TAXSY : A RULE BASED EXPERT SYSTEM SHELL DEVELOPED WITH SAS®SOFTWARE

Paul DARIUS – K. U. LEUVEN

Abstract

Expert systems use explicitly coded knowledge (often in the form of IF-THEN rules) to solve problems for which a (numerical) algorithmic solution is not appropriate. There are many specialized expert system development environments available. TAXSY is an expert system shell written completely in SAS. It consists of a set of SAS programs which, with the addition of datasets with rules and code, form a flexible system for knowledge-based consultation.

Introduction

Expert systems (or knowledge-based systems) are computer programs that try to emulate human expert behavior in a certain domain through the use of techniques based on experience and heuristic reasoning, rather than on algorithmic methods only. During the last decade, many expert systems have been developed for widely varying applications.

From a technical point of view, a major feature that makes expert systems different from traditional programs is their structure, which consists of two separate parts: the knowledge and the software that exploits the knowledge. The knowledge itself is encoded in a standardized format. One format often used is IF-THEN rules. The software that exploits the knowledge is called the inference engine. Due to the separation, it is possible to use the same inference engine in many different expert systems, as long as the rules have been coded in the same format. An "empty" environment with an inference engine but no rules is called an expert system shell. This paper discusses such a shell, integrated within the SAS (registered) system.

The SAS (registered) environment provides algorithmic tools for a wide variety of tasks. One of these (but by no means the only one) is statistical processing. Expert system techniques could be used to add non-algorithmic knowledge in order to support and automate these tasks. In the case of knowledge-based statistical processing, much work has already been done (Gale 1986, Haux 1989). It has been argued that in this area, two major goals are (1) provide guidance, (2) prevent misuse.

The actual implementation can be done in several ways. First it should be noted that there are a number of alternative approaches, like the use of macro's or hierarchical menu systems. However, in these approaches the knowledge is deeply buried within the programs where it is neither transparent nor easily accessible.

A knowledge-based approach can be implemented outside the statistical package, in which case it provides information and advice on the use of the package (e.g. Hand 1987). Or it can be implemented within a specialized environment and communicate with the package through the operating system (e.g. Wolstenholme 1988, Gale 1986). If the environment of the package is rich and flexible enough, one can try to implement the knowledge based part within this environment (this is the approach described here). And finally, one can build a knowledge-based statistical system from scratch (e.g. Oldford and Peters 1988).

An overview of the system

TAXSY consists of several parts: Fig 1 gives a schematic overview. The heart of the system is the inference engine. It is capable of backward chaining on rules of a certain format. Backward chaining means that one needs to specify an attribute as the goal of the inference process (e.g. the name of the appropriate test). The inference engine will then repeatedly invoke the rules, in order to find a value for the goal attribute. If more information is needed, it will first look for other rules that could provide it. When there are no such rules it will invoke an interface (to the user or to the statistical procedures, depending on the information needed). The answer is then processed and used in guiding the continuation of the search.

In TAXSY, the inference engine is implemented as an AF-application. It uses a number of "housekeeping" SAS datasets to store the current state of the search. To operate, it must be provided with datasets that contain the data and the knowledge for the specific problem at hand. These datasets are not part of TAXSY (TAXSY is a shell). Hence TAXSY could be used in any application domain, provided it is fed with the appropriate rule and interface datasets.
TAXSY needs a rule base in the form of a SAS dataset, with rules of the following (rather restricted) format:

IF (attribute) (operator) (value)
AND (attribute) (operator) (value)

... THEN (attribute) IS (value).

In addition, for each rule a title can be specified (which will appear in the trace window when the rule is considered), and some reference note (for documentation). Each IF, AND, THEN, title or reference part is coded as one observation.

To obtain a value that cannot be inferred from rules, TAXSY should be able to invoke an appropriate interface. The PROMPTS dataset should contain, for each such attribute, the name of the AF-application TAXSY has to start. TAXSY simply gives control to that AF-application, and expects to find (when the application ends), the requested value in a given macro-variable. In this way, TAXSY can get access (through the SUBMIT feature of the SCL language) to all the features of the SAS system. It makes the system open-ended: the PROMPTS can point to simple menus, or to sophisticated applications involving the construction of SAS-programs based on information previously obtained and processing of their results.

A number of other features have been added to TAXSY because they were found to be very useful in developing rule bases for statistical problems. For these problems, the user has to specify the name of the SAS(registered) dataset that contains his data. He also has the option to specify the name of a STRUCTURE dataset. This is a dataset containing metadata (data about the data). Its purpose is to store that information about the data that is relevant for statistical processing, but for which there is no room in the "traditional" SAS dataset representation. It can store (using specified character variables) information about variables and observations (e.g. yield is a response variable), and about relations between variables. When TAXSY looks for information, it first looks in the STRUCTURE dataset (if there is one), and only then goes to PROMPTS. The presence of a STRUCTURE dataset can reduce the number of questions asked to the user considerably, and hence speed up the consultation process.
Finally, it was found that the inference process generally needs in a given stage only a limited number of rules. Hence the possibility to split the rule dataset in a number of modules was implemented through the STRATEGY dataset. The STRATEGY dataset uses metarules to specify the sequence of inferences to be made. It contains observations like:

if (conditions about CONTEXT)
    then use (dataset with rules)
    and (dataset with prompts)
    to obtain value for (goal).

(here CONTEXT contains the values already obtained at that moment). Through the use of the STRATEGY dataset, we can work with smaller datasets, hence speed up the search processes, and have our knowledge split in modules which are easier to manage and to change.

TAXSY was designed with two kinds of users in mind: first there is the person who provides the knowledge in the form of rules and interface code, or who adapts the knowledge provided by someone else. He must have expertise about the application domain, as well as a good knowledge about the SAS system. We will call him/her the application builder. The second kind of user only uses the knowledge provided by someone else. He only needs to know how to start the system, and provide the name of his data and the strategy he wants to use. We will call him/her the end user. Obviously, in some situations the application builder and the end user may be the same person.

TAXSY from the end user point of view

When the end user starts up TAXSY, a screen appears asking whether this is a new session or the continuation of one already begun. If it is a new session, clear versions of the bookkeeping datasets are created and the end user is asked for the names of the datasets with the strategy, the data and (optionally) the metadata. The end user can also specify the level of tracing activity wanted. Tracing results in messages appearing in a window (the higher the tracing level, the more detailed the messages). Above a certain level, every inference is accompanied by a window asking the user to stop or continue. If he chooses "stop", he is back in the normal SAS session. He can then do whatever he wants (check his data, perform additional analyses, ...). To continue, he invokes TAXSY again and specifies it is the continuation of a session. The inference process then continues from the point where it was halted. In this way it is easy to alternate between guided and free sequences of analyses.

Since the state of the inference process is stored in SAS datasets with a timestamp for each observation, it is also possible to spread a TAXSY session over several computer sessions, and to correct errors or explore alternatives by restarting TAXSY from a previous point in time.

If the combination of answers/results obtained are such that the rule base cannot handle them, the user is informed of this and the search halted.

TAXSY from the point of view of the applications developer

The application developer is responsible for the construction of meaningful consistent rulebases with appropriate, user-friendly interfaces. This is not a trivial task, and a number of facilities have been developed to assist him. One is a rule editor, another one a cross-checking facility to check if given PROMPTS and RULES datasets are complete with regard to each other. A rulebase to assist the developer in writing rulebases and constructing SCL programs is under development.

Conclusions

TAXSY provides knowledge-based facilities in the familiar SAS environment. The knowledge is stored in the form of SAS datasets and catalogs, hence allows easy access and facilities for experimentation. The user is not confined to follow the system, but can break out any moment and resume later. Hence the knowledge used in TAXSY serves as an addition to, and not a selection of the capabilities provided in the SAS(r) system.
The current version of TAXSY runs under SAS Version 6.03 under DOS, and is operational on a 80286 based PC with 640K and 1MB extended memory.

References


For further information, please contact:
Dr. P. DARIUS
Kard. Mercierlaan 92, B-3030 Heverlee BELGIUM