A two layers incremental discretization based on order statistics

Christophe Salperwyck and Vincent Lemaire

Abstract Large amounts of data are produced today: network logs, web data, social network data... The data amount and their arrival speed make them impossible to be stored. Such data are called streaming data. The stream specificities are: (i) data are just visible once and (ii) are ordered by arrival time. As these data can not be kept in memory and read afterwards, usual data mining techniques can not apply. Therefore to build a classifier in that context requires to do it incrementally and/or to keep a subset of the information seen and then build the classifier. This paper focuses on the second option and proposed a two layers approach based on order statistics. The first layer uses the Greenwald et al. quantiles summary and the second layer a supervised method such as MODL.

Key words: Incremental learning, discretization, order statistics

1 Introduction

Many companies produce today large amounts of data. Sometimes data can be kept into a database, sometimes their arrival speed makes them impossible to be stored. In that specific case mining data is called stream mining. The stream specificities are that: (i) data are just visible once and (ii) are ordered by arrival time. Such an amount of data leads to the impossibility to keep them in memory and to read them afterwards. Therefore to build a classifier in that context requires to do it incrementally and/or to keep a subset of the information seen and then build the classifier. In this paper the focus is on the second option. This can be achieved by: keeping

Christophe Salperwyck
Orange Labs, Lannion, France - LIFL, Université de Lille 3, Villeneuve d'Ascq, France e-mail: christophe.salperwyck@orange-ftgroup.com

Vincent Lemaire
Orange Labs, Lannion, France e-mail: vincent.lemaire@orange-ftgroup.com

a subset of the stream examples, calculating a density estimation or having order statistics.

This work focuses on numeric attributes discretization based on order statistics. The discretization in this paper is used for supervised classification.

2 Related works

Incremental discretization has two main applications: data mining for large data set and Data Base Management Systems (DBMS) for building query plans.

2.1 Data mining field

Gaussian density approximation: The data distribution is supposed to follow a Gaussian law that will be approximated. Only two parameters are needed to store a Gaussian law: mean and standard deviation. The incremental version required one more parameter: the number of elements. An improved version for supervised classification on stream can be found in [6].

PiD: Gama and Pinto in [4], proposed a two layers incremental discretization method. The first layer is a mix of "EqualWidth" and "EqualFrequency" (algorithm details: [4] p. 663) and is updated incrementally and has much more bins than the second layer. The second layer uses information of the first one to build a second discretization. Many methods can be used on the second layer such as: EqualWidth, EqualFrequency, Entropy, Kmeans...

2.2 DBMS field

On line histogram: Ben-Haim et al [1], presented an incremental and on line discretization for decision trees. Their algorithm is based on three methods: (i) update - add a new example in an existing histogram, (ii) merge - merge two bins in one, (iii) uniform: use a trapezoid method to build the EqualFrequency bins.

Online quantiles: Greenwald and Khanna [5] proposed a method to estimate quantiles with a maximum error bound. Their method is based on keeping a list of tuples which have three values: v - a value in the stream, g - the number of values before v, Δ - the maximum error on the quantile position. This method has a theoretical bound on the error ε and on the required space: $O(\frac{1}{\varepsilon} log \varepsilon N)$ where N is the size of stream.

3 Our proposal

Objective: Our proposal aims to used at best the data mining field and the DBMS field to propose an incremental discretization method which intrinsically realizes a

compromise between the error ε and the memory M used. This method will also have to be robust and accurate for classification problems.

Proposal: The idea is to use a two layer incremental discretization method as PiD [4] bound by a memory M parameter. The first layer is a summary of the data and the second layer the final discretization. The memory will be used at best to have the lowest error. For the first layer, the Greenwald and Khanna quantiles summary (called GK from now) suits this requirement the best and provides order statistics. For the second layer, among methods using order statistics two are particularly interesting considering their performances: MDLP (Recursive Entropy Discretization) [3] and MODL [2]. One of these two methods is used to build the final discretization. They are supervised and known to be robust. The choice to use GK for the first layer and MDLP/MODL in the second layer is coherent since the complete structure is based on order statistics.

4 First experiment

The MODL discretization was adapted to use GK quantiles summary and then a naive Bayes classifier is built. The Delta training dataset from the large scale learning challenge[1] is used for a first experiment. It shows that with a given amount of memory our method performances are similar to the one loading all the data into memory. The AUC performances of the GK quantiles summary were studied with 10 and 100 quantiles: GK10 and GK100. The reservoir sampling approach [7] is also used to compare performances with a bounded memory technique. The two sizes used are equivalent to the memory consumed by GK10 and GK100. Figure 1 shows the comparison of theses approaches.

With limited memory GK methods are performing much better than reservoir sampling. Compared to Khiops using all data, the GK performance with 100 quantiles is almost the same.

5 Conclusion and future works

The first experiment validates that with large data sets and bounded memory, our two layers discretization has a strong interest. Our approach is controlling its error level and can be set up to use a fixed memory size. The following works are in progress and will be presented in the conference.

Firstly, experiments on the robustness of the discretization will be conducted. A synthetic data set can be used such as the crenel pattern used in [2]. Such a study quantifies how many bins should have our modified GK discretization in order to well identify the crenels of the pattern we were learning.

Secondly, a dialog between the two layers to supervise the first layer discretization will be proposed. Some very early experiments with the crenel pattern showed

[1] http://largescale.ml.tu-berlin.de

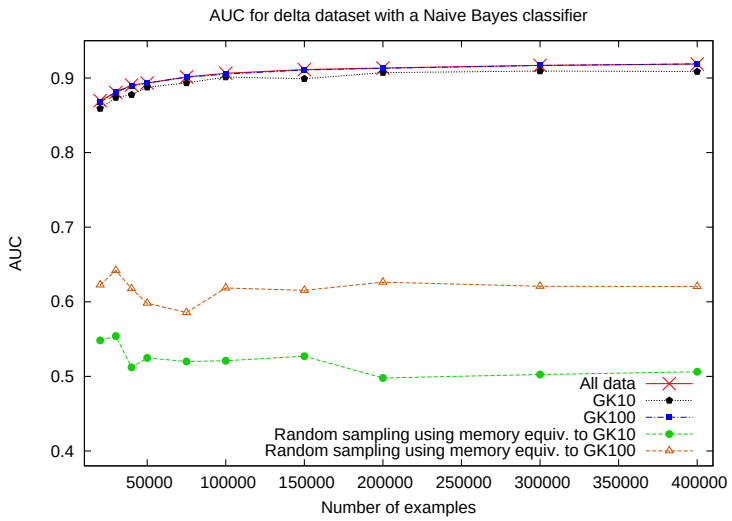

Fig. 1 Naive Bayes AUC performances with all data, GK and reservoir sampling.

that it can be beneficial to have a dialog between the two layers. As the second layer is supervised it can provide to the first layer the regions which have more interest. The first layer can then adapt and focus on these regions by allocating them more bins - ie: a more precised discretization for them.

References

1. Ben-Haim, Y., Tom-Tov, E.: A streaming parallel decision tree algorithm. Journal of Machine Learning **11**, 849–872 (2010)
2. Boullé, M.: MODL: A Bayes optimal discretization method for continuous attributes. Machine Learning **65**(1), 131–165 (2006)
3. Fayyad, U., Irani, K.: Multi-interval discretization of continuous-valued attributes for classification learning. Proceedings of the International Joint Conference on Uncertainty in AI pp. 1022–1027 (1993)
4. Gama, J., Pinto, C.: Discretization from data streams: applications to histograms and data mining. In: Proceedings of the 2006 ACM symposium on Applied computing, pp. 662–667 (2006)
5. Greenwald, M., Khanna, S.: Space-efficient online computation of quantile summaries. ACM SIGMOD Record **30**(2), 58–66 (2001)
6. Pfahringer, B., Holmes, G., Kirkby, R.: Handling numeric attributes in hoeffding trees. Advances in Knowledge Discovery and Data Mining pp. 296–307 (2008)
7. Vitter, J.S.: Random sampling with a reservoir. ACM Transactions on Mathematical Software **11**(1), 37–57 (1985)