ON CHANGING CONTINUOUS ATTRIBUTES INTO ORDERED DISCRETE ATTRIBUTES

J. CATLETT
Basser Department of Computer Science, University of Sydney, N.S.W. 2006, Australia
Email: jason@cs.su.oz.au

Abstract

The large real-world datasets now commonly tackled by machine learning algorithms are often described in terms of attributes whose values are real numbers on some continuous interval, rather than being taken from a small number of discrete values. Many algorithms are able to handle continuous attributes, but learning requires far more CPU time than for a corresponding task with discrete attributes. This paper describes how continuous attributes can be converted economically into ordered discrete attributes before being given to the learning system. Experimental results from a wide variety of domains suggest this change of representation does not often result in a significant loss of accuracy (in fact it sometimes significantly improves accuracy), but offers large reductions in learning time, typically more than a factor of 10 in domains with a large number of continuous attributes.

Keywords Discretisation, empirical concept learning, induction of decision trees

1. Introduction

As Subramanian (1989) concisely stated, "Present day learners depend on careful vocabulary engineering for their success." Many authors since Amarel (1968) have demonstrated the critical importance of the framework used to represent a given learning task, and the benefits to be gained from an appropriate change of representation. Recent work such as Rendell (1989) has advanced the state of the art of constructive induction, where the attributes given to a selective induction system such as ID3 (Quinlan 1979, 1983) are transformed into better attributes. Usually "better" is taken to mean "higher level", allowing the target concept to be expressed more concisely, making the product of the learning more accurate. The transformations often involve investigation of many complex combinations of the raw attributes, and can be computationally expensive, but

this work may be rewarded by error rates lower than could be obtained from induction directly on the raw attributes.

This paper too addresses the question of changing the representation of tasks given to an inductive learning system, but with the aim of lowering the computational cost of the learning rather than improving the accuracy of the product. The ideal representation would be one that allows the learner both to express the final concept most accurately (or concisely, or comprehensibly), and to compute that concept in the least time possible, but these two goals are in conflict. Thus the client for whom the learning is performed faces a trade-off between speed and accuracy. Changing the representation complicates this decision, because any change entails a computational cost.

The situations in which the client would want to reduce the cost of learning are becoming increasingly common as machine learning becomes a commercial technology. The main spur comes from problems of scale: training set of hundreds of thousands of instances are now being attacked (e.g. Sejnowski 1987), making learning time a limiting factor. As commercial knowledge acquisition tools that interface learning systems to corporate databases become popular, the management of the extraction and transformation of learning data will become a difficult logistic task, much of it needing to be transformed from numeric to symbolic form. Finally, in the past three years a class of applications that use incremental learning has come under the spotlight (for example, Utgoff 1988). Where this is done in real time with new examples becoming available during the induction, it may be advantageous to finish the induction earlier.

Various researchers have attacked the question of how to speed up the learning task. Selective induction algorithms such as ID3 that partition the instance space are generally very efficient and difficult to improve. Quinlan's first descriptions of ID3 (1979) included a method called windowing to speed up ID3 on large training sets, but Wirth & Catlett (1988) showed that on noisy data it usually costs rather than saves CPU time. Breiman, Friedman, Olshen & Stone (1984, pp 163-7) proposed a method of choosing the attribute to partition the space based on a random subset of the available instances, but this still retains much of the cost of dealing with continuous attributes.

The algorithm we describe for discretising a dataset could be used as before a variety of learning algorithms; here we use ID3, to provide a familiar basis for comparison. However, the rationale for some of the design decisions in the algorithm is based on the assumption that the learning system partitions the training set, and may not be appropriate for learners that build around a seed example, such as AQ15 (Michalski,