Prolog to

Computational Intelligence Methods For Rule-Based Data Understanding

An introduction to the paper by Duch, Setiono, and Zurada

The central challenge facing computational intelligence approaches is the problem of extracting knowledge from data. How does one combine extracted knowledge with available symbolic knowledge and refine the resulting knowledge-based expert systems? Black-box statistical approaches may be good at deriving predictions from data, but formulating understandable rules from the analysis of data is something entirely different from formulating predictive models from that data. To build predictive data models, many data analysis methods can be employed: naive Bayesian methods in statistics, linear discrimination methods, support vector machines, and multilayered perceptron (MLP) neural networks. And yet the discovery of class structures, association patterns, and causal relationship sequences was never an explicit goal of such methods. The use of predictive nonparametric classification and approximation methods can lead to several dangers: the models may overfit the data; irrelevant attributes may figure into the solution; and combining the models with *a priori* knowledge may be difficult, all of which results in the unsuitability of using black-box models in expert systems, which would pose unacceptable risks for medical, industrial, and financial applications. Reasoning with logical rules is a preferable alternative to black-box systems because it is more comprehensible to humans and capable of being validated.

The initial goal of machine learning, a subfield of artificial intelligence, was to formulate symbolic inductive methods and to discover logical rules that could be expressed in natural language. Machine learning has since broadened its scope to include all methods that learn from data. Besides symbolic description, other methods for understanding data include intuitive understanding (memorized pattern recognition) and visualization. The best way to explain data varies depending on the type of problem, the intention of the user, and the discourse of a given field. In order to understand the class structure of objects, we can focus on classification rules in their simplest, propositional form. Those rules are derived from data sets that contain structure information (assuming that a set of symbolic or continuous-valued predicate functions has been established for some objects).

Various types of logical rules can be discussed in the context of the decision borders these rules create in multidimensional feature spaces. Standard crisp propositional IF. . .THEN rules provide overlapping hyperrectangular covering areas, threshold logic rules are equivalent to separating hyperplanes, while fuzzy rules based on real-valued predicate functions (called membership functions) provide more complex decision borders. The admission of predicate functions that perform tests on more than one attribute leads to the shapes of decision borders that depend on the distance functions used. Rough set theory can also be used to derive crisp logical propositional rules. Logical rules treated as a classification model provide approximation to posterior probabilities.

The design of a rule-based system is always a tradeoff between the flexibility of decision borders and the comprehensibility of the rules. One should always first try the simplest models based on crisp logic rules, then move toward more complex forms if they fail.

Propositional logic has limited expressive power and should only be applied to domains where attribute-value language is sufficient to express knowledge, whereas more complex objects should be treated with first-order or higher order logic. Because this poses computational difficulties, various restrictions have been proposed to increase computational effectiveness.

In order to verbalize knowledge, logical rules require symbolic inputs called linguistic variables, which implies that input data must be quantized (data features are identified and subranges labeled). Two types of linguistic variables are in use: universal, context-independent variables, and context-dependent variables. The simplest way to select initial linguistic variables is to analyze histograms obtained by displaying data for all classes for each feature. Global and local discretization methods are useful for creating linguistic variables. Local discretization methods may also use neuro-fuzzy algorithms, adjusting adaptive parameters of a network to model probability density functions.

Decision trees are an important tool in rule generation and data mining. They are fast and easy to use despite their somewhat limited power. This paper discusses various methods and algorithms using information gain, methods using the separability of split values, methods using random perturbation, and the hybrid connectionist–symbolic method.

Many inductive learning algorithms or "concept learning" algorithms have been produced to extract logical rules. Many only work for symbolic inputs, so continuous features have to be discretized first. This paper briefly surveys AQ covering algorithms, CN2, RIPPER, Version spaces, and Inductive logic programming.

Digital Object Identifier 10.1109/JPROC.2004.826608

Because software implementations of inductive machine learning algorithms are not readily available, it is difficult to compare them with neural networks or decision trees.

Neural networks also can be used for rule extraction. To better understand them, one must understand what neural networks really do and how to use them to extract logical rules describing the data. Despite the notorious difficulty of interpreting neural networks, a typical network may be simplified and approximated by applying logical rules. Neural-inspired algorithms have important advantages, especially for continuous inputs. Good linguistic variables may be determined simultaneously with logical rules, and selection and aggregation of features into a smaller number of useful features may be incorporated in the neural model. Adaptation mechanisms are built in, and wide-margin classification by neural networks can produce more robust logical rules. Global and local methods are discussed, as well as the process for simplifying rule extraction. The MLP2LN algorithm, for instance, facilitates extraction of logical rules from an MLP network. Search-based procedures can be used as an alternative to gradient-based back-propagation training.

The field of logical rule extraction rarely addresses the need to control the tradeoff between comprehensibility and accuracy, the need to optimize linguistic variables and final rules, and the need to estimate the reliability of rules. An important part of rule optimization involves simplification and symbolic operations on rules. Optimized linguistic variables may be used to extract better rules in an iterative process. Tests of classification accuracy should be performed using stratified cross validation. The process of rule extraction is illustrated using the Iris dataset of 150 vectors evenly distributed in three classes. Each vector has four features. Other illustrative applications on various datasets are provided to show the usefulness of different rule extraction algorithms.

This paper only reviews one approach to data understanding: the extraction of crisp and fuzzy logical rules from data. A good strategy to start with is to extract crisp rules first, then move over to fuzzy rules if results are not satisfactory. If the number of logical rules is too high or the accuracy too low, one should switch to other classification methods. Logical rules can expose problems with the data itself. Although there are many methods to extract logical rules from the data, neural networks have important advantages. With proper regularization, they can create decision borders that are equivalent to logical rules.

After extracting rules, various cost functions for additional optimization of linguistic variables may be used, creating hierarchical sets of logical rules with different reliability, rejection rate, or different specificity and sensitivity. A great advantage of fuzzy logic is the soft evaluation of probabilities of different classes, instead of binary *yes* or *no* crisp logic answers.

The entire process of logical data description and creation of expert systems from extracted rules is still far from being automatic, and perhaps will remain so for a long time.

—Jim Esch