N89-20080

EQUATING AN EXPERT SYSTEM TO A CLASSIFIER IN ORDER TO EVALUATE THE EXPERT SYSTEM

Final Report

NASA/ASEE Summer Faculty Fellowship Program - 1988

Johnson Space Center

Prepared By:	Patrick L. Odell
Academic Rank:	Professor
University & Department:	Baylor University Mathematics Waco, Texas 76798

NASA/JSC	
Directorate:	Engineering
Division:	Systems Development & Simulation
Branch:	Intelligent Systems
JSC Colleague:	Richard P. Heydorn, Ph.D.
Date Submitted:	July 22, 1988
Contract Number:	NGT 44-005-803

ABSTRACT

A strategy to evaluate an expert system is formulated. The strategy proposed is based on finding an equivalent classifier to an expert system and evaluate that classifier with respect to an optimal classifier, a Bayes classifier.

This paper shows that for the rules considered that an equivalent classifier exists. Also, a brief consideration of meta and meta-meta rules is included. Also, a taxonomy of expert systems is presented and an assertion made that an equivalent classifier exists for each type of expert system in the taxonomy with associated sets of underlying assumptions.

1. INTRODUCTION

It is the purpose of this paper to formulate a formal mathematical relationship between an expert system and a classifier. If indeed the relationship is unique and constructable, then one will be able to evaluate an expert system by comparing that system to a Bayesian classifier which is by definition the best estimate.

In order to formulate a general result one must develop a definition of an expert system and a taxonomy of special types of expert systems. I have selected my glossary from a document prepared for Electric Power Research Institute entitled, Approaches to the Verification and Validation of Expert Systems [1].

The major purpose of this paper is to develop a theoretical foundation for evaluating expert systems.

2. DESCRIPTION OF EXPERT SYSTEMS

A number of definitions of expert systems are found in Artificial Intelligence literature, but the most informative, yet concise one seems to be the following. First, they perform large, tediously complicated and sometimes difficult tasks at expert levels of performance. Second, they emphasize domain specific problem-solving strategies over the more general 'weak methods' of AI. Third, they employ rules of self knowledge to dynamically consider their own inference process and provide explanations or justifications for conclusions reached.

An expert system usually consists of four main parts, as shown in Figure 1.

1. Knowledge Base(s) of structured domain facts and their relationships and heuristics (problem-solving rules),
2. Inference Engine(s) for controlling the application of facts, relationships, and heuristics in solving the problem(s) at hand, usually under the control of an overall Meta-Controller module,
3. Problem Data Base of information about the problem being solved and the history of the solution process, and
4. User Interface providing results/status displays, explanations, and interaction facilities for user inputs.

Either AI or Conventional software techniques can be used for the storage management of the problem data base and the control of the use interface, but AI programming techniques are always used for representing the knowledge base(s) and developing the inference engine(s).

In the sections that follow, the differences between expert systems and conventional software and the types of expert systems, with respect to the ease of verifying and validating them, will be discussed.

2.1 Differences From Conventional Software

Computer scientist claim that expert systems differ from conventional software programs both in the types of problems they solve and their internal structure. Like a human expert, the expert system has to accommodate information that is incomplete, erroneous, or misleading. Yet, a choice of an action or decision among several alternatives must be made. They also claim that a conventional software program works correctly only when inputs are complete, of the proper syntax,

and the problem is ambiguous. These claims are not at all obvious to others in other discipline who view AI and AI tools as simply computer language and an approach to helping solve important engineering problems. However, expert systems and their rule base does pose some new specific problems in evaluation of software. The problems expert systems are designed to solve generally fall into the following categories: interpretation, prediction, diagnosis, design, planning, monitoring, debugging, repair, instruction and control. Notice, each of these categories contain a function of observing, recognizing and may include a call for action.

Expert systems have many structural similarities with conventional software in the modules which perform conventional tasks such as input problem data processing, data management, and user interface display processing. However, the core of expert systems, including the knowledge bases and inference engines, can be different. Symbolic representation techniques are primarily used to represent knowledge versus conventional software's numeric or table-based techniques for representing information. These symbolic representations include information about relationships between data items and data item aggregations and information about heuristics (procedures and rules) for problem solving, which are not represented in conventional data bases. Expert systems are developed to solve problems which may not have easily formalized or algorithmic solutions, so the problem-solving apparatus (inference engines) must use unconventional methods, including heuristic-guided search, symbolic inferencing, generation and test of solutions, and constraint-based reasoning.

It is these differences from conventional software which make expert systems interesting and applicable, and yet also less easily tested, verified and validated. Because their solution methods are often potentially unbounded and not prescribed in a straightforward "recipe" format like conventional solution methods, they are therefore harder to "prove" correct.

2.2 Expert System Types

There are many ways of dividing expert systems into types, by application, by method of reasoning, etc. Examples of these groupings are found in Table 2-1 and Figure 2-1. In this discussion, expert systems will be divided into types relative to the complexity and difficulty of performing V&V on each type. The types are listed in Table 2-0.

TABLE 2-0.- EXPERT SYSTEM TYPES

NUMBER	NAME
1	Simple, based on Codified Knowledge
2	Simple with Uncertainty Handling
3	Simple, based on Elicited Knowledge
4	Elicited with Uncertainty Handling
5	Complex
6	Complex with Uncertainty Handling

The first type of expert system, Simple, is developed through the straightforward encoding of validated and verified decision tables and/or procedure trees. Its search space is small and examined with exhaustive search techniques or large and factorable and examined with

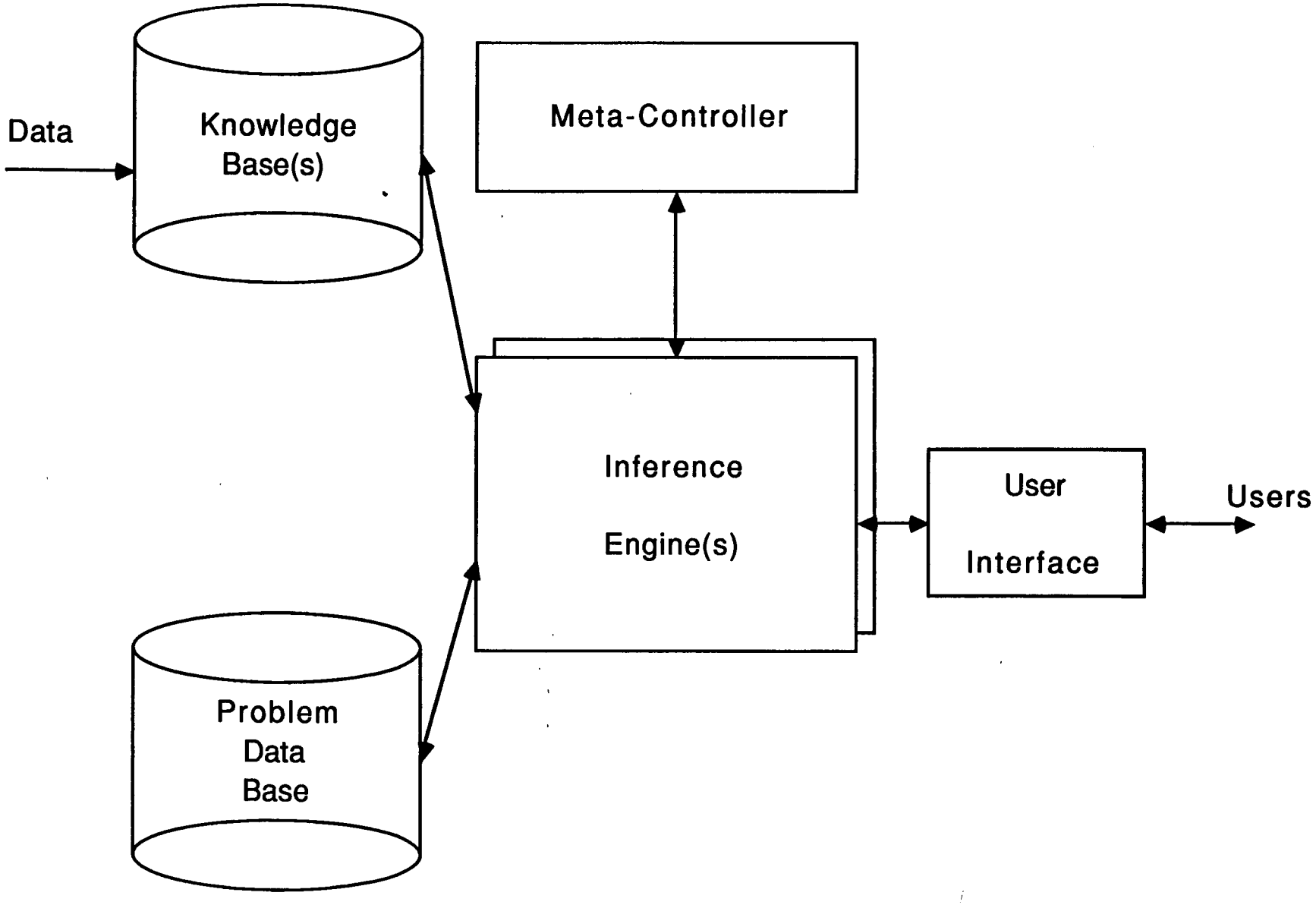

Figure 1-1. Typical Expert System Components

Table 2-1

GENERIC CATEGORIES OF KNOWLEDGE ENGINEERING APPLICATIONS

Category	Problem Addressed
Interpretation	Inferring situation descriptions from sensor data
Prediction	Inferring likely consequences of given situations
Diagnosis	Inferring system malfunctions from observables
Design	Configuring objects under constraints
Planning	Designing actions
Monitoring	Comparing observations to plan vulnerabilities
Debugging	Prescribing remedies for malfunctions
Repair	Executing a plan to administer a prescribed remedy
Instruction	Dianosing, debugging, and repairing student behavior
Control	Interpreting, predicting, repairing, and monitoring system behaviors

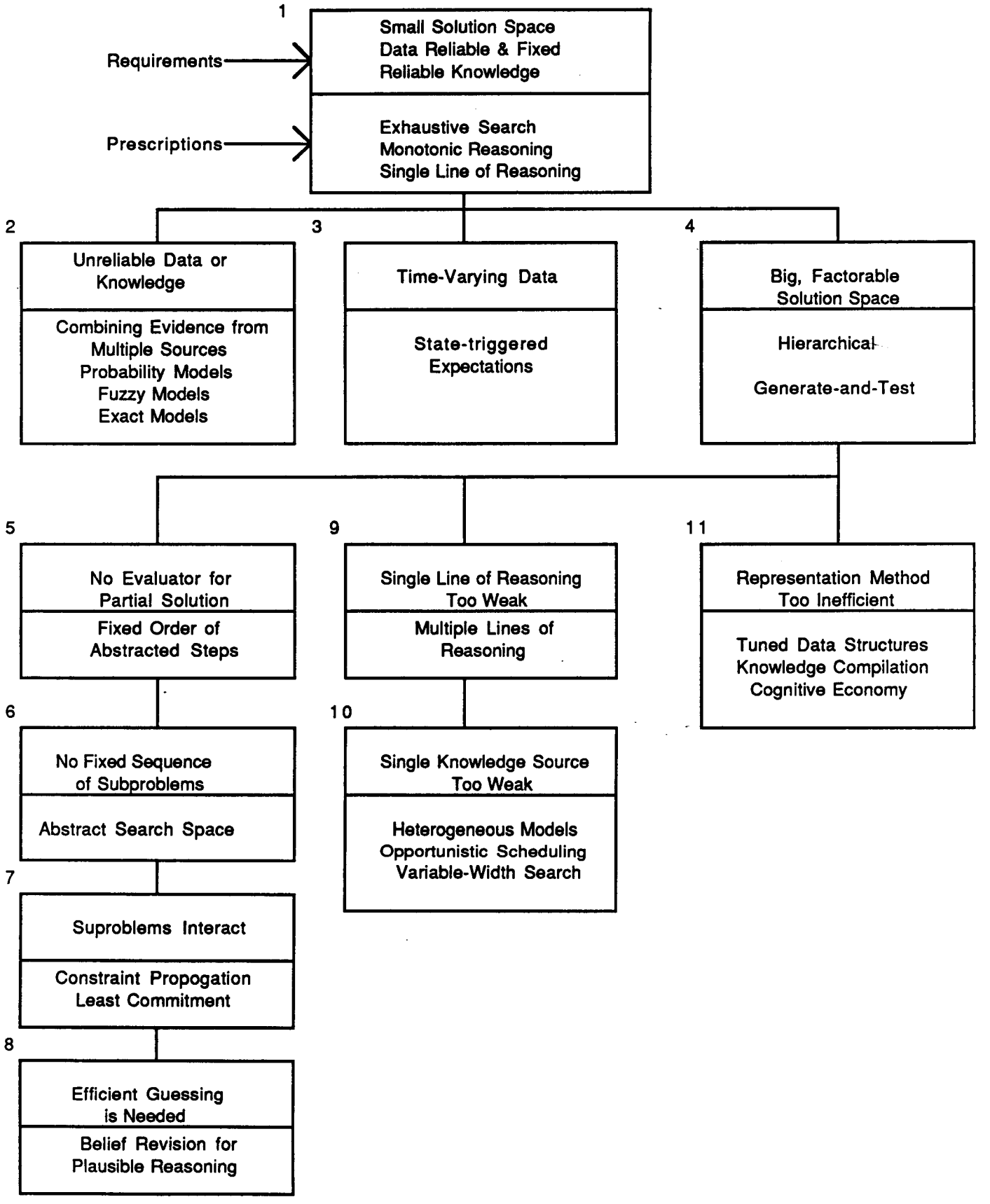

Figure 2-1. Expert System Types Based on Reasoning Methods

enumerative search techniques, for which proofs exist of the best answer. A single line of reasoning is used and reasoning is monotonic, approaching a single, best answer. This type roughly corresponds to types 1,3, and 4 in Figure 2-1.

The most important feature besides simplicity for this type is that the knowledge to be encoded into the expert system has already been written down in some form and tested for correctness. If the codified decision tables or procedure trees must be augmented with non-proved knowledge, then the expert system does not fall into this type category. So, V&V of this type of expert system involves merely proving that it correctly represents and uses the knowledge, not that the knowledge itself is correct.

The second type of expert system, Simple with Uncertainty Handling, has the features of simplicity and proven knowledge in common with the first, but also includes certainty factors, fuzzy logic, probabilities, or some other method of dealing with uncertain information. This type of expert system roughly corresponds to type 2 in Figure 2-1. There are two general types of uncertainty, uncertainty on the existence or value of knowledge items or their relationships and uncertainty on the rules, reflecting the expert's uncertainty of the applicability of the rule on the antecedent conditions or the appropriateness of the conclusions. Expert systems may incorporate either or both forms of uncertainty. There are various methods of combining certainty factors which may be used alone or in combination in an expert system. The decision of which combination function(s) to use is complex and requires analysis and judgment on the part of the knowledge engineer and expert and effects the reliability of the system.

Even if there exist pre-proven uncertainty values on knowledge items or rules, the method of combining those factors most-likely is not pre-proven and tested. And, usually, the values themselves do not pre-exist and must be elicited from experts. An example is the Probability Risk Assessment (PRA) prior event probabilities. Even though these probabilities pre-exist and have been determined to be correct, to the best knowledge of experts, how the probabilities can be combined and what probabilities to place on rules have not been predetermined and tested. Therefore, uncertainty handling complicates the V&V of simple expert systems by adding the need to V&V the certainty factors and their combination method(s).

The third type of expert system, Simple based on Elicited Knowledge, is probably the most common type in existence. The simplicity of the previous two types is still evident, probably because an expert system building tool or environment was used to develop the system. However, the knowledge (items and rules), or some portion of it, does not pre-exist in a tested form and must be elicited from usually one expert. Successive prototypes are developed and tested to verify that the knowledge required to solve the problem is actually being elicited and that it is encoded properly.

The fourth type of expert system, Elicited with Uncertainty Handling, is a combination of the second and third. Its knowledge has been elicited from an expert and it includes certainty factors. An example of this type of expert system is the Mycin system in the field of medical diagnosis developed by Stanford University. V&V of this type of expert system would include testing the correctness of the knowledge, the correctness of the knowledge implementation, the correctness of the uncertainty factors, and the correctness of the

uncertainty combination functions(s).

The fifth type of expert system, Complex, eliminates all or some of the simplicity assumptions of the previous four, as listed in the description of the first type of expert system. The search space(s) can be infinite or large and unfactorable, subproblems within the problem may interact in various ways, constraint-based reasoning may be used to limit search and multiple lines of reasoning may be pursued to independently produce candidate answers. This type of expert system roughly corresponds to types 5-11 in Figure 2-1. Usually the knowledge is elicited from more than one expert, so conflicting heuristics may arise. Also, usually expert system building tools are not used to implement these systems because the tools do not allow the complexities to be built in.

Examples of the fifth type of expert system include autonomous vehicle control and battle management applications, such as those in the Strategic Computing Program funded by the Defense Advanced Research Projects Agency (DARPA, 1986). Expert systems of this type are just now beginning to work in a primitive form, and V&V of these systems is still a research issue. It is very unlikely that this type of expert system will be developed for non-military application any time in the near future, so V&V of this type of expert system will not be discussed.

The sixth type of expert system, Complex with Uncertainty Handling, adds the complicating factor of uncertainty handling to the fifth, making V&V even more impossible and even more of a research issue. This type also will not be discussed.

3. THE MAIN RESULT

Consider a simple expert system represented by the following rules involving two conditions C_1 and C_2 which can occur or fail to occur and four actions A_0, A_1, A_2, and A_3.

R0 If $C_1 \wedge C_2$, then A_0,

R1 If $C_1 \wedge \bar{C}_2$, then A_1,

R2 If $\bar{C}_1 \wedge C_2$, then A_2, and

R3 If $\bar{C}_1 \wedge \bar{C}_2$, then A_3,

which can also be represented in vector notation

R0 If (0, 0), then (0)

R1 If (0, 1), then (1)

R2 If (1, 0), then (2)

R3 If (1, 1), then (3)

Note that the set of rules { R0, R1, R(2), R(3) } can be thought of and modeled by a function R

$$R: (r_1, r_2) \rightarrow \{ 0,1,2,3 \}$$

where r_1 and $r_2 \in \{ 0,1 \}$

Suppose further that

$$p_0 = \Pr\,[C_1 \wedge C_2]\ ,$$

$$p_1 = \Pr\,[C_1 \wedge \bar{C}_2]\ ,$$

$$p_3 = \Pr\,[\bar{C}_1 \wedge C_2]\ ,\ \text{and}$$

$$p_4 = \Pr\,[\bar{C}_1 \wedge \bar{C}_2]\ ,$$

are the a priori probabilities that the system are in the respective states,

$$S_0 = C_1 \wedge C_2\ ,$$

$$S_1 = C_1 \wedge \bar{C}_2\ ,$$

$$S_3 = \bar{C}_1 \wedge C_2\ ,\ \text{and}$$

$$S_4 = \bar{C}_1 \wedge \bar{C}_2\ ,$$

Also let data be taken to make a decision as to which state the system belongs. The data can be vector valued, dependent or independent, known exactly (deterministic) or observable (random). In order to determine if C_1 occurs, we follow the usual logic of statistical classification theory [2]. Here if

$$X = (X_1, X_2, \ldots, X_p)^T \quad \text{and} \quad Y = (Y_1, Y_2, \ldots, Y_q)^T$$

denotes the observations, then the recognition rules are

RR1 If $X \in R_1$, then $C_1 = 0$.

If $X \notin R_1$, then $C_1 = 1$.

RR2 If $Y \in R_2$, then $C_2 = 0$.

If $Y \notin R_2$, then $C_2 = 1$.

We use the notation $\hat{C}_j$ to denote that we estimate the value of C_j and do not know that value exactly. The classification regions R1 and R2 are selected by the expert or optimally as Bayes regions.

Note that the set of rules when described as above is equivalent to a classical statistical classification problem. We know the solution to this problem; that is, the best classifier is a Bayes classifier. A Bayes classifier is one in which the regions R_1 and R_2 are selected judiciously to assure that the expected costs of misclassification are minimized.

Probabilities that an expert system commits errors can now be computed when the probability density functions f(x) and f(y) of X and Y are known. That is,

$$\Pr\,[\hat{C}_1 = 0 \mid C_1 = 1] = \int_{R_1} f_1(x)dx$$

$$\Pr\,[\hat{C}_2 = 0 \mid C_2 = 1] = \int_{R_2} f_1(y)dy$$

and the probability of not making an error is given

$$\Pr\,[\hat{C}_1 = 0 \mid C_1 = 0] = \int_{R_1} f_0(x)dx$$

$$\Pr\,[\hat{C}_2 = 0 \mid C_2 = 0] = \int_{R_2} f_0(y)dx$$

Note that if X and Y are independent then various combined probabilities can be computed easily. If X and Y are not independent then the joint probability density function is required.

One can stack the observations into a new random vector $Z = (x,y)^T$ and establish Z as the data vector. This form allows one to consider the effect of the covariance structure of X and Y.

Suppose one introduce some "beliefs" concerning the rules. Let us model them in the following form.

RB_0 $\quad \Pr[R0 \text{ is true}] = q_0$

RB_1 $\quad \Pr[R1 \text{ is true}] = q_1$

RB_2 $\quad \Pr[R2 \text{ is true}] = q_2$

RB_3 $\quad \Pr[R3 \text{ is true}] = q_3$

This in many applications can affect the action, that is, the actions are modified to

R0 If $C_1 \wedge C_2$, then $\Pr[A_0] = q_0$

and $\Pr[A_0'] = 1-q_0$,

R1 If $C_1 \wedge \bar{C}_2$, then $\Pr[A_1] = q_1$

and $\Pr[A_1'] = 1-q_1$,

R2 If $\bar{C}_1 \wedge C_2$, then $\Pr[A_2] = q_2$

and $\Pr[A_2'] = 1-q_2$, and

R3 If $\bar{C}_1 \wedge \bar{C}_2$, then $P[A_3] = q_3$

and $\Pr[A_3'] = 1-q_3$

These rule can be modelled as stochastic actions which are applied with a specified probability. This formulation introduces the concept of stochastic actions. This in turn generates a requirement for conflict resolution rules for handling uncertainties of this type. To example consider the following two types of conflict resolution rules.

CRR1: If $q > 1-q$, then select action A, if $q \leq 1-q$, select action A'.

CRR2: If the expected cost of A_0 is greater than expected cost of A_0', select A_0'; if not A_0.

CRR3: Generate a uniform random number N on the interval [0,1], if $0 \leq N \leq q$ then select action A_0; if $q < N \leq 1$, then select A_0'.

The first two rules are deterministic rules while CRR3 is a stochastic rule. Clearly, these rules can be extended to more than two actions per rule.

What we have done is given a precise formulation of an expert system which in turn gives a mathematical model for evaluating the expert system as compared with on optimal Bayesian classifier.

Let us now consider the problem of evaluation. Let

$$R_1 = \{X;\ \underline{X} \leq X \leq \bar{X}\}$$

$$R_2 = \{Y;\ \underline{Y} \leq Y \leq \bar{Y}\}$$

then we see that directly

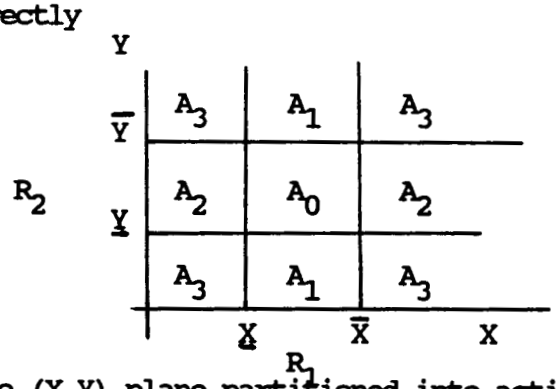

Figure 3.1. The (X,Y) plane partitioned into action regions based on an Expert System if

$X \in R_1$ and $Y \in R_2$, then A_0

$X \in R_1$ and $Y \notin R_2$, then A_1

$X \notin R_1$ and $Y \in R_2$, then A_2

$X \notin R_1$ and $Y \notin R_2$, then A_3

where R_1 and R_2 are the regions defined by the expert.

However, a Bayesian classifier would give different regions R_1 and R_2, the optimal regions. See Figure 3.2.

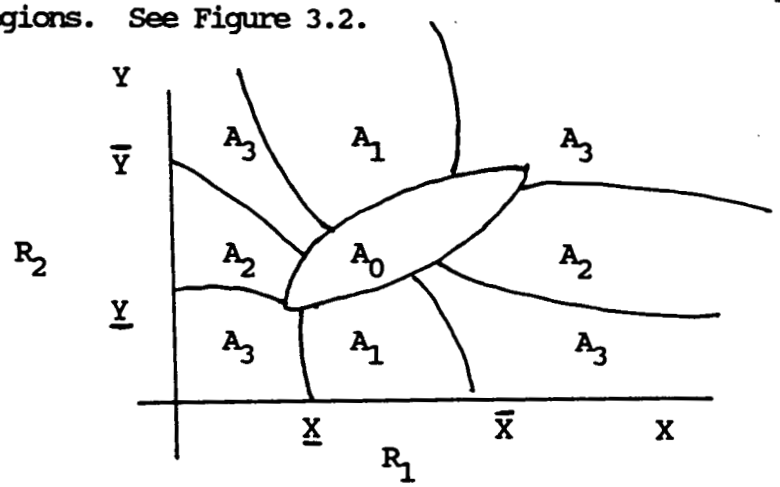

Figure 3.2. The (X,Y) plane partition into action regions based on Bayesian classifier.

Now consider the following meta rule and a meta-meta rule:

MR1: If after K trials

$$\hat{C}_1 \wedge \hat{C}_2 \text{ occurs } K_0 \text{ times}$$
$$\hat{C}_1 \wedge \hat{\bar{C}}_2 \text{ occurs } K_1 \text{ times}$$
$$\hat{\bar{C}}_1 \wedge \hat{C}_2 \text{ occurs } K_2 \text{ times}$$
$$\hat{\bar{C}}_1 \wedge \hat{\bar{C}}_2 \text{ occurs } K_3 \text{ times}$$

then replace (a's are preselected known parameters),

$$P_0 \text{ with } a_0 P_0 + (1-a_0)\ K_0/K$$
$$P_1 \text{ with } a_1 P_1 + (1-a_1)\ K_1/K$$
$$P_2 \text{ with } a_2 P_2 + (1-a_2)\ K_2/K$$
$$P_3 \text{ with } a_3 P_3 + (1-a_3)\ K_3/K$$

and compute new regions R_1 and R_2.

MMR1: Let $K(j) = K + j\ 50$.

The argument appears reasonable that an expert system can be modelled mathematically by finding an equivalent classifier and the expert system can then be evaluated by evaluating that classifier.

4. CONCLUDING REMARKS

The types of expert systems listed in Table 2.2 and discussed in Section 2 of this paper may now be defined precisely by defining an associated classifier when different amount and/or kind of knowledge differs.

This allows evaluation of expert systems to be done in a direct way compatible with the theory of classification.

5. REFERENCES

Groundwater, E.H., Donnell, M.L., and Archer, M.A., "Approaches to the Verification and Validation of Expert Systems for Nuclear Power Plants, Electric Power Research Institute, EPRI Np-5239 Final Report, July 1987.

Anderson, T.W., An Introduction to Multvariate Statistical Analysis, John Wiley and Sons, Second Edition (1984) pp. 195-241.