Improving modular classification rule induction with G-Prism using dynamic rule term boundaries

Improving Modular Classification Rule Induction with G-Prism Using Dynamic Rule Term Boundaries

Improving Modular Classification Rule Induction with G-Prism using Dynamic Rule Term Boundaries

Manal Almutairi[1], Frederic Stahl[1], and Max Bramer[2]

[1] Department of Computer Science, University of Reading
Whiteknights. PO Box 225, Reading, RG6 6AY, UK,
Manal.Almutairi@pgr.reading.ac.uk, F.T.Stahl@reading.ac.uk
[2] School of Computing, University of Portsmouth, Buckingham Building, Lion Terrace, PO1 3HE, Max.Bramer@port.ac.uk

Abstract. Modular classification rule induction for predictive analytics is an alternative and expressive approach to rule induction as opposed to decision tree based classifiers. Prism classifiers achieve a similar classification accuracy compared with decision trees, but tend to overfit less, especially if there is noise in the data. This paper describes the development of a new member of the Prism family, the G-Prism classifier, which improves the classification performance of the classifier. G-Prism is different compared with the remaining members of the Prism family as it follows a different rule term induction strategy. G-Prism's rule term induction strategy is based on Gauss Probability Density Distribution (GPDD) of target classes rather than simple binary splits (local discretisation). Two versions of G-Prism have been developed, one uses fixed boundaries to build rule terms from GPDD and the other uses dynamic rule term boundaries. Both versions have been compared empirically against Prism on 11 datasets using various evaluation metrics. The results show that in most cases both versions of G-Prism, especially G-Prism with dynamic boundaries, achieve a better classification performance compared with Prism.

Keywords: Modular Classification Rule Induction, Dynamic Rule Term Boundaries, Gaussian Probability Density Distribution

1 Introduction

The general consensus in the data mining community is that there is no single best technique that can work successfully on every dataset. However, decision tree induction is one of the most popular and most widely used algorithms. It produces classification rules in the form of a tree structure and uses a 'divide-and-conquer' strategy to construct the tree from training data. A considerable amount of literature has been published discussing and referring to this approach and a popular and widely used algorithm is C4.5 [1]. However, decision tree based algorithms suffer from several drawbacks such as redundant rule terms, overfitting, and replicated subtrees [2]. This paper will revisit some of these problems in Section 2. Decision rules can be extracted from a decision tree [3] by transforming each leaf in the tree into a rule [4]. Despite its simplicity, this process ends up with a set of rules that may inherit all the shortcomings of decision trees and thus might become more difficult to understand [5, 4]. The author of [3] argues that the major cause of overfitting problem is the tree representation itself and suggests that the solution is to look at another representation which extracts rules directly from data. Examples of classifiers that are based on the induction of classification rules directly from training dataset are, among others, RIPPER [6] CN2 [7] and Prism [2]. Cendrowska's original Prism algorithm started a range of different Prism variations and improvements over the years, also known as the Prism family of algorithms. Some of the members of the Prism family are PrismTCS which improves original Prism's computational efficiency [8] and PMCRI [9], a parallel version of Prism. Originally Prism was only applicable on categorical data, however, all aforementioned Prism variations are also applicable on numerical attributes as will be explained in Section 2.

In [10] we provided a proof of concept (evaluated only on 2 datasets) for a potentially efficient method to induce such rule terms based on Gauss Probability Density Distribution (GPDD) of attribute values. The method was termed G-Prism. There are two contributions in this paper: (1) a more dynamic rule term boundary allowing larger rule terms to be built (in terms of data coverage) and (2) a thorough empirical evaluation of both, the original G-Prism, the new version of G-Prism with dynamic rule term boundaries and original Prism.

This paper is organised as follows: Section 2 introduces Prism and Section 3 describes and positions the development of a new version of Prism based on Gauss Probability Density Distribution. Section 4 provides an empirical evaluation of G-Prism in comparison with Prism. Section 5 describes our ongoing and future work and concluding remarks are provided in Section 6.

2 Related Work: The Prism Family of Algorithms for Inducing Modular Classification Rules

A major critique of rule representation in the form of trees is the replicated subtree problem. First discussed in [2] and later termed replicated subtree problem in [5]. For example, consider a training dataset with 4 attributes *a, b, c* and *d.* Each attribute can take 2 possible values T (true) and F (false). There are also two possible class values *stop* and *go.* The rules below encode a pattern that predicts class *stop* and all remaining instances would lead to class *go.*

```
IF a AND b → Stop
IF c AND d → Stop
```

Labelling instances *go* and *stop* using a tree will require replicated subtrees to be induced as illustrated in Figure 1. An alternative to decision trees are classifiers that induce modular IF-THEN classification rules directly from a training dataset. Each rule can be separately handled or even removed without needing to reconstruct the whole classifier or affect its accuracy. Cendrowska's Prism algorithm [2] can induce such modular IF-THEN rules that do not necessarily contain any redundancies.

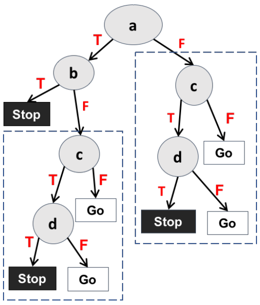


Fig. 1: Replicated subtree problem

Algorithm 1 depicts Prism based on Cendrowska's original publication [2]. It also incorporates a method of handling continuous attributes using a local discretisation technique called cut-points calculations [3], as the original version of Prism does not consider numerical attributes for inducing rule terms. ChiMerge [11] and Chi Square [12] are alternative discretisation methods that could convert the values of continuous attributes into a small number of intervals as a pre-processing step. Prism generates rules by appending rule terms (using a logical *AND*) that maximise the conditional probability with which the rule covers a target class. A rule is complete if it only covers instances of the target class or if it cannot be specialised further. Once the rule has been generated all instances covered by the rule are removed from the training data and the next rule is constructed from the remaining instances. This is repeated until no instances that match the target class remain. Then the same process is repeated for the next target class for the entire original training dataset. This is also known as 'separate-and-conquer' approach. Basically Prism classifiers generate rule terms from numerical attributes α through binary splitting [3], potentially resulting in rule term combinations such as $(10 \leq \alpha)$ or $(20 > \alpha)$ to describe an interval of attribute values. Binary splitting is also very inefficient due to a potentially large number of probability calculations which can be quantified as $N \cdot m \cdot 2$, where N is the number of training instances and m the number of numerical attributes. A better way of representing such a rule term is $(10 \leq \alpha < 20)$ instead of two separate rule terms. This would greatly enhance readability of the individual rules and potentially reduce overfitting. The in this paper presented G-Prism approach, is able to induce such more readable rule terms.

Another interesting property of the the Prism family of algorithms is that it by default does not force a classification. If a data instance is unknown to the classifier, i.e. it is not covered by a Prism rule, it will simply remain unclassified. We refer to this property as abstaining. Abstaining may be desirable in application where an incorrectly classified data instance could be potentially very costly, such as in financial applications, or risky, such as in medical applications.

Algorithm 1: Learning classification rules from labelled data instances using Prism.

```
1  for $i = 1 \rightarrow C$ do
2      D ← Dataset;
3      while D does not contain only instances of class $\omega_i$ do
4          forall attributes $\alpha_j \in D$ do
5              if attribute $\alpha_j$ is categorical then
6                  Calculate the conditional probability, $\mathbb{P}(\omega_i|\alpha_j = x)$ for all
                     possible attribute-value $(\alpha_j = x)$ from attribute $\alpha$;
7              else if attribute $\alpha_j$ is numerical then
8                  sort D according to $x$ values;
9                  foreach $x$ value of $\alpha_j$ do
10                     calculate $\mathbb{P}(\omega_i|\alpha_j \leq x)$ and $\mathbb{P}(\omega_i|\alpha_j > x)$;
                   end
12
               end
           end
15         Select the $(\alpha_j = x)$, $(\alpha_j > x)$, or $(\alpha_j \leq x)$ with the maximum
             conditional probability as a rule term;
16         D ← S, create a subset S from D containing all the instances covered by
             selected rule term at line 15;
       end
18     The induced rule $R$ is a conjunction of all selected $(\alpha_j = x)$, $(\alpha_j > x)$, or
         $(\alpha_j \leq x)$ at line 15;
19     Remove all instances covered by rule $R$ from original Dataset;
20     repeat
21         lines 2 to 19;
       until all instances of class $\omega_i$ have been removed;
23     Reset input Dataset to its initial state;
   end
25 return induced rules;
```

3 G-Prism: Inducing Rule Terms directly from Numerical Attributes

This section describes rule induction directly from numerical attributes using Gauss Probability Density Distribution (GPDD) termed G-Prism. Section 3.1 introduces the in [10] published proof of concept for G-Prism with fixed rule term boundaries and Section 3.2 introduces the new version of G-Prism using dynamic rule term boundaries.

3.1 Prism using GPDD to induce Rule Terms for Numerical Attributes

The work described in this section is inspired by a data stream classifier that also uses GPDD to induce rule terms from numerical real-time data sources [13]. We have incorporated this rule term structure into G-Prism. The Gaussian distribution is calculated with mean μ and variance σ^2 for the values of a numerical attribute α matching a given target class ω_i in the training dataset. The most relevant value for a numerical attribute α for the given target class ω_i is obtained from this Gaussian distribution. Equation 1 can be used to calculate the conditional probability for class ω_i for a given attribute value α_j:

$$\mathbb{P}(\alpha_j|\omega_i) = \mathbb{P}(\alpha_j|\mu, \sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}} exp(-\frac{(\alpha_j - \mu)^2}{2\sigma^2}) \tag{1}$$

A value for $\mathbb{P}(\omega_i|\alpha_j)$ or $log(\mathbb{P}(\omega_i|\alpha_j))$ can be calculated using Equation 2. This value is then used to acertain the probability of class label ω_i for a valid value of attribute α_j.

$$\log(\mathbb{P}(\omega_i|\alpha_j)) = log(\mathbb{P}(\alpha_j|\omega_i)) + log(\mathbb{P}(\omega_i)) - log(\mathbb{P}(\alpha_j)) \tag{2}$$

The Gaussian distribution for a class label can then be used to determine the probability of an attribute value α_j belonging to class label ω_i, assuming that α_j lies between an upper and lower bound Ω_i. This is based on the assumption that the values close to μ represent the most common values of numerical attribute α_j for ω_i. This is depicted in Figure 2, values in the shaded area are more relevant for ω_i than those outside the shaded area.

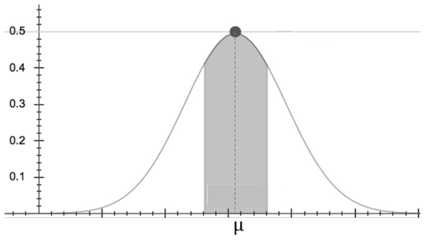


Fig. 2: Gaussian distribution of a classification from a continuous attribute

G-Prism uses the next smaller attribute value x and next larger attribute value y from μ to build a rule term $(x < \alpha_j \leq y)$. Then G-Prism calculates $\mathbb{P}(\omega_i|x < \alpha_j \leq y)$. G-Prism does this for each numerical attribute and selects the rule term with the highest conditional probability to specialise the rule further. In this process G-Prism also considers categorical rule terms which are induced in the same way as in the original Prism algorithm. Otherwise G-Prism follows the same rule specialisation process as outlined in Algorithm 1.

A test of normal distribution may be applied prior to applying this method. Attributes that are not normally distributed can alternatively be dealt with by binary splitting as outlined in the Prism algorithm in Section 2.

3.2 Prism using GPDD with Dynamic Rule Term Boundaries

As explained in Section 3.1 above, Gaussian distribution is calculated for a continuous attribute α with mean μ and σ^2. The range of values which extends to both sides from μ of the distribution should represent the most common values of attribute α_j for the target class ω_i. For each continuous attributes in a dataset, the original G-Prism uses the class conditional density probability of the Gaussian distribution to find a rule term in the form of $(x < \alpha \leq y)$, which can maximise the probability with which the rule term covers the target class. As illustrated in Figure 3 (a), the mean value (μ) of the attribute in the middle of the shaded area represents the highest posterior class probability while x and y are the next smaller and larger values from μ. A rule term is produced using these two values.

However, this is a very conservative strategy for finding good rule term boundaries as the GPDD for a range beyond the current fixed boundaries may still be very high. Thus the approach taken in G-Prism may result in the rules only covering few instances, which in turn may lead to overfitting of rules and more rules to be induced.

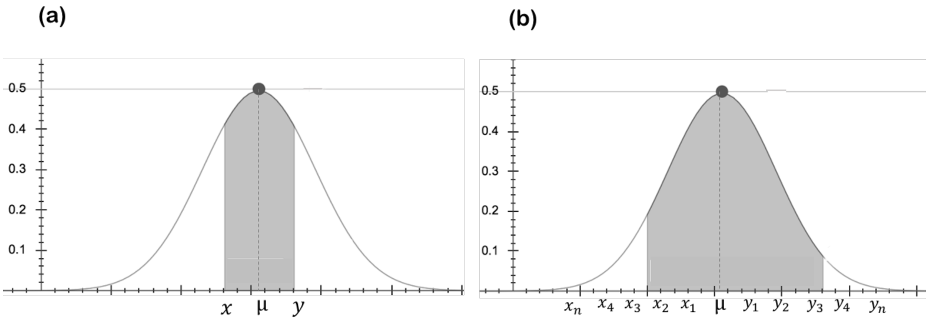


Fig. 3: The shaded area represents a range of values of attributes α_j for class ω_i. **(a)** 68% of all possible values, **(b)** 95% of all possible values

To overcome this problem, the shaded area under the curve is expanded as shown in Figure 3 (b). Thus, more attribute values are tested before choosing a highly relevant range of values that maximises the coverage of a rule for a target class. This means that instead of generating one range in the form of $(x < \alpha \leq y)$ by selecting the next lower bound (x) and the next upper bound (y), more ranges can be dynamically produced in such as $(x_1 < \alpha \leq y_3)$, $(x_3 < \alpha \leq y_5)$, $(x_2 < \alpha \leq y_4)$,... $(x_n < \alpha \leq y_k)$.

Algorithm 2: Learning classification rules from labelled data instances using G-Prism with Dynamic Bounds.

```
1  for $i = 1 \rightarrow C$ do
2      D ← input Dataset;
3      while $D$ does not contain only instances of class $\omega_i$ do
4          forall attributes $\alpha_j \in D$ do
5              if attribute $\alpha_j$ is categorical then
6                  Calculate the conditional probability, $\mathbb{P}(\omega_i|\alpha_j)$ for all possible
                     attribute-value $(\alpha_j = x)$ from attribute $\alpha$;
7              else if attribute $\alpha_j$ is numerical then
8                  calculate mean $\mu$ and variance $\sigma^2$ of continuous attribute $\alpha$ for class $\omega_i$;
9                  foreach value $\alpha_j$ of attribute $\alpha$ do
10                     calculate $\mathbb{P}(\alpha_i|\omega_i)$ based on created Gaussian distribution created in
                         line 8;
                   end
12                 for $n = maxBound \rightarrow 1$ do
13                     for $k = 1 \rightarrow maxBound$ do
14                         calculate $\mathbb{P}(x_n < \alpha_j \leq y_k)$ ;
                       end
                   end
               end
           end
19         select $(\alpha_j = x)$ or $(x_n < \alpha_j \leq y_k)$ with the maximum conditional probability as a
             rule term ;
20         D ← S, create a subset S from D containing all the instances covered by selected
             rule term at line 19;
       end
22     The induced rule $R$ is a conjunction of all selected rule terms built at line 19;
23     Remove all instances covered by rule $R$ from original Dataset;
24     repeat
25         lines 2 to 23;
       until all instances of class $\omega_i$ have been removed;
27     Reset input Dataset to its initial state;
   end
29 return induced rules;
```

In the current implementation the user can choose the maximum upper and lower bound considered from μ and any combination of rule terms within these bounds is considered. For example, if the user has chosen a bound of 3, then there are (3^2) possible rule terms that are considered for this attribute. Theoretically it is possible not to impose a bound (apart form the minimum and maximum values of the attribute) and allow any possible rule term combinations fanning out from μ. However, this is likely to result in a computationally very expensive approach especially if there are many distinct values for a particular attribute. Also it is statistically unlikely that rule terms spanning far from the μ will cover many instances of the target class. In our current implementation the default boundary is 6, which is also used in all of the experiments presented in Section 4. This setting worked well in most case. However, the user is able to specify a different lower and upper boundary. We termed this improved version of G-Prism *Dynamic Bound G-Prism*, which is illustrated in the Algorithm 2. The algorithm uses the Prism rule induction strategy as outlined in Algorithm 1 combined with the in this Section presented generation of rule terms which is contained in lines 7-19 in Algorithm 2. As the algorithm follows a 'separate-and-conquer' strategy the number of training instances decreases over time. Thus after each iteration μ and σ^2 are updated and the bounds are selected from the currently available values of the numerical attribute.

As mentioned in Section 2 the Prism family, including the new Dynamic Bound G-Prism has the ability to abstain from a classification if it is uncertain. This is desirable in applications where incorrectly classified instances are either costly or risky. This abstaining property is retained in both versions of G-Prism.

4 Empirical Evaluation

The main goal of the experimental evaluation is to compare the performance of the dynamic rule term boundary approach with binary splitting in Prism and the fixed boundary approach in G-Prism. Binary splitting is the discretisation method that has been applied to Prism in [3] to deal with numerical attributes. Therefore, our comparison is against this implementation of Prism.

4.1 Experimental Setup

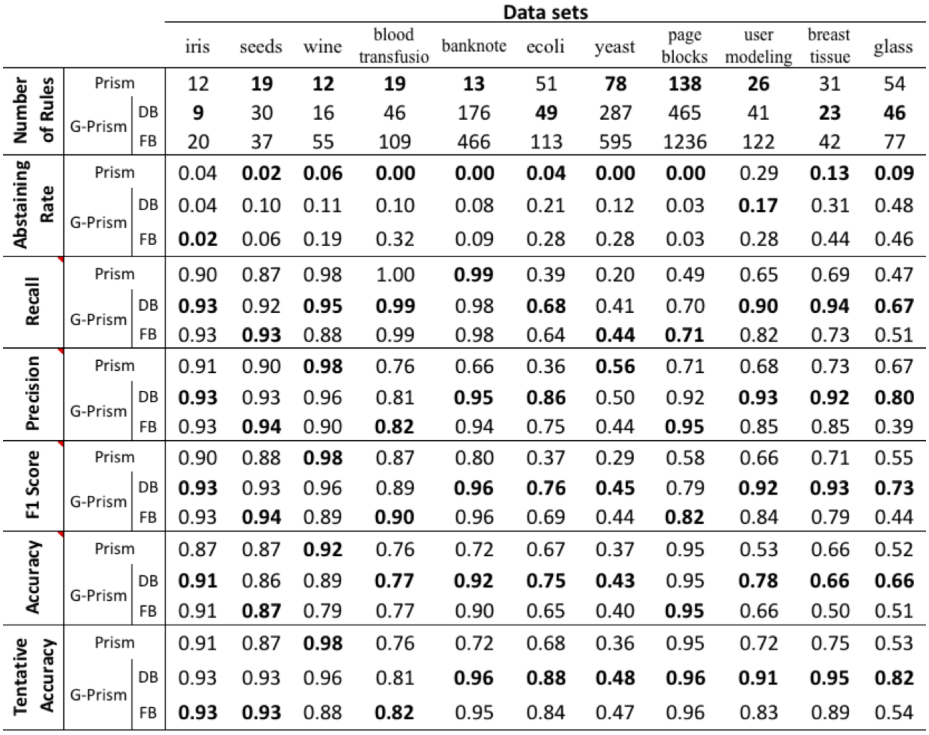
The algorithms used for the evaluation are Prism [14] incorporating the rule induction strategy for numerical attributes as outlined in Section 2 as a baseline, G-Prism as published in [10] with fixed size rule term boundaries which is termed G-Prism-FB and G-Prism as described in this paper with dynamic rule term boundaries which is termed G-Prism-DB. Please note that the original publication of G-Prism-FB [10] produced a proof of concept with limited empirical evaluation, thus this paper also aims to provide a more detailed empirical analysis of G-Prism-FB. All algorithms have been implemented in the statistical programming language R [15]. The fixed sized boundary of G-Prism-FB was set to one value smaller and one value larger than μ, which is how is was originally implemented in [10]. The dynamic sized boundary was set to allow a range up to 6 smaller and 6 larger values from μ. The algorithms have been applied to 11 datasets from the UCI repository [16]. These datasets were chosen randomly from all datasets in the UCI repository that comprise numerical attributes only and require classification tasks. The reason for choosing datasets with numerical attributes only is because the G-Prism-FB and G-Prism-DB algorithms are distinct from the baseline Prism only with respect to processing numerical attributes. The datasets have been randomly sampled without replacement into train and test datasets, whereas the testset comprises 30% of the data. On each of the datasets the algorithms were evaluated against 6 evaluation metrics for classifiers which are described below:

- *Abstaining Rate:* Prism, G-Prism-FB and G-Prism-DB abstain from a classification if a case is not covered in the ruleset. This would be very useful in applications where a wrong classification potentially leads to costly or dangerous consequences. The abstain rate is the ratio of instances that remain unclassified in the testset. A low abstain rate may be desired, however, this may be at the expense of accuracy. This is a number between 0 and 1.
- *Accuracy*: This is the ratio of data instances that have been correctly classified. Unclassified instances are classified using the majority class strategy. A high classification accuracy is desired. This is a number between 0 and 1.

- *Tentative Accuracy (Precision)*: Different compared with the accuracy above, the tentative accuracy is the ratio of correctly classified instances based only on the number of instances that have been assigned a classification. A high tentative accuracy is desired. This is a number between 0 and 1.
- *Recall*: The recall is the probability with which a data instance is classified correctly. A high recall is desired. This is a number between 0 and 1.
- *F1 Score*: This is the product of recall and tentative accuracy divided by their average. This is also known as the harmonic mean of precision and recall. A high F1 Score is desired. This is a number between 0 and 1.
- *Number of rules induced*: This is simply the total number of rules induced.

One should note that there is a direct relationship between accuracy, tentative accuracy and abstaining rate as abstained instances are counted as misclassifications in the accuracy measure, but are not considered at all in the tentative accuracy measure. Thus a higher abstain rate will also result in a lower accuracy and a higher tentative accuracy.

4.2 Empirical Results



			Data sets										
			iris	seeds	wine	blood transfusio	banknote	ecoli	yeast	page blocks	user modeling	breast tissue	glass
Number of Rules	Prism		12	**19**	**12**	**19**	**13**	51	**78**	**138**	**26**	31	54
	G-Prism	DB	**9**	30	16	46	176	**49**	287	465	41	**23**	**46**
		FB	20	37	55	109	466	113	595	1236	122	42	77
Abstaining Rate	Prism		0.04	**0.02**	**0.06**	**0.00**	**0.00**	**0.04**	**0.00**	**0.00**	0.29	**0.13**	**0.09**
	G-Prism	DB	0.04	0.10	0.11	0.10	0.08	0.21	0.12	0.03	**0.17**	0.31	0.48
		FB	**0.02**	0.06	0.19	0.32	0.09	0.28	0.28	0.03	0.28	0.44	0.46
Recall	Prism		0.90	0.87	0.98	1.00	**0.99**	0.39	0.20	0.49	0.65	0.69	0.47
	G-Prism	DB	**0.93**	0.92	**0.95**	**0.99**	0.98	**0.68**	0.41	0.70	**0.90**	**0.94**	**0.67**
		FB	0.93	**0.93**	0.88	0.99	0.98	0.64	**0.44**	**0.71**	0.82	0.73	0.51
Precision	Prism		0.91	0.90	**0.98**	0.76	0.66	0.36	**0.56**	0.71	0.68	0.73	0.67
	G-Prism	DB	**0.93**	0.93	0.96	0.81	**0.95**	**0.86**	0.50	0.92	**0.93**	**0.92**	**0.80**
		FB	0.93	**0.94**	0.90	**0.82**	0.94	0.75	0.44	**0.95**	0.85	0.85	0.39
F1 Score	Prism		0.90	0.88	**0.98**	0.87	0.80	0.37	0.29	0.58	0.66	0.71	0.55
	G-Prism	DB	**0.93**	0.93	0.96	0.89	**0.96**	**0.76**	**0.45**	0.79	**0.92**	**0.93**	**0.73**
		FB	0.93	**0.94**	0.89	**0.90**	0.96	0.69	0.44	**0.82**	0.84	0.79	0.44
Accuracy	Prism		0.87	0.87	**0.92**	0.76	0.72	0.67	0.37	0.95	0.53	0.66	0.52
	G-Prism	DB	**0.91**	0.86	0.89	**0.77**	**0.92**	**0.75**	**0.43**	0.95	**0.78**	**0.66**	**0.66**
		FB	0.91	**0.87**	0.79	0.77	0.90	0.65	0.40	**0.95**	0.66	0.50	0.51
Tentative Accuracy	Prism		0.91	0.87	**0.98**	0.76	0.72	0.68	0.36	0.95	0.72	0.75	0.53
	G-Prism	DB	0.93	0.93	0.96	0.81	**0.96**	**0.88**	**0.48**	**0.96**	**0.91**	**0.95**	**0.82**
		FB	**0.93**	**0.93**	0.88	**0.82**	0.95	0.84	0.47	0.96	0.83	0.89	0.54

Fig. 4: Overview of empirical results. DB denotes Dynamic Boundaries and FB denotes Fixed Boundaries

Figure 4 gives an overview of the results obtained using the datasets and evaluation metrics explained in Section 4.1. A more detailed breakdown of the results is given in the following illustrations in this section. For easier description we refer to G-Prism with Dynamic Boundaries as G-Prism-DB and G-Prism with Fixed Boundaries as G-Prism-FB.

Figure 5 illustrates the difference of the accuracies and tentative accuracies of G-Prism-DB and G-Prism-FB compared with Prism. As can be seen in the figure G-Prism-DB achieves a better accuracy compared with Prism in 7 out of 11 datasets. In 7 out of 11 cases G-Prism-DB is considerably better and in one case (blood transfusion) only marginally better. However, in the cases where G-Prism-DB has a lower accuracy this accuracy is only marginally lower. G-Prism-FB does not seem to outperform Prism, in 5 cases it has a higher accuracy and in 7 it does not. With respect to tentative accuracy both versions of G-Prism clearly outperform Prism, both achieve a higher tentative accuracy in 10 out of 11 cases. Furthermore G-Prism-DB in comparison with G-Prism-FB achieves a better accuracy in 7 out of 11 cases and in all cases where G-Prism-DB did not perform better than G-Prism-FB, it achieved only marginally lower accuracy.

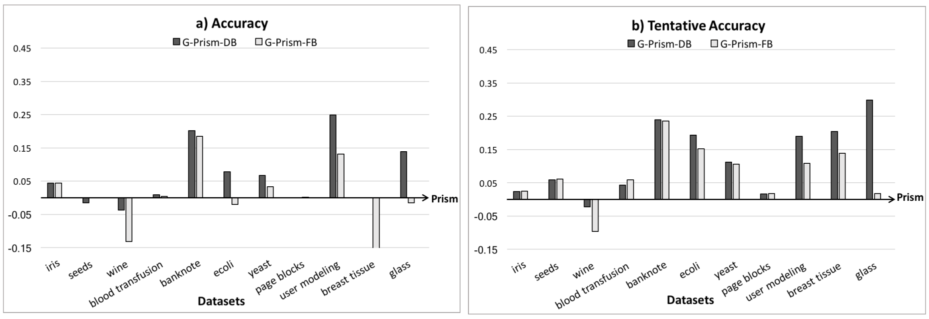


Fig. 5: Difference of Accuracy and Tentative Accuracy of G-Prism-DB and G-Prism-FB compared with Prism

Figure 6 illustrates the difference of the F1 Score and Recall of G-Prism-DB and G-Prism-FB compared with Prism. Both versions of G-Prism outperform Prism in 8 out of 11 cases. G-Prism-DB seems to be the better of the two G-Prism versions with G-Prism-DB achieving a higher recall in 7 out of 11 cases compared with G-Prism-FB. With respect to the F1 Score G-Prism achieves a higher score compared with Prism in 10 out of 11 cases. Again, G-Prism-DB seems to be the better of the two G-Prism versions with G-Prism-DB achieving a higher F1 Score in 7 out of 11 cases compared with G-Prism-FB.

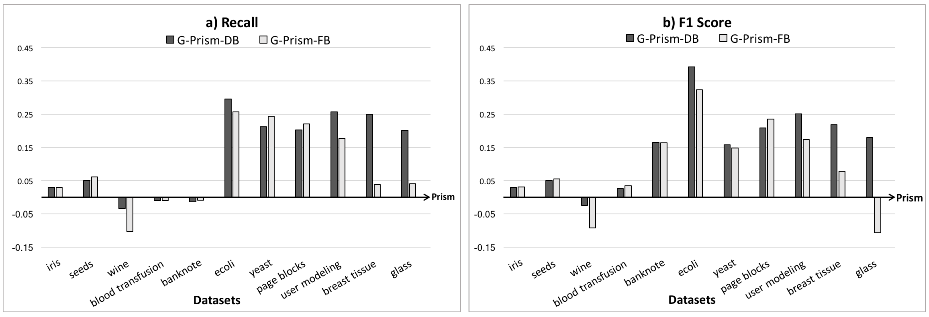


Fig. 6: Difference of Recall and F1 Score of G-Prism-DB and G-Prism-FB compared with Prism

Regarding abstaining rate as illustrated in Figure 7 the predecessor Prism seems to have a lower abstain rate in most cases (9 out of 11). Comparing both version of G-Prism, G-Prism-DB achieves a lower abstain rate compared with G-Prism-FB in 8 out of 11 cases.

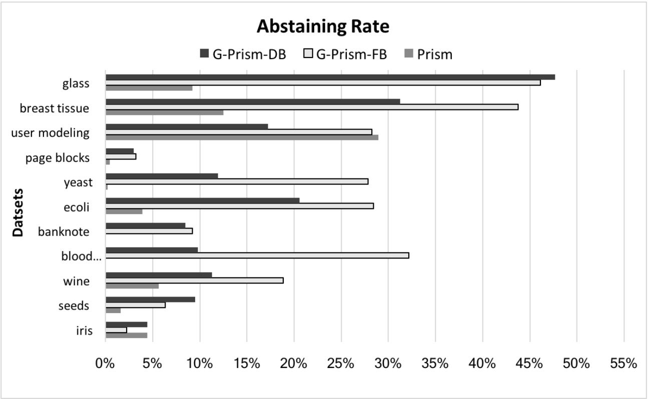


Fig. 7: Abstaining rates of G-Prism-DB, G-Prism-FB and Prism

4.3 Summary of Evaluation

Overall it can be seen that G-Prism outperforms Prism in most cases with regards to all evaluation metrics except for the abstaining rate where Prism performs better. This could potentially be linked to the high number of rules induced by G-Prism approaches as can be seen in Figure 4. A high number of rules suggests that the rule terms for each rule covers a lower number of instances compared with Prism. However, what can also be seen is that G-Prism-DB induces less rules compared with G-Prism-FB and also has a lower abstain rate compared with G-Prism-FB. The difference between both G-Prism approaches is that G-Prism-DB had a dynamic rule term boundaries that are either the same or cover a wider range and thus also produce potentially less rules but covering a larger number of training instances. Overall G-Prism-DB achieves a better classification performance compared with Prism and G-Prism-FB. Alternative descritisation methods, such as ChiMerge [11] and Chi square [12], prior to the application of Prism will also be considered in the future.

5 Ongoing

Current ongoing work comprises four aspects of the current G-Prism with Dynamic Boundaries approach, which are (1) an improved parameter settings for the dynamic boundaries and (2) the assumption of normally distributed attributes. With respect to (1), the parameter settings for dynamic boundaries, currently there is a user defined threshold of maximum boundary values to be considered left and right of μ, which is by default set to 6. However, the optimal number of maximum steps to be considered may be higher or lower than the by the user define threshold and this may also be dependent on the number of training instances as a larger number of training instances is likely to produce a larger number of distinct values. For attributes with a smaller number of distinctly different values it is more likely that the maximum boundary threshold is further away from μ compared with attributes with a larger number of distinctly different values. In order to resolve this limitation we are currently considering implementing a version of the algorithm using the interquartile range as an upper rule term boundary to limit the rule term boundary search space. We expect that this will produce rules with a larger coverage of data and thus reduce the abstain rate. With respect to (2), the assumption of normally distributed attributes, we have not tested for normal distribution in the data used in our experiments, yet G-Prism-DB outperforms its predecessor Prism in many respects. Thus there is the possibility that G-Prism may not perform as well on attributes that are not normally distributed compared with its predecessor Prism. Therefore, we are currently implementing a hybrid approach that tests if an attribute's values are normally distributed, if it they are, then the algorithm would use the G-Prism approach, otherwise it would use the Prism approach to induce rule terms from that a particular attribute. Moreover, future work comprises the development of novel methods for rule term induction taking different underlying attribute value distributions into consideration, such as i.e. Poison distribution. Also the implementation of other discretisation methods based on ChiMerge [11] and Chi square [12] for a further comparisons is planned in the future work.

6 Conclusion

The paper introduced a new rule term induction method based on Gauss Probability Density Distribution to produce a new rule term structure that improves classification performance of the Prism family of algorithms. The new rule term structure is an alternative structure to the currently used rule terms in the Prism family of algorithms. The basic idea of the new rule term structure had been introduced by the authors in [10] in a short paper but only limited evaluation was conducted at the time. This paper offers two contributions (1) a thorough evaluation of the originally proposed G-Prism algorithm and (2) an improvement to the G-Prism rule term induction method by using more dynamic maximum rule term boundaries. Both G-Prism approaches (with fixed and dynamic rule term boundaries) and their predecessor Prism have been evaluated empirically and comparatively using various metrics and datasets. Overall G-Prism with dynamic boundaries outperforms Prism and G-Prism with fixed boundaries in most cases. Regarding abstain rate we saw a larger number of rejected test instances by using G-Prism than by using Prism. We also observed that G-Prism (either of the two versions) produces more rules and we assume that this is related to a higher abstaining rate of G-Prism. Thus we are currently working on a more elastic dynamic rule term boundary selection that will likely lead to a higher coverage of data instances and thus is expected to reduce the abstaining rate of G-Prism. Other ongoing work comprises also the development of approaches that can be used to induce rule terms if an attribute is not normally distributed.

References

[1] J Ross Quinlan. *C4. 5: programs for machine learning.* Elsevier, 2014.

[2] Jadzia Cendrowska. “PRISM: An algorithm for inducing modular rules”. In: *International Journal of Man-Machine Studies* 27.4 (1987), pp. 349–370.

[3] Max Bramer. *Principles of data mining.* Vol. 131. Springer, 2016.

[4] Jiawei Han, Jian Pei, and Micheline Kamber. *Data mining: concepts and techniques.* Elsevier, 2011.

[5] Ian H Witten et al. *Data Mining: Practical machine learning tools and techniques.* Morgan Kaufmann, 2016.

[6] William W Cohen. “Fast effective rule induction”. In: *Proceedings of the twelfth international conference on machine learning.* 1995, pp. 115–123.

[7] Peter Clark and Tim Niblett. “The CN2 induction algorithm”. In: *Machine learning* 3.4 (1989), pp. 261–283.

[8] M A Bramer. “An Information-Theoretic Approach to the Pre-pruning of Classification Rules”. In: *Intelligent Information Processing.* Ed. by B Neumann M Musen and R Studer. Kluwer, 2002, pp. 201–212.

[9] Frederic Stahl and Max Bramer. “Computationally efficient induction of classification rules with the PMCRI and J-PMCRI frameworks”. In: *Knowledge-Based Systems* 35 (2012), pp. 49–63.

[10] Manal Almutairi et al. “Towards Expressive Modular Rule Induction for Numerical Attributes”. In: *Research and Development in Intelligent Systems XXXIII: Incorporating Applications and Innovations in Intelligent Systems XXIV.* Springer. 2016, pp. 229–235.

[11] Randy Kerber. “Chimerge: Discretization of numeric attributes”. In: *Proceedings of the tenth national conference on Artificial intelligence.* Aaai Press. 1992, pp. 123–128.

[12] IF Imam, RS Michalski, and L Kerschberg. “Discovering attribute dependence in databases by integrating symbolic learning and statistical analysis techniques”. In: *Proceeding of the AAAI-93 Workshop on Knowledge Discovery in Databases, Washington DC.* 1993.

[13] Thien Le et al. “Computationally efficient rule-based classification for continuous streaming data”. In: *Research and Development in Intelligent Systems XXXI.* Springer, 2014, pp. 21–34.

[14] Max Bramer. “Automatic induction of classification rules from examples using N-Prism”. In: *Research and development in intelligent systems XVI.* Springer, 2000, pp. 99–121.

[15] R Core Team. *R: A Language and Environment for Statistical Computing.* R Foundation for Statistical Computing. Vienna, Austria, 2014. URL: http://www.R-project.org/.

[16] M. Lichman. *UCI Machine Learning Repository.* 2013. URL: http://archive.ics.uci.edu/ml.