Computationally Efficient Rule-Based Classification for Continuous Streaming Data

Thien Le, Frederic Stahl, João Bártolo Gomes, Mohamed Medhat Gaber and Giuseppe Di Fatta

Abstract Advances in hardware and software technologies allow to capture streaming data. The area of Data Stream Mining (DSM) is concerned with the analysis of these vast amounts of data as it is generated in real-time. Data stream classification is one of the most important DSM techniques allowing to classify previously unseen data instances. Different to traditional classifiers for static data, data stream classifiers need to adapt to concept changes (concept drift) in the stream in real-time in order to reflect the most recent concept in the data as accurately as possible. A recent addition to the data stream classifier toolbox is *eRules* which induces and updates a set of expressive rules that can easily be interpreted by humans. However, like most rule-based data stream classifiers, eRules exhibits a poor computational performance when confronted with continuous attributes. In this work, we propose an approach to deal with continuous data effectively and accurately in rule-based classifiers by using the Gaussian distribution as heuristic for building rule terms on continuous attributes. We show on the example of eRules that incorporating our method for continuous attributes indeed speeds up the real-time rule induction process while maintaining a similar level of accuracy compared with the original eRules classifier. We termed this new version of eRules with our approach G-eRules.

T. Le (✉) · F. Stahl · G.D. Fatta
School of Systems Engineering, University of Reading, Whiteknights, PO Box 225, Reading RG6 6AY, UK
e-mail: t.d.le@pgr.reading.ac.uk

F. Stahl
e-mail: f.t.stahl@reading.ac.uk

G.D. Fatta
e-mail: g.difatta@reading.ac.uk

J.B. Gomes
Institute for Infocomm Research (I2R), A*STAR 1 Fusionopolis Way Connexis, Singapore 138632, Singapore
e-mail: bartologjp@i2r.a-star.edu.sg

M.M. Gaber
School of Computing Science and Digital Media, Robert Gordon University, Riverside East, Garthdee Road, Aberdeen AB10 7GJ, UK
e-mail: m.gaber1@rgu.ac.uk

M. Bramer and M. Petridis (eds.), *Research and Development in Intelligent Systems XXXI*, DOI 10.1007/978-3-319-12069-0_2

1 Introduction

One of the problems in the area of Big Data Analytics is concerned with is the analysis of high velocity data streams. A data stream is defined as data instances that are generated at a high speed and thus challenges our computational capabilities for the processing of this data [1]. There are many applications that generate such fast and possibly infinite data streams, such as sensor networks, communication networks, Internet traffic, stock markets [2, 3].

Unlike traditional data mining systems that process static (batch) data, data stream mining algorithms analyse the streaming data on the fly in order to avoid storing these possibly infinite amounts of data. Data stream classification is only one but very important data mining task on streaming data. Data stream classifiers learn and adapt to changes in the data (concept drifts) in real-time and thus require only a single pass through the training data. A concept drift occurs if the current data mining model (i.e. the classifier) is no longer valid through a change of the distribution in the data stream.

A recent data stream classifier development is eRules [4]. eRules is based on a sliding window approach [5] and uses the rule-based Prism classifier [6] to work on streaming data. eRules has shown good classification accuracy and adaptability to concept drifts [4]. eRules induces rules of the form *IF condition THEN classification* that are expressive and compact and effectively represent information for classification. The condition is a conjunction of rule terms of the form ($Attribute = Value$) for categorical attributes and ($Attribute < Value$) or ($Attribute \geq Value$) for continuous data. Compared with other stream classifiers (such as Hoeffding Trees) eRules tends to leave a data instance unclassified rather than forcing a possibly wrong classification. This feature is highly desirable in many applications where the results from miss-classification are costly and un-reversible, e.g.: medical and financial applications. Further weaknesses of the decision tree structure in comparison with rulesets have been analysed, and the reader is referred to [4] for further reading.

However, eRules method of processing continuous attributes is computationally very expensive and hence results in long processing times, which is a disadvantage on the application on data streams. In this paper, we propose a new heuristic method based on the Gaussian distribution in order to make eRules computationally more efficient. We show empirically that our method improves eRules' processing times and competes well with other data stream classifiers. We termed this new version of the classifier G-eRules, where G stands for the use of the **G**aussian distribution. Our method is not only applicable to eRules but could potentially be adopted in other rule and decision tree based stream classifiers as well.

The paper is organised as follows. Section 2 highlights related work in the field of rule-based data stream classification. Section 3 discusses the principles of the eRules classifier and our new approach for dealing with continuous attributes based on the Gaussian distribution. An empirical analysis of eRules classifier with our approach, termed G-eRules is presented in Sect. 4. Concluding remarks and a brief description of ongoing work are given in Sect. 5.

2 Related Work

Many algorithms for data stream classification have been proposed. Gaber et al. have identified such techniques and their features in [7]. Among these techniques, possibly the best established methods, are the Hoeffding tree based approaches [8]. Nevertheless, tree based classification approaches have been criticised in the past for both static data classification [6, 9] as well as for data stream classification [4, 10]. In particular, the drawbacks of decision tree based techniques in dealing with concept drift are mentioned in [10] and the fact that decision trees cannot abstain from a potentially wrong classification are discussed in [4]. Hence, alternative rule-based approaches have been developed for data stream classification, such as the aforementioned eRules algorithm [4] or Very Fast Decision Rules (VFDR) [11], both based on the separate and conquer rule induction approach.

eRules induces expressive *IF-THEN* rules and performs well in terms of classification accuracy and adaptation to concept drifts, also it is able to abstain from classification. However, eRules drawback is its computational efficiency when dealing with continuous attributes. To the best of our knowledge, there is no single optimised method to deal with continuous attributes for rule induction based algorithms. Methods for working with continuous attributes have been studied extensively in the past 2 decades for batch data. An overview of well-known methods for data discretisation in batch data is described in [12] and most of these methods cannot be adopted for data streams without significant modifications, and consequently they are unlikely to perform as good as they do on batch data. The work presented in this paper addresses the problem of low computational efficiency in generating rules from continuous attributes, especially for eRules as a representative technique, by developing a heuristic method based on the Gaussian distribution.

3 Computationally Efficient Rule Term Induction for Continuous Attributes in Data Streams

This section presents a new method for inducing rule terms from continuous attributes that is computationally more efficient, assuming a Gaussian distribution of the values of continuous attributes. We first highlight eRules conceptually and discuss its underlying rule induction mechanism. Then we present an alternative approach of dealing with continuous attributes with a different rule term structure. We argue and show empirically that this approach is computationally more efficient.

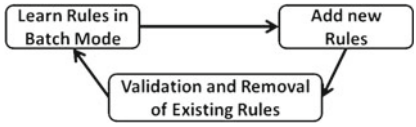

Fig. 1 The three main processes in eRules

3.1 eRules and Its Rule Induction Approach

The eRules classifier has three main processes as illustrated in Fig. 1.

Essentially eRules makes use of Cendrowska's Prism algorithm [6] to learn classification rules from data streams by using a sliding window approach [5]. The three main processes of eRules illustrated in Fig. 1 are outlined as follow:

Learn Rules in Batch Mode—An initial ruleset is learnt using Prism on the first batch of incoming data instances. Later instances are added to a buffer if they are not covered by the current ruleset. If the number of instances in the buffer reaches a user defined threshold then its instances are used to induce new rules (using Prism) and the buffer is emptied.

Add New Rules—Whenever new rules are generated from the aforementioned buffer then they are added to the current ruleset and thus adapt to new emerging concepts in the data stream.

Validate and Remove Existing Rules—eRules also removes learnt rules from the model if the concerning rules are not relevant to the current concept anymore, i.e. if a concept drift occurred. This is quantified by the deterioration of the individual rule's classification accuracy over time.

Algorithm 1 shows the basic Prism rule induction method employed by eRules. ω_i denotes a target class for a rule and C is the number of classes; α_j is a rule term of the form $(\alpha < v)$ or $(\alpha \geq v)$ for continuous attributes or $(\alpha = v)$ for categorical attributes, where α denotes an attribute name and v a valid value for this attribute.

Algorithm 1: Prism classification rule induction algorithm

```
1  for i = 1 → C do
2      D ← original Dataset
3      while D contains classes other than ω_i do
4          forall the attributes α in D do
5              Calculate probability of occurrence, p(ω_i|α_j) for each possible rule term α_j;
           end
7          Select the α_j with the maximum probability of occurrence, p(ω_i|α_j) as rule term;
8          Create subset S of D containing all the instances covered by α_j;
9          D ← S;
       end
11     The induced rule, R is a conjunction of all α_j at line 7;
12     Remove all instances covered by rule R from original Dataset;
13     repeat
14         lines 2 to 12
       until all instances of ω_i have been removed from original Dataset;
16     Reset original Dataset to its initial state;
   end
```

3.2 Current Rule Term Induction Method Based on Conditional Probability

The original Prism approach [6] does only work on categorical data for inducing rule terms of the form $(\alpha = v)$. On the other hand, eRules is also able to work on continuous attributes in a very similar way compared with the VFDR [11] algorithm.

For continuous attributes eRules produces rule terms of the form $(\alpha < v)$, or $(\alpha \geq v)$ and VFDR produces rule terms of the form $(\alpha \leq v)$, or $(\alpha > v)$. The process of eRules dealing with a continuous attribute is outlined in the following steps:

1. Sort the dataset according to the attribute value.
2. For each possible value v of the continuous attribute α, calculate the probability for a target class for both terms $(\alpha < v)$ and $(\alpha \geq v)$.
3. Return the term, which has overall highest probability for the target class.

It is evident that this method of dealing with continuous attributes in eRules requires many cutpoint calculations for the conditional probabilities $p(\omega_i|\alpha < v)$ and $p(\omega_i|\alpha \geq v)$ for each attribute value v, where ω_i is the target class. Thus, this method decreases the computational efficiency of eRules and VFDR considerably. We propose to use the Gaussian distribution of the attribute values associated with the target class in order to create rule terms of the form $(x < \alpha \leq y)$, and thus avoid frequent cutpoint calculations, as described in Sect. 3.3.

3.3 Using Gaussian Distribution to Discriminant Classification

For each continuous attribute in a dataset, we can generate a Gaussian distribution as shown in Fig. 2 to represent all possible values of that continuous attribute for a target class.

Assume a dataset with classifications, $\omega_1, \ldots, \omega_i$. If we have a measurement vector (of attribute values) x then we can compute the attribute value that is the most relevant one to a particular classification based on the Gaussian distribution of the values associated with this particular classification.

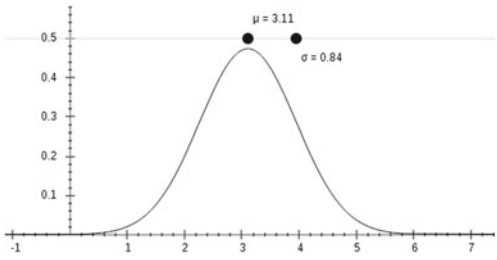

Fig. 2 Gaussian distribution of a classification from a continuous attribute

The Gaussian distribution is calculated for a continuous attribute α with mean μ and variance σ^2 from all attribute values with classification, ω_i. The class conditional density probability is then given by:

$$p(\alpha_j|\omega_i) = p(\alpha_j|\mu, \sigma^2) = \frac{1}{\sqrt{2r\sigma^2}}\exp(-\frac{(\alpha_j - \mu)^2}{2\sigma^2}) \quad (1)$$

Then a heuristic measurement of posterior class probability, $p(\omega_i|\alpha_j)$, or equivalently $\log(p(\omega_i|\alpha_j))$ can be calculated and used to determine the probability of a target class for a valid value of a continuous attribute.

$$\log(p(\omega_i|\alpha_j)) = \log(p(\alpha_j|\omega_i)) + \log(p(\omega_i)) - \log(p(\alpha_j)) \quad (2)$$

We calculate the probability of regions Ω_i for these attribute values such that if $x \in \Omega_i$ then x belongs to class ω_i. This approach may not necessarily capture the full details of the intricate continuous distribution, but it is highly efficient in computation and memory perspectives. This is because the Gauss distribution only needs to be calculated once and can then be updated when new data stream instances are received, by simply recalculating mean μ and variance σ^2.

The range of values, which extends to both sides from the mean μ of the distribution should represent the most common values of attribute α for a target class ω_i. A candidate rule term can be generated by selecting an area under the curve for a range of values for which the density class probability $p(x < \alpha \leq y|\omega_i)$ is the highest, where x and y are valid values of attribute α from the Gauss distribution for the target class ω_i. As shown in Fig. 3, the shaded area represents the highest density class probability $p(x < \alpha \leq y|\omega_i)$ of a subset from the training dataset.

Globally, the area right in the middle under the curve represents the most common values of the attribute for a target class. For example, the shaded area of one standard deviation of the mean ($\mu \pm 1\sigma$), as illustrated in Fig. 4a, covers 68 % of all possible values of the attribute for the target class; or as illustrated in Fig. 4b 95 % of all possible values of the attribute for the area of ($\mu \pm 1.96\sigma$) [13].

However, distributions for different classifications can overlap each other and an area of a distribution for a target class sometimes cannot be used to precisely distinguish a classification as shown in Fig. 5.

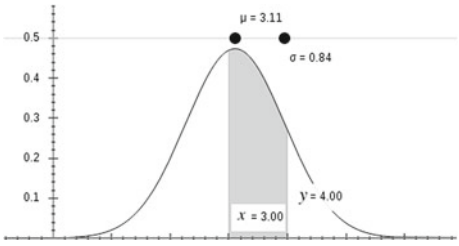

Fig. 3 *Shaded area* represents a range of values of attribute α for class ω_i

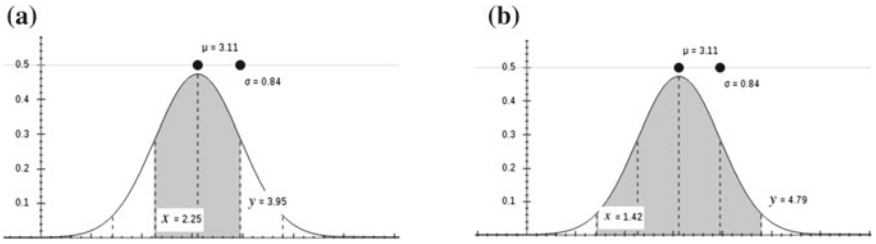

Fig. 4 *Shaded area* represents a range of values of attribute α for class ω_i. **a** 68 % of all possible values. **b** 95 % of all possible values

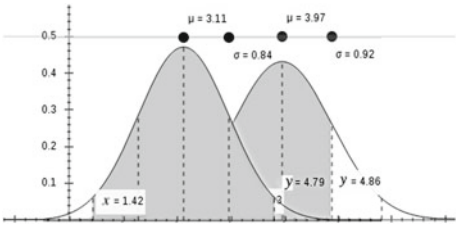

Fig. 5 Distributions of different classification overlap each other

However, we are interested to find a rule term, which can maximise the coverage of the rule for a target class. Therefore, our approach uses density estimation to discover a rule term in the form of $(x < \alpha \leq y)$ by selecting only a highly relevant range of values from a continuous attribute, which can then be used to represent a subset of instances for the target class along with other rule terms. Our approach for rule induction from continuous attributes can be described with the following steps:

1. For each continuous attribute, calculate a Gaussian distribution with mean μ and variance σ^2 for each classification.
2. Calculate class conditional density and posterior class probability for each continuous attribute value for the target class, using Eqs. (1) and (2).
3. Select the value of the attribute with greater posterior class probability.
4. Select the next smaller and larger values from the value chosen in **step 3** which have the greatest posterior class probability.
5. Calculate density probability with two values from **step 4** from the normal distribution for the target class.
6. Select the range of the attribute $(x < \alpha \leq y)$ as the rule term for which density class probability is the maximum.

Algorithm 2 outlines eRules using our approach for extracting rule terms from continuous attributes. The resulting algorithm is termed G-eRules, where G stands for Gauss distribution.

Algorithm 2: G-eRules induction approach for continuous attributes

```
1  for i = 1 → C do
2      D ← original Dataset
3      while D contains classes other than ω_i do
4          forall the α in D do
5              calculate mean μ and variance σ^2 of continuous attribute α for class ω_i;
6              foreach value α_j of attribute α do
7                  Calculate p(α_j|ω_i);
               end
9              Select α_j of attribute α, which has highest value of p(α_j|ω_i);
10             For values < α_j select the value x that has the highest probability density in
               this range, and for values ≥ α_j select the value y that has the highest
               probability density in this range;
11             Calculate p(x < α ≤ y|ω_i);
           end
13         Select (x < α ≤ y) for which p(x < α ≤ y|ω_i) is a maximum;
14         Create subset S of D containing all the instances which has (x < α ≤ y);
15         Build a rule term describing S;
16         D ← S;
       end
18     The induced rule, R is a conjunction of all the rule terms built 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 ω_i have been removed;
23     Reset original Dataset to its initial state;
   end
```

4 Evaluation

The main aim of our experimental evaluation is to study the effectiveness of our proposed approach in terms of its computational performance but also its competitiveness in terms of its accuracy compared with other established data stream classifiers. The implementation of our experiments were realised in the Java based Massive Online Analysis (MOA) [14] framework, a workbench for evaluating data stream mining algorithms. MOA was chosen as it already implements a wide range of data stream classifiers.

4.1 Experimental Setup

In our experiments, we used four different classifiers for a comparative analysis:

1. *Very Fast Decision Rules (VFDR)*—[11] a rule-based data stream classifier available in MOA.

2. *Hoeffding Tree*—[8] is the state-of-art decision tree classifier implemented in MOA for data streams.
3. *eRules*—is outlined in Sect. 3.1, it is the only data stream classifier that is able to abstain from classifying when uncertain.
4. *G-eRules*—inherited from original eRules but this version of the classifier uses our continuous rule term format and induction approach outlined in Sect. 3.3.

The reason for these choices is that Hoeffding tree has been studied by the community [15] for the last decade as the state-of-the-art in data stream classification and VFDR is one of few rule-based classifiers, which can generate rules directly from a data stream, and thus shares similarities with eRules. We used the default settings for eRules and G-eRules in the experiments, which are: a sliding window of size 500; a particular rule is removed if it falls below an individual accuracy 0.8 and a has had a minimum number of 5 classification attempts. We use both, artificial (stream generators in MOA) and real datasets in our experiments:

SEA Generator—This artificial dataset is introduced in [16], it generates an artificial data stream with 2 classes and 3 continuous attributes, whereas one attribute is irrelevant for distinguishing between the 2 classes. The interested reader is referred to [16] for more information. This data generator has been used in empirical studies in [4, 11, 17] amongst others. For our evaluation we used the default generator settings which are concept function 1, a random instance seed of 1, allowing unbalanced classes and a noise level of 10 %. We generated 500,000 instances.

Random Tree Generator—This generator is introduced in [8] and is based on a randomly generated decision tree and the user can specify the number of attributes and classes. New instances are generated by assigning uniformly distributed random values to attributes and the class label is determined using the tree. Because of the underlying tree structure, decision tree based classifiers should perform better on this stream. We have generated 2 versions of this data stream, one with 5 categorical and 5 continuous attributes, and 3 classifications called RT Generator 5-5-3; and the other one also with 3 classifications but no categorical and only 4 continuous attributes called RT Generator 0-4-3. Both versions comprised 500,000 instances. The remaining settings were left at their default values, which are a tree random seed of 1, instance random seed option of 1, 5 possible values for each categorical attribute, a maximum tree depth of 5; minimum tree level for leaf nodes of 3, and the fraction of leaves per level from first leaf Level onwards is 0.15.

Covertype—is a dataset from US Forest Service (USFS) Region 2 Resource Information System (RIS) which contains the forest cover type for 30×30 m cells. There are 581,012 instances with 54 attributes and 10 of them are continuous attributes. This dataset has been used in several papers on data stream classification, i.e. in [18]. This real dataset was downloaded from the MOA website [19] and used without any modifications.

Airlines—this data source was created based on the regression dataset from Elena Ikonomovska, which was extracted from Data Expo 2009, which consists of about 500,000 flight records and the task is to use the information of scheduled departure to predict whether a given flight will be delayed. The datset comprises 3 continuous

and 4 categorical attributes. The dataset was also used in one of Elena's studies about data streams [20]. This real dataset was downloaded from the MOA website [19] and used without any modifications.

Instead of initialising each artificial generator for each experiment anew, we have created static datasets and streamed these datasets to the classifiers. We did not initialise each artificial data stream anew as the generators are non-deterministic. This way we ensure that each classifier is presented with exactly the same instances and the same order of instances in the stream.

4.2 Results

The evaluation is focussed on the scalability of G-eRules to fast data streams but also shows its competitiveness in terms of classification accuracy. We used 'Interleaved Test-Then-Train' strategy [14] to calculate the mean-accuracy of each of the algorithms. All parameters for artificial stream generators were left at default values unless stated otherwise.

4.2.1 Overall Accuracy, Tentative Accuracy and Learning Time

One feature of eRules is the ability to abstain from classification and G-eRules inherited this feature of eRules. Therefore, we also record tentative accuracy for eRules and G-eRules, which is the accuracy for instances where the classifier is confident.

We can see that G-eRules achieves a very similar classification accuracy compared with its original eRules classifier, however, it is significantly faster. Both, eRules and G-eRules suffer from higher abstain rates when confronted with data streams with many continuous attributes (i.e. Random Tree Generator datasets and SEA), however, G-eRules has a lower abstain rate on continuous data. Nevertheless, G-eRules not only has a lower abstain rate when confronted with continuous attributes, but also a higher accuracy.

G-eRules generally outperforms, in terms of accuracy and running time, existing rule-based classifier VFDR. In terms of accuracy G-eRules outperforms VFDR in 3 out of 4 cases and in general processes the data stream much faster. For the RT-Generator 0-4-3 stream VFDR is unexpectedly slow. This could be explained by VFDR not reaching a stable ruleset that keeps changing over time. Regarding the Airlines stream eRules and G-eRules need much longer to process the stream, however, again, this could be due to the fact that both algorithms are unstable on this particular dataset due to a fixed sliding window size. However, this problem can be addressed by using a sliding window that adapts its size dynamically to concept changes (see Sect. 5).

In comparison to Hoeffding trees, we achieve a comparable accuracy on the two real datasets Covertype and Airlines, on the MOA data generators Hoeffding tree

Table 1 G-eRules compared with eRules, Hoeffding trees and VFDR

Dataset	Classifier									
	eRules			G-eRules			Hoeffding tree		VFDR	
	Ab (%)	T.Ac (%)	T (ms)	Ab (%)	T.Ac (%)	T (ms)	Ac (%)	T (ms)	Ac (%)	T (ms)
CoverType	27.993	81.12	90,098	35.77	80.68	12,484	80.61	10,631	61.29	20,722
Airlines	9.63	61.76	366,859	16.68	62.55	116,975	65.20	4,289	61.02	4,818
RT generator 5-5-3	10.52	69.28	333,163	26.47	64.77	36,486	89.78	2,422	55.51	79,153
RT generator 0-4-3	96.35	34.94	47,790	65.65	81.21	4,283	96.39	2,391	90.10	3,883,291
SEA generator	94.06	68.58	22,043	40.84	95.87	3,227	99.9.3	1,265	62.70	176,705

The abbreviation *Ab* stands for abstain rate, *T.Ac* stands for tentative accuracy, *T* stands for execution time and *Ac* stands for accuracy

performs better. However, one needs to note that Hoeffding trees are less expressive classifiers than G-eRules and eRules. The rulesets generated by our techniques can easily be interpreted and examined by the users. On the other had, Hoeffding trees need a further processing step to explain the single path that led to a particular decision that may well be quite long for interpretation by decision takers. As such we argue that G-eRules is the most expressive rule-based technique for data stream classification.

In order to examine G-eRules' computational advantage on continuous attributes we have compared eRules and G-eRules on the same base datasets as in Table 1, but with increasing numbers of continuous attributes. SEA and Random Tree Generator (with continuous attributes only) were used, both with a gradual concept drift lasting 1,000 instances starting at the 250,000th instance in order to trigger adaptation and thus make the problem computationally more challenging. The two real datasets Covertype and Airlines have been chosen. For all datasets the continuous attributes were gradually increased by duplicating the existing attributes. The reason for duplication is that the concept encoded in the stream stays the same, but the computational effort needed to find the rules is increased.

As expected, Figs. 6a, b and 7a, b show that the execution times of G-eRules do increase at a much smaller rate than the execution times of eRules do. In fact, G-eRules' increase in execution time when confronted with an increasing number of continuous attributes seems to be very close to that of Hoeffding trees. The reason for this is that eRules has to work out $p(\omega_i|\alpha < v)$ and $p(\omega_i|\alpha \geq v)$ for each value of a continuous attribute and then this process is repeated during the learning.

On the other hand, for each continuous attribute, G-eRules only has to calculate Gaussian distributions for each classification once and then the classifier can update and look up $p(\omega_i|x < \alpha \leq y)$ value from created distributions. Please note we have omitted the execution times for VFDR in Figs. 6a, b and 7a, b, as they are generally

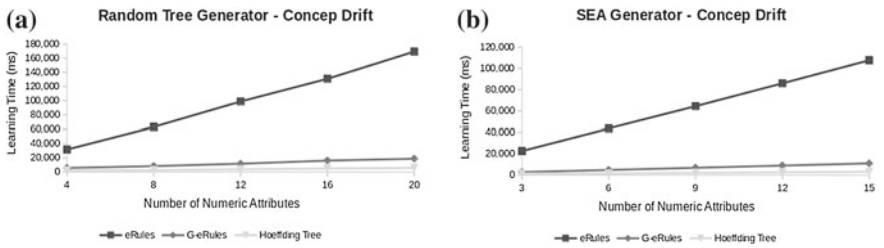

Fig. 6 Learning time of random tree and SEA generators with concept drift

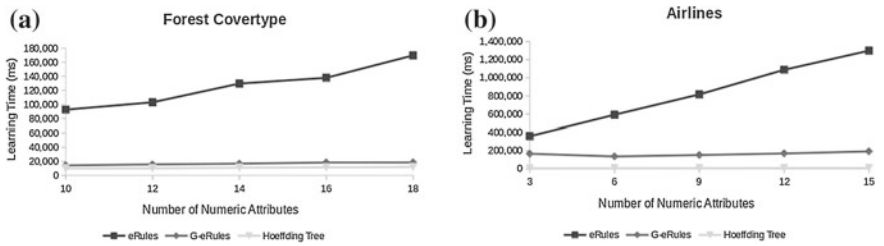

Fig. 7 Learning time of real datasets, CoverType and airlines

much greater compared with eRules and G-eRules on these particular cases, as shown in Table 1.

Moreover, we can observe that eRules tends to produce rules with irrelevant terms for continuous attributes, resulting in some of the rules produced by eRules being too specific. In other words, eRules encounters the problem of overfitting in dealing datasets with lots of continuous attributes. But rule terms in form of $(x < \alpha \leq y)$ produced from G-eRules for continuous attributes tend to cover more instances matching the target class and thus need to produce less rule terms. Although, G-eRules may not dramatically improve the levels of accuracy from original eRules, the processing time is clearly improved.

5 Conclusion

In this paper we presented a new computationally efficient method of extracting rule terms in the form of $(x < \alpha \leq y)$ from continuous attributes in a data stream for classification algorithms. The new method is based on the density class probability from Gaussian distribution. This generic way of extracting continuous rule terms can potentially be used in any rule-based data stream classifier in order to increase the data throughput of the algorithm. We have implemented this approach in the eRules algorithm, a simple yet competitive rule-based data stream classifier, that allows abstaining from a classification, but is computationally inefficient when dealing with continuous attributes. We termed this new version of the classifier G-eRules. Our

evaluation shows that G-eRules with the new method for generating continuous rule terms achieves comparable levels of accuracy compared with original eRules, but is much faster when dealing with continuous attributes, which is desired in mining high velocity data streams.

We are currently developing a new rule-based classifier motivated by the scalability of our density class probability based approach for inducing continuous rule terms. Currently eRules and G-eRules are using a fixed size sliding window, however, this does not take the length of a concept drift into account. We are currently exploring metrics that could be used to dynamically re-size the sliding window to the optimal number of data instances for generating rules. For this the Hoeffding bound is considered similarly to the Hoeffding Tree algorithm [8]. However, we are also planning to change the nature of the rules from conjunctive combinations of rule terms to also allow disjunctive combinations. Thus the classifier could naturally capture concepts for classification that encode not only 'AND' but also 'OR' relationships between different attributes.

References

1. Gaber, M.M., Zaslavsky, A., Krishnaswamy, S.: Mining data streams: a review. SIGMOD Rec. **34**(2), 18–26 (2005)
2. de Aquino, A.L.L., Figueiredo, C.M.S., Nakamura, E.F., Buriol, L.S., Loureiro, A.A.F., Fernandes, A.O., Coelho, Jr. C.J.N.: Data stream based algorithms for wireless sensor network applications. In: AINA, pp. 869–876. IEEE Computer Society, New York (2007)
3. Thuraisingham, B.M.: Data mining for security applications: Mining concept-drifting data streams to detect peer to peer botnet traffic. In: ISI, IEEE (2008)
4. Stahl, F., Gaber, M.M., Salvador, M.M.: eRules: amodular adaptive classification rule learning algorithm for data streams. In: Bramer, M., Petridis, M., (eds.) SGAI Conference, pp. 65–78. Springer, Berlin (2012)
5. Babcock, B., Babu, S., Datar, M., Motwani, R., Widom, J.: Models and issues in data stream systems. In: Proceedings of the Twenty-first ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems, PODS'02, pp. 1–16. ACM, New York (2002)
6. Cendrowska, J.: PRISM: an algorithm for inducing modular rules. Int. J. Man-Mach. Stud. **27**(4), 349–370 (1987)
7. Gaber, M.M., Zaslavsky, A., Krishnaswamy, S.: A survey of classification methods in data streams. In: Data Streams, pp. 39–59. Springer, Berlin (2007)
8. Domingos, P., Hulten, G., Mining high-speed data streams. In: Ramakrishnan, R., Stolfo, S.J., Bayardo, R.J., Parsa, I. (eds.) KDD, pp. 71–80. ACM (2000)
9. Witten, I.H., Eibe, F.: Data Mining: Practical Machine Learning Tools and Techniques with Java Implementations, 2nd edn. In: Kaufmann, M. (2005)
10. Wang, H., Fan, W., Yu, P.S., Han, J.: Mining concept-drifting data streams using ensemble classifiers. In: Getoor, L., Senator, T.E., Domingos, P., Faloutsos, C. (eds.) KDD, pp. 226–235. ACM (2003)
11. Gama, J., Kosina, P.: Learning decision rules from data streams. In: Walsh, T. (ed.) IJCAI 23, pp. 1255–1260. IJCAI/AAAI (2011)
12. Han, J., Kamber, M.: Data Mining. Concepts and Techniques, 2nd edn. In: Kaufmann, M. (2006)
13. Bruce, N., Pope, D., Stanistreet, D.: Quantitative methods for health research: a practical interactive guide to epidemiology and statistics. Wiley, Chichester (2008), 2013

14. Bifet, A., Holmes, G., Kirkby, R., Pfahringer, B.: MOA: massive online analysis. J. Mach. Learning Res. **11**, 1601–1604 (2010)
15. Hoeglinger, S., Pears, R.: Use of hoeffding trees in concept based data stream mining. In: Third International Conference on Information and automation for sustainability, ICIAFS 2007, pp. 57–62. IEEE, New York (2007)
16. Street, W.N., Kim, Y.: A streaming ensemble algorithm (SEA) for large-scale classification. In: Lee, D., Schkolnick, M., Provost, F.J., Srikant, R. (eds.) KDD, pp. 377–382. ACM (2001)
17. Bifet, A., Holmes, G., Pfahringer, B., Kirkby, R., Gavaldà, R.: New ensemble methods for evolving data streams. In: Elder IV, J.F., Fogelman-Soulié, F., Flach, P.A., Zaki, M., (eds.) KDD, pp. 139–148. ACM, New York (2009)
18. Abdulsalam, H., Skillicorn, D.B., Martin, P.: Streaming random forests. In: IDEAS, pp. 225–232. IEEE Computer Society, Washington, DC (2007)
19. Datasets from moa (massive online analysis) website (online). Accessed Apr 2014
20. Ikonomovska, E., Gama, J., Dzeroski, S.: Learning model trees from evolving data streams. Data Min. Knowl. Discov. **23**(1), 128168 (2011)

Springer

http://www.springer.com/978-3-319-12068-3

Research and Development in Intelligent Systems XXXI
Incorporating Applications and Innovations in Intelligent Systems XXII
Bramer, M.; Petridis, M. (Eds.)
2014, XIV, 344 p. 114 illus., Softcover
ISBN: 978-3-319-12068-3