Granular Computing
https://doi.org/10.1007/s41066-018-0097-2

ORIGINAL PAPER

Granular computing-based approach of rule learning for binary classification

Han Liu[1] · Mihaela Cocea[2]

Received: 21 February 2018 / Accepted: 26 April 2018

Abstract
Rule learning is one of the most popular types of machine-learning approaches, which typically follow two main strategies: 'divide and conquer' and 'separate and conquer'. The former strategy is aimed at induction of rules in the form of a decision tree, whereas the latter one is aimed at direct induction of if–then rules. Due to the case that the divide and conquer strategy could result in the replicated sub-tree problem, which not only leads to overfitting but also increases the computational complexity in classifying unseen instances, researchers have thus been motivated to develop rule learning approaches through the separate and conquer strategy. In this paper, we focus on investigation of the Prism algorithm, since it is a representative one that follows the separate and conquer strategy, and is aimed at learning a set of rules for each class in the setting of granular computing, where each class (referred to as target class) is viewed as a granule. The Prism algorithm shows highly comparable performance to the most popular algorithms, such as ID3 and C4.5, which follow the divide and conquer strategy. However, due to the need to learn a rule set for each class, Prism usually produces very complex rule-based classifiers. In real applications, there are many problems that involve one target class only, so it is not necessary to learn a rule set for each class, i.e., only a set of rules for the target class needs to be learned and a default rule is used to indicate the case of non-target classes. To address the above issues of Prism, we propose a new version of the algorithm referred to as PrismSTC, where 'STC' stands for 'single target class'. Our experimental results show that PrismSTC leads to production of simpler rule-based classifiers without loss of accuracy in comparison with Prism. PrismSTC also demonstrates sufficiently good performance comparing with C4.5.

Keywords Machine learning · Decision tree learning · Rule learning · Rule-based classification · Granular computing

1 Introduction

Rule learning is one of the most popular types of machine learning approaches, which can be achieved through two strategies, namely 'divide and conquer' and 'separate and conquer'. The former strategy is aimed at learning rules that are represented in the form of a decision tree, so this strategy is also referred to as 'top-down induction of decision trees (TDIDT)'. The latter strategy is aimed at learning if–then rules directly from training data, so this strategy is also referred to as 'covering approach'. In the rest of the paper, we refer to TDIDT as decision tree learning and to the covering approach as rule learning.

As discussed in Liu and Cocea (2018b) and Cendrowska (1987), the main difference between decision tree learning and rule learning is in terms of their strategy of specializing a rule through appending rule terms. In particular, decision tree learning is attribute oriented, i.e., it is aimed at selecting an attribute for labelling each non-leaf node and several branches are then split from this node towards growing each branch of the tree. In this context, each branch of a decision tree can be converted into a rule and the growth of different branches (specializing different rules) are in parallel. In contrast, rule learning is attribute–value oriented, i.e., it is aimed at selecting an attribute–value pair towards specializing a

✉ Han Liu
LiuH48@cardiff.ac.uk

Mihaela Cocea
mihaela.cocea@port.ac.uk

[1] School of Computer Science and Informatics, Cardiff University, Queen's Buildings, 5 The Parade, Cardiff CF24 3AA, UK

[2] School of Computing, University of Portsmouth, Buckingham Building, Lion Terrace, Portsmouth PO1 3HE, United Kingdom

Published online: 04 May 2018

rule. In addition, rules are learned sequentially, which means that the learning of one rule cannot start until the completion of learning of the previous rule. In addition, rules may not necessarily fit into a tree structure.

Popular algorithms of decision tree learning include ID3 (Quinlan 1986) and C4.5 (Quinlan 1993). As argued in Cendrowska (1987), decision tree learning is likely to result in the replicated sub-tree problem, which could not only lead to overfitting of training data but also increase the computational complexity for a rule-based classifier to classify unseen instances, due to the production of a large number of complex rules with many redundant terms. To address the above issue, researchers have been motivated to develop rule-learning algorithms, such as Prism (Cendrowska 1987).

In this paper, we explore further the significance of the rule-learning strategy in comparison with decision tree learning and highlight some granular computing perspectives of the Prism algorithm. Furthermore, we identify some limitations of Prism and propose a new version of this algorithm (referred to as PrismSTC) in the setting of granular computing. The contributions of this paper include that PrismSTC leads to a reduction of complexity without loss of accuracy in comparison with Prism and that the PrismSTC shows sufficiently good performance of classification in comparison with C4.5 (the most popular algorithm of decision tree learning).

The rest of this paper is organized as follows: Sect. 2 provides an overview of decision tree learning and rule-learning methods. In Sect. 3, we describe the procedure of PrismSTC and justify its significance from the perspectives of granular computing and machine learning. In Sect. 4, we report an experimental study and compare PrismSTC with Prism as well as C4.5 in terms of their performance and complexity (i.e., number of rules and terms). In Sect. 5, we summarize the contributions of this paper and suggest further directions for this research area towards further advances in rule learning.

2 Related work

Decision tree learning has been used as a popular approach of machine learning in various application areas, due to the fact that decision tree models are represented in a white box manner. In other words, decision tree models are so transparent that people can clearly identify how outputs are mapped from inputs (Liu et al. 2015, 2016c). In practice, applications of decision tree learning are extensive and varied, such as text classification (Khan et al. 2015), biomedicine (Tayefi et al. 2017), intelligent tutoring systems (Crockett et al. 2017) and transient stability assessment (Rahmatian et al. 2017).

In terms of algorithmic development, decision tree learning has been highly competitive and technically sound, since the ID3 algorithm was developed by Quinlan (1986) with very good performance especially on the chess end games dataset (Quinlan 1983). However, the ID3 algorithm cannot deal with continuous attributes without discretization of the attributes. To be capable of learning directly from continuous attributes, the C4.5 algorithm was developed as an extension of ID3 for effectively dealing with continuous attributes and replacing missing values (Quinlan 1993, 1996). Further to C4.5, a commercial version of decision tree learning algorithm was developed, which is referred to as C5.0 (Kuhn and Johnson 2013).

Decision tree learning methods were extended through the use of pruning methods and a comparison of different pruning methods was made in Esposito et al. (1995). In addition, decision tree learning methods have also been used in the context of ensemble learning (Liu et al. 2016a) for increasing the overall accuracy of classification, such as random forests (Breiman 2001) and gradient boosted trees (Ogutu et al. 2011), which are based on Bagging (Breiman 1996) and Boosting (Freund and Schapire 1996). Over the last decade, decision tree learning methods have also been extended in other different ways. One way is through incorporating cost functions into heuristics for attribute selection towards minimizing the risk of incorrect classification. Some more recent work has been presented in Zhao and Li (2017), Min and Zhu (2012), and Li et al. (2015). Another popular way of extending decision tree learning methods is through fuzzification of continuous attributes towards the generation of fuzzy decision trees, which is essentially based on fuzzy set theory (Zadeh 1965). Some more recent studies have been reported in Lertworaprachaya et al. (2014), Altay and Cinar (2016), and Lertworaprachaya et al. (2010).

The nature of decision tree learning is to generate a set of non-overlapping rules, which constrains that these rules must have at least one common attribute to make the rules fit in a tree structure. Due to the above constraint, decision tree learning often results in complex trees being generated and difficulty for people to understand the information extracted from the trees (Furnkranz 1999). To simplify decision trees, Quinlan investigated the use of pruning methods (Quinlan 1987), such as reduced error pruning (Elomaa and Kriinen 2001). However, even if decision trees are simplified using pruning methods, it is still difficult to avoid the case that decision trees are too cumbersome, complex and inscrutable to provide insight into a domain for people to use as knowledge (Quinlan 1993; Furnkranz 1999). In addition, complex decision trees are more likely to overfit training data than simple trees (Furnkranz 1999). On the other hand, Cendrowska (1987) pointed out that the nature of decision tree learning may result in the replicated subtree problem due to the constraint that rules must have at least one common attribute to represent these rules in a tree structure, i.e., rules that have no common attribute cannot fit in a tree structure. It is also mentioned in Cendrowska (1987) that the replicated subtree problem may result in the need to examine the whole tree for extracting rules about a single classification in the worst case, which makes it difficult to manipulate for expert systems.

Algorithm 1: Prism Algorithm (Liu *et al.*, 2016b)

```
Input : a training set $T$, a subset $T' \subseteq T$, an
        attribute set $AS$, an instance $t \in T$,
        dimensionality $d$, an attribute $A_x$, an
        attribute value $v_{xm}$, class $C_i$, number of
        classes $n$, max-probability $p_{max}$
Output: a rule set $RS$, a result set of instances $T$"
        covered by a rule $R \in RS$
Initialize: $T' = T$, $T" = T'$, $i = 0$, $p_{max} = 0$;
for $i < n$ do
    while $\exists t : t \in T' \wedge t \in C_i$ do
        while $\exists t : t \in T' \wedge t \notin C_i$ do
            $x = 0$; $j = 0$; $p_{max} = 0$; while $x < d$ do
                for each value $v_{xm}$ of $A_x$ do
                    Calculate $P(C_i|A_x = v_{xm})$;
                    if $P(C_i|A_x = v_{xm}) > p_{max}$ then
                        $p_{max} = P(C_i|A_x = v_{xm})$;
                        $j = x$; $k = m$;
                    end
                end
                $x++$;
            end
            assign $A_j = v_{jk}$ to $R$ as a rule term; $AS$=
            $AS$ - $\{A_j\}$; $d$= $d - 1$;
            $\forall t : T" = T" - \{t\}$, if $t \in T'$ and $t$ does
            not comprise $A_j = v_{jk}$;
        end
        $RS$= $RS \cup \{R\}$; $T'$= $T'$ - $T"$;
    end
    $T' = T$; $i++$;
end
```

To overcome the replicated subtree problem, the Prism algorithm (Cendrowska 1987) has been developed, which has led to the motivation for developing rule learning methods through the separate and conquer strategy. A comprehensive review of rule-learning methods can be found in Furnkranz (1999). As illustrated in Algorithm 1, the nature of the Prism algorithm is to select a target class, and then to learn a rule by selecting attribute–value pairs iteratively for specializing this rule, until all the instances covered by this rule belong to the target class. However, it is generally unknown whether the selected target class can lead to the generation of a high-quality rule. In fact, the separate and conquer strategy involves the learning of rules on a sequential basis, i.e., the learning of a rule must not start until the learning of the last rule is completed. In this context, the outcome of learning one rule impacts greatly on the outcome of learning the next rule. In the case of the Prism algorithm, the selected target class may not be suitable for learning a high quality rule, and even leads to the generation of an inconsistent rule, which means that the instances covered by this rule belong to different classes (Liu and Gegov 2016; Liu et al. 2016b). In addition, the learning of the subsequent rules would be impacted greatly, due to the unexpected outcome that the last rule learned is inconsistent.

The Prism algorithm introduced in Cendrowska (1987) is designed to simply keep selecting the same class as the target class towards learning a set of rules, all of which are assigned this class as the rule consequent, and then repeat the same procedure by having another class as the target class for learning a different set of rules. The above design is based on the assumption that all the classes are important, i.e., the classification task is to detect features of each class in the setting of pattern recognition, such as human activities recognition, where each class represents an activity and it is needed to identify features that can recognize any of these classes. In real applications, it is not always necessary to select each of the classes as the target class in turn for learning rules, since it is very possible that only one or some but not all of the classes are of interest. Some examples of such applications are provided in Sect. 3 for justification.

3 Granular computing-based rule learning

In this section, we propose a variant of the Prism algorithm referred to as PrismSTC, where 'STC' stands for 'single target class'. In particular, we describe the procedure of the PrismSTC algorithm and show how it is different from the original version of Prism. We also justify how the PrismSTC algorithm shows granular computing perspectives and argue the significance of this algorithm from machine-learning perspectives.

3.1 Procedure of PrismSTC

The procedure of the PrismSTC algorithm is illustrated in Algorithm 2, which is essentially to select an attribute–value pair that has the maximum posterior probability given a target class, as illustrated in Eq. 1.

$$P(\text{class} = \text{TC}|A_i = v_j) = \frac{F(A_i = v_j, \text{class} = \text{TC})}{F(A_i = v_j)} \tag{1}$$

where A_i represents an attribute and i is the index of this attribute; v_j represents a value of the attribute A_i and j is the index of the attribute value v_j. In addition, $F(A_i = v_j, \text{class} = \text{TC})$ denotes the joint frequency that not only an instance meets the condition $A_i = v_j$ but also belongs to the target class TC, whereas $F(A_i = v_j)$ denotes the independent frequency that an instance meets the condition $A_i = v_j$.

Algorithm 2: PrismSTC Algorithm

```
Input : a training set $T$, a subset $T' \subseteq T$, an
        attribute set $AS$, an instance $t \in T$,
        dimensionality $d$, an attribute $A_x$, an
        attribute value $v_{xm}$, the target class $TC$,
        max-probability $p_{max}$
Output: a rule set $RS$, a result set of instances $T$"
        covered by a rule $R \in RS$
Initialize: $T' = T$, $T" = T'$, $p_{max} = 0$;
while $\exists t : t \in T' \wedge t \in TC$ do
    while $\exists t : t \in T' \wedge t \notin TC$ do
        $x = 0$; $j = 0$; $p_{max} = 0$; while $x < d$ do
            for each value $v_{xm}$ of $A_x$ do
                Calculate $P(TC|A_x = v_{xm})$;
                if $P(TC|A_x = v_{xm}) > p_{max}$ then
                    $p_{max} = P(TC|A_x = v_{xm})$; $j = x$;
                    $k = m$;
                end
            end
            $x++$;
        end
        assign $A_j = v_{jk}$ to $R$ as a rule term;
        $AS$= $AS$ - $\{A_j\}$; $d$= $d - 1$;
        $\forall t : T" = T" - \{t\}$, if $t \in T'$ and $t$ does not
        comprise $A_j = v_{jk}$;
    end
    $RS$= $RS \cup \{R\}$; $T'$= $T'$ - $T"$;
end
```

We also provide an illustrative example using the contact lenses dataset (Cendrowska 1987) detailed in Table 1. For the age attribute, the values 1, 2 and 3 represent 'young', 'pre-presbyopic' and 'presbyopic', respectively. For the spectacle-prescrip attribute, the values 1 and 2 represent 'myope' and 'hypermetrope', respectively. For the astigmatism attribute, the values 1 and 2 represent 'no' and 'yes', respectively. For the tear-prod-rate attribute, the two values 1 and 2 represent 'reduced' and 'normal', respectively. In addition, the three classes ('hard lenses', 'soft lenses' and 'no lenses') relating to contact lenses are expressed as 1, 2 and 3, respectively. In this illustration, we choose the 'no lenses' (3) class as the target class.

According to Table 1, we can get a frequency table for each attribute, i.e., we have four frequency tables for the four attributes: 'age' (see Table 2), 'spectacle-prescrip' (see Table 3), 'astigmatism' (see Table 4) and 'tear-prod-rate' (see Table 5).

Based on the frequency tables, the conditional probabilities for each attribute–value pair of each attribute can be calculated. We display these here for ease of explanation—in the normal course of the algorithm the probabilities would be calculated when needed, not in advance.

According to Table 2, we can derive the conditional probability for each of the three values of attribute 'age', towards the target class 'no lenses'(3).

Table 1 Contact lenses data

Age	Spectacle-prescrip	Stigmatism	Tear-prod-rate	Class
1	1	1	1	3
1	1	1	2	2
1	1	2	1	3
1	1	2	2	1
1	2	1	1	3
1	2	1	2	2
1	2	2	1	3
1	2	2	2	1
2	1	1	1	3
2	1	1	2	2
2	1	2	1	3
2	1	2	2	1
2	2	1	1	3
2	2	1	2	2
2	2	2	1	3
2	2	2	2	3
3	1	1	1	3
3	1	1	2	3
3	1	2	1	3
3	1	2	2	1
3	2	1	1	3
3	2	1	2	2
3	2	2	1	3
3	2	2	2	3

Table 2 Frequency table for age

Class label	Age = 1	Age = 2	Age = 3
1	2	1	1
2	2	2	1
3	4	5	6
Total	8	8	8

Table 3 Frequency table for spectacle-prescrip

Class label	Spectacle-prescrip = 1	Spectacle-prescrip = 2
1	3	1
2	2	3
3	7	8
Total	12	12

$$P(\text{class} = 3|\text{age} = 1) = \frac{4}{8}$$
$$P(\text{class} = 3|\text{age} = 2) = \frac{5}{8}$$
$$P(\text{class} = 3|\text{age} = 3) = \frac{6}{8}$$

Table 4 Frequency table for astigmatism

Class label	Astigmatism = 1	Astigma-tism = 2
1	0	4
2	5	0
3	7	8
Total	12	12

Table 5 Frequency table for tear-prod-rate

Class label	Tear-prod-rate = 1	Tear-prod-rate = 2
1	0	4
2	0	5
3	12	3
Total	12	12

According to Table 3, we can derive the conditional probability for each of the two values of attribute 'spectacle-prescrip', towards the target class.

$$P(\text{class} = 3|\text{spectacle-prescrip} = 1) = \frac{7}{12}$$
$$P(\text{class} = 3|\text{spectacle-prescrip} = 2) = \frac{8}{12}$$

According to Table 4, we can derive the conditional probability for each of the two values of attribute 'astigmatism', towards the target class.

$$P(\text{class} = 3|\text{astigmatism} = 1) = \frac{7}{12}$$
$$P(\text{class} = 3|\text{astigmatism} = 2) = \frac{8}{12}$$

According to Table 5, we can derive the conditional probability for each of the two values of attribute 'tear-prod-rate', towards the target class.

$$P(\text{class} = 3|\text{tear-prod-rate} = 1) = \frac{12}{12}$$
$$P(\text{class} = 3|\text{tear-prod-rate} = 2) = \frac{3}{12}$$

Since the attribute–value pair (tear-prod-rate = 1) has the maximum conditional probability, i.e., $P(\text{class} = 3|\text{tear-prod-rate} = 1) = 1$, it is selected and appended into the first rule being learned as a rule term. In addition, the conditional probability has been equal to 1, so it indicates that there is no uncertainty any more towards classifying instances covered by the current rule and the learning of the first rule is thus complete. The first rule is finally expressed as: if tear-prod-rate= 1 then class = 3. Following the completion of learning the first rule, all the 12 instances with the attribute–value pair tear-prod-rate = 1 are deleted from the training set, and the learning of the second rule is started on the reduced training set.

The main difference between the original Prism algorithm and PrismSTC is that the original one needs to have each class selected in turn as the target class towards learning a set of rules, as illustrated in Algorithm 1. For example, the contact lenses dataset involves three classes, so it is needed to repeat three times the above learning process (illustrated for PrismSTC) for learning three sets of rules respectively for the three classes.

3.2 Justification

The design of the Prism algorithm is essentially in the setting of granular computing, which is a paradigm of information processing (Yao 2005b). From philosophical perspectives, granular computing is a way of structured thinking (Yao 2005b). From practical perspectives, it is adopted as a way of structured problem solving (Yao 2005b).

In general, granular computing is achieved through two operations, namely granulation and organization (Yao 2005a). The former operation is aimed at decomposition of a whole into different parts, whereas the latter operation is aimed at integrating several parts into a whole. In computer science, the concepts of granulation and organization have been popularly used to achieve the top-down and bottom-up approaches, respectively (Liu and Cocea 2017; Liu et al. 2017).

In granulation and organization, the main aim is to deal with granules and granularity (Pedrycz 2011; Pedrycz and Chen 2011, 2015a, b, 2016), which are two main concepts of granular computing. A granule generally represents a large particle, which consists of smaller particles that can form a larger unit. In the setting of the Prism rule-learning algorithm, each class is selected as the target class in turn towards learning a set of rules for this class. In this context, for each class, a set of rules are learned, and the sets of rules for all these classes make up a rule-based classifier learned from a training set. From this point of view, a rule based classifier can be treated as a granule in the top level of granularity, and each set of rules learned for a specific class would be treated as a granule (sub-classifier) in the second level of granularity. In addition, each class can also be viewed as a granule in the second level of granularity, corresponding to a rule set (sub-classifier).

In real applications, there are many examples of binary classification that only involve one target class. For example, cyberbullying detection only involves two classes ('yes' and 'no'), and the aim is just to detect if a text message contains abusive behaviours (Zhao et al. 2016; Reynolds et al. 2011), i.e., people are only interested to identify any features of abusive languages from text messages. From this point of view, it is not necessary to learn a set of rules for both classes. Instead, only a set of rules for the 'yes' class needs to be learned from training data, and the rule-based classifier would classify an instance (text message) to the 'no' class by default, unless any rules fire implying that some features of abusive languages have been detected from this instance.

From machine-learning perspectives, the design of the PrismSTC algorithm would lead to the production of simpler rule-based classifiers by means of a smaller number of simpler rules, which can not only result in reduction of computational complexity in classifying unseen instances but also avoid overfitting of training data, in comparison with the Prism algorithm. Moreover, in some application areas such as cyberbullying detection, it is the natural case that training data is highly imbalanced, where the target class is the minority class with a very low weight (less than 10% of the whole training data) (Reynolds et al. 2011). Since PrismSTC is aimed at learning a set of rules from the target (minority) class only, the rule-based classifier would be much less sensitive to the class imbalance problem, i.e., the classifier is not likely to be biased on the majority class.

4 Experiments, results and discussion

In this section, we report an experimental study conducted using 12 datasets from the UCI repository (Lichman 2013). The characteristics of the datasets are shown in Table 6. In particular, all the datasets are for binary classification tasks, and the aim of the experimental study is to investigate the impact on classification accuracy and model complexity, when learning a set of rules only for one of the two classes, in comparison with learning two sets of rules respectively for the two classes. In other words, we aim to show that the model complexity is reduced significantly, but the classification accuracy is not lost, when only one class is selected as the target class for rule learning.

In this experimental study, we compare PrismSTC with Prism as well as C4.5. In terms of classification accuracy, each dataset is randomly partitioned into a training set and a test set in the ratio of 70:30. The experiment on each dataset is repeated 100 times in terms of the data partitioning, and the average accuracy is taken for comparative validation. The classification accuracy is measured according to Eq. 2, and Eq. 3 shows that the error rate is the complement to 1 of Accuracy. Both measures involve TP, TN, FP and FN, which stand for true-positive rate, true-negative rate, false-positive rate and false-negative rate, respectively, and positive and negative represent two classes in a classification task. In terms of model complexity, each whole dataset is used to train a rule-based classifier (model) towards counting the rules and rule terms, for comparison of models trained using different algorithms.

$$\text{Accuracy} = \frac{\text{TP+TN}}{\text{TP+TN+FP+FN}} \tag{2}$$

$$\text{Error rate} = \frac{\text{FP+FN}}{\text{TP+TN+FP+FN}} \tag{3}$$

The results on classification accuracy are shown in Table 7. In particular, PrismSTC1 and PrismSTC2 represent that class 1 and class 2 are selected as the target classes, respectively. On some datasets, class 1 is the minority class and class 2 is the majority class, but some other datasets show the opposite situation. The details about the frequency distribution between the two classes are available in Table 6.

The results shown in Table 7 indicate that the performance of PrismSTC is highly comparable or even better than the one of Prism in 10 out of 12 cases, no matter which one of the two classes is selected as the target class. On the 'ionosphere' and 'labor' datasets, the performance is somewhat lower when the majority class is selected as the target class. However, when the minority class is used as the target

Table 6 Characteristics of datasets

Dataset	Attribute types	#Attributes	#Instances (distribution)	#Classes
Breast-cancer	Discrete	9	286 (201:85)	2
Breast-w	Continuous	10	699 (458:241)	2
Credit-a	Discrete, continuous	15	690 (307:383)	2
Credit-g	Discrete, continuous	20	1000 (700:300)	2
Cylinder-bands	Discrete, continuous	40	540 (228:312)	2
Diabetes	Discrete, continuous	20	768 (500:268)	2
Hepatitis	Discrete, continuous	20	155 (32:123)	2
Ionosphere	Continuous	34	351 (126:225)	2
kr-vs-kp	Discrete	36	3196 (1669:1527)	2
Labor	Discrete, continuous	17	57 (20:37)	2
Mushroom	Discrete	22	8124 (4208:3916)	2
Vote	Discrete	16	435 (267:168)	2

class, the performance is very similar to the one of Prism. The main reason behind the phenomenon is likely that the attributes involved in the two datasets are more relevant for the minority class than for the majority class. In comparison with C4.5, the performance of PrismSTC is also sufficiently good. In particular, there are 5 out of 12 cases that PrismSTC1 performs marginally worse than C4.5, i.e., PrismSTC1 shows better or the same performance as C4.5 in all the other cases. Similarly, there are only 3 out of 12 cases that PrismSTC2 performs marginally worse than C4.5, except for the case of the 'ionosphere' dataset.

The results also show that the classification performance is usually better when the minority class is selected instead of the majority class for learning a set of rules. For example, on the 'breast-cancer', 'breast-w', 'credit-a', 'ionosphere' and 'labor' data sets, class 1 is the minority class, and the performance is better when it is selected instead of class 2 as the target class. Also, on the 'kr-vs-kp' dataset, class 2 is the minority class, and the performance is higher when it is selected instead of class 1 as the target class. On the other hand, on the 'credit-g', 'cylinder-bands', 'diabetes' and 'hepatitis' datasets, the performance is just marginally worse when the minority class is selected for rule learning. The above phenomenon is highly expected in practice, since the target class is likely to be the minority class in a dataset in various application areas, such as cyberbullying detection, as mentioned in Sect. 3.2.

The results on model complexity are shown in Table 8. In particular, the results show that the complexity of the rule-based classifier is equal to the sum of the ones of PrismSTC1 and PrismSTC2 in all the 12 cases, which is consistent with our justification given in Sect. 3.2 from granular computing perspective. In other words, the classifiers trained respectively using PrismSTC1 and PrismSTC2 are essentially the sub-classifiers of the classifier trained using Prism. In comparison with C4.5, PrismSTC produces a smaller number of simpler rules in 11 out of the 12 cases, no matter which one of the two classes is selected as the target class.

The results shown in Table 8 also indicate that the rule-based classifier learned from the instances of the minority class is usually simpler than the classifier learned from the instances of the majority class, with an exception on the vote dataset. Although it is normally expected to occur that learning from instances of the minority class leads to production of a simpler model, the nature of the Prism algorithm (as well as PrismSTC) could lead to the opposite case, when the attributes involved in a data set are highly relevant for the majority class but are less relevant or irrelevant for the minority class. In other words, as illustrated in Sect. 3.1, the nature of Prism-based rule learning is to select iteratively an attribute–value pair that is highly relevant for the target class towards specializing a rule. If some highly relevant attribute–value pairs are found, the rule learning could be completed with a smaller number of iterations, leading to the

Table 7 Classification accuracy

Data set	C4.5 (%)	Prism (%)	PrismSTC1 (%)	PrismSTC2 (%)
Breast-cancer	67	67	64	67
Breast-w	94	93	94	96
Credit-a	83	80	81	79
Credit-g	68	74	71	70
Diabetes	72	69	70	72
Hepatitis	76	76	76	80
Ionosphere	89	90	92	81
kr-vs-kp	99	98	97	99
Labor	80	88	87	80
Mushroom	100	98	100	100
Vote	95	93	94	94

Table 8 Number of rules and terms

Data set	C4.5		Prism		PrismSTC1		PrismSTC2	
	#Rules	#Terms	#Rules	#Terms	#Rules	#Terms	#Rules	#Terms
breast-cancer	152	645	110	329	72	223	38	106
Breast-w	23	124	36	87	18	52	18	35
Credit-a	101	546	148	392	74	185	74	207
Credit-g	359	2262	310	882	179	499	131	383
Cylinder-bands	430	432	310	312	153	153	157	159
Diabetes	22	120	223	680	128	375	95	305
Hepatitis	16	81	30	51	11	22	19	29
Ionosphere	18	121	39	92	14	19	25	73
kr-vs-kp	43	379	101	387	53	229	48	158
Labor	13	44	12	15	4	5	8	10
Mushroom	25	67	27	34	15	20	12	14
Vote	19	97	27	88	13	38	14	50

production of a smaller number of simpler rules, where each rule covers a larger number of training instances.

5 Conclusions

In this paper, we identified some limitations of the Prism algorithm and proposed its variant referred to as PrismSTC. We conducted an experimental study for comparing PrismSTC with Prism as well as C4.5 in terms of their performance. The experimental results show that PrismSTC leads to a reduction of model complexity without loss of classification accuracy in comparison with the other two algorithms.

In addition, another contribution of this paper is an in-depth analysis of Prism-based rule learning from the perspectives of both granular computing and machine learning. The reduction of model complexity without loss of classification accuracy through using the proposed PrismSTC algorithm is also beneficial and helpful towards speeding up the process of classifying unseen instances in practical applications.

In future, we will investigate the use of PrismSTC in some real applications, such as cyberbullying detection, where the data collected typically involve only one target class and the minority class is usually the target class. In addition, we will investigate the adoption of Prism-based rule learning for multi-class classification tasks. In particular, we will identify more effective ways for the selection of the target class for learning each single rule of as high quality as possible, in the setting of granular computing (Liu et al. 2017; Liu and Cocea 2018a; Liu et al. 2016d; Ahmad and Pedrycz 2017). Furthermore, it is worth to adopt fuzzy set theory for fuzzification of continuous attributes (Chen 1996; Chen et al. 2014; Mendel et al. 2006; Lee and Chen 2008; Chen and Lee 2010) for improving the quality of rules, and employ optimization techniques for searching an optimal set of rules in terms of rule quality (Chen and Chung 2006; Tsai et al. 2008, 2012).

Acknowledgements The authors acknowledge support for the research reported in this paper through the research development fund at the University of Portsmouth.

References

Ahmad SSS, Pedrycz W (2017) The development of granular rule-based systems: a study in structural model compression. Granul Comput 2(1):1–12
Altay A, Cinar D (2016) Fuzzy decision trees. In: Kahraman C, Kabak O (eds.) Fuzzy statistical decision-making: theory and applications, vol 343. Springer, Switzerland, pp 221–261
Breiman L (1996) Bagging predictors. Mach Learn 24(2):123–140
Breiman L (2001) Random forests. Mach Learn 45(1):5–32
Cendrowska J (1987) Prism: an algorithm for inducing modular rules. Int J Man Mach Stud 27:349–370
Chen S-M (1996) A fuzzy reasoning approach for rule-based systems based on fuzzy logics. IEEE Trans Syst Man Cybern Part B Cybern 26(5):769–778
Chen S-M, Chung N-Y (2006) Forecasting enrollments using high-order fuzzy time series and genetic algorithms. Int J Inf Manag Sci 17(3):1–17
Chen S-M, Lee L-W (2010) Fuzzy multiple criteria hierarchical group decision-making based on interval type-2 fuzzy sets. IEEE Trans Syst Man Cybern Part A Syst Hum 40(5):1120–1128
Chen S-M, Lin T-E, Lee L-W (2014) Group decision making using incomplete fuzzy preference relations based on the additive consistency and the order consistency. Inf Sci 259:1–15
Crockett K, Latham A, Whitton N (2017) On predicting learning styles in conversational intelligent tutoring systems using fuzzy decision trees. Int J Hum Comput Stud 97:98–115
Elomaa T, Kriinen M (2001) An analysis of reduced error pruning. J Artif Intell Res 15(1):163–187
Esposito F, Malerba D, Semeraro G (1995) Simplifying decision trees by pruning and grafting: new results. In: 8th European conference on machine learning, vol 912. Crete, Greece, pp 287–290
Freund Y, Schapire RE (1996) Experiments with a new boosting algorithm. In: 13th international conference on machine learning. Bari, Italy, pp 148–156
Furnkranz J (1999) Separate-and-conquer rule learning. Artif Intell Rev 13:3–54
Khan K, khan RU, Alkhalifah A, Ahmad N (2015) Urdu text classification using decision trees. In: International conference on high-capacity optical networks and enabling/emerging technologies. Islamabad, Pakistan, pp 56–59
Kuhn M, Johnson K (2013) Applied predictive modeling. Springer, New York
Lee L-W, Chen S-M (2008) Fuzzy multiple attributes group decision-making based on the extension of topsis method and interval type-2 fuzzy sets. In: Proceedings of the 2008 international conference on machine learning and cybernetics, vol 6, Kunming, China. pp 3260–3265
Lertworaprachaya Y, Yang Y, John R (2010) Interval-valued fuzzy decision trees. In: IEEE international conference on fuzzy systems, Barcelona, Spain. pp 1–7
Lertworaprachaya Y, Yang Y, John R (2014) Interval-valued fuzzy decision trees with optimal neighbourhood perimeter. Appl Soft Comput 24:851–866
Li X, Zhao H, Zhu W (2015) A cost sensitive decision tree algorithm with two adaptive mechanisms. Knowl Based Syst 88:24–33
Lichman M (2013) UCI machine learning repository. http://archive.ics.uci.edu/ml. Accessed 11 Feb 2018
Liu H, Cocea M (2017) Fuzzy information granulation towards interpretable sentiment analysis. Granul Comput 2(4):289–302
Liu H, Cocea M (2018a) Granular computing based machine learning: a big data processing approach. Springer, Berlin
Liu H, Cocea M (2018b) Induction of classification rules by gini-index based rule generation. Inf Sci 436–437:227–246

Liu H, Gegov A (2016) Induction of modular classification rules by information entropy based rule generation. Springer, Switzerland, pp 217–230

Liu H, Gegov A, Cocea M (2015) Network based rule representation for knowledge discovery and predictive modeling. In: IEEE international conference on fuzzy systems. Istanbul, Turkey, pp 1–8

Liu H, Gegov A, Cocea M (2016a) Ensemble learning approaches. In: Liu H, Gegov A, Cocea M *(eds) Rule based systems for big data: a machine learning approach, vol 13. Springer, Switzerland, pp 63–73

Liu H, Gegov A, Cocea M (2016b) Generation of classification rules. In: Liu H, Gegov A, Cocea M (eds) Rule based systems for big data: a machine learning approach, vol 13. Springer, Switzerland, pp 29–42

Liu H, Cocea M, Gegov A (2016c) Interpretability of computational models for sentiment analysis. In: Pedrycz W, Chen S-M (eds) Sentiment analysis and ontology engineering: an environment of computational intelligence, vol 639. Springer, Switzerland, pp 199–220

Liu H, Gegov A, Cocea M (2016d) Rule based systems: a granular computing perspective. Granul Comput 1(4):259–274

Liu H, Cocea M, Mohasseb A, Bader M (2017) Transformation of discriminative single-task classification into generative multi-task classification in machine learning context. In: International conference on advanced computational intelligence. Doha, Qatar, pp 66–73

Mendel JM, John RI, Liu F (2006) Interval type-2 fuzzy logic systems made simple. IEEE Trans Fuzzy Syst 14(6):808–821

Min F, Zhu W (2012) A competition strategy to cost-sensitive decision trees. In: Li T, Nguyen HS, Wang G, Grzymala-Busse J, Janicki R, Hassanien AE, Yu H (eds) Rough sets and knowledge technology: 7th international conference, RSKT 2012, Chengdu, China, August 17–20, 2012. Proceedings, vol 7414. pp 359–368

Ogutu JO, Piepho H-P, Schulz-Streeck T (2011) A comparison of random forests, boosting and support vector machines for genomic selection. BMC Proc 5(3):S11

Pedrycz W (2011) Information granules and their use in schemes of knowledge management. Sci Iran 18(3):602–610

Pedrycz W, Chen S-M (2011) Granular computing and intelligent systems: design with information granules of higher order and higher type. Springer, Heidelberg

Pedrycz W, Chen S-M (2015a) Granular computing and decision-making: interactive and iterative approaches. Springer, Heidelberg

Pedrycz W, Chen S-M (2015b) Information granularity, big data, and computational intelligence. Springer, Heidelberg

Pedrycz W, Chen S-M (2016) Sentiment analysis and ontology engineering: an environment of computational intelligence. Springer, Heidelberg

Quinlan JR (1983) Learning efficient classification procedures and their application to chess end games. In: Michalski RS, Carbonell JG, Mitchell TM (eds) Machine learning: an artificial intelligence approach. Springer, Heidelberg, pp 463–482

Quinlan JR (1987) Simplifying decision trees. Int J Man Mach Stud 27:221–234

Quinlan JR (1996) Improved use of continuous attributes in c4.5. J Artif Intell Res 4:77–90

Quinlan JR (1986) Induction of decision trees. Mach Learn 1(1):81–106

Quinlan JR (1993) C4.5: programs for machine learning. Morgan Kaufmann Publishers, San Francisco

Rahmatian M, Chen YC, Palizban A, Moshref A, Dunford WG (2017) Transient stability assessment via decision trees and multivariate adaptive regression splines. Electr Power Syst Res 142:320–328

Reynolds K, Kontostathis A, Edwards L (2011) Using machine learning to detect cyberbullying. In: Proceedings of the 10th international conference on machine learning and applications, pp 241–244

Tayefi M, Esmaeili H, Karimian MS, Zadeh AA, Ebrahimi M, Safarian M, Nematy M, Parizadeh SMR, Ferns GA, Ghayour-Mobarhan M (2017) The application of a decision tree to establish the parameters associated with hypertension. Comput Methods Progr Biomed 139:83–91

Tsai P-W, Pan J-S, Chen S-M, Liao B-Y, Hao S-P (2008) Parallel cat swarm optimization. In: Proceedings of the 2008 international conference on machine learning and cybernetics, vol 6, Kunming, China, pp 3328–3333

Tsai P-W, Pan J-S, Chen S-M, Liao B-Y (2012) Enhanced parallel cat swarm optimization based on the Taguchi method. Expert Syst Appl 39(7):6309–6319

Yao J (2005a) Information granulation and granular relationships. In: IEEE international conference on granular computing. Beijing, China, pp 326–329

Yao Y (2005b) Perspectives of granular computing. In: Proceedings of 2005 IEEE international conference on granular computing, Beijing, China, pp 85–90

Zadeh L (1965) Fuzzy sets. Inf Control 8(3):338–353

Zhao H, Li X (2017) A cost sensitive decision tree algorithm based on weighted class distribution with batch deleting attribute mechanism. Inf Sci 378:303–316

Zhao R, Zhou A, Mao K (2016) Automatic detection of cyberbullying on social networks based on bullying features. In: Proceedings of the 17th international conference on distributed computing and networking, vol, 43. pp 1–43:6