Available online at www.sciencedirect.com
ScienceDirect
Expert Systems with Applications 34 (2008) 2783–2788
Expert Systems with Applications
www.elsevier.com/locate/eswa

Rule effectiveness in rule-based systems: A credit scoring case study

Arie Ben-David *

Management Information Systems, Department of Technology Management, Holon Institute of Technology, 52 Golomb Street, P.O. Box 305, Holon 58102, Israel

Abstract

Rule-based systems may sometimes grow very large, making their acceptance by users and their maintenance quite problematic. One therefore needs to make rule-bases as compact as possible. The classical definition of rule redundancy in the literature is based upon logic and graph theory. Another, complementary, view of redundancy is proposed here. The suggested approach is based on the contribution of individual rules to the overall system's accuracy.

It is shown here, though an analysis of a real-world credit scoring rule-based system, that by taking into account system's accuracy, one can sometimes significantly reduce the size of a rule-base; even one which is already free from logic-related abnormalities. The approach taken here is not proposed as a substitution to classical logic and graph-based methods. Rather, it complements them.

Keywords: Rule-based systems; Rule reduction; Verification; Validation; Classification accuracy

1. Introduction

Rule-based expert systems have been in use for a couple of decades, and their usefulness has been demonstrated in many domains. However, they also drew a lot of criticism. In particular, deriving the "right" rules proved a very difficult task. Another issue has been the need to adjust rule-based systems to cope with changing conditions. Some rule-bases, such as the legendary Mycin, R1, and Prospector, grew to significant sizes, making system implementation and maintenance very difficult.

As rule-based systems grew larger, it became increasingly impractical to manually check each rule for consistency, redundancy, etc. This topic, thus, drew the attention of the research community, and algorithmic tools began to emerge. Gradually, verification and validation of rule-bases have evolved into a sub-field of software engineering and computer science.

While different approaches towards rule-base validation and verifications currently exist, there is a consensus among researchers that all other things being equal, smaller rule-bases are preferred to larger ones (Domingos, 1999; Hayes-Roth, 1983; Mitchell, 1997). There are many reasons why one should generally prefer a small rule-base: Inconsistencies among rules, for instance, are easier to detect and resolve when the number of rules is relatively small. The rationale behind a compact set of rules is easier to explain to employees, simplifying its integration into a company's manuals and culture. Clearly, compact rule-bases also consume less space and generally give quicker response time.

But are the current approaches towards validation and verification of rule-bases sufficient for making them compact enough to be comprehensible? Since real-world expert systems are usually kept secret for fear of competition, it is practically impossible to study many of them in a single context. This research, thus, takes a case study of one real-world credit scoring rule-based expert system, which has undergone all relevant logical tests (e.g., redundancy checks, contradiction elimination, etc.), and yet, was found by its users to contain too many rules to be comprehensible and easy to maintain. It is shown here how a logically correct rule-base can be reduced to a fraction of its original size by taking into account the effectiveness or contribution

* Tel.: +972 3 5026744/6; fax: +972 3 5716481.
E-mail address: hol_abendav@bezeqint.net

0957-4174/$ - see front matter
doi:10.1016/j.eswa.2007.05.003

of each rule to the overall system's accuracy. As the name of this publication suggests, since this research was based on a single case study, it does not pretend to provide a universal answer to the above (possibly) philosophical, difficult question. This research does provide, however, an additional approach towards rule-base simplification. As any other approach, it is not guaranteed to be always effective. It should, therefore, be regarded, as an additional tool in one's toolkit while trying to simplify his/her rule-base.

2. Background and related work

The idea of deleting rarely used rules from a rule-base is not new. It is almost as old as the concept of rule-bases (Hayes-Roth, 1983). This approach is very useful when one has a relatively long history log handy. This is, however, often not the case during an early stage of a system's life cycle. Long history about the number of times each rule was actually used or fired, is simply not available yet. Another limitation of this approach stems from the fact that it does not consider the accuracy of the system as a criterion for rule deletion. A rarely used rule, if eliminated from a rule-base, may sometimes cause critical, errors.

Another approach towards identifying unnecessary rules in rule-bases comes from the area of verification and validation. The origin of this approach also dates to the early days of rule-bases. A comprehensive early review of this topic can be found in Nazareth (1989). In this publication, Nazareth wrote: "As expert system technology spreads, the need for verification of system knowledge assumes greater importance". While referring to the question of verification, Nazareth adopted predicate calculus notation and identified two major sources of abnormalities in rule-bases: redundancy and incompleteness. The former includes pairwise redundancy (identical rules, for example), redundancy in chains of rules, and conflicts (same antecedent but different consequences, circularity, etc.).

One of the early tools for automated verification of rule-bases was embedded in EMYCIN (van Melle, 1981). It included algorithms for detecting redundancy, subsumption and conflict through pairwise rule comparison. This was not an easy task considering the $\binom{n}{2}$ pairs that had to be tested. Chain errors were not even tested in EMYCIN, mainly due to time complexity considerations. Later environments, such as ONCOCIN (Suwa, Scott, & Shortliffe, 1982) also considered a limited version of completeness checks by partitioning the rule space and an exhaustive search. The approach was feasible only for relatively small prototypes. The LES environment (Nguyen, Perkins, Laffey, & Pecora, 1985) also included checks for circularity, dead ends, and unreachable clauses. Larichev and his colleagues (Larichev, Moshkovich, & Furems, 1986) have built a decision support system called CLASS, which helped in generating consistent and irredundent rule-bases. CLASS was assisting knowledge engineers, but was not an automatic tool, as it entirely relied upon its user's choices. McGuire (McGuire, 1990) described an approach for uncovering redundancies and inconsistencies using deduction. Chander, Shinghal, and Radhakrishnan (1997) proposed using goals in order to verify rule-bases. Yang, Lee, Chu, and Yang (1998) suggested methods for rule-base verification using Petri nets. Orsolya and Varkonyi-Koczy (2002) proposed methods for rule-reduction in fuzzy rule-bases.

Verification and validation tools, such as those described above, may help in making rule-bases more compact indeed; in particular, when redundancies, contradictions or subsumptions are detected. However, minimizing the number of rules is not the ultimate goal of these techniques. Consequently, rule-base size reduction is not guaranteed when these tools are used. It might so happen that after all logical bugs are fixed – the resulting rule-base grows even larger.

Using error criteria for choosing candidates for rule elimination has been used in machine learning for quite a long period. The main motivation for controlling the size of the concept (i.e., model) formed by any machine learning algorithm was avoiding the well-known phenomenon of over-fitting of the resulting learned concepts to noisy data. A notable example of this approach can be found in Quinlan's ID3 and C4.5 algorithms (Quinlan, 1986, 1993). The basic idea is to add branches to a decision tree only when they are "informative" enough in a forward mode, or delete "uninformative" branches in a backward pruning. Another example of concept-building using some overall error criteria can be found in Pao (1989) with respect to Artificial Neural Networks. The basic idea is similar: adding a node to a Neural Network is done only when the error over some testing sample is reduced. Typically, neither the comprehensibility of the resulting model nor some validation issues are taken into account while using the latter approach. For instance, a branch in C4.5's decision tree can contradict another when decisions are to be monotonous with each other (Ben-David, 1995).

While machine learning techniques were found very useful and important, there have been no reports about their usefulness in the context of rule-based systems that were derived from human experts using "conventional" knowledge acquisition techniques. Perhaps this report is a first attempt in this direction. In the case study, to be shortly described, there has been no point in deleting rules from an existing rule-base via rule firing statistics, simply since no long enough history was available (see the coming section for details). On the other hand, all relevant logical tests on this rule-base have already been carried out. Yet, it seemed to domain experts that there were too many rules in the rule-base to be comprehensible. The full implementation phase of this project was postponed till a more compact rule-base will somehow be formed. Could the most "informative" rules in the current rule-base be identified and ranked according to their estimated contribution to the overall system's accuracy? As the following case study shows, by adopting some concepts from machine learning, this goal could be achieved algorithmically.