Rule Induction with CN2:  
Some Recent Improvements

Peter Clark      Robin Boswell
The Turing Institute, 36 N.Hanover St., Glasgow
email: {pete,robin}@turing.ac.uk

Abstract

The CN2 algorithm induces an ordered list of classification rules from examples using entropy as its search heuristic. In this short paper, we describe two improvements to this algorithm. Firstly, we present the use of the Laplacian error estimate as an alternative evaluation function and secondly, we show how unordered as well as ordered rules can be generated. We experimentally demonstrate significantly improved performances resulting from these changes, thus enhancing the usefulness of CN2 as an inductive tool. Comparisons with Quinlan's C4.5 are also made.

Keywords: learning, rule induction, CN2, Laplace, noise

1 Introduction

Rule induction from examples has established itself as a basic component of many machine learning systems, and has been the first ML technology to deliver commercially successful applications (eg. the systems GASOIL [Slocombe et al., 1986], BMT [Hayes-Michie, 1990], and in process control [Leech, 1986]). The continuing development of inductive techniques is thus valuable to pursue.

CN2 is an algorithm designed to induce ‘if...then...’ rules in domains where there might be noise. The algorithm is described in [Clark and Niblett, 1989] and [Clark and Niblett, 1987], and is summarised in this paper. The original algorithm used entropy as its search heuristic, and was only able to generate an ordered list of rules. In this paper, we demonstrate how using the Laplacian error estimate as a heuristic significantly improves the algorithm’s performance, and describe how the algorithm can also be used to generate unordered rules. These improvements are important as they enhance the accuracy and scope of applicability of the algorithm.
2 An Improved Evaluation Function

2.1 The Original Entropy Function

The CN2 algorithm consists of two main procedures: a search algorithm performing a beam search for a good rule (shown in Appendix 2) and a control algorithm for repeatedly executing the search (shown later in Figure 1).

During the search procedure, CN2 must evaluate the rules it finds to decide which is best. One possible metric of rule quality is its accuracy on training data (eg. an option for AQ15 [Michalski et al., 1986]). An alternative is entropy, used by ID3 and the original CN2, which behaves very similarly to apparent accuracy. Entropy also prefers rules which cover examples of only one class.

The problem with these metrics is that they tend to select very specific rules covering only a few examples, as the likelihood of finding rules with high accuracy on the training data increases as the rules become more specific. In the extreme case, a maximally specific rule will just cover one example and hence have an unbeatable score using the metrics of accuracy (scores 100% accuracy) or entropy (scores 0.00, a perfect score). This is undesirable as rules covering few examples are unreliable, especially with noise in the domain. Their accuracy on the training data does not adequately reflect their true predictive accuracy (ie. accuracy on new test data) which may appear.

2.2 Significance Testing: A Partial Solution

To avoid selecting highly specific rules, CN2 uses a significance test (see [Clark and Niblett, 1989]) which ensures that the distribution of examples among classes covered by the rule is significantly different from that which would occur by chance. In this way, many rules covering only a few examples are eliminated, as the significance test deems their apparent high accuracy likely to be simply due to chance.

However, while a significance test eliminates rules which are below a certain threshold of significance, there is still the problem that rules which just pass the significance test will tend to be preferred over more general and reliable but less apparently accurate rules. Consider a domain with two equally likely classes C1 and C2, and consider three rules R1, R2 and R3, where:

- R1 covers 1000 examples of class C1 and 1 of C2 (we denote this by [1000, 1])
- R2 covers 5 examples of C1 and 0 of C2 (ie. [5, 0])
- R3 covers [1, 0])

Here, the algorithm should ideally prefer R1 as its accuracy on new test data is likely to be the best - rules R2 and R3 only cover a few examples and their apparent accuracies of 100% are not fully reflective of performance on new test data. However, although a 99% significance test eliminates R3, R2 will just pass and be selected in preference to R1. Raising the significance level further does not solve the problem as a rule R1.5 (say) may exist which again just passes the raised significance threshold.