Extending Rule-Based Classifiers to Improve Recognition of Imbalanced Classes

Jerzy Stefanowski and Szymon Wilk

1 Introduction

Knowledge discovery in general, and data mining in particular, have received a growing interest both from research and industry in recent years. Its main aim is to look for previously unknown relationships or patterns representing knowledge hidden in real-life data sets [16]. The typical representations of knowledge discovered from data are: associations, trees or rules, relational logic clauses, functions, clusters or taxonomies, or characteristic descriptions of concepts [16, 29, 21]. In this paper we focus on the rule-based representation. More precisely, we are interested in *decision* or *classification rules* that are considered in *classification* problems. In data mining other types of rules are also considered, e.g., association rules or action rules [16, 29, 34], however, in the text hereafter we will use the general term "rules" to refer specifically to decision rules.

Rules represent functions mapping examples (objects), described by a set of *attributes* (features) to *decision classes* (concepts) and they are expressed in the form: **if** P **then** Q, where P is the *condition part* formed as a conjunction of elementary conditions – tests on values of attributes, and Q is the *decision part* of the rule, which indicates the assignment of an example satisfying the condition part to a specific decision class.

The rule-based representation due to its symbolic expressiveness is considered to be more comprehensible and human-readable than other representations (see discussions in [28, 29, 33]). Although, such characteristic is shared by the tree-based representation, a set of rules may be more compact than a decision tree [33]. Moreover, rules constitute "blocks" of knowledge, and experts can more easily analyze individual rules [28, 39]. Finally, rules were successfully used in several applications as demonstrated in a review paper by Simon and Langley [23].

Induction of rules has been intensively studied in machine learning [29, 31] and many algorithms have been proposed, for reviews see [10, 31, 29]. The majority of them try to generate rules following a *sequential covering* strategy. They are focused

Jerzy Stefanowski and Szymon Wilk
Institute of Computing Science, Poznań University of Technology,
ul. Piotrowo 2, 60–965 Poznań, Poland
e-mail: jerzy.stefanowski@cs.put.poznan.pl,
szymon.wilk@cs.put.poznan.pl

on creating a *minimal set of rules*, which means that learning examples are covered by the smallest number of non-redundant rules.

Sets of rules induced from learning examples are usually applied to *predict class labels* for new examples. In such a classification-oriented perspective, rules and a strategy of using them constitute a *classifier*.

Although sequential covering algorithms have been shown to be quite effective from this perspective, there are also other algorithms which provide “richer” sets of rules. Such rules are also often characterized by better descriptive properties, e.g., they are supported by a larger number of learning examples (see [42, 16]). In general they could better characterize some regularities hidden in data, what corresponds to so-called descriptive perspective of knowledge discovery [50].

On the other hand, such rules could be also useful for handling more difficult classification problems. One of the main reasons for difficulty is *class imbalance* in learning data, i.e., a situation when one class (further called the *minority class*) includes much smaller number of examples comparing to other *majority* classes. The minority class is usually of primary interest in a given problem and it is required to recognize its members as accurately as possible. The imbalanced distribution of classes constitutes a difficulty for standard learning algorithms because they are biased toward the majority classes. As a result examples from the majority classes are more likely to be classified correctly by created classifiers, whereas examples from the minority class tend to be misclassified.

The problem of dealing with the class imbalance receives a growing research interest in machine learning and data mining communities (for a review see [3]). Several methods have been proposed to improve performance of various types of classifiers, not only rule-based ones. In general, one can distinguish two kinds of approaches [20]. The first approach, which is classifier-independent, relies on transforming an original data set to change the balance between classes, e.g., by re-sampling . The second approach involves modifying classifiers in order to improve their sensitivity to the minority class.

In the paper we discuss how rule-based classifiers can be adopted to deal with imbalance in the learning set. We consider two ways of doing it – either by modifying a classification strategy, or by using a different approach to induce rules for the minority class. The main aim of this study is to present a new method of extending the structure of a rule-based classifier in order to improve its sensitivity to the minority class. The main principle of the proposed method is that a minimal set of rules for the minority class is replaced by a new set of stronger rules. Such rules are discovered by a special algorithm, called EXPLORE, which was previously introduced by Stefanowski and Vaderpooten in [42]. Thus, using such rules for the minority class, while preserving the original minimal set of rules for the majority classes, improves the chance that an example from the minority class is correctly recognized.

Within our approach we do not only preserve the comprehensible representation of decision knowledge for the minority class, but we even make it more comprehensive by discovering additional rules, still hidden in data, which have not been revealed in a minimal set. Discussing the usefulness of the EXPLORE algorithm for providing such comprehensible patterns is the second goal of this paper. Maintaining comprehensible representation of knowledge is consistent with arguments behind rule paradigms. It further distinguishes our proposal from other known approaches, which also aim at improving the minority class prediction, however, at the cost of making extensive changes in data.

Finally, we present results of experiments where the performance of our approach is compared against LEM2 – a typical sequential covering algorithm, and its modification for handling imbalanced data on several benchmark data sets.

The paper is organized as follows. We begin with a brief review of two main categories of rule induction algorithms and classification strategies. Then, in Section 3 we describe the EXPLORE algorithm, which is employed by our approach. Our previous experience with using this algorithm is also reported. The next section contains a short review of methods for handling imbalanced data. Then, we present our approach to changing the structure of a rule-based classifier. In Section 6 we experimentally evaluate its usefulness in a comparative study. Final remarks and discussion on future research are provided in the last section.

We would like to note that this paper is a summary, which includes partial results from other papers by Stefanowski and coauthors on the EXPLORE algorithms [42] and from our joint research with Grzymala-Busse on handling imbalanced data by modifying rule-based classifiers [15].

2 Approaches to Rule Induction

This chapter gives basic information on rule induction and classification strategies which are necessary for presenting our method. More comprehensive descriptions can be found in [10, 12, 21, 39]

2.1 Basic Notation

For classification problems data sets include examples described by attributes and assigned to decision classes. We assume that these examples are represented in a *decision table* $DT = (U, A \cup \{d\})$, where U is a set of examples, A is a set of condition attributes describing them, and $d \notin A$ is a decision categorical attribute that partitions examples into a set of disjoint decision classes $\{K_j : j = 1,...,k\}$. A decision rule r assigning examples to a class K_j is represented in the following form:

$$\textbf{if } P \textbf{ then } Q,$$

where $P = p_1 \wedge p_2 \wedge \ldots \wedge p_n$ is the *condition part* of r, and Q is the *decision part* of r indicating that an example should be assigned to a class K_j. The condition part is a conjunction of elementary conditions p_i. Each condition represents a test on a value of a corresponding attribute. For a symbolic attribute the test compares its value to a constant, and for a numerical attribute other relations (e.g., greater than) are possible.

A decision table DT contains *learning* examples for inducing rules, therefore, it is called a *learning* set. For a given class K_j, learning examples from this class are called its *positive* examples, while examples belonging to the remaining classes are called *negative examples* of K_j .

Using these terms we briefly present some definitions of basic rule properties. $[P]$ is a *cover* of the condition part of a rule r in DT, i.e., it is a set of examples, which descriptions (values of condition attributes) satisfy elementary conditions in P. Let $[K_j]$ be a set of positive examples of a class K_j. A rule r is *discriminant* (also called *certain* or *consistent*) if it distinguishes positive examples of K_j from its negative examples, i.e., $[P] \subseteq [K_j]$. Moreover, P should be a minimal conjunction of elementary conditions satisfying this requirement.

A set of decision rules R *completely* covers (describes) all positive examples of a class K_j, if each positive example is covered by at least one decision rule from R. Moreover, if there is no other $R' \subset R$ that covers all positive examples of K_j, we say that R is the minimal cover of K_j. In other words it completely describes positive examples of this class by the smallest number of rules.

If the learning set contains noisy or inconsistent examples, also so-called *partially discriminant* or *possible* rules can be constructed. Besides positive examples such rules cover a limited number of negative examples.

2.2 Perspectives of Rule Induction and Evaluation

In general induction of decision rules can be performed according to different perspectives. The most common ones are [39, 50]:

- classification-oriented induction,
- descriptive-oriented induction.

The aim of the *classification-oriented induction* is to create from learning examples a set of rules which will be further used to *classify* new objects. Rules are then combined with a strategy defining how to use them to produce the final prediction for a new object – such a combination constitutes a *classifier*. This perspective has been extensively studied in machine learning and several approaches for deriving *rule-based classifiers* have been proposed.

The aim of the *descriptive-oriented induction* is to *extract* from learning examples information patterns (regularities or sometimes exceptions or anomalies) which may be *interesting* and *useful* for different users [16]. These patterns (represented as *rules*) aim at clarifying dependencies between values of attributes and decision classes [42] and usually are much more comprehensive than rules created following the classification-oriented perspective. The descriptive-oriented induction has been conceived and considered within the field of knowledge discovery, however, there has been successful research on building classifiers using rules constructed according to this approach [16, 39].

The two perspectives of rule induction do not only have different goals – there are other profound differences between them. One of the main distinctions consists in different evaluation criteria [42] of constructed rules. In the classification-oriented induction, a *complete set of rules is evaluated* as a classifier. An evaluation criterion is usually single and defined as the classification (predictive) accuracy or similar prediction measure (e.g., based on a confusion matrix – see Section 4) of a rule-based classifier using these rules. This criterion is evaluated in an experimental and automatic way.

In the descriptive-oriented induction, *each rule is evaluated individually and independently* as possible representation of an interesting pattern, which is definitely a more difficult task. Depending on a rule induction algorithm, the user may obtain quite a large number of rules to interpret. Selecting some of them is a non-trivial issue, it is also partly subjective as it generally depends on the problem at hand and on interests and expertise of users. To support the selection, several *quantitative measures* (also called *interestingness* measures) have been proposed and studied, each capturing different characteristic of rules. Many of these measures characterize relationships between the condition and the decision parts of a rule and a data set, from which the rule has been discovered. Generality, support, confidence, logical sufficiency or necessity are examples of widely approved and used measures. Their systematic review is available, e.g., in [17]. Below we present two of the most commonly used measures, i.e. *support* and *confidence* of a rule.

The support of the condition part P, denoted as $sup(P)$, is equal to the number of examples in U satisfying P, i.e., its equal to $\mid [P] \mid$, where where $\mid . \mid$ denotes the cardinality of a set. In a similar way we define the support of the decision part Q and denote it as $sup(Q) = \mid [Q] \mid$.

The support of a rule r denoted as $sup(r)$, is equal to the number of objects in U satisfying the condition and the decision parts (P and Q respectively), i.e., $sup(r) = \mid [P \cap Q] \mid$. The support could be given in relation to the number of examples in U as

$$sup(r) = \frac{\mid [P \cap Q] \mid}{\mid [U] \mid}.$$

The confidence of a rule r shows the degree to which P implies Q and it is defined as

$$conf(r) = \frac{\mid [P \cap Q] \mid}{\mid [P] \mid}.$$

This measure is also known as *certainty factor*, *accuracy* or *discrimination level*.

Let us notice that both these measures characterize two different properties of a rule – support corresponds to the generality of a pattern represented by the rule in data, while confidence estimates the certainty of assignment to a decision class indicated by the rule.

Another measure of rule generality is called *coverage* or rule strength, and it is used in the description of the EXPLORE algorithm. The coverage of a rule r is defined as

$$cov(r) = \frac{|\,[P \cap Q]\,|}{|\,[Q]\,|}.$$

The other major distinction between the classification and descriptive perspectives corresponds to different rule induction algorithms. The former perspective employs algorithms inducing minimal sets of rules, while the latter requires different methods (producing non-minimal sets of rules). These two groups of algorithms are briefly discussed in the following subsections.

2.3 Induction of Minimal Sets of Rules

The majority of rule induction algorithms employed by the classification-oriented perspective follow the sequential covering strategy, which historically comes from the early Michalski's works on the family of the AQ algorithms. It is also known as the *separate-and-conquer* strategy and used in several inductive logic programs [10].

Figure 1 shows the basic idea of the sequential covering strategy. It sequentially generates a *minimal set* of decision rules for each decision class.[1] In each run it accepts as input a set of positive and negative examples of a class K_j and provides as output a set of rules R covering all positive examples of this class and not covering any of its negative examples (if the learning set does not contain any inconsistent examples). The strategy iteratively creates the best possible rule based on the "best" conjunction of elementary conditions according to selected criteria (see the function *find_single_best_rule*). Then, it stores the rule and excludes from consideration all positive examples that match this rule. This process is repeated if at least one positive example of the decision concept remains uncovered.

The function *find_single_best_rule* produces a candidate for a rule, which in general should cover as many positive examples of the target class as possible and no negative ones (for consistent data), or a limited number of negative examples (for inconsistent or noisy data). This function can be formulated in different ways depending on a particular version of the algorithm. In majority of them the condition part of a candidate rule is constructed by successively adding new elementary conditions to the conjunction (the process starts with an empty condition part). This process is repeated until a selected acceptance criterion has been fulfilled, e.g., the current condition part does not cover any of the negative examples (e.g., see the description of AQ [29] or LEM2 [12]).

The search for the best elementary condition to be added to the conjunction is driven by specific evaluation criteria. The number of proposals is quite large, for a review see [10]. For instance, in the LEM2 algorithm Grzymala-Busse proposed to select conditions in the following way [12]:

[1] The are also some versions of this strategy, which do not sequentially go through classes but attempt to consider all classes together, however, still maintaining the principle of recursively learning the best rule, removing covered examples, etc.

```
procedure sequential_covering(input $K_j$: class;
$E$ : its positive learning examples; $N$ : and its negative examples
output $R$: set of rules)
begin
    R $\leftarrow \emptyset$    { initialize rules };
    while $E \neq \emptyset$ do
    begin
        $r \leftarrow$ find_single_best_rule($K_j, E, N$)
        $[r]_E \leftarrow$ set of positive examples covered by $r$
        if rule_stopping_conditions($r$) then exit;
        $E \leftarrow E \setminus [r]_E$;
        $R \leftarrow R \cup r$
    end
    $R \leftarrow$ post-process($R$)
end
```

Fig. 1 Sequential covering strategy

1. Choose a condition that results in the maximum support of a candidate rule,
2. If a tie occurs, choose a condition that results in the largest confidence of a candidate rule.

Several other criteria are considered, the most common choices are: entropy-based measures calculated over the distribution in the examined cover [6, 37], Laplace estimate or more flexible m-estimate [11]. Weighted formulas are useful as well, e.g., weighted information gain used by Quinlan in FOIL, J-measures and many others (for a review see again [10]).

The basic covering strategy presented above reveals drawbacks if data is noisy. Rules for noisy examples may be too complicated (*overfitted to noise*) and lead to low predictive accuracy while classifying new examples. In general, there are several solutions to overcome the overfitting, which usually rely on *pruning*. They allow induced rules not to cover all positive examples or to cover some negative ones. Efficient techniques for rule post-pruning were employed in the RIPPER algorithm [7], which is one of the most popular techniques of rule induction. Different rule pruning techniques are summarized in [9].

Finally, we would like to note that rules are also successfully applied inside multiple classifiers (ensembles). For instance, basic concepts of RIPPER were adopted inside SLIPPER [8], similarly, MODLEM was used inside extended bagging and pairwise coupling [40].

2.4 Induction of Non-minimal Sets of Rules

Minimal sets of rules usually contain only a *limited number* of interesting rules, they may also include some rules of very little or no interest, which is undesirable from the discovery-oriented perspective. These shortcomings result directly from the sequential covering strategy described in the previous section. This strategy excludes from consideration learning examples that have been already covered by generated rules, thus, some interesting rules cannot be discovered. This happens especially when different patterns are shared by a large number of examples. Moreover, the sequential covering strategy aims at covering all positive learning examples, therefore, in last iterations it may produce very specific rules, consisting of many elementary conditions, which refer only to one or very few learning examples that have been left uncovered. More detailed discussion on this problem is given in [42].

In order to overcome the above limitations other induction strategies and algorithms have been proposed. The most radical solution is to produce a so-called *exhaustive* set of rules, which contains *all* rules that can be induced on the basis of positive examples of a class. Examples of such approach include the *dropping condition* technique described in [12] or *Boolean reasoning* approach to looking for local object reducts [36] (specific for rough set theory). However, time complexity for the latter technique is exponential and using it may be not practical for larger data sets, so approximate algorithms are employed. Moreover, the data analyst could be "overloaded" by getting too many rules to be considered. In fact, only a small number of them is usually interesting (these approaches may generate many specific rules supported by few learning examples).

Another category of induction algorithms employs "more efficient" search strategy leading to less numerous sets of rules, which should be also characterized by better values of evaluation measures. A good example is the BRUTE algorithm introduced in [35]. The name comes from authors' motivation to perform a massive, brute-force search for accurate rules in place of the greedy hill-climbing search typical for the iterative sequential covering. Briefly speaking, BRUTE conducts an exhaustive depth-bounded search for the most accurate and shortest rules. It optimizes the search by introducing canonical order in possible conditions. Moreover, it limits the search to rules not exceeding the maximum number of conditions in their condition parts. Finally, it outputs only a limited number of rules that have been most accurate on a learning set. Experimental results showed that a classifier using the top 50 rules outperformed CART and C4 trees [35]. A similar idea is present *Data Surveyor system* described in [18].

Finally, the last group of approaches includes adaptation algorithms for association rule mining. Such rule are transformed into a form where the right hand side of a rule contains the decision class. Rules should also satisfy predefined requirements for the minimum support (such rules are called frequent ones) and the minimum confidence.[2] The key issue is to adopt in a proper way search strategies derived from algorithms for mining frequent items, e.g., to construct an iterative sequential extension approach similar to Apriori, which efficiently prunes candidates. In [16] there is a short review of some proposals, e.g., a method of associative classification by Liu et al. [26].

[2] These requirements are similar to the ones presented in EXPLORE - see Section 3.

2.5 Classification Strategies

In the classification-oriented perspective a set of rules is used to classify new examples, i.e., examples unseen in the learning phase, by matching them to the condition parts of rules. Sets of rules can be either ordered or unordered. In the first case rules are organized into a priority list. The matching is done starting from the first rule. The first matched rule from the list is used to classify a new example and the remaining rules are skipped. The last rule is a default rule and it is used if no other rule has been matched.

For unordered sets of rules matching a new object may lead to three situations. The first one is a unique match to one or more rules from the same class. The two other situations are *matching multiple rules* indicating different classes or *not matching* any rules at all. In both situations a suggestion is ambiguous, thus, proper resolution strategy is necessary. Review of different strategies is given in [39]. Below we briefly summarize a classification strategy introduced by Grzymala-Busse in LERS [13] as it is employed in our experiments. In case of ambiguous multiple matching the decision how to classify an example e is made on the basis of voting and e is assigned to the strongest class (i.e., the class that has received most votes). For each matched rule its absolute support is considered as a basic score. The total *support* for a class K_i and an example e is defined with the following expression:

$$sup(K_i,e) = \sum_{rules for\ K_i\ matching\ e} sup(r).$$

The class K_j with the largest support is the winner and the example e is assigned to it.

If complete matching is impossible, all *partially matching rules* are identified. These are rules with at least one elementary condition matching an example e. The total support is then calculated from the support of identified rules, and from their matching factors, defined as a ratio of conditions matched by e to all conditions in a rule (or to the length of a rule):

$$sup(K_i,e) = \sum_{rules\ for\ K_i\ partially\ matching\ e} sup(r) \times match(r,e).$$

Again, the class K_j with the largest support is the winner and an example e is classified as its member.

3 EXPLORE Algorithm

In this chapter we present the EXPLORE algorithm that extracts from data all rules that satisfy requirements defined by the user. Thus, EXPLORE is able to generate rules which are general, simple, accurate and relevant. This makes it very useful not only from the descriptive-oriented perspective but also also from the classification-oriented one, especially when imbalanced data has to be dealt with.

3.1 Presentation of the Algorithm

The *EXPLORE* algorithm, first presented in [30], is a procedure that extracts from data all decision rules that satisfy certain requirements. In this study we focus on the following ones:

- *support* or *coverage* of a rule: the user can expect that the general and strong rule should cover a large enough number of positive examples,
- *consistency* of a rule represented by its *confidence*: the rule should cover no or very few negative examples,
- *simplicity* of a rule represented by its *length*: generally the rule should be short,
- *total number of rules* for all decision classes: the resulting set of rules should be limited in size for cognitive reasons.

These requirements are used to impose restrictions on the rule space explored during induction. The algorithm can handle inconsistent examples either by using rough set theory to define approximations of decision classes, or by determining appropriate threshold for confidence of induced rules to be used in pre-pruning.

Exploration of the rule space is performed using a procedure, which is *repeated* for each class K_j to be described. The main part of the algorithm is based on a *breadth-first search*, which amounts to generating rules of increasing size, starting from one-condition rules. Exploration of a specific branch is stopped as soon as a rule satisfying the requirements is obtained or any of stopping conditions *SC*, reflecting the impossibility to fulfill the requirements, has been met. EXPLORE is formally presented in Figure 2. A short description of the algorithm is given below, more enhanced discussion is provided in [42], and its implementation details can be found in [30]).

An initial list *LS* representing elementary conditions is created by analyzing positive examples provided as input to EXPLORE (for more precise description see [38]). Obviously, conditions in *LS* must cover at least one example from K_j; they may also be subject to specific constrains on their syntax. This initial list is first pruned to discard conditions, which directly correspond to rules, as well as those which already satisfy *SC*, and thus cannot give rise to rules (procedure *good_candidates*). Conditions remaining in *LS* are then combined to form *complexes* (i.e., conjunctions of elementary conditions), which are candidates for the condition parts of rules. This is achieved by procedure *extend*, which at iteration k creates conjunctions of size $k+1$ by extending candidate conjunctions of size k with conditions from *LS*. While extending the conjunctions we can use the monotonicity principle known from the Apriori algorithm, stating that all subsets of a candidate conjunction must also be sufficiently strong [16]. The resulting conjunctions are then tested by procedure *good_candidates*.

In general, stopping conditions *SC* can be defined according to requirements expressing various expectations of the user, e.g., imposed on coverage, length, number of rules, etc. In our experiments presented in Section 6 we mainly consider requirements referring to the minimal coverage l. The corresponding stopping condition for a conjunction C currently examined is thus simply: $cov(C) < l$. Let us remark

```
**procedure** EXPLORE
(**input** $LS$: list of valid elementary conditions; SC: stopping conditions;
**output** $R$: set of rules)
**begin**{Main search procedure}
    $R \leftarrow \emptyset$
    good_candidates($LS$, $R$);    {$LS$ is a list of valid elementary conditions $s_1, s_2, \ldots, s_n$
            ordered according to decreasing coverage}
    $Q \leftarrow LS$; {Copy current $LS$ to a queue $Q$}
    **while** $Q \neq \emptyset$ **do**
    **begin**
        select the first conjunction $C$ in $Q$;
        $Q \leftarrow Q \setminus \{C\}$; {remove it from the queue}
        extend($C$,$LC$); {generate $LC$ – a list of extended conjunctions}
        good_candidates($LC$,$R$);
        $Q \leftarrow Q \cup LC$ {place all conjunctions from $LC$ at the end of $Q$}
    **end**
**end**;

**procedure** extend(**input** $C$: complex;
    **output** $L$: list of conjunctions)
{ This procedure puts in list $L$ extensions of conjunctions $C$ that are potential candidates for rules.}
**begin**
    Let $k$ be the size of $C$ and $h$ be the highest index of the elementary condition involved in $C$;
    $L \leftarrow \{C \wedge s_{h+i}$ where $s_{h+i} \in LS$ and such that all the $k$ subconjuctions of $C \wedge s_{h+i}$
            of size $k$ and involving $s_{h+i}$ belong to $Q$ $(i = 1, \ldots, n-h)\}$
**end**;

**procedure** good_candidates(**input** $L$: list of conjunctions;
    **output** $R$: set of rules)
{ This procedure prunes list $L$, discarding:
    - conjunctions, which cannot give rise to rules due to $SC$,
    - conjunctions corresponding to rules, which are stored into $R$. }
**begin**
    **for** each $C \in L$ **do**
    **begin**
        **if** $C$ satisfies SC **then** $L \leftarrow L \setminus \{C\}$
        **else**
            **if** $conf(C) \geq \alpha$ **then**      {$\alpha = 1$ for totally discriminant rules}
            **begin**
                $R \leftarrow R \cup \{C\}$;
                $L \leftarrow L \setminus \{C\}$
            **end**
    **end**
**end**;
```

Fig. 2 The main procedure of the EXPLORE algorithm

that stopping conditions restrict exploration space and reduce computational costs. If the user does not define any requirements, the algorithm will produce all rules, which is at the risk of exponential complexity (see the evaluation of complexity in [39]). Examples of using EXPLORE and tuning *SC* are presented in the next two subsections.

Besides basic requirements represented by the stopping conditions, EXPLORE can be easily adopted to handle additional expectations of the user with regard to the syntax of the condition parts of rules. For instance, the user can express her preferences for some specific elementary conditions or attributes to be used in a rule (or to be excluded from a rule). It is also possible to focus the search on some specific subsets of examples. Such an approach is typical for interactive knowledge discovery tools and it has been described in [41, 39].

3.2 Example: Using EXPLORE for Technical Diagnostics

To demonstrate the benefits of a non-minimal set of rules induced by the EXPLORE algorithm we describe a real life problem of technical diagnostics of a homogeneous fleet of buses [52]. 76 buses were described by 8 diagnostic symptoms (attributes) and divided into two classes depending on their technical conditions (good or bad). The following symptoms were chosen: *s1* – maximum speed, *s2* – compression pressure, *s3* – blacking components in exhaust gas, *s4* – torque, *s5* – summer fuel consumption, *s6* – winter fuel consumption, *s7* – oil consumption and *s8* – maximum horsepower of the engine. All these attributes were numeric.

We started with inducing a minimal set of rules using the MODLEM algorithm (this algorithm follows the sequential covering strategy and is well suited for numerical data [37]). The generated set contained the three following rules covering all learning examples (numbers in brackets correspond to positive learning examples covered by each rule) :

1. if ($s2 \geq 2.4$ MPa) & ($s7 < 2.1$ *l*/1000km) then (technical state=good) [46]
2. if ($s2 < 2.4$ MPa) then (technical state=bad) [29]
3. if ($s7 \geq 2.1$ *l*/1000km) then (technical state=bad) [24]

Prediction accuracy of a classifier using these rules was evaluated using the leaving-one-out technique, and it was equal to 98.7%. Although it was a very accurate predictor of the technical condition, the analysis of the syntax of these three rules showed that only two symptoms were important. In particular, the compression level was crucial as it nearly perfectly discriminated buses from the two considered classes. On the other hand, practical measurements of this symptom were the most difficult at the diagnostic stand. Thus, the experts were interested in discovering other rules, formulated using symptoms that were easier to collect. Therefore, they decided to discover strong rules covering more than 50% of buses in each class. EXPLORE found 11 rules satisfying the above requirements, they are listed below.

1. if ($s1 > 85$ km/h) then (technical state=good) [34]
2. if ($s8 > 134$ KM) then (technical state=good) [26]
3. if ($s2 \geq 2.4$ MPa) & ($s3 < 61$ %) then (technical state=good) [44]
4. if ($s2 \geq 2.4$ MPa) & ($s4 > 444$ Nm) then (technical state=good) [44]
5. if ($s2 \geq 2.4$ MPa) & ($s7 < 2.1$ *l*/1000km) then (technical state=good) [46]
6. if ($s3 < 61$ %) & ($s4 > 444$ Nm) then (technical state=good) [42]
7. if ($s1 \leq 77$ km/h) then (technical state=bad) [25]

8. if ($s2$<2.4 MPa) then (technical state=bad) [29]
9. if ($s7$≥2.1 l/1000km) then (technical state=bad) [24]
10. if ($s3$≥61 %) & ($s4$≤444 Nm) then (technical state=bad) [28]
11. if ($s3$≥61 %) & ($s8$<120 KM) then (technical state=bad) [27]

These rules provided much more information about values of symptoms than the previous minimal set of rules. We used them to construct a new classifier and estimated its accuracy again in the leaving-one-out test. The classification accuracy was exactly the same as for the classifier with the minimal set of rules.

3.3 Other Experience with EXPLORE

Let us remind that setting proper thresholds for the stopping conditions *SC* is crucial for the EXPLORE algorithm. An iterative procedure based on stepwise changes of the *rule coverage threshold* and observing its influence on the set of rules was presented in [42]. Experiments on several data sets from the UCI repository [1] showed that it was possible to determine a range of values for this threshold, which led to good sets of rules in terms of their classification accuracy, the average coverage, the average length and their number. In Table 1 we present sample results obtained for the *congress voting* data. The last line lists results for the minimal set of rules generated by LEM2. One can notice that threshold values between 20% and 30% led to sets of rules, which had significantly better descriptive properties (e.g., the average support was twice as high as for rules in the minimal set) and not worse classification properties at the same time.[3]

Table 1 Characteristics of rules induced by EXPLORE vs. the minimal set or rules induced by LEM2 for voting data (*SC* – rule support threshold, N_R – number of rules, *cov* – average rule coverage (absolute number of examples), *len* – average rule length (number of elementary conditions), *acc* – overall classification accuracy [%])

SC	N_R	*Cov*	*Len*	*Acc*
5%	231	45.86	3.36	97.91
10%	138	66.96	3.19	97.67
15%	125	75.46	3.71	96.98
20%	103	82.75	3.81	96.07
25%	80	86.95	3.95	95.38
30%	63	95.16	3.75	92.61
40%	21	133.00	2.76	80.23
LEM2	26	43.77	3.69	95.87

[3] We would like to clarify that the main aim of these experiments was not to get the most accurate classifier. The classification accuracy was just an additional criterion to evaluate the "quality" of a set of rules.

Continuing our discussion, we would like to stress that for large values of the coverage threshold EXPLORE may induce a set of rules covering only a subset of learning examples. Some "difficult" examples (e.g., located in sparse subregions of classes) may not be covered by any strong rule. In [43] we proposed a solution to this problem and introduced a *hybrid approach*, where the first level of representation is constituted by rules and the second level is a set of learning examples not covered by these rules. This first level can be obtained either by rule pruning or by using EXPLORE with large values of the coverage threshold (the stepwise tuning mentioned above could be used to establish the threshold – for more details see [39, 43]). The classification strategy for new examples is a two stage approach. A new example is first classified by rules. If there is no match or matching is ambiguous, then the example is classified according to the k-nearest neighbor principle on the basis of stored examples.

This idea was verified in the problem of evaluating business loans [43]. The interesting observation was that the hybrid approach led to the highest classification accuracy of 81%, while the rule level itself gave 77% and other classifiers (e.g., a decision tree) around 74%. Furthermore, we noticed that this approach slightly increased the sensitivity for the minority class, which corresponded to the most risky loans leading to questionable or lost liabilities. Similar improvements were observed in a medical case study.

Such an impact of the threshold tuning procedure on the sensitivity for the minority class has been a direct inspiration for our current research on dealing with imbalanced data, which is presented in the following sections.

4 Handling Imbalanced Data

Many learning algorithms are formulated with an explicit or implicit assumption that learning sets are balanced. However, this is not always the case. Imbalanced data sets are quite common as many processes produce certain observations with different frequencies. A good example is medicine, where databases regarding a rare but dangerous disease usually contain a smaller group of patients requiring special attention, while there is much larger number of members of other classes – patients who do not require special treatment. Similar situations occur in other domains, e.g., in technical diagnostics or continuous fault-monitoring tasks, where non-faulty examples may heavily outnumber faulty ones. Survey papers [49, 3] report other real technical or engineering problems, e.g., detection of oil spills in satellite radar images, detection of fraudulent telephone calls or credit card transactions, prediction of telecommunication equipment failures, and information retrieval and filtering.

If the imbalance in the class distribution is extensive, i.e., some classes are *heavily under-represented*, these learning methods do not work properly. They are "somehow biased" to focus searching on the more frequent classes while "missing" examples from the minority class. As a result constructed classifiers are also biased toward recognition of the majority classes and they usually have difficulties (or even are unable) to classify correctly new examples from the minority class. In [25] authors described an information retrieval system, where the minority class (being of primary importance) contained only 0.2% of examples. Although the classifiers achieved accuracy close to 100%, they were useless because they failed to deliver requested documents from this class. Similar degradation of classifier's performance for the minority class was reported for other imbalanced problems [4, 14, 20, 22, 49].

The class imbalance also affects rule-based classifiers – especially classifiers using minimal sets of rules are biased toward the majority classes. Rules induced for the majority classes are more general and cover more learning examples, while rules for the minority class are usually more specific and "weaker" in terms of their cover. As a result new examples from the minority class tend to be misclassified.

Imbalanced data constitutes a problem not only when inducing rules to be used in a classifier, but also when evaluating its performance. Indeed, overall classification accuracy is not the only and the best criterion characterizing performance of a classifier. Satisfactory recognition of the minority class may be often more preferred, thus, a classifier should be characterized rather by its *sensitivity* and *specificity* for the minority class. Sensitivity (also called a true-positive rate) is defined as the ratio of correctly recognized examples from the minority class and specificity is the ratio of correctly excluded examples from the majority classes. More attention is usually given to sensitivity than to specificity [14]. However, in general there is trade-off between these two measures, i.e., improving sensitivity may lead to deterioration of specificity – see experimental results in [45]. Thus, some measures summarizing both points of view are considered. One of them is *G-means* [22], calculated as a geometric mean of sensitivity and specificity.

Several authors also use the *ROC (Receiver Operating Characteristics) curve* analysis. A ROC curve is a graphical plot of a true positive rate (sensitivity) as a function of a false positive rate ($1 -$ specificity) along different threshold values characterizing performance of a studied classifier. The quality of the classifier performance is reflected by the area under a ROC curve (so-called AUC measure) [3, 49].

A small number of examples in the minority class ("the lack of data") is not the only source of difficulties in inducing classifiers. Several researchers claim that besides the size of this class it is necessary to go deeper into its other characteristics. Quite often the minority class overlaps heavily the majority classes. In particular, boundaries between classes are ambiguous. Both boundaries and the inside of the minority class may be affected by noisy examples from other classes, which cause incorrect classification of many examples from the minority class. Their influence is more critical for this class than for the majority ones – see [22, 24] for experiments and discussion. Japkowicz in her experimental study [19] also showed that the class imbalance becomes even more difficult problem particularly when the minority class contains very small subclusters, which are difficult to be learned (so-called, a small disjunct problem).

Several methods have been proposed to improve performance of classifiers learned from imbalanced data, for a review see [20, 49]. In general, one can distinguish two types of approaches. The first category includes pre-processing techniques that change the distribution of examples among classes by appropriate sampling. Simple random *over-sampling*, which replicates examples from the minority class, or random *under-sampling* that randomly eliminates examples from the majority classes until a required degree of balance between classes is reached are not the best solutions. Focused methods like SMOTE, one-side-sampling, NCR or selective filtering attempt to take into account internal characteristics of regions around examples from the minority class. Thus, they modify only these examples from majority classes, which most likely lead to misclassifying their minority class neighbors and in a more sophisticated way over-sample or introduce synthetic examples in local sub-regions of the minority class. These methods and their combinations were experimentally shown to be quite good [2, 22, 45, 47].

Other approaches proposed in the literature modify either induction or classification strategy, assign weights to examples, and use boosting or other combined classifiers [48]. Some researchers transform the problem of learning from imbalanced data to the problem of cost learning (although it is not the same and misclassification costs are unequal and unknown) and use techniques from the ROC curve analysis.

Considering the approach we propose later in this paper, the most related research is the work by Grzymala-Busse [14] on increasing sensitivity of LEM2 rule classifiers by changing the LERS classification strategy – as described in Section 2.5. Necessary changes of the strategy are limited to formulas for calculating support for a given class that are presented in Section 2.5. The main idea of this approach is to multiply the support of all minority class rules by the same real number, called a *support multiplier*, while not changing the support of rules from the majority classes. This support multiplier is a positive number greater or equal to 1 – for the majority classes it should be equal to 1, while for the minority class it should be greater than 1. As a result, during classification of a new example, such minority class rules have a better chance to influence the voting, so the minority class is finally predicted for the new object.

Another problem is selecting a value for the support multiplier. In general, the sensitivity of a classifier increases with increase of the support multiplier. However, at the same time specificity decreases, thus, it is important to identify a proper value of this parameter. In [14] Grzymala-Busse proposed to maximize a measure called *gain* = *sensitivity* + *specificity* − 1. Following this proposal, a value of the support multiplier was established experimentally in a loop, where in each iteration the support multiplier was increased, the classifier was evaluated on extra validation examples, and the loop stopped as soon as a value of the gain measure decreased. The value of the support multiplier resulting in the best gain was used in the final classifier. Experimental results confirmed that this approach outperformed the standard LEM2 classifier for many imbalanced medical data sets [14, 15].

Some other researchers tried to develop a *less greedy search* strategy while looking for rules (an example is a version of the BRUTE algorithm described in [35], or a specific genetic algorithm [49]), or to change the *inductive bias of the algorithm*, e.g., Holte at al. modified the rule induction algorithm CN2 to improve its performance for small disjuncts corresponding to rare examples from the minority class. Moreover, Weiss describes hybrid and two-phase rule induction [49], where one phase focuses on optimizing sensitivity, while the other optimizes specificity. Other approaches may use knowledge about prior distribution of probabilities or transforming the task to cost sensitivity learning [49].

5 Replacing a Set of Rules for the Minority Class

In this section we briefly describe our classifier-specific approach to handle imbalanced data – we evaluate it experimentally in Section 6 together with Grzymala-Busse's proposal of modifying the rule support for the minority class. Let us remark that both approaches assume an initial classifier uses a minimal set of rules. In general, it can be induced by any sequential covering algorithm, but in this paper we have chosen the LEM2 algorithm [13] and the classification strategy described in Section 2.5.

The new approach is inspired by the observation that in a minimal set of rules the average support of rules pointing at the majority classes is greater than the average support of rules for the minority class, so when classifying a new example the minority class may be easily outvoted. Such situation results in deteriorated sensitivity of a classifier.

The new approach, called *Replacing Rules for the Minority Class*, has been sketched for the first time in [43]. Generally speaking, unlike the Grzymala-Busse's approach, which addresses this issue by artificially increasing the support of rules for the minority class, it improves sensitivity for this class by replacing the minimal set of rules by a non-minimal set of stronger rules generated by the EXPLORE algorithm. Since these rules have better (greater) support than the original ones and are shorter (i.e. easier to be matched by a new example), there is no need for any modification of the classification strategy.

When inducing rules with EXPLORE, the stopping condition *SC* specifies the minimum required coverage or the support for constructed rules (rules with coverage below a given threshold are discarded). Setting the right values of this threshold is crucial. If the threshold is very low, EXPLORE may generate a very large set of rules for the minority class, that easily outvote rules for the majority classes what leads to high sensitivity at a cost of low specificity. On the other hand, if the threshold is very high, EXPLORE generates a very small set of very strong rules. Such rules well describe most common learning examples, however, fail to capture less frequent ones, thus many new examples are classified using partially matched rules. Then, the rules for the majority classes by the virtue of their number have better chance to win in the voting, what results in higher specificity and lower sensitivity.

We establish the range for the coverage threshold by checking the minimum and maximum coverage of the initial minimal set of rules for the minority class. The maximum coverage of rules generated by LEM2 and EXPLORE should be the same, thus, there is no sense in examining larger values (EXPLORE would generate no rules in such case). Moreover, the minimum coverage indicates the prevalence of the least frequent pattern in learning data, so it is not necessary to check smaller thresholds. We iteratively examine possible coverage thresholds within the

```
procedure replace_rules (input $K_{min}$: the minority class;
R : initial minimal set of rules;
L : learning examples; T : validation examples;
output $R^{final}$: resulting set of rules)
begin
        $min\_sup \leftarrow$ minimum coverage in $R$ for $K_{min}$
        $max\_sup \leftarrow$ maximum coverage in $R$ for $K_{min}$
        $R_{maj} \leftarrow$ rules from $R$ pointing at the majority classes
        $R_{min}^{min\_sup} \leftarrow$ use EXPLORE to induce rules from $L$ for $K_{min}$
            with minimum required coverage set to $min\_sup$
    for $sup = min\_sup$ to $max\_sup$ do
    begin
        $R_{min}^{sup} \leftarrow$ select these rules from $R_{min}^{min\_sup}$ for which coverage $\geq sup$
        $R^{sup} \leftarrow R_{min}^{sup} \cup R_{maj}$
        $gain \leftarrow$ evaluate $R^{sup}$ on $T$
        memorize $gain$ and $R^{sup}$
    end
    $R^{final} \leftarrow R^{sup}$ corresponding to the best observed $gain$
end
```

Fig. 3 Replacing rules for the minority class

identified range. In each iteration of the loop we use EXPLORE to generate rule for the minority class with the minimum coverage equal to the current threshold. Then, we combine these rules with the minimal rules for the majority classes and evaluate the resulting classifier on extra validation examples using the gain measure as in the Grzymala-Busse's approach. Finally, we select the set of rules that resulted in the highest gain to be embedded in the final classifier. In order to avoid repeating induction for various coverage thresholds, it is sufficient to create a set of rules for the minimal threshold and filter it appropriately in subsequent iterations of the loop. Figure 3 illustrates a basic version of our approach.

6 Experimental Evaluation

To evaluate the usefulness of our approach we experimentally compared it to a baseline classifier using a minimal set of rules generated by LEM2 [12]. Moreover, we considered a variant of such a classifier with a modified classification strategy expanded with the support multiplier (this modification proposed by Grzymala-Busse is particularly suited to deal with imbalanced data – see its description in Section 4).

We decided to examine the three measures: sensitivity, specificity and G-mean because they are more intuitive than AUC measure and they correspond to the fully deterministic algorithms (which is a case of our rule-based classifiers). Additionally, we report overall classification accuracy. Values of all these measures are presented as percentages. They are estimated as means in the *k*-fold cross validation. Moreover, to minimize the influence of splitting data sets on the classification results

Table 2 Characteristics of data sets used for experiments (N – number of examples, N_{Pos} – number of examples in the minority class, N_{Oth} – number of examples in the majority classes, $R_{Pos} = N_{Pos}/N$ – ratio of examples in the minority class)

Data set	N	N_{Pos}	N_{Oth}	R_{Pos}
Abdominal Pain	723	202	521	27.9%
Breast Slovenia	294	89	205	30.3%
Breast Wisconsin	625	112	513	17.9%
Bupa	345	145	200	42.0%
German	666	209	457	31.4%
Hepatitis	155	32	123	20.6%
Pima	768	268	500	34.9%
Scrotal Pain	201	59	142	29.4%
Urology	498	155	343	31.1%

Table 3 Results for the original LEM2 algorithm (*sens* – sensitivity, *spec* – specificity, *GM* – G-mean, *acc* – overall accuracy, N_R – number of rules)

Data set	*Sens*	*Spec*	*GM*	*Acc*	N_R
Abdominal Pain	58.42	92.90	73.67	83.26	20.0
Breast Slovenia	36.47	88.56	56.83	73.08	20.5
Breast Wisconsin	31.25	92.59	53.79	81.60	29.5
Bupa	32.41	74.00	48.97	56.52	42.0
German	30.14	84.68	50.51	67.57	42.5
Hepatitis	43.75	95.12	64.51	84.52	6.5
Pima	39.18	82.60	56.89	67.45	66.0
Scrotal Pain	54.24	83.10	67.14	74.63	12.0
Urology	12.18	82.27	31.65	60.40	28.0

obtained for all approaches, the division into folds was performed only once and the same subsets of examples were used to construct all three variants of classifiers.[4]

Experiments were conducted on 9 imbalanced data sets, which are coming from UCI repository except two data sets *abdominal pain* and *scrotal pain* – these are coming from our practical case studies [51, 27]. Let us notice that nearly all data sets, except *German credit*, come from a medical domain. Data sets, which originally included more than two classes, were transformed to binary ones, by collapsing all the majority classes into one. Moreover, some of the original data sets contained numerical attributes, which was a disadvantage for LEM2, thus, these attributes were discretized by a Grzymala-Busse's method based on clustering with merging intervals [5]. Table 2 lists the data sets along with their basic characteristics.

[4] In our previous joint research with Grzymala-Busse [15] we conducted some experiments, thus, some of results for LEM2 come from that paper.

Table 4 Best results of increasing rule support by multipliers (*mult* – support multiplier, *sens* – sensitivity, *spec* – specificity, *GM* – G-mean, *acc* – overall accuracy)

Data set	*Mult*	*Sens*	*Spec*	*GM*	*Acc*
Abdominal Pain	5	80.69	84.84	82.74	83.68
Breast Slovenia	1	36.47	88.56	56.83	73,08
Breast Wisconsin	5	57.14	86.74	70.41	81.44
Bupa	3	55.86	58.50	57.17	57.39
German	4	57.89	64.11	60.92	62.16
Hepatitis	18	84.38	77.24	80.73	78.71
Pima	3.5	59.33	76.40	67.32	70.44
Scrotal Pain	3	67.80	80.99	74.10	77.11
Urology	14	51.92	49.42	50.65	50.52

Table 5 Results for the *Replacing Rules* approach (*SC* – coverage threshold, *sens* – sensitivity, *spec* – specificity, *GM* – G-mean, *acc* – overall accuracy, N_R – number of rules)

Data set	*SC*	*Sens*	*Spec*	*GM*	*Acc*	N_R
Abdominal Pain	8.0	83.14	83.68	83.41	83.54	88.0
Breast Slovenia	3.0	47.09	84.11	62.93	73.08	37.0
Breast Wisconsin	2.0	63.85	81.60	72.18	78.57	158.5
Bupa	2.0	42.75	63.00	51.90	54.50	61.5
German	5.0	62.71	72.65	67.50	69.50	73.5
Hepatitis	4.0	75.30	81.56	78.37	80.02	76.5
Pima	2.0	68.78	67.89	68.33	68.10	341.5
Scrotal Pain	4.0	68.87	87.24	77.51	81.56	12.5
Urology	4.0	71.73	43.20	55.67	51.61	691.5

Results for all compared approaches are presented in Tables 3 – 5. The approach to extend the classification strategy with the support multiplier is consistent with the Grzymala-Busse's proposal [14] of optimizing the gain measure (see also its description in Section 4) – the best values of the multiplier are listed in Table 3. For our approach we additionally present values of the rule support (coverage) threshold for the minority class and the number of rules generated by EXPLORE to replace the initial minimal set for the minority class. We compare results of both these approaches to the standard LEM2 rule-based classifier using the Wilcoxon Signed Ranks test (with $\alpha = 0.05$). Considering sensitivity and G-mean both approaches (based on the multiplier and replace techniques) outperform it, and the difference between them is significant for sensitivity, what emphasizes superiority of our approach. Moreover our approach significantly outperforms the multiplier approach with respect to G-mean. On the other hand, we should be aware of the fact that number of rules for the minority class significantly increases (especially for abdominal pain, hepatitis, pima and urology). According to discussion in [15] it may be possible to impose an upper limit on the number of replaced rules so it is closer to the number in the original minimal set.

7 Conclusions

Our study focuses on using rule induction algorithms on imbalanced data to create improved rule-based classifiers. Following a comprehensive discussion of the most common rule induction algorithms and classification strategies we have shown they are too biased towards the majority classes – both during learning and classification phases. This is attributed mainly to a greedy search strategy of the sequential covering employed by many rule induction algorithms. However, this bias can be avoided either by changing the classification strategy or by using less greedy search for rules for the minority class.

The main research contribution of our study involves introducing a new approach to constructing rules for a rule-based classifier, where minimal sets of rules are induced for the majority classes, while for the minority class we create a non-minimal set of rules (more numerous, and characterized by higher average coverage) that improves a chance of a classification strategy to recognize the minority class. We have proposed to use the EXPLORE algorithm to generate rules for this class. As opposed to algorithms based on sequential covering, EXPLORE performs less greedy search and induces all rules that satisfy specific requirements (e.g., coverage greater than a given threshold).

In a series of experiments we have compared our approach to a baseline classifier with the minimal set of rules and the basic classification strategy, and a classifier with the classification strategy expanded with the strength multiplier. Experimental results have shown that both our approach and the approach with the support multiplier have increased sensitivity in comparison to the baseline classifier. However, let us notice that the multiplier approach is similar to over-sampling of learning data and in some cases it may lead to quite extensive changes in balance between classes (see Table 4). On the other hand, our approach does not modify learning data, rules discovered by EXPLORE correspond to really existing patterns and they are still comprehensible for human experts.

Further directions for our research include expanding our approach by post-pruning rules for the majority classes and manipulating learning examples in an "intelligent" way. The latter is a subject of our current work and we have already introduced a new approach to selective pre-processing of imbalanced data that aims at improving sensitivity of an induced classifier, while keeping overall accuracy at an acceptable level [45]. Briefly speaking, it combines selective filtering of difficult examples from the majority classes (either by removing examples, which may contribute to misclassification of examples from the minority class, or by relabeling some of them) with limited over-sampling of the minority class. In the first experimental study presented in [45] this approach was successfully combined with MODLEM. The more advanced research [46] also involved the use of C4.5 decision trees [33] and RIPPER rules [7]. We conducted comprehensive experiments, where this approach was compared against other pre-processing methods, such as SMOTE, NCR, or simple random under- and over-sampling, showing its advantages. Unfortunately, more elaborated presentation is beyond the scope and limit of this paper.

Acknowledgements. This research is supported by the grant N N519 3505 33. We also acknowledge cooperation of J.Stefanowski with D. Vanderpooten from University Paris Dauphine on introducing the EXPLORE algorithm and of both of us with J.Grzymala-Busse from the University of Kansas on comparative experiments with LEM2.

References

1. Asuncion, A., Newman, D.J.: UCI Machine Learning Repository. University of California, School of Information and Computer Science, Irvine (2007), `http://www.ics.uci.edu/~mlearn/MLRepository.html`
2. Batista, G., Prati, R., Monard, M.: A study of the behavior of several methods for balancing machine learning training data. ACM SIGKDD Explorations Newsletter 6(1), 20–29 (2004)
3. Chawla, N.: Data mining for imbalanced datasets: an overview. In: Maimon, O., Rokach, L. (eds.) The Data Mining and Knowledge Discovery Handbook, pp. 853–867. Springer, Heidelberg (2005)
4. Chawla, N., Bowyer, K., Hall, L., Kegelmeyer, W.: SMOTE: synthetic minority oversampling technique. Journal of Artificial Intelligence Research 16, 341–378 (2002)
5. Chmielewski, M.R., Grzymala-Busse, J.W.: Global discretization of continuous attributes as preprocessing for machine learning. In: Lin, T.Y., Wildberger, A. (eds.) Soft Computing: Rough Sets, Fuzzy Logic, Neural Networks, Uncertainty Management, Knowledge Discovery, pp. 294–297. Simulation Councils Inc. (1995)
6. Clark, P., Niblett, T.: The CN2 induction algorithm. Machine Learning 3, 261–283 (1989)
7. Cohen, W.: Fast effective rule induction. In: Proc. of the 12th International Conference on Machine Learning (ICML 1995), pp. 115–123 (1995)
8. Cohen, W., Singer, Y.: A simple, fast and effective rule learner. In: Proc. of the 16th National Conference on Artificial Intelligence (AAAI 1999), pp. 335–342. AAAI Press, Menlo Park (1999)
9. Furnkranz, J.: Pruning algorithms for rule learning. Machine Learning 27(2), 139–171 (1997)
10. Furnkranz, J.: Separate and conquer rule learning. Artificial Intelligence Review 13(1), 3–54 (1999)
11. Dzeroski, S., Cestnik, B., Petrovski, I.: Using the m-estimate in rule induction. Journal of Computing and Information Technology 1, 37–46 (1993)
12. Grzymala-Busse, J.W.: LERS - a system for learning from examples based on rough sets. In: Slowinski, R. (ed.) Intelligent Decision Support. Handbook of Applications and Advances of the Rough Sets Theory, pp. 3–18. Kluwer, Dordrecht (1992)
13. Grzymala-Busse, J.W.: Managing uncertainty in machine learning from examples. In: Proc. of the 3rd International Symposium in Intelligent Systems, Wigry, Poland, pp. 70–84. IPI PAN Press (1994)
14. Grzymala-Busse, J.W., Goodwin, L.K., Grzymala-Busse, W.J., Zheng, X.: An approach to imbalanced data sets based on changing rule strength. In: AAAI Workshop at the 17th Conference on AI, AAAI 2000, Learning from Imbalanced Data Sets, Austin, TX, July 30–31, pp. 69–74 (2000)

15. Grzymala-Busse, J.W., Stefanowski, J., Wilk, S.: A comparison of two approaches to data mining from imbalanced data. In: Negoita, M.G., Howlett, R.J., Jain, L.C. (eds.) KES 2004. LNCS, vol. 3213, pp. 757–763. Springer, Heidelberg (2004)
16. Han, J., Kamber, M.: Data mining: Concepts and techniques. Morgan Kaufmann, San Francisco (2000)
17. Hilderman, R.J., Hamilton, H.J.: Knowledge Discovery and Measures of Interest. Kluwer Academic, Boston (2002)
18. Holsheimer, M., Kersten, M.L., Siebes, A.: Data Surveyor: searching the nuggets in parallel. In: Fayyad, U.M., et al. (eds.) Advances in Knowledge Discovery and Data Mining, pp. 447–467. AAAI/MIT Press, Cambridge (1996)
19. Japkowicz, N., Stephen, S.: The class imbalance problem: a systematic study. Intelligent Data Analysis 6(5), 429–450 (2002)
20. Japkowicz, N.: Learning from imbalanced data sets: a comparison of various strategies. In: AAAI Workshop at the 17th Conference on AI, AAAI 2000, Learning from Imbalanced Data Sets, Austin, TX, July 30–31, pp. 10–17 (2000)
21. Klosgen, W., Żytkow, J.M.: Handbook of Data Mining and Knowledge Discovery. Oxford Press, Oxford (2002)
22. Kubat, M., Matwin, S.: Addressing the curse of imbalanced training sets: one-side selection. In: Proc. of the 14th International Conference on Machine Learning (ICML 1997), pp. 179–186 (1997)
23. Langley, P., Simon, H.A.: Fielded applications of machine learning. In: Michalski, R.S., Bratko, I., Kubat, M. (eds.) Machine learning and data mining, pp. 113–129. John Wiley & Sons, Chichester (1998)
24. Laurikkala, J.: Improving identification of difficult small classes by balancing class distribution. Technical Report A-2001-2, University of Tampere (2001)
25. Lewis, D., Catlett, J.: Heterogeneous uncertainty sampling for supervised learning. In: Proc. of 11th International Conference on Machine Learning (ICML 1994), pp. 148–156 (1994)
26. Liu, B., Hsu, W., Ma, Y.: Integrating classification and association rule mining. In: Proc. of the 4th International Conference on Knowledge Discovery and Data Mining, KDD 1998 (1998)
27. Michalowski, W., Wilk, S., Farion, K., Pike, J., Rubin, S., Slowinski, R.: Development of a decision algorithm to support emergency triage of scrotal pain and its implementation in the MET system. INFOR 43(4), 287–301 (2005)
28. Michalski, R.S.: A theory and methodology of inductive learning. In: Michalski, R.S., Carbonell, J.G., Mitchell, T.M. (eds.) Machine Learning: An Artificial Intelligence Approach, pp. 83–134. Morgan Kaufman, San Francisco (1983)
29. Michalski, R.S., Bratko, I., Kubat, M. (eds.): Machine learning and data mining. John Wiley & Sons, Chichester (1998)
30. Mienko, R., Stefanowski, J., Toumi, K., Vanderpooten, D.: Discovery-oriented induction of decision rules. Cahier du Lamsade no. 141, Paris, Université Paris Dauphine (September 1996)
31. Mitchell, T.: Machine learning. McGraw-Hill, New York (1997)
32. Pawlak, Z.: Rough sets. In: Theoretical Aspects of Reasoning about Data. Kluwer Academic Publishers, Dordrecht (1991)
33. Quinlan, J.R.: C4.5: Programs for Machine Learning. Morgan Kaufmann, San Francisco (1992)
34. Ras, Z., Wieczorkowska, A.: Action rules: how to increase profit of a company. In: Zighed, D.A., Komorowski, J., Żytkow, J.M. (eds.) PKDD 2000. LNCS (LNAI), vol. 1910, pp. 587–592. Springer, Heidelberg (2000)

35. Riddle, P., Segal, R., Etzioni, O.: Representation design and brute-force induction in a Boening manufacturing fomain. Applied Artificial Intelligence Journal 8, 125–147 (1994)
36. Skowron, A.: Boolean reasoning for decision rules generation. In: Komorowski, J., Raś, Z.W. (eds.) ISMIS 1993. LNCS (LNAI), vol. 689, pp. 295–305. Springer, Heidelberg (1993)
37. Stefanowski, J.: The rough set based rule induction technique for classification problems. In: Proc. of the 6th European Conference on Intelligent Techniques and Soft Computing EUFIT 1998, Aachen, pp. 109–113 (1998)
38. Stefanowski, J.: Handling continuous attributes in discovery of strong decision rules. In: Polkowski, L., Skowron, A. (eds.) RSCTC 1998. LNCS (LNAI), vol. 1424, pp. 394–401. Springer, Heidelberg (1998)
39. Stefanowski, J.: Algorithims of rule induction for knowledge discovery. Habilitation Thesis published as Series Rozprawy no. 361. Poznan Univeristy of Technology Press, Poznan (2001) (in Polish)
40. Stefanowski, J.: On combined classifiers, rule induction and rough sets. In: Peters, J., et al. (eds.) Transactions on Rough Sets VI. LNCS, vol. 4374, pp. 329–350. Springer, Heidelberg (2007)
41. Stefanowski, J., Borkiewicz, R.: Interactive rule discovery of decision rules. In: Proc. of the VIIIth Intelligent Information Systems, June 1999, pp. 112–116. Wyd. Instytutu Podstaw Informatyki PAN, Warszawa (1999)
42. Stefanowski, J., Vanderpooten, D.: Induction of decision rules in classification and discovery-oriented perspectives. International Journal of Intelligent Systems 16(1), 13–28 (2001)
43. Stefanowski, J., Wilk, S.: Evaluating business credit risk by means of approach integrating decision rules and case based learning. International Journal of Intelligent Systems in Accounting, Finance and Management 10, 97–114 (2001)
44. Stefanowski, J., Wilk, S.: Rough sets for handling imbalanced data: combining filtering and rule-based classifiers. Fundamenta Informaticae 72, 379–391 (2006)
45. Stefanowski, J., Wilk S.: Improving rule based classifiers induced by MODLEM by selective pre-processing of imbalanced data. In: Proc. of the RSKD Workshop at ECML/PKDD, Warsaw, pp. 54–65 (2007)
46. Stefanowski, J., Wilk, S.: Selective pre-processing of imbalanced data for improving classification performance. In: Song, I.-Y., Eder, J., Nguyen, T.M. (eds.) DaWaK 2008. LNCS, vol. 5182, pp. 283–292. Springer, Heidelberg (2008)
47. Van Hulse, J., Khoshgoftarr, T., Napolitano, A.: Experimental perspectives on learning from imbalanced data. In: Proc. of the 24th International Conference on Machine Learning (ICML 2007), pp. 935–942 (2007)
48. Wang, B., Japkowicz, N.: Boosting support vector machines for imbalanced data sets. In: An, A., Matwin, S., Raś, Z.W., Ślezak, D. (eds.) Foundations of Intelligent Systems. LNCS (LNAI), vol. 4994, pp. 38–47. Springer, Heidelberg (2008)
49. Weiss, G.M.: Mining with rarity: a unifying framework. ACM SIGKDD Explorations Newsletter 6(1), 7–19 (2004)
50. Weiss, S.M., Indurkhya, N.: Predicitive Data Mining. Morgan Kaufmann, San Francisco (1999)
51. Wilk, S., Slowinski, R., Michalowski, W., Greco, S.: Supporting triage of children with abdominal pain in the emergency room. European Journal of Operational Research 160(3), 696–709 (2005)
52. Zak, J., Stefanowski, J.: Determining maintenance activities of motor vehicles using rough sets approach. In: Proc. of Euromaintenance 1994 Conference, Amsterdam, pp. 39–42 (1994)