can conclude that the last rule r_m is consistent with $trnSet$ and maximally general by performing analogous reasoning as in Theorem III.2.

Also, in analogous way as in Theorem III.2 with the use of Lemma III.3, we obtain that the following implication holds: if $g\text{-}rule\,(tst, trn)$ is inconsistent with $trnSet$, then in $MaxRules(GenRules, trnSet)$ there is no rule covering tst and trn. $\square$

Let us note that the set $MaxRules(GenRules, trnSet)$ is defined for the given values c_a for $a \in A_{sym}$ (in the testing procedure we assume $c_a = a(tst)$). The idea behind construction of $MaxRules(SimRules, trnSet)$ was to compute all maximally general rules in advance for the later use in the classification process. In order to construct $MaxRules(GenRules, trnSet)$, this should be done for all possible combinations of all possible values for all symbolic attributes. It would increase the number of generated rules by the factor no more than b^k, where b is the maximal cardinality of $|V_a|$ for $a \in A_{sym}$ and k is the number of symbolic attributes.

From Theorem III.4 it follows that it is sufficient to generate g-rules for all training examples and then check their consistency with $trnSet$ (instead of computing the support sets for rules from $MaxRules(GenRules, trnSet)$ covering a new test case). The lazy Rule Induction Algorithm (RIA) realises this idea.

Algorithm 4: RIA(tst, $trnSet$, $\{\varrho_a\}_{a\in A_{sym}}$)

```
Input: test example $tst$, training set $trnSet$, family of
       pseudometrics for symbolic attributes $\{\varrho_a\}_{a\in A_{sym}}$
Output: predicted decision for $tst$
begin
    foreach decision $v \in V_d$ do
        $supportSet(v) = \emptyset$
    end
    foreach $trn \in trnSet$ do
        $v = d(trn)$
        if $isCons\left(g\text{-}rule\left(tst, trn, \{\varrho_a\}_{a\in A_{sym}}\right), trnSet\right)$ then
            $supportSet(v) = supportSet(v) \cup \{trn\}$
        end
    end
    return $\arg\max_{v\in V_d} |supportSet(v)|$
end
```

$isCons(r, verifySet)$ is defined in Algorithm 2 (see Subsection II-B). The RIA (see Algorithm 4) and LAZY (see Algorithm 3) algorithms differ only in line 7, namely in Algorithm 3 the rule $s\text{-}rule(tst, trn)$ is used and in Algorithm 4 this is the rule $g\text{-}rule\left(tst, trn, \{\varrho_a\}_{a\in A_{sym}}\right)$.

RIA computes the measure $Strength$ for $MaxRules = MaxRules(GenRules, trnSet)$ what directly follows from Theorem III.4. Hence, the results of RIA are equivalent to the results of the algorithm based on calculating $MaxRules$ with $Strength$ as a strategy for conflict resolution. In this way, we obtain the corollary analogous to Corollary II.2 (see Subsection II-B).

Corollary III.5. *For any test object tst, and the classifier $decision_{MaxRules}(tst)$ (see Eqn. 8), where*

$$MaxRules = MaxRules(GenRules, trnSet)$$

and

$$GenRules = GenRules\left(\{(\varrho_a, a(tst))\}_{a\in A_{sym}}\right),$$

we have

$$RIA(tst, trnSet, \{\varrho_a\}_{a\in A_{sym}}) = decision_{MaxRules}(tst).$$

B. Combining instance-based learning and rule methods – RIONA

In this section, we additionally assume that Agr is an aggregation function defined as sum of individual metrics.

Let us recall that RIONA is based on a combination of instance-based learning and rule-based methods. The primary observation used in the development of RIONA concerns the property of the widely used kNN method. kNN has quite good performance, usually for small values of k. Hence, one may expect that that only training examples close to a given test case are important in the process of inducing (inferring) the final decision. The intuition supporting this claim is that the training examples which are far from a given test object are less relevant for classification than the closer ones. Contrary to this, in the case of rule-based methods, in general, all training examples are used in the process of rule generation. Hence, instead of considering all training examples in constructing the support set in the case of rule-based approach, like in the RIA algorithm, one can bound it to a certain neighbourhood of a test example. In the case of RIONA algorithm, the classification of a given test case is based on training objects from a neighbourhood of this example.

Our approach to inducing of decision for a given test case is basing on a combination of instance-based learning and lazy rule learning (see Section III-A). The core idea concerns the strategy for conflict resolution based on $Strength$ measure (see Eqn. 7) slightly modified by bounding it to the neighbourhood of the test case:

$$\begin{aligned} &LocStrength(tst, v, k, \varrho) = \\ &= \left| \bigcup_{r\in MatchR(tst,v)} locSuppSet(r) \right|, \end{aligned} \tag{9}$$

where most notation is as in Eqn. 7; additionally $\varrho = Agr(\{\varrho_a\}_{a\in A})$ is the aggregated pseudometric and k is the number indicating the size of the neighbourhood, and $locSuppSet(r) = supportSet(r) \cap N(tst, trnSet, k, \varrho)$. One can observe that the change from $supportSet(r)$ to $locSuppSet(r)$ causes that only those examples covered by the rules matched by a test object that are in a specified neighbourhood of the test example are considered. The predicted decision based on $LocStrength$ is analogous to the previous one (see Eqn. 8):

$$\begin{aligned} &decLoc_{MaxRules}(tst, k, \varrho) = \\ &\arg\max_{v\in V_d} LocStrength(tst, v, k, \varrho). \end{aligned} \tag{10}$$

Let us note that the size k of the neighbourhood is optimised in the learning phase (see [5]) while in the classification process, we assume that number k for the neighbourhood $N(tst, k)$ is set to this optimal value.

The above measures can be calculated for a given $MaxRules$ by bounding the support sets of the rules from $MaxRules$ covering a test example to the specified neighbourhood of a given test example. Hence, the algorithm based on maximally general rules with $LocStrength$ can be used here.

It is worthwhile mentioning that $LocStrength$ can also be calculated using the lazy rule learning methodology. This can be done analogously to computing by RIA the measure $Strength$ (see Corollary III.5). For this purpose we modified Algorithm 4 as follows (i) in line 5 of the algorithm, only examples $trn \in N(tst, k)$ should be considered, (ii) it is not necessary to consider all the examples from the training set to check the consistency of the $g\text{-}rule\left(tst, trn, \{\varrho_a\}_{a \in A_{sym}}\right)$ (see line 7 of Algorithm 4). This follows from the next proposition.

Proposition III.6. *Suppose that ϱ_a (for $a \in A_{num}$) in a given pseudometric decision system are defined as normalised Euclean metric, $\varrho = Agr(\{\varrho_a\}_{a \in A})$ and Agr is defined either by sum of metrics or weighted sum of metrics. If $trn' \in trnSet$ satisfies g-rule $\left(tst, trn, \{\varrho_a\}_{a \in A_{sym}}\right)$, then $\varrho(tst, trn') \le \varrho(tst, trn)$.*

Proof. If trn' satisfies $g\text{-}rule\left(tst, trn, \{\varrho_a\}_{a \in A_{sym}}\right)$, then we have (see Definition III.2 of g-rule):

- for any $a \in A_{sym}$ we have $a(trn') \in B(a(tst), r_a)$, where $r_a = \varrho_a(a(tst), a(trn))$. Hence, from definition of the closed ball it follows that $\varrho_a(a(tst), a(trn')) \le \varrho_a(a(tst), a(trn))$.
- for any $a \in A_{num}$ we have $a(trn') \in [min_a, max_a]$, where $min_a = min(a(tst), a(trn))$, $max_a = max(a(tst), a(trn))$. Hence $|a(tst) - a(trn')| \le |a(tst) - a(trn)|$. Thus, using definiton of metric for numerical attributes (normalised Euclidean metric) we have $\varrho_a(a(tst), a(trn')) = \frac{|a(tst) - a(trn')|}{a^{\max} - a^{\min}} \le \frac{|a(tst) - a(trn)|}{a^{\max} - a^{\min}} = \varrho_a(a(tst), a(trn))$.

Hence, for any $a \in A$ we have $\varrho_a(a(tst), a(trn')) \le \varrho_a(a(tst), a(trn))$. In consequence, we obtain the following inequality between the global distances[4] for Agr defined by sum of metrics $\varrho(tst, trn') = \sum_{a \in A} \varrho_a(a(tst), a(trn')) \le \sum_{a \in A} \varrho_a(a(tst), a(trn)) = \varrho(tst, trn)$.

One can observe that we have the same result also for aggregation function defined by weighted sum of metrics. This is because adding weights for each attribute preserves the above inequality. □

From the above considerations it follows that the examples distanced from tst more than the training example trn cannot cause inconsistency of $g\text{-}rule\left(tst, trn, \{\varrho_a\}_{a \in A_{sym}}\right)$. Hence, one can use $N(tst, trnSet, k, \varrho)$ instead of $trnSet$ in line 7 of Algorithm 4.

The description of classification algorithm RIONA is presented in Algorithm 5. Later on we prove that Algorithm 5 computes $LocStrength$ (see Theorem IV.2). Algorithm 5 returns the most common class corresponding to decisions on the training examples covered by the rules satisfied by tst and belonging to the specified neighbourhood. One should note that all pseudometrics in the argument of Algorithm 5 are given (used for computation of the final pseudometric). However, in g-rule only pseudometrics for symbolic attributes are used (see Definition III.2 and note after it).

Algorithm 5: RIONA-classify(tst,$trnSet$,k,$\{\varrho_a\}_{a \in A}$)

```
Input: test example $tst$, training set $trnSet$, positive
       integer $k$, family of pseudometrics for attributes
       $\{\varrho_a\}_{a \in A}$
Output: predicted decision for $tst$
1  begin
2      $\varrho = Agr(\{\varrho_a\}_{a \in A})$
3      $nSet = N(tst, trnSet, k, \varrho)$
4      foreach decision $v \in V_d$ do
5          $supportSet(v) = \emptyset$
6      end
7      foreach $trn \in nSet$ do
8          $v = d(trn)$
9          if
             $isCons\left(g\text{-}rule\left(tst, trn, \{\varrho_a\}_{a \in A_{sym}}\right), nSet\right)$
           then
10             $supportSet(v) = supportSet(v) \cup \{trn\}$
11         end
12     end
13     return $\arg\max_{v \in V_d} |supportSet(v)|$
14 end
```

For every decision value, RIONA computes the support set restricted to the neighbourhood $N(tst, k)$ rather than the whole support set of the maximally general rules covering tst (as in the case of RIA algorithm). This is done as follows. For any $trn \in trnSet$ from $N(tst, k)$ RIONA constructs the rule $g\text{-}rule\left(tst, trn, \{\varrho_a\}_{a \in A_{sym}}\right)$ based on trn and tst. Next, RIONA is testing whether g-rule is consistent with the examples from the neighbourhood $N(tst, k)$. If g-rule is consistent with $N(tst, k)$ then the support set of the decision $d(trn)$ is extended by trn. Finally, RIONA returns the decision value with the support set of the highest cardinality.

IV. Relationships of RIONA to other approaches

A specific combination of kNN approach and lazy rule induction allowed us to develop the algorithm RIONA. One can observe that only the line 9 of RIONA, where is examined the consistency of the rule determined by the training and testing example, differs from Algorithm 1 (kNN).

The relationships between RIONA, RIA and kNN for $k = 1$ are as follows.

Proposition IV.1. *Let us assume that* 1NN *is the nearest neighbour algorithm for $k = 1$ with a distance defined by pseudometric $\varrho = Agr(\{\varrho_a\}_{a \in A})$. Then for any test object tst we have*

$$RIONA(tst, trnSet, k, \{\varrho_a\}_{a \in A}) = \begin{cases} RIA(tst, trnSet, \{\varrho_a\}_{a \in A_{sym}}) \text{ for } k \ge |trnSet| \\ 1NN(tst, trnSet, \varrho) \text{ for } k = 1, |N(tst, trnSet, k, \varrho)| = 1. \end{cases}$$

[4]We assume that pseudometrics used for grouping symbolic attributes are the same as the pseudometrics composing the aggregated pseudometric used for measuring distance between examples. The analogous assumption is used for numerical attributes: real values are grouped using interval contained in the ball $B(a(tst), \varrho_a(a(tst), a(trn)))$ determined by the Euclidean metric. The same Euclidean metric (however normalised) is used for components of the final pseudometric.

Proof. If $k \geq |trnSet|$ then the $neighbourSet = trnSet$, where $neighbourSet$ is defined in the RIONA algorithm (see Algorithm 5). Hence, RIONA works exactly as the RIA algorithm (see Algorithm 4).

If $k = 1$ and $|N(tst, trnSet, 1, \varrho)| = 1$[5] then the $neighbourSet$ in the RIONA algorithm is a singleton set and consistency checking can be omitted. Hence, RIONA works exactly as 1NN (see Algorithm 1). $\square$

RIONA behaves like the RIA algorithm for the maximal neighborhood (and from the Corollary III.5 as the algorithm based on the maximally general rules with the $Strength$ strategy for conflict resolution). By taking a neighbourhood based on the one nearest training example, the nearest neighbour algorithm is obtained. RIONA is positioned between the rule-based classifier based on the maximally general rules and the nearest neighbour classifier. If a small neighborhood is chosen then it acts more like a kNN classifier and if a large neighbourhood is chosen it works more like a rule-based classifier based on inducing maximally general rules. Selection of a neighbourhood that is not the maximal may be interpreted as taking more specific rules instead of maximally general rules consistent with the training examples.

Below, we present more properties of RIONA.

Theorem IV.2. *The following equality holds:*
$RIONA(tst, trnSet, k, \{\varrho_a\}_{a\in A}) =$
$decLoc_{MaxRules}(tst, k, \varrho)$,
where

(i) *tst is any test object,*
(ii) *$decLoc_{MaxRules}(tst, k, \varrho)$ is the output of classifier from Eqn. 10 with*
   - $MaxRules = MaxRules(GenRules, trnSet)$,
   - $GenRules = GenRules\left(\{(\varrho_a, a(tst))\}_{a\in A_{sym}}\right)$,
   - $\varrho = Agr(\{\varrho_a\}_{a\in A})$.

Proof. We have $RIA(tst, trnSet, \{\varrho_a\}_{a\in A_{sym}})$ $= decision_{MaxRules}(tst)$ (see Corollary III.5). This is based on the following fact: RIA computes measure $Strength$ for $MaxRules = MaxRules(GenRules, trnSet)$, i.e. for each $v \in V_d$ $supportSet(v)$ (in line 11 of Algorithm 4) $=$ $Strength(v)$ (see Theorem III.4). RIONA works only on training examples from $N(tst, trnSet, k, \varrho)$ and examples consistent with $N(tst, trnSet, k, \varrho)$ are also consistent with the whole training set, $trnSet$ (see Proposition III.6). In computing of the measure $LocStrength(tst, v, k, \varrho)$ are used only training examples from the neighbourhood $N(tst, trnSet, k, \varrho)$. Hence, $supportSet(v)$ (see line 13 of Algorithm 5) $=$ $LocStrength(tst, v, k, \varrho)$ for any $v \in V_d$. From this the equation of the theorem follows. $\square$

Theorem IV.3. *The following equality holds:*
$RIONA(tst, trnSet, k, \{\varrho_a\}_{a\in A}) =$
$decLoc_{MaxRules}(tst, k, \varrho)$,
where

(i) *tst is any test object,*
(ii) *$decLoc_{MaxRules}(tst, k, \varrho)$ is the output of classifier from Eqn. 10 with*
   - $MaxRules =$ $MaxRules(GenRules, N(tst, trnSet, k, \varrho))$,
   - $GenRules = GenRules\left(\{(\varrho_a, a(tst))\}_{a\in A_{sym}}\right)$,
   - $\varrho = Agr(\{\varrho_a\}_{a\in A})$.

Proof. The following equality holds (see Theorem IV.2):
$RIONA(tst, N(tst, trnSet, k, \varrho), k, f) =$
$decLoc_{MaxRules}(tst, k, \varrho)$,
where $trnSet$ is substituted by a new training set $N(tst, trnSet, k, \varrho))$,
and
$f = \{\varrho_a\}_{a\in A}$,
$MaxRules = MaxRules(GenRules, N(tst, trnSet, k, \varrho))$.
We also have
$RIONA(tst, N(tst, trnSet, k, \varrho), k, f) =$
$RIONA(tst, trnSet, k, f)$
what ends the proof. $\square$

From the last two theorems the following interesting corollary follows.

Corollary IV.4. *Let us assume that there are given* $\{\varrho_a\}_{a\in A}$, $\varrho = Agr(\{\varrho_a\}_{a\in A})$, $trnSet$, $MaxRules = MaxRules(GenRules, trnSet)$, $MaxLocalRules =$ $MaxRules(GenRules, N(tst, trnSet, k, \varrho))$, *where* $GenRules = GenRules\left(\{(\varrho_a, a(tst))\}_{a\in A_{sym}}\right)$. *Then the outputs returned by the following classifiers are the same for any tst object:*

1. $RIONA(tst, trnSet, k, \{\varrho_a\}_{a\in A})$,
2. $decLoc_{MaxRules}(tst, k, \varrho)$,
3. $decLoc_{MaxLocalRules}(tst, k, \varrho)$,
4. $decision_{MaxLocalRules}(tst)$ *with a new training set* $trnSet' = N(tst, trnSet, k, \varrho)$.

Proof. From Theorems IV.2 and IV.3 it follows the equivalence of the first three classifiers. To obtain the equivalence of the third and fourth classifiers let us observe that
$trnSet' = N(tst, trnSet, k, \varrho) =$
$N(tst, N(tst, trnSet, k, \varrho), k, \varrho) = N(tst, trnSet', k, \varrho)$.
We also have $supportSet(r) \subseteq trnSet'$. Hence, from the previous equation we obtain $supportSet(r) \subseteq N(tst, trnSet', k, \varrho)$. Then $locSuppSet(r) =$ $supportSet(r) \cap N(tst, trnSet', k, \varrho) = supportSet(r)$.
Now one can see that Eqn. 9 becomes Eqn. 7, what implies that Eqn. 10 becomes Eqn. 8. $\square$

Summarising, the conclusions are the following: (i) RIONA calculates the $LocalStrength$ measure (see Eqn. 9). (ii) The $LocStrength$ measure is the $Strength$ measure with $N(tst, k)$ as the local training set (fourth algorithm). (iii) The algorithm presented in Eqn. 8 after substitution of a

[5]This assumption can't be omitted. If $|N| > 1$ then it may happen (even for consistent training set) that N has (equally distanced from test example) two cases with different decisions leading to inconsistency.

new training set $trnSet' = N(tst, trnSet, k, \varrho)$ instead of $trnSet$ becomes the fourth algorithm. (iv) One can consider the RIONA algorithm as an algorithm for computing all maximally general, consistent rules locally and using (locally) $Strength$ for conflict resolution.

In Table I and Table II is presented comparison of these algorithms (the third algorithm is omitted because it is very similar to the fourth).

Table I
A GENERAL COMPARISON OF THREE ALGORITHMS FROM COROLLARY IV.4: ALGORITHM (1) RIONA, ALGORITHM (2) BASED ON THE MEASURE $LocStrength$ AND ALGORITHM (4) BASED ON THE MEASURE $Strength$ COUNTED LOCALLY.

RIONA	algorithm (2) based on the measure $LocStrength$	algorithm (4) based on the measure $Strength$ counted locally
	counting rules	
no need to count rules explicitly	counts $MaxRules$ globally once at the beginning	counts $MaxRules$ locally for each test case
	counting support	
counts support using lazy local rules	counts support locally	counts support locally

Table II
A COMPARISON SCHEME OF THREE ALGORITHMS FROM COROLLARY IV.4: ALGORITHM (1) RIONA, ALGORITHM (2) BASED ON THE MEASURE $LocStrength$ AND ALGORITHM (4) BASED ON THE MEASURE $Strength$ COUNTED LOCALLY.

RIONA	algorithm (2) based on the measure $LocStrength$	algorithm (4) based on the measure $Strength$ counted locally
Global input: $trnSet$, $k \in \mathbb{N}$		
1.	count $MaxRules$ for $trnSet$	
Input: test case tst		
2. $nSet = N(tst, k)$		
3.	$RuleBase = MaxRules$	count (locally) $MaxRules(nSet)$ $RuleBase = MaxRules(nSet)$
4.	consider rules from $RuleBase$ with premise satisfied by tst	
5. for each decision d		
6. consider $trn \in nSet$ with decision d	consider rules from step 4 with decision d	
7. count the number of trn from step 6 forming consistent rules with tst	count the number of $trn \in nSet$ supporting rules from step 6	
8. choose the decision with the maximal count (maximally supported)		

A. RIONA and rules

Some important properties of instance-based classifiers and rule-based classifiers are inherited by the RIONA algorithm. Even though rule-based classifiers produce less accurate classifications, there are several features of them that users prefer over instance-based classifiers. The ability for a human, non-computer science professional, to interpret rules is one of these crucial features. He or she can check to see if the information found in such rules is non-trivial, accurate, and revealing brand-new features of the considered case. A rule includes an explanation for making the specific decision that is simple enough for a human to comprehend.

Here, we assume that the RIONA algorithm's parameter k is fixed (potentially learnt [5]). Let's now concentrate on algorithm (4) from Sect. III. Because the local complete set of consistent and maximally general decision rules must be computed for each test case tst, the direct computation of $MaxLocalRules$ may initially appear to be highly expensive and impractical. However, if we assume that the size of N is k, then the size of the local training sample is much smaller than the size of the entire training sample being reduced from $n = |trnSet|$ to k. As a result, the total cost of computing $MaxRules$ (globally or locally) is decreased from $O(2^n)$ to $O(m \cdot 2^k)$, where m is the number of test cases. We don't just present this strategy from a theoretical standpoint only. When a classifier's decision needs to be explained, this kind of method might be useful. In this way, the RIONA algorithm shares characteristics with rule algorithms and quick lazy learning algorithms, i.e. its parameters can be converted into rules.

Additionally, algorithm (4) might be extended to construct all rules globally once at the beginning, analogously to algorithm (2) from Corollary IV.4, except that the rules would be based on the local neighborhood only. Such rules would mimic the RIONA algorithm's behavior. The use of such a strategy has several benefits. First, a set of rules could be immediately provided to explain the predicted decision on a particular test object. Second, the usefulness of the knowledge acquired might be tested against all potential rules generated at the beginning.

The approach for construction of these rules is analogous as in algorithm 4 from Corollary IV.4. One could just construct $MaxRules$ locally for each training case and use each training example as a test example. It could be seen as a computation of specific local reducts i.e. reducts constructed during generation of maximally general rules for a given object (see e.g. [33], [34], [35], [32]). Usually, in construction of local reducts one should be aware to keep discernibility for objects with various decisions. Only objects with different decisions and at a distance of no more than determined by k would be required to be discernible in this case.

V. CONCLUSION

The presented findings indicate some important relationships of classifiers generated by the RIONA learning algorithm with instance- and rule-based classifiers. For example, it is proved the relative to classification equivalence of the RIONA algorithm to the algorithm generating all consistent and maximally general rules from a training set including the close training cases to a given test case. As a result, the classification by RIONA classifier can be performed by a relatively small set of rules that are simple for a person to comprehend. It might be applied in circumstances where it's crucial to provide an explanation for the decision that was reached by classifier. Finally, it is worthwhile to mention that the RIONA algorithm, based on hybridization of instance- and rule-based techniques, has the following properties (i) it is efficient as well as effective from the point of view of classification, (ii) it can be used as a high quality tool in the process of explanation of the predicted decisions.

References

[1] S. J. Russell and P. Norvig, *Artificial Intelligence: A Modern Approach*, 4th ed. Hoboken, NJ: Pearson Education, 2021.

[2] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*, 2nd ed. New York, NY: Springer, 2009. [Online]. Available: https://doi.org/10.1007/978-0-387-84858-7

[3] T. M. Mitchell, *Machine Learning*. New York, NY: McGraw-Hill, 1997.

[4] G. Góra and A. Wojna, "RIONA: A New Classification System Combining Rule Induction and Instance-Based Learning," *Fundamenta Informaticae*, vol. 51, no. 4, pp. 369–390, 2002. [Online]. Available: https://doi.org/10.5555/2371138.2371141

[5] ——, "RIONA: A Classifier Combining Rule Induction and K-nn Method with Automated Selection of Optimal Neighbourhood," in *Proceedings of the 13th European Conference on Machine Learning (ECML 2002)*. Heidelberg: Springer-Verlag, 2002, pp. 111–123. [Online]. Available: https://doi.org/10.1007/3-540-36755-1_10

[6] G. Góra, "Combining instance-based learning and rule-based methods for imbalanced data," Ph.D. dissertation, University of Warsaw, Warsaw, 2022, [Online]. Available: https://www.mimuw.edu.pl/sites/default/files/gora_grzegorz_rozprawa_doktorska.pdf.

[7] J. Fürnkranz, D. Gamberger, and N. Lavrac, *Foundations of Rule Learning*, ser. Cognitive Technologies. Heidelberg: Springer, 2012. [Online]. Available: https://doi.org/10.1007/978-3-540-75197-7

[8] A. Skowron and D. Ślęzak, "Rough sets turn 40: From information systems to intelligent systems," in *Proceedings of the 17th Conference on Computer Science and Intelligence Systems, FedCSIS 2022, Sofia, Bulgaria, September 4-7, 2022*, ser. Annals of Computer Science and Information Systems, M. Ganzha, L. A. Maciaszek, M. Paprzycki, and D. Ślęzak, Eds., vol. 30, 2022, pp. 23–34. [Online]. Available: https://doi.org/10.15439/2022F310

[9] C. C. Aggarwal, "Instance-Based Learning: A Survey," in *Data Classification: Algorithms and Applications*, 1st ed., C. C. Aggarwal, Ed. New York: Chapman & Hall/CRC, 2014, pp. 157–186. [Online]. Available: https://doi.org/10.1201/b17320

[10] C. Cornelis, "Hybridization of fuzzy sets and rough sets: Achievements and opportunities," in *Proceedings of the 17th Conference on Computer Science and Intelligence Systems, FedCSIS 2022, Sofia, Bulgaria, September 4-7, 2022*, ser. Annals of Computer Science and Information Systems, M. Ganzha, L. A. Maciaszek, M. Paprzycki, and D. Ślęzak, Eds., vol. 30, 2022, pp. 7–14. [Online]. Available: https://doi.org/10.15439/2022F302

[11] A. Adadi and M. Berrada, "Peeking Inside the Black-Box: A Survey on Explainable Artificial Intelligence (XAI)," *IEEE Access*, vol. 6, pp. 52 138–52 160, 2018. [Online]. Available: https://doi.org/10.1109/ACCESS.2018.2870052

[12] F. K. Došilović, M. Brčić, and N. Hlupić, "Explainable artificial intelligence: A survey," in *2018 41st International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO)*. Croatian Society MIPRO, 2018, pp. 0210–0215. [Online]. Available: https://doi.org/10.23919/MIPRO.2018.8400040

[13] H. Hagras, "Toward Human-Understandable, Explainable AI," *Computer*, vol. 51, no. 9, pp. 28–36, 2018. [Online]. Available: https://doi.org/10.1109/MC.2018.3620965

[14] D. W. Aha, Ed., *Lazy Learning*, 1st ed. Dordrecht: Springer, 1997. [Online]. Available: https://doi.org/10.1007/978-94-017-2053-3

[15] A. Wojna and R. Latkowski, "Rseslib 3: Library of Rough Set and Machine Learning Methods with Extensible Architecture," in *Transactions on Rough Sets XXI*, J. F. Peters and A. Skowron, Eds. Berlin, Heidelberg: Springer, 2019, pp. 301–323.

[16] N. Dey, S. Borah, R. Babo, and A. S. Ashour, Eds., *Social Network Analytics: Computational Research Methods and Techniques*, 1st ed. London: Academic Press, 2019. [Online]. Available: https://doi.org/10.1016/C2017-0-02844-6

[17] L. Grama and C. Rusu, "Choosing an accurate number of mel frequency cepstral coefficients for audio classification purpose," in *Proceedings of the 10th International Symposium on Image and Signal Processing and Analysis (ISPA 2017)*, 2017, pp. 225–230. [Online]. Available: https://doi.org/10.1109/ISPA.2017.8073600

[18] R. de Oliveira Almeida and G. T. Valente, "Predicting metabolic pathways of plant enzymes without using sequence similarity: Models from machine learning," *The Plant Genome*, vol. 13, no. 3, p. e20043, 2020. [Online]. Available: https://doi.org/10.1002/tpg2.20043

[19] C. Rusu and L. Grama, "Recent developments in acoustical signal classification for monitoring," in *2017 5th International Symposium on Electrical and Electronics Engineering (ISEEE)*, 2017, pp. 1–10. [Online]. Available: https://doi.org/10.1109/ISEEE.2017.8170705

[20] L. Grama and C. Rusu, "Adding audio capabilities to TIAGo service robot," in *2018 International Symposium on Electronics and Telecommunications (ISETC)*, 2018, pp. 1–4. [Online]. Available: https://doi.org/10.1109/ISETC.2018.858389

[21] A. Almasri, E. Celebi, and R. S. Alkhawaldeh, "EMT: Ensemble Meta-Based Tree Model for Predicting Student Performance," *Scientific Programming*, vol. 2019, pp. Article No. 3 610 248, 1–13, 2019. [Online]. Available: https://doi.org/10.1155/2019/3610248

[22] N. R. Howes, *Modern Analysis and Topology*. New York: Springer Science+Business Media, 1995. [Online]. Available: https://doi.org/10.1007/978-1-4612-0833-4

[23] P. Domingos, "Unifying instance-based and rule-based induction," *Machine Learning*, vol. 24, no. 2, pp. 141–168, 1996. [Online]. Available: https://doi.org/10.1007/BF00058656

[24] H. S. Nguyen and A. Skowron, "Quantization of Real Value Attributes – Rough Set and Boolean Reasoning Approach," in *Proceedings of the 2nd Joint Annual Conference on Information Sciences (JCIS 1995)*, 1995, pp. 34–37.

[25] A. Skowron and C. Rauszer, "The Discernibility Matrices and Functions in Information Systems," in *Intelligent Decision Support: Handbook of Applications and Advances of the Rough Sets Theory*, R. Słowiński, Ed. Dordrecht: Springer, 1992, pp. 331–362. [Online]. Available: https://doi.org/10.1007/978-94-015-7975-9_21

[26] C. Sammut and G. I. Webb, Eds., *Encyclopedia of Machine Learning and Data Mining*, 2nd ed. USA: Springer, 2017. [Online]. Available: https://doi.org/10.1007/978-1-4899-7687-1

[27] H. S. Nguyen, "Approximate Boolean Reasoning: Foundations and Applications in Data Mining," in *Transactions on Rough Sets V*, J. F. Peters and A. Skowron, Eds. Heidelberg: Springer, 2006, pp. 334–506. [Online]. Available: https://doi.org/10.1007/11847465_16

[28] J. G. Bazan and M. Szczuka, "RSES and RSESlib – A Collection of Tools for Rough Set Computations," in *Rough Sets and Current Trends in Computing (RSCTC 2001)*. Heidelberg: Springer, 2001, pp. 106–113. [Online]. Available: https://doi.org/10.1007/3-540-45554-X_12

[29] J. W. Grzymala-Busse, "LERS-A System for Learning from Examples Based on Rough Sets," in *Intelligent Decision Support: Handbook of Applications and Advances of the Rough Sets Theory*, R. Słowiński, Ed. Dordrecht: Springer, 1992, pp. 3–18. [Online]. Available: https://doi.org/10.1007/978-94-015-7975-9_1

[30] S. H. Nguyen, "Regularity Analysis and its Applications in Data Mining," in *Rough Set Methods and Applications: New Developments in Knowledge Discovery in Information Systems*, L. Polkowski, S. Tsumoto, and T. Y. Lin, Eds. Heidelberg: Physica-Verlag, 2000, pp. 289–378. [Online]. Available: https://doi.org/10.1007/978-3-7908-1840-6_7

[31] R. S. Michalski, I. Mozetic, J. Hong, and N. Lavrac, "The Multi-Purpose Incremental Learning System AQ15 and Its Testing Application to Three Medical Domains," in *Proceedings of the 5th AAAI National Conference on Artificial Intelligence*. AAAI Press, 1986, pp. 1041–1045.

[32] J. G. Bazan, "Discovery of Decision Rules by Matching New Objects Against Data Tables," in *Rough Sets and Current Trends in Computing (RSCTC 1998)*. Heidelberg: Springer, 1998, pp. 521–528. [Online]. Available: https://doi.org/10.1007/3-540-69115-4_72

[33] Z. Pawlak and A. Skowron, "A Rough Set Approach to Decision Rules Generation," in *Proceedings of the Workshop W12: The Management of Uncertainty at the 13th International Joint Conference on Artificial Intelligence (IJCAI 1993)*. Chambéry: Morgan Kaufmann, 1993, pp. 114–119.

[34] J. Wróblewski, "Covering with Reducts – A Fast Algorithm for Rule Generation," in *Rough Sets and Current Trends in Computing (RSCTC 1998)*. Heidelberg: Springer, 1998, pp. 402–407. [Online]. Available: https://doi.org/10.1007/3-540-69115-4_72

[35] J. G. Bazan, H. S. Nguyen, S. H. Nguyen, P. Synak, and J. Wróblewski, "Rough Set Algorithms in Classification Problem," in *Rough Set Methods and Applications: New Developments in Knowledge Discovery in Information Systems*, L. Polkowski, S. Tsumoto, and T. Y. Lin, Eds. Heidelberg: Physica-Verlag, 2000, pp. 49–88. [Online]. Available: https://doi.org/10.1007/978-3-7908-1840-6_3