

Extending Rule-based Classifiers to Improve Recognition of Imbalanced Classes

Jerzy Stefanowski¹ and Szymon Wilk^{1,2}

¹ Institute of Computing Science, Poznań University of Technology,
ul. Piotrowo 2, 60-965 Poznań, Poland
`jerzy.stefanowski@cs.put.poznan.pl, szymon.wilk@cs.put.poznan.pl`

² Telfer School of Management, University of Ottawa,
55 Laurier Ave East, K1N 6N5 Ottawa, Canada
`wilk@telfer.uottawa.ca`

Abstract. This paper deals with inducing rule-based classifiers from imbalanced data, where one class (a minority class) is under-represented in comparison to the remaining classes (majority classes). We discuss reasons for bias of standard classifiers toward recognition of examples from majority classes and misclassification of the minority class. To avoid limitations of sequential covering approaches, we present a new approach to improve sensitivity of the rule based classifier. It includes a modification of the structure of sets of rules, where for majority classes minimal sets of rules are still induced while the rule set for the minority class is generated by the algorithm, called EXPLORE. This algorithm produces rules being more general and supported by more learning examples than rules from the minimal set. The usefulness of the new approach is verified in comparative experiments on several imbalanced data sets.

1 Introduction

Knowledge discovery in general, and one of its sub-part – data mining – in particular, have received a growing interest both from research and industry in last years. Its main aim is to look for previously unknown relationships or patterns representing knowledge hidden in real-life data sets [14]. The typical representations of knowledge discovered from data are: associations, trees or rules, relational logic clauses, functions, clusters or taxonomies, characteristic descriptions of concepts [14, 26, 19]. In this paper we focus on the rule representation. More precisely, we are interested in *classification* problems, where discovered knowledge represents a function mapping objects (examples), described by a fixed set of *attributes* (features) to *decision classes* (concepts). Rules considered in such problems are called *decision* or *classification rules* and they are expressed in the form:

IF (*condition part*) THEN (*decision class*),

where condition part is formed as a conjunction of elementary conditions – tests on values of attributes and a decision part of the rule indicates the assignment of an object, which satisfies the condition part, to the given decision class.

In data mining other types of rules are also considered, e.g., association rules, action rules [14, 26]. In the text hereafter we will use the general term “rules” to refer specifically to decision rules, as they are of our primary interest.

Symbolic expressiveness of the rule-based representation is treated as more comprehensible and human-readable than other representations (see discussions in [25, 26, 30]). A set of rules may be also more compact than a decision tree [30]. Moreover, rules constitute “blocks” of knowledge, and experts can more easily analyse single rules [25, 36]. Finally, rules were successfully used in several applications – a review paper was written by Simon and Langley [21].

Induction of rules has been intensively studied in machine learning [26, 28] and many algorithms have been proposed, for reviews see [8, 28, 26]. The majority of them try to generate rules following a *sequential covering* strategy. They are focused on 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 coming objects. In this - classification oriented - perspective, rules and the strategy of using them form a *classifier*. Although sequential covering algorithms have shown to be quite effective for this perspective, there are also other algorithms which provide more numerous sets of rules than the former ones. Such rules are also often characterized by better descriptive properties, e.g. they are supported by a larger number of learning examples (see [37, 14]). In general they could better characterize some regularities hidden in data what corresponds to, so called, descriptive perspective of knowledge discovery [43].

On the other hand, such rules could be also useful for handling more difficult classification problems. One of the main sources of difficulty is a *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 has been receiving a growing research interest the machine learning and data mining communities (for some reviews see, [3]). Several methods have already been proposed to improve performance of various types of classifiers, not only rule-based ones. In general, one can distinguish two kinds of approaches [18]. The first approach, which is classifier-independent, relies transforming the original data set to change the balance between classes, e.g., by resampling. 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 imbalanced in the learning set. We will consider two ways of doing it: either changing the rule induction phase or modifying classification strategy.

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. Its main principle 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 Vanderpooten in [37]. Thus, using such rules for the minority class, while preserving the original set of rules for the majority classes, improves the chance that an example from the minority class is correctly recognized.

Within our approach we try to maintain a comprehensible structure of the symbolic knowledge representation by discovering additional rules, still hidden in data which was not revealed in the minimal set. This property is consistent with some arguments behind rule paradigms. It additionally 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. Discussing the usefulness of the EXPLORE algorithm for providing such comprehensible patterns is the second aim of this paper.

Finally, we present results of experiments where the performance of our approach is compared against a typical sequential covering algorithm LEM2 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 basic categories of rule induction algorithms and strategies for building final classifiers on the basis of rules. Then, in section 3 we describe of the EXPLORE algorithm, which is employed by our approach. Our previous experience with using this algorithm is also described. The next section is devoted to a short review of methods for handling imbalanced data. Then, we present our approach to change the structure of a rule-based classifier. In section 6 we experimentally evaluate its usefulness in a comparative study. Final remarks and discussion of future research are provided in the last section.

We should remark that this is kind of a summary paper, which partly includes some results other papers of Stefanowski and coauthors on the EXPLORE algorithms [37] and from our joint research with Grzymala-Busse on handling imbalanced data by modifying rule-based classifiers [13].

2 Various Approaches to Rule Induction

This chapter should give only basic information on rule induction and classification strategies which are necessary for presenting a new method. More comprehensive reviews on this topic could be found in [8, 19, 36]

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 $\{K_j : j=1, \dots, k\}$. A decision rule r assigning objects to a class K_j is represented in the following form:

$$\text{if } P \text{ then } Q$$

where $P = p_1 \wedge p_2 \wedge \dots \wedge p_n$ is a *condition part* of r and Q is a *decision part* of r indicating that an object 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 numerical attribute other relations (e.g., greater than) are possible.

We can say that a decision table DT contains *learning* examples for inducing rules, thus it is called a *learning* set. Thus, 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.

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 rules 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 ones.

2.2 Different Perspectives of Rule Induction and Evaluation

In general induction of decision rules can be performed according to different perspectives. The most common ones [36, 43]:

- 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. The rules are then combined with an additional strategy defining how to use them to produce a final prediction for a new object – such combination constitutes a *classifier*. This perspective has been extensively studied in *machine learning* and several approaches for deriving *rule-based classifiers* have already 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 decision users [14]. These patterns (represented as *rules*) aim at clarifying dependencies between values of attributes and decision classes [37] 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 [14, 36].

The two perspectives of rule induction do not only have different goals – here are other profound differences between them. One of the main distinctions consists in different evaluation criteria [37] of constructed rules. In the classification-oriented induction, a *complete set of rules is evaluated* as a classifier. The evaluation criterion is usually single and defined as the classification (predictive) accuracy or similar prediction measures (e.g., based on confusion matrix – see Section 4) of a rule-based classifier using these rules. The criterion is evaluated in experimental way by splitting data into learning and testing sets.

In the descriptive-oriented induction, *each rule is evaluated individually and independently* as possible representation of an interesting pattern. This is definitely 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 the interests and expertise of users. To support them, 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 condition and decision parts of a rule and a data set in 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 [15]. Below we present two measures: *support* and *confidence* of a rule, as we refer to them in the further text.

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

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., $sup(r) = |[P \cap Q]|$. The support could be given in relation to the number of examples in U as

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

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

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

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

Let us notice that both these measures characterize two different properties of a rule: support corresponds to the generality of a rule pattern in data while confidence estimate the certainty of assignment to a decision class.

We mention another measure of rule generality, called *coverage* or rule strength, as it is used in the description of the EXPLORE algorithm. It 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 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 follows the sequential covering strategy which historically comes from early Michalski's works on the family of the AQ algorithms. It is also known as *separate-and-conquer* strategy and used in several inductive logic programs [8].

Figure 1 shows the basic idea of the sequential covering strategy. This strategy sequentially generates a *minimal set* of decision rules for each decision class.³ In each run it accepts as input a set of positive and negative examples of the class K_j and provides as output a set of rules R covering all positive examples of the class and not covering any 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 all positive examples that match this rule from consideration. This process is being repeated while some positive examples of the decision concept remain uncovered.

The function *Find single best rule* produces a candidate for a rule which in general should cover as many as possible positive examples of the target class and no negative ones (for consistent data) or a limited number of negative ones (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 specialization 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 [26] or LEM2 [10]).

³ There are also some versions of this strategy which do not sequentially go through classes but attempt to consider all classes together in one general loop, however still maintaining the principle idea of recursively learning best rule, removing covered examples, etc.

Fig. 1. Sequential covering strategy

```

Procedure Sequential_covering input ( $K_j$ : class;
 $E$  : its positive learning examples;  $N$  : and its negative examples
output  $\mathcal{R}$ : set_of_rules);
begin
   $R \leftarrow \emptyset$  { initialize rules };
  while  $E \neq \emptyset$  do
    begin
       $r \leftarrow \text{Find\_single\_best\_rule}(K_j, E, N)$ 
       $\text{Cover}(r) \leftarrow \text{set\_of\_positive\_examples\_covered\_by\_}r$ 
      if rule_stopping_conditions( $r$ ) then exit;
       $E \leftarrow E - \text{Cover}(r)$ ;
       $R \leftarrow R \cup r$ 
    end
   $R \leftarrow \text{Postprocess}(R)$ 
end

```

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 [8]. For instance, in LEM2 algorithm, which will be considered in the experimental section of this paper, Grzymala proposed to select conditions in the following way [10]:

1. Choose a condition that results in the maximum support of a candidate rule,
2. If tie occurs, choose a condition that leads to the maximum 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, 33], Laplace estimate or more flexible m -estimate [9]. 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 [8].

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. Different rule pruning techniques are summarized in [7].

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 limited or no interest, which is undesirable from the discovery-oriented perspective. These shortcomings result directly

from following the sequential covering strategy described in the previous section. This strategy excludes from consideration learning examples that has been already covered by generated rules, thus, some interesting rules cannot be discovered. This happens especially when different patterns are shared by a large proportion of common examples. Moreover, the covering strategy aims at covering all positive learning examples, therefore, in the 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 these phenomena is given in [37].

In order to overcome the above limitations, other induction strategies and algorithms have been proposed. The most radical solution is to produce so called *exhaustive* set of rules, which contains *all* (often discriminant only) rules that can be induced on the basis of positive examples of the class. Examples of such approach include the *dropping condition* technique, described in [10] or *Boolean reasoning* approach to look for local object reducts [32] – specific for rough set theory. However, time complexity for the second choice is exponential and using this kind of approach may be not practical for larger input data files – 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 part of them is usually interesting for him (e.g. notice that in this option may generate many specific rules supported by too less 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 from this category is the BRUTE algorithm introduced in [31]. 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 short 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 the most accurate rules on learning set. Experimental results of using the set of the top 50 rules in a classifier showed that it outperformed CART and C4 trees. Quite similar idea is present *Data Surveyor system* described in [16].

Finally, the last group of approaches includes adaptations algorithms for association rule mining. These rule are changed into the form where the right hand size of the rule contains the class label. Rules should also satisfy a predefined minimum support (they are called frequent ones) and a minimum confidence ⁴. The key issue is to adopt in a proper way search strategies coming from algorithms for mining frequent items, e.g. to construct iterative sequential extension approach similar to Apriori which efficiently prune the candidates. In [14] there is a short review of some proposals, e.g. the Liu et al method of associative classification [24]. It is also showed that such rule sets were empirically found to be more accurate than decision trees.

⁴ These requirements are similar to ones present in EXPLORE - see section 3

2.5 Classification Strategies

In a the classification-oriented perspective a set of rules is used to classify new objects, i.e. objects unseen in the learning phase by matching these objects to condition parts of the 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 object 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 cases. The first one is unique matching to one or more rules from a single class. Two other two cases are matching to multiple rules indicating different classes or not matching any rules at all. In both cases a suggestion is ambiguous, thus proper resolution strategy is necessary. Review of different strategies is given in [36]. Below we briefly summarize the classification strategy introduced by Grzymala in LERS [11] as it is employed in our experiments. In case of ambiguous matching the decision to which class classified object e belongs is made on the basis of voting and e is assigned to the strongest class. 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 as the following expression:

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

The class K_j for which such defined support is the largest 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 not only on the bases of the support of identified rules, but also 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):

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

Again, the class K_j with the largest support is the winner and the 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, where imbalanced data has to be dealt with.

3.1 Presentation of the Algorithm

The *EXPLORE* algorithm, first presented in [27], 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 a *coverage* of a rule : the user can expect that 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*: the rule should be generally short
- *number of rules*: total number rules for all decision classes – this additional criterion which refers to the set of rules (not a single rule), it means that the number of rules which should be limited for cognitive reasons.

These requirements are used to impose restrictions on the rule space explored during induction. Finally, the algorithm can handle inconsistent examples either by using rough set approach or by defining appropriate threshold for confidence of induced rules.

Exploration of the rule space is performed using a procedure which is *repeated* for each concept K 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 a stopping condition SC , reflecting the impossibility to fulfill the requirements, is met. A formal description of *EXPLORE* is presented in figure 2. Brief comments to this description are given below (more precise discussion is presented in [37] and more implementation details can be found in [27]).

An initial list LS representing elementary conditions is created by analysis description of positive examples and given as an input to *EXPLORE*, for more precise description see [34]. Obviously, conditions in LS must cover at least one example from K ; they may also be subject to specific constraints on the syntax. This initial list is first pruned so as to discard conditions which directly correspond to rules as well as those which already satisfy SC and thus cannot give rise to rules (subprocedure *Good.Candidates*). Conditions remaining in LS are then combined to form *complexes* (i.e. conjunctions of elementary conditions) which will be candidates for being the condition part of rules. This is achieved by subprocedure *Extend* which, at iteration k , creates conjunctions complexes of size $k + 1$ by extending candidate conjunctions of size k stored into a queue using conditions from LS (some necessary conditions for the newly created conjunctions to be candidates are already checked at this stage - see description). While extending conjunctions we can use the monotonicity principle known from the Apriori algorithm, saying that all subsets of a candidate conjunction must also be sufficiently strong [14]. The resulting conjunctions are then tested by subprocedure *Good.Candidates*.

Fig. 2. The main procedure of the EXPLORE algorithm

```

Procedure Explore
input ( $LS$ : list_of_valid_elementary_conditions;  $SC$ : stopping_conditions;
output  $\mathcal{R}$ : set_of_rules);
begin{Main search procedure}
     $\mathcal{R} \leftarrow \emptyset$ 
    Good_Candidates( $LS, \mathcal{R}$ ); { $LS$  is a list of valid elementary conditions  $s_1, s_2, \dots, s_n$ 
        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( $LC, \mathcal{R}$ );
             $Q \leftarrow Q \cup LC$  {place all conjunctions from  $LC$  at the end of  $Q$ }
        end
    end;

Procedure Extend( $C$ : complex; var  $L$ : list_of_conjunctions);
{ This procedure puts in list  $L$  extensions of conjunctions  $C$  that are potential candidates}
begin
    Let  $k$  be the size of  $C$  and  $h$  be the highest index of the elementary condition involved in  $C$ ;
     $L \leftarrow \{C \wedge s_{h+i} \text{ where } s_{h+i} \in LS \text{ and such that all the } k \text{ subconjunctions of } C \wedge s_{h+i}$ 
        of size  $k$  and involving  $s_{h+i}$  belong to  $Q$  ( $i = 1, \dots, n - h$ )\}
end;

Procedure Good_Candidates(var  $L$ : list_of_conjunctions; var  $\mathcal{R}$ : set_of_rules);
{ This procedure prunes list  $L$ , discarding:
    - conjunctions whose extension cannot give rise to rules due to  $SC$ 
    - conjunctions corresponding to rules, which are stored into  $\mathcal{R}$  }
begin
    for each  $C \in L$  do
        begin
            if  $C$  satisfies  $SC$  then  $L \leftarrow L \setminus \{C\}$ 
            else
                if  $confidence(C) \geq \alpha$  then { $\alpha = 1$  for totally discriminant rules}
                    begin
                         $\mathcal{R} \leftarrow \mathcal{R} \cup \{C\}$ ;
                         $L \leftarrow L \setminus \{C\}$ 
                    end
                end
            end
        end
    end;

```

In general stopping conditions SC can be defined according to requirements expressing various expert's expectations, e.g. imposed on coverage, length, number of rules, etc. In our experiments presented in section 6, we mainly consider requirements referring to a minimal coverage level l (smallest percentage of positive examples that a rule must cover). The corresponding stopping condition for a conjunction C currently examined is thus simply: $coverage(C) < l$. Let us remark that stopping conditions restricts exploration space and reduces computational costs. If the user does not define any requirements the algorithm will produce all rules, which is at the risk of exponential costs, see the evaluation of complexity in [36]. An examples of using EXPLORE and tuning SC is presented in the two next subsections.

Besides basic requirements represented in the stopping conditions, EXPLORE can be easily adopted to handle additional expectations of the decision maker 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 possible to zoom search on some specific subsets of examples. Such approach is typical for interactive knowledge discovery tools and it is described in [35, 36].

3.2 An Example of Comparing Minimal Sets of Rules vs. Rules Induced by EXPLORE

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 the homogeneous fleet of buses [44]. 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 [33]). The generated set contained the three following rules covering all learning examples (numbers in brackets refer 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 is a very accurate classifier 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 the technical

condition of examined busses. On the other hand, practical measurements of this symptom were the most difficult at the diagnostic stand. So, the diagnostic experts were interested in discovering other rules, formulated in terms of symptoms that were easier to collect. They also wanted to get strong rules covering more than 51% 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 ($s7 \geq 2.1$ l/1000km) then (technical state=bad) [24]
10. if ($s3 \geq 61$ %) & ($s4 \leq 444$ Nm) then (technical state=bad) [28]
11. if ($s3 \geq 61$ %) & ($s8 < 120$ 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 Experiences with Using EXPLORE

Let us repeat 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 [37]. Experiments on several data sets from the UCI repository [1] shown that it is possible to determine a range of values for this threshold which lead to good rules regarding their classification accuracy, the average coverage, the average length of the rules and the number of the rules. In table 1 we present exemplary results obtained for the *Congress Voting* data. The last line contains results for the minimal set of rules obtained 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 ⁵.

Going on with 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 [38] we proposed a solution to this problem and introduced a *hybrid approach*, where the first level

⁵ 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.

Table 1. Characteristics of rules induced by EXPLORE vs the minimal set of rules induced by LEM2

Data set	Stopping Condition (min. relative strength)	Number of rules	Average rule coverage v [# examples]	Average rule length [# conditions]	Classification accuracy in [%]
Voting	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

of representation are 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 of it see [36, 38]). The classification strategy for new examples is a two stage approach. The new example is first classified by rules. If there is no matching or the matching is ambiguous, 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 [38]. 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. The similar improvements were observed for another medical case study.

Such an additional effect of the threshold tuning procedure on the sensitivity to the minority class has been a direct inspiration for our current research on imbalance data which is presented in the next sections.

4 Handling Imbalanced Data

Many learning algorithms are constructed 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 a different frequency. A good example is medicine, where databases regarding a rare, but dangerous, disease usually contain a smaller group of patients requiring a special attention, while there is a 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 examples. Survey papers as [42, 3] report other real technical or engineering problems as, e.g., detection of oil spills in satellite radar images, detection of fraudulent telephone calls or credit card transactions, prediction of telecommunication equipment failures, information retrieval and filtering.

If the imbalance in the class distribution is high, 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 final classifiers are also biased toward recognition of the majority classes and they usually have difficulties (or even are unable) to classify correctly new objects from the minority class. In [23] authors described an information retrieval system, where the minority class (being of a primary importance) contains only 0.2% examples. Although the classifiers achieved the accuracy close to 100%, they were useless because they failed to deliver the requested documents from this class. Similar degradation of classifier’s performance for the minority classes was reported for other imbalanced problems [4, 12, 18, 20, 42].

The class imbalance also affects set 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, covering 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 for a classifier, but also when evaluating its performance. Indeed, an 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 [12]. However, in general there is trade-off between these two measures, i.e., improving the sensitivity too much may lead to deterioration of specificity at the same time - see experimental results in [40]. Thus, some measures summarizing both points of view are considered. One of them is *G-means* [20], calculated as geometric mean 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 false positive rate ($1 - \text{Specificity}$) along different threshold values characterizing the performance of the studied classifier. The quality of the classifier performance is reflected by the area under a ROC curve (so called AUC measure) [3, 42].

A small number of examples in the minority class (“the lack of data”) is not the only source of difficulties for 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 the majority ones – see [20, 22] for experiments and discussion. Japkowicz in her experimental study [17] also showed that the class imbalance becomes even more difficult problem particularly when the minority class contains a number of 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 [18, 42]. In general, one can distinguish two kind 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* which 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 on taking 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 or their combinations, were experimentally shown to be quite good [2, 41, 20, 40].

Other approaches proposed in the literature modify either induction or classification strategy, assign weights to examples, and use boosting or other combined classifiers. 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 Grzymala’s work [12] on increasing sensitivity of LEM2 rule classifiers by changing LERS classification strategy - as described in subsection 2.5. Necessary changes of the strategy are somehow limited to formulas for calculating support for a given class that are presented in Section 2.5. The main idea of the Grzymala’s 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. So. for the majority classes it should be equal to 1 while for the minority class it should be greater than 1.0. As a result, during classification of a new coming object, such minority class rules have an increased chance to predict the final classification for the new object.

Another problem is selecting a proper value for the support multiplier. In general, the sensitivity of a classifier increases with the increase of support multiplier. However at the same time, specificity decreases, thus it is important to

identify a proper value of this parameter leading to a kind of trade off between. In [12] Grzymala proposed to maximize a measure called $gain = sensitivity + specificity - 1$. Following it, a value of support multiplier was established experimentally in a loop, where in each iteration the support multiplier was stepwise increased, the classifier was evaluated on extra validation examples, and the loop stopped as soon as a value of the gain measure had decreased. The value of support multiplier resulting in the best gain was used in the final classifier. Experimental results confirmed that this approach outperformed the use standard LEM2 classifier for many imbalanced medical data sets [12, 13].

Few other researchers tried to develop *less greedy search* strategy while looking for rules (an example is a version of the Brute algorithm described in [31] or a specific genetic algorithm [42]) or to change the *inductive bias of the algorithm*, e.g. Holte et al. modified the rule induction algorithm CN2 to improve its performance for small disjuncts referring to rare examples from the minority class. Moreover, Weiss describes hybrid and two-phase rule induction [42], where one part focuses on optimizing sensitivity while the other corresponds to optimizing specificity. Other approaches may use knowledge about prior distribution of probabilities or transforming the task to cost sensitivity learning [42].

5 A Method for Extending A Set of Rules for the Minority Class

In this section we briefly describe our classifier-specific approaches to handle imbalanced data – that together with Grzymala’s proposal modifying rule support for the minority class we evaluate experimentally in Section 6. Let us remark that both approaches assume an initial classifier uses a minimal set of rules. In general it could be induced by any good sequential covering algorithm but in this paper we have chosen the LEM2 algorithm [11] and a classification strategy described in Section 2.5.

Both approaches are 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.

New approach, called *Replacing Rules for the Minority Class*, has been sketched the first time in [38]. Generally speaking, unlike the first approach which addresses this issue by artificially increasing the support of rules for the minority it improves sensitivity for the minority class by replacing the minimal set of rules for this class by a non-minimal set of stronger rules generated by the EXPLORE algorithm. Since these rules have better (greater) support than the original ones, unlike in the previous approach 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 support / coverage below a given threshold are discarded). Setting the right value

Fig. 3. Replacing rules for the minority class

Procedure Replace_rules (**input** K_{min} : the minority class;
 \mathcal{R} : initial minimal set of rules;
 L : learning examples; T : validation examples;
output \mathcal{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}
 $\mathcal{R}_{maj} \leftarrow$ rules from R pointing at the majority classes
 $\mathcal{R}_{min}^{min_sup} \leftarrow$ 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
 $\mathcal{R}_{min}^{sup} \leftarrow$ select these rules from $\mathcal{R}_{min}^{min_sup}$ for which coverage $\geq sup$
 $\mathcal{R}^{sup} \leftarrow \mathcal{R}_{min}^{sup} \cup \mathcal{R}_{maj}$
 $gain \leftarrow$ evaluate \mathcal{R}^{sup} on T
memorize $gain$ and \mathcal{R}^{sup}
end
 $\mathcal{R}^{final} \leftarrow \mathcal{R}^{sup}$ corresponding to the best observed $gain$
end

of this threshold is a crucial point. 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 lower thresholds. We iteratively examine possible coverage thresholds within the identified range (have a kind of step value). In each iteration of the loop we use EXPLORE to generate rule for the minority class with the required coverage set 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 with the gain measure, as in the Grzymala's approach (at least to be consistent for comparative experiments). 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 minimal set of rules generated by LEM2 [10]. Moreover, we considered a variant of such LEM2 classifier with a modified classification strategy expanded with a support multiplier (such modification 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 determined algorithms (which is a case of our rule classifiers). Additionally, we report the total classification accuracy. Values of all these measures are presented as percentage. They are estimated as means in the k -fold cross validation. Moreover, to minimize the influence of splitting a data set on the classification results obtained for all approaches, the division into folds was performed only once and the same subsets of objects were used in experiments ⁶.

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 class, $R_{Pos} = N_{Pos}/N$ – 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%

Experiments were carried out 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. Let us notice that nearly all data sets, except *german credit*, are from a medical domain. Data sets, which originally included more than two classes, were transformed to binary ones, by

⁶ In our previous joint research with J.Grzymala-Busse [13] - we have already carried out some experiments. So, some of values for LEM2 are coming from [13]

collapsing all 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’s method based on clustering with merging intervals [5]. Table 2 lists the data sets along with their basic characteristics.

Table 3. Results for the original LEM2 algorithm

Data set	Sensitivity	Specificity	G-mean	Overall accuracy	number of rules
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

Table 4. Best results of increasing rule support by multipliers

Data set	Multiplier	Sensitivity	Specificity	G Mean	Accuracy
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

Results for all compared approaches are presented in Tables 3 - 5. The approach to extend the classification strategy with a support multiplier is consistent with the Grzymala’s proposal [12] of optimizing the gain measure (see also its description in section 4) – the best values of the multiplier are listed in 3. For our approach we additionally present values of the rule support threshold for the minority class and the number of replaced rules. We compare results of both these approaches suited to imbalanced data to the standard rule-based classifier using the Wilcoxon Signed Ranks test (with $\alpha = 0.05$). Considering sensitivity

Table 5. Results for the replacing rule approach

Data set	Support threshold	Sensitivity	Specificity	G-mean	Overall accuracy	number of rules
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

and G-mean both approaches outperformed it, and the difference was 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 was also significantly increased. According to discussion from [13] it may be possible to block increasing the number of replaced rules - to be more similar to the number in the original minimal set. It concerns the following data sets: Abdominal pain, hepatitis, pima and urology.

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 showed they are too much biased toward 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 support) that improves a chance of a classification strategy to indicate 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 (i.e., coverage greater than a given threshold).

In a series of experiments we have compared our approach to a baseline classifier with minimal rules and a basic classification strategy, and a classifier with a

classification strategy expanded with a strength multiplier. Experimental results have showed that both our approach and the approach with a 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 the balance between classes. On the other hand, our approach does not modify learning data, rules discovered by EXPLORE correspond to really existing patterns and they are potentially still comprehensible for human experts.

Further directions for our research could 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 [40]. Briefly speaking, it combines selective filtering of the majority classes with limited over-sampling of the minority class. Unfortunately, more elaborated presentation is beyond the scope and limit of this paper.

Acknowledgment: 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 University of Kansas in comparative experiments with LEM2.

References

1. Asuncion, A., Newman, D.J., UCI Machine Learning Repository [<http://www.ics.uci.edu/~mlearn/MLRepository.html>]. Irvine, CA: University of California, School of Information and Computer Science (2007).
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), (2004) 20–29
3. Chawla, N.: Data mining for imbalanced datasets: An overview. In: Maimon O., Rokach L. (eds.) *The Data Mining and Knowledge Discovery Handbook*, Springer 2005, 853–867.
4. Chawla, N., Bowyer, K., Hall, L., Kegelmeyer, W.: SMOTE: Synthetic Minority Over-sampling Technique. *J. of Artificial Intelligence Research*, vol. 16 (2002) 341–378.
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*, Simulation Councils Inc., (1995), 294–297.
6. Clark P., Niblett T.: The CN2 induction algorithm. *Machine Learning*, vol. 3, (1989) 261–283.
7. Furnkranz J.: Pruning algorithms for rule learning. *Machine Learning*, 27(2) (1997) 139–171.
8. Furnkranz J.: Separate-and-conquer rule learning. *Artificial Intelligence Review*, vol. 13 (1) February (1999), 3–54.

9. Dzeroski S, Cestnik B., Petrovski I.: Using the m-estimate in rule induction, *Journal of Computing and Information Technology*, (1993), vol. 1, 37-46.
10. 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*, Kluwer, (1992) 3-18.
11. Grzymala-Busse J.W.: Managing uncertainty in machine learning from examples. In: *Proc. 3rd Int. Symp. in Intelligent Systems*, Wigry, Poland, IPI PAN Press, (1994) 70-84.
12. Grzymala-Busse, J.W., Goodwin, L.K., Grzymala-Busse, W.J., Zheng, X., An approach to imbalanced data sets based on changing rule strength. *Learning from Imbalanced Data Sets*, AAAI Workshop at the 17th Conference on AI, AAAI-2000, Austin, TX, July 30-31, 2000, 69-74.
13. Grzymala-Busse J.W., Stefanowski J. Wilk Sz.: A comparison of two approaches to data mining from imbalanced data. In *Proc. of the KES 2004 - 8-th Int. Conf. on Knowledge-based Intelligent Information & Engineering Systems*, Springer LNCS vol. 3213, (2004) 757-763.
14. Han J., Kamber M.: *Data mining: Concepts and techniques*, San Francisco, Morgan Kaufmann, (2000).
15. Hilderman R.J., Hamilton H.J.: *Knowledge Discovery and Measures of Interest*. Kluwer Academic, Boston, (2002).
16. Holsheimer M., Kersten M.L., Siebes A., *Data Surveyor: Searching the Nuggets in Parallel*, In: Fayyad et al. (eds.) *Advances in Knowledge Discovery and Data Mining*, AAAI/MIT-Press Cambridge, Mass. (1996), 447-467.
17. Japkowicz N., Stephen, S., *The Class Imbalance Problem: A Systematic Study*. *Intelligent Data Analysis*, 6 (5), (2002) 429-450.
18. Japkowicz, N., *Learning from imbalanced data sets: a comparison of various strategies*. *Learning from Imbalanced Data Sets*, AAAI Workshop at the 17th Conference on AI, AAAI-2000, Austin, TX, July 30-31, 2000, 10-17.
19. Klossen W., Żytkow J.M.: *Handbook of Data Mining and Knowledge Discovery*. Oxford Press, (2002).
20. Kubat M., Matwin S.: Addressing the curse of imbalanced training sets: one-side selection. In: *Proc. of the 14th Int. Conf. on Machine Learning*, (1997) 179-186.
21. P. Langley, H.A. Simon, *Fielded applications of machine learning*, In: R.S. Michalski, I. Bratko, M. Kubat (eds.), *Machine learning and data mining*, John Wiley & Sons, (1998), 113-129.
22. Laurikkala, J.: *Improving identification of difficult small classes by balancing class distribution*. Tech. Report A-2001-2, University of Tampere (2001).
23. Lewis D., Catlett J.: *Heterogenous uncertainty sampling for supervised learning*. In *Proc. of 11th Int. Conf. on Machine Learning*, (1994), 148-156.
24. Bing Liu, Wynne Hsu, and Yiming Ma.: *Integrating classification and association rule mining*. In: *Proc. of the 4th Int. Conference on Knowledge Discovery and Data Mining (KDD-98)*, 1998.
25. Michalski R.S.: *A theory and methodology of inductive learning*, In: R.S. Michalski, J.G. Carbonell and T.M. Mitchell, (eds.), *Machine Learning: An Artificial Intelligence Approach*, Morgan Kaufman, 1983, 83-134.
26. Michalski R.S. , Bratko I., Kubat M.(eds.), *Machine learning and data mining*, John Wiley & Sons, (1998).
27. Mienko R. , Stefanowski J., Toumi K., Vanderpooten D.: *Discovery-Oriented Induction of Decision Rules*. Cahier du Lamsade no. 141, Paris, Université Paris Dauphine, septembre 1996.

28. Mitchell T.: Machine learning, McGraw Hill, (1997).
29. Pawlak Z., Rough sets. Theoretical aspects of reasoning about Data. Kluwer Academic Publishers, (1991).
30. Quinlan J.R., C4.5: Programs for Machine Learning, Morgan Kaufmann, (1992).
31. Riddle P., Segal R., Etzioni O., Representation design and Brute-force induction in a Boeing manufacturing fomain. Applied Artificial Intelligence Journal, vol. 8, (1994), 125-147.
32. Skowron A.: Boolean reasoning for decision rules generation. In J. Komorowski and Z.W. Ras, editors, Methodologies for Intelligent Systems, LNAI 689, Springer Verlag, Berlin, (1993), 295-305.
33. 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'98, Aachen, (1998) 109-113.
34. Stefanowski J.: Handling continuous attributes in discovery of strong decision rules, In: Proc. of the RSCTC98, LNAI 1424, Springer Verlag, 1998, 394-401.
35. Stefanowski J., Borkiewicz R.: Interactive rule discovery of decision rules, w: Proceedings of VIIIth Intelligent Information Systems, June 1999, Wyd. Instytutu Podstaw Informatyki PAN, Warszawa, 112-116.
36. Stefanowski J.: Algorithms of rule induction for knowledge discovery. (In Polish), Habilitation Thesis published as Series Rozprawy no. 361, Poznan Univeristy of Technology Press, Poznan (2001).
37. Stefanowski J., Vanderpooten D.: Induction of decision rules in classification and discovery-oriented perspectives. International Journal of Intelligent Systems vol. 16 (1), (2001), 13-28.
38. 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, vol. 10 (2001) 97-114.
39. Stefanowski, J., Wilk, S.: Rough sets for handling imbalanced data: combining filtering and rule-based classifiers. Fundamenta Informaticae, vol. 72, (2006) 379-391.
40. 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, (2007) 54-65.
41. Van Hulse J., Khoshgoftarr T., Napolitano A.: Experimental perspectives on learning from imbalanced data. In: Proceedings of ICML 2007, (2007), 935-942.
42. Weiss, G.M.: Mining with rarity: a unifying framework. ACM SIGKDD Explorations Newsletter, vol. 6 (1), (2004) 7-19.
43. Weiss S.M., Indurkha N., Predictive Data Mining, San Francisco, Morgan Kaufmann, (1999).
44. Zak J., Stefanowski J., Determining maintenance activities of motor vehicles using rough sets approach, In: Proc. of Euromaintenance'94 Conference, Amsterdam (1994), 39-42.