Modeling and debugging engineering decision procedures with machine learning

Yoram Reich,[1] Miguel Medina,[2] M. ASCE, Tung-Ying Shieh,[3] Timothy Jacobs,[4] A. M. ASCE,

Journal of Computing in Civil Engineering, ASCE, (in press), 1995

Abstract

This paper reports on the use of machine learning systems for modeling existing engineering decision procedures. In this activity, various models of an existing decision procedure are constructed by using different machine learning systems as well as by changing their operational parameters and input. Individual models serve to focus on different aspects of the decision procedure and their combined use thus improves the understanding of the decision procedure which, in turn, can assist in its evaluation and subsequent debugging and improvement. This important modeling role of machine learning systems is exemplified by modeling an existing decision procedure that is used by engineers in selecting among available techniques for modeling groundwater flow and contaminant transport in a process of environmental decision making. This decision procedure was corrected and improved in the course of this work. The example demonstrates the practical utility of the modeling role of machine learning for engineering applications.

Key words: machine learning, engineering modeling, decision procedures, knowledge acquisition, groundwater flow modeling, computational support for modeling, practical impact.

1 Introduction

A decision procedure (DP) can be defined as a sequence of actions executed for arriving at some final decision. Most often, the final decision is the selection among competing alternatives. A DP can be represented as a graph whose nodes are the actions or choices and its branches connect between consecutive actions. The leaf nodes of the graph are the final decisions. The actions may range from simple choices between several options presented by the system, to costly experiments followed by choices (e.g., "What is the actual soil structure in a contaminated site? Select next step based on measurements augmented by judgment."), and the final decision may be which model to select for a particular analysis.

DPs are often developed manually by engineers and researchers without the utilization of tools from decision theory or from other disciplines. DPs are developed for various purposes such as recording regulations or transferring some expertise between individuals or organizations. When such regulations or expertise evolve, DPs are modified manually. As with any other manual activity, the construction of DPs is prone to errors. Indeed, many flaws may exist in DPs including a wrong or incomplete structure of the decision graph. Solutions to these problems can be approached by

[1]Senior Lecturer, Department of Solid Mechanics, Materials and Structures, Faculty of Engineering, Tel-Aviv University, Tel-Aviv 69978, Israel. This work was done while this author was visiting the Department of Civil and Environmental Engineering at Duke University.

[2]Associate Professor, Department of Civil and Environmental Engineering, Box 90287, Duke University, Durham, NC 27708-0287, USA

[3]Graduate Research Assistant, Department of Civil and Environmental Engineering, Box 90287, Duke University, Durham, NC 27708-0287, USA

[4]Assistant Professor, Department of Civil and Environmental Engineering, Box 90287, Duke University, Durham, NC 27708-0287, USA

a variety of ways. For example, one can attempt to traverse a DP carefully in order to detect these problems; however, if a DP is complex, this may prove a time-consuming and error-prone process. Other approaches include the use of software verification techniques or the translation of a DP into rules and, subsequently, the use of some knowledge-based systems verification techniques. Unfortunately, these mechanisms will only deal with some syntactic issues, such as detecting contradictions between final decisions, and not with the semantics of various choices, branches, and their relationships. In order to address semantic problems a different approach of evaluation is required.

This paper illlustrates the use of machine learning (ML) tools for modeling existing DPs for improving their understanding, thus supporting their evaluation, debugging, and improvement. By using ML tools to learn models of the DP from examples of its use, and the use of these models to debug and improve an existing DP, we demonstrate a practical application of ML.

ML algorithms developed in artificial intelligence accept *input* data in several forms and transform it into an output, referred to as "*knowledge*;" this knowledge captures *important properties* of the input and can be used to explain the input data or be used for decision making. For example, the *input* of a program that learns to diagnose bridge failures may consist of descriptions of bridge structure, traffic pattern, and historical performance data, and its output may be *knowledge* that can explain previous failures concisely, or predict when another bridge is suspect of failing. This knowledge may be presented in the form of decision rules which encode a causal relationship between an important attribute of the bridge (e.g., bridge type and traffic patterns) and its expected failure type. Intuitively, the knowledge is constructed by analyzing the co-occurrences of certain bridge failures with certain bridge types and traffic patterns. An overview of ML in civil engineering is presented in (Arciszewski and Rossman, 1992). A sample of studies in civil engineering includes: deriving feasibility information from design examples (Arciszewski et al, 1987), extracting synthesis knowledge from designs (Garrett and Ivezic, 1992; Reich and Fenves, 1992), learning traffic control knowledge (Arciszewski et al, 1994), and learning evaluation knowledge (Julien et al, 1992). A recent examination of the role of ML techniques in engineering design can be found in (Reich et al, 1993).

The remainder of this paper is organized as follows. Section 2 discusses the modeling role of ML. Section 3 describes a case study of using ML techniques for modeling and debugging one particular DP and Section 4 concludes the paper.

2 ML systems as modeling tools

A *model of an object* (physical such as a beam or conceptual such as a procedure) can be defined as an abstract description of it that attains all the important features of the actual object and is constructed for some predefined purpose. The purposes of modeling may be divided into two interrelated and overlapping categories: *understanding* and *utilization*. The purpose of understanding is to gain insight about the original object (i.e., to learn additional knowledge about it). The purpose of utilization is to use the model instead of the original object for decision making. Of course, improved understanding allows for better utilization and vice versa.

Our purpose of evaluating a given DP belongs to the understanding category. Thus, ML systems are not meant to generate knowledge for decision making; rather, the goal of their use is to create models that capture some properties of the DP in a comprehensible form that can be inspected and be useful for making observations about the DP. Since the knowledge structure must be comprehensible, subsymbolic learning methods such as neural networks are inappropriate for this role.

Many ML systems produce comprehensible knowledge structures, the common forms being rules or decision trees. Increasing the number of modeling tools improves understanding because each learning system provides a different perspective of the DP under study due to its inherent bias toward a particular knowledge representation. Furthermore, for each system, different parameters can yield different models. We begin the study with the most common type of learning: *symbolic, supervised, concept learning* methods. These methods can utilize input in several forms. The first source is a set of examples of decision scenarios created by an actual DP where each scenario consists of the selections made at various nodes of the algorithm and the final decision, all described by a list of attribute-value pairs. This source can be readily used by concept learning techniques, supervised and unsupervised, the former being able to use the final decision of a scenario as the label.

The second source of input to a learning system consists of a formal logical model of the structure of the DP, e.g., a decision tree whose structure is as close as possible to the structure of the DP. This model can be used in conjunction with the first source by various ML systems to different degrees. Some systems may perform a search in the space of possible knowledge structures using both sources of input (e.g., hierarchy correction scheme in Ecobweb (Reich and Fenves, 1991) or the `dgraphpawn` option in IND (Buntine and Caruana, 1992; Oliver, 1993)).

The third source consists of user preferences that can be supplied interactively or *a priori*. For example, IND allows the user to intervene in the learning process by selecting an attribute to branch off, different from the one the algorithm proposes. In addition, preferences can be supplied as ordering between attributes (e.g., CN2 (Boswell, 1990; Clark and Niblett, 1989) or IND).

When a learning program is chosen, the selection of its operational parameter values have a great impact on the results. As indicated before, in the modeling role, this is an advantage. Therefore, in the course of modeling, many parameters should be experimented with.

When ML systems are used for knowledge creation, the main performance measure is constituted by various *quantitative* statistical performance tests. In such tests, the knowledge created by the ML system is used to classify new cases. These tests can assess the accuracy of models but cannot tell whether the models are useful for *understanding* the source of the data. Thus, it is not necessary that better understanding of the original DP is facilitated by more accurate ML models; sometimes, crisp and concise insight can be derived from primitive or inaccurate models. Since the goal of the modeling process is to understand the original DP so as to uncover potential flaws in its structure, we focus on manual *qualitative* evaluations of the knowledge structures that different ML systems generate. Next, we discuss how the above steps are operationalized through an example of modeling a particular DP.

3 A case study: Modeling the CHOICE Algorithm

This section explores ML systems as a means of modeling and analyzing an algorithm, CHOICE, that assists engineers in the proper selection of mathematical models within a comprehensive groundwater quality modeling and management advisory system. These models are based on analytical, semi-analytical and numerical solutions to solute transport in one to three-dimensional saturated groundwater flow.

3.1 An overview of the CHOICE Algorithm

CHOICE is part of an interactive, menu-driven management program that executes a large number of supporting decision algorithms and mathematical models. The models currently incorporated into CHOICE are listed in Table 1. The mathematical details of the models and the theoretical aspects of the management modules are presented elsewhere (Medina et al, 1988a; Medina et al, 1988b; Marin et al, 1989; Medina et al, 1989). Criteria for choosing among transport models have also been a topic of regulatory interest (U.S. EPA, 1988), but without guiding the user to a specific model. Several authors have compared the performance of numerical codes to analytical solutions, using benchmark data sets and real site data (e.g., Kinzelbach, 1987; Kinzelbach, 1988; Lobo Ferreira, 1988). The algorithm for choosing among the numerical codes is based in part on such comparisons and in part on the expertise of its developers in groundwater modeling. CHOICE was initially developed to aid regulators in the selection of transport models to be used for evaluating groundwater discharge permit applications (Medina et al, 1988b). It is currently undergoing modification for the U.S. Air Force and the U.S. Army Corps of Engineers, to include selection of transport models used to predict the effectiveness of alternative remediation schemes, optimizing for cost/effectiveness. Its development process followed common program development practices with no reference to AI techniques in general or ML techniques specifically. In retrospect, there are not enough empirical data available that can support the (semi-)automated generation of knowledge for selecting between groundwater models by ML programs.

The structure of the algorithm is shown in Figure 1. A summary of CHOICE's attributes is given in Appendix III. The algorithm can be roughly subdivided into 3 branches: branch I which allows experienced users to select the model of their choice, branch II which deals with analytical models, and branch III which deals with numerical models. The user responds to screen queries about whether analytical solutions are known to be appropriate or inappropriate (in the latter case, whether the region modeled is homogeneous or heterogeneous). If the complexities require a numerical model, the algorithm then jumps to the branch depicted in Figure 1(b), to select between the two available numerical codes. The model recommended depends upon various site conditions input by the user.

3.2 A general outline of the modeling activities

Three types of modeling techniques are reviewed among those that have been conducted with CHOICE and that demonstrate the modeling role of ML techniques (for a more complete description see (Shieh, 1993)): (1) naive modeling — based solely on frequency counts, (2) modeling with CN2 (Boswell, 1990; Clark and Niblett, 1989) — a rule creation program, and (3) modeling with IND (Buntine and Caruana, 1992) — a suite of decision-tree creation programs. Appendix II briefly describes these two programs. Both CN2 and IND are symbolic, supervised, concept learning programs. The selection of programs was based on functionality, availability, and diversity in knowledge representation (i.e., rules and decision trees).

The input to these ML systems, consisting of a collection of DP examples, was created by simulating CHOICE such that at each decision point, all branches were traversed with equal probability. If the DP was a balanced tree, each leaf would have also been selected with equal probability. This is a reasonable default strategy for modeling a "black box" DP, and we employed it here although we know exactly that CHOICE is not a balanced tree. Each training example was created by recording as a list of attribute-value pairs the sequence of choices made while traversing the algorithm. The recommendation of CHOICE, encoded as the value of the final decision attribute, was considered as the category of the example. Since there are altogether 32 distinct choices, each example was described by 32 attributes and belonged to one of 9 classes which are the models depicted in Table 1. Nevertheless, since in a particular simulation only the attributes on a particular path were visited, the remaining attributes were assigned an immaterial value. In this study, 500 examples were created by 500 simulations of the algorithm. (We note in passing that modeling with 5000 examples gave comparable results. This can be explained in the context of ML programs' learning curve: for our purpose, 500 examples were sufficient to generate good models of the DP.)

Most ML systems have several operational parameters that govern their behavior. These may include threshold values for handling noise in data or types of criteria for selecting how to search in the space of rules or trees that the programs generate. Different parameter values give rise to different models. While diversity of models is useful for providing different perspectives of the data, an exhaustive combination of all different parameter values is not feasible. Therefore, we decided to select the operational parameters that lead to the best results in performance tests of the ML programs. The performance test we used is the standard 10-fold cross-validation (CV), an excellent test when the input contains more than 100 examples (Weiss and Kapouleas, 1989).

In the following sections, naive modeling, modeling with rules, and with decision trees that demonstrate the potential of ML techniques for modeling and debugging DPs are reviewed.

3.3 Naive modeling with frequency counts

Table 2 shows the number of times each model appeared as the final recommendation in the 500 examples. The table clearly shows the bias of CHOICE towards selecting numerical or semi-analytical models over analytical models. This bias can be contrasted with the detailed part of the algorithm dealing with analytical models (branch II in Figure 1) and the relative simplicity of selecting between the non-analytical models (branch III in Figure 1). The bias reflects an intention (even if implicit at the time of development) of the developers (i.e., Medina, Marin, and Butcher, 1988b) that the analytical models are rarely suitable for modeling realistic groundwater flow.

Table 3 displays the frequencies in which each attribute is specified in the 500 examples. Notice that four attributes were never used and others used very infrequently, indicating branches never or hardly ever visited. The results reported in the table raise several issues. First, problems might arise from such data due to its differing nature from common input for ML systems (which consist of almost completely specified attributes). Further, in our case the unspecified attributes are not unknown; rather, they are *known to be immaterial* for the specific example. Such attributes require special treatment by ML techniques.

The second observation is that it is expected that attributes whose frequency is close to 500 are placed close to the root of the algorithm or placed at different relevant locations inside the decision process. If this is not the case, attributes may be immaterial to the decision or may be placed wrongly throughout the decision process.

Attribute A24 (see Appendix III for a description of the attributes) appears so frequently because of the aforementioned bias towards more complex numerical models. Several attributes irrelevant to the final decision have been observed in this activity. Since the data were generated by running CHOICE under the assumption that no preliminary analysis using a simplified groundwater screening model had been undertaken, the two attributes dealing with such analysis (A17 and A18, which were omitted from Figure 1) could be removed.

Attribute A25 (which was also omitted from Figure 1) appears near leaf nodes, yet appears in many examples, suggesting it may be immaterial to the final decision. In fact, A25 merely differentiates between an environmental standard and human exposure and was indeed immaterial to model selection.

Attributes A11-A13, A14, A16, A19, or A23 should not have ranked as immaterial to the final decision as they appear to be from their infrequent appearance. This infrequency was due to the method of example generation and the relative placement of the attributes in CHOICE. This points to the need to employ multiple ways of input generation as well as examine whether any of these attributes should be promoted to higher nodes at CHOICE.

3.4 Modeling with CN2

In this case, the 500 examples were used by CN2 for creating rules. In the frequency count modeling the present data differ significantly with common input to ML systems. In the initial runs with CN2, the rules created often did not have any physical meaning because they included tests on attribute values that were meaningless to the final decision of the rules. In its natural way of operation, CN2 tried to differentiate between the 9 example categories (denoted by the models' names). CN2 included in some category descriptions, attributes immaterial to those categories definitions only because these attributes were used for the definition of *other* categories. We call this problem, the 'don't-care' value problem. (This is different from the 'don't-know' problem in which some of the attribute values are not known because here, they are *known* to be immaterial.)

For example, the rule:

```
IF    Aquifer can be characterized as not drastically inhomogeneous in the
       region of interest, but may be affected by constant head boundaries
       or pumping
  AND Aquifer can be characterized as approximately infinite.
  AND Perimeter of interest is located close to the edge of the source,
       so that source can be treated as approximately horizontally infinite.
  AND Regional flow regime is important to the transport of contaminant(s).
  AND A22 = 7 (i.e., User selects the model RESSQ.)
  AND Type of impact standard is human exposure.
THEN model=RESSQ [0 0 0 0 0 0 221 0 0]
```

includes the A22 attribute which is meaningless in the context of the other attributes. A close comparison of this rule with Figure 1 reveals that choices relevant to two different branches of the DP (i.e., the sequences (A3, RESSQ) and (A1, A24, A4, A5, A15, A9, RESSQ)) were incorporated into a single rule that explained 221 out of the 500 examples with 100% accuracy.

There are several approaches to solving the don't-care value problem. The first, improving the particular ML system employed in the modeling, is not relevant to the context of this paper, even though this would be an important undertaking. A second option is using the facilities available in CN2 to try to circumvent the problem. The approach selected—a way of using background knowledge in the learning process—introduces ordering relationships between the attributes as they appear in the decision process to compensate for the inability of the attribute-value pair representation to capture the order relationship. For example, given that A1 precedes A2, A4, and the A5 attributes, we could force CN2 to include a condition with A1 whenever a condition with either A2, A4, or A5 was considered.

The ordering was based on the precedence of choices as they appear in the CHOICE flowchart. A steep decline in the performance measure calculated by the 10-fold CV test occurred when introducing an improper precedence constraint through a statement that enforces a constraint that is true *locally* (in one branch) on the *whole* DP. For example, A6 is before A9 only when traversing the sequence of choices (A1, A24, ratio, A6, A9) but not (A1, A24, A5, A15, A9). With the exclusion of such constraints, the results converged, albeit never got, to 100%. Note that such a decline in accuracy can also be the result of a discrepancy between the algorithm flowchart and its computer implementation. The ordering that causes the decline focuses attention on a particular branch of the algorithm and can thus be used to help to debug it.

The 100% accuracy was not approached due to CN2 inability to accept ordering constraints not only between attributes but also between attribute values and the lack of provision for specifying conditions on orderings. Nevertheless, ordering overcame much of the don't-care value problem.

Another modeling activity involved a detailed examination of the ability of rules induced to classify the training examples after removing the category attribute (also called a resubstitution test). When the DP as in flowchart (a) (see Figure 1) was used in generating the examples and enforcing some ordering constraints on attributes, CN2 created an ordered set of rules that when used to classify the examples gave the results in Table 4. (CN2 can create an ordered set of rules that must be executed sequentially or an unordered set which can be run in parallel. See Appendix II.)

The entry in row i, column j of the matrix is the number of examples of class i classified by the rules induced by CN2 as belonging to class j. In the matrix only 38 classifications of MOC examples match the actual examples and up to 77 (2+41+34) examples constitute a mismatch for MOC. The classifications of these 77 examples into the RESSQ or R_W classes demonstrate that the rules induced for MOC cannot differentiate well between RESSQ, MOC or R_W.

The ordered rule set model had better accuracy in the 10-fold CV test than the unordered set of rules, but its comprehensibility was significantly inferior to the unordered set because ordered rules are executed in sequence and are therefore dependent on each other, whereas the unordered rule set is decomposable into single rules such that each could be inspected independently. Thus, not necessarily is a more accurate model for prediction better for comprehension.

Based on the previous discussion, several modifications were performed to the modeling activity. (1) The exercise of ordering that applies to the whole DP; and (2) interpreting the DP as in flowchart (b) of Figure 1, rather than as in (a). These were meant to lead to models that better resemble the structure of CHOICE. (3) Performing modeling with unordered rules. This was aimed at obtaining more comprehensible rules. Finally, (4) modeling without user selection (eliminating branch I). This was employed to eliminate obvious algorithm branches from consideration in the modeling. (If we divided these and performed them one at a time it was hard to tell which contribute more to the subsequent results.) Item (4) required us to generate 500 new examples from the DP without branch I. Table 5 shows the matrix obtained from the resulting activities.

The matrix shows significant improvement in differentiating between MOC and R_W: both accuracies of MOC and R_W are higher than 95%. There are several explanations for the better separation. First, in the 500 examples, no example whose final decision was the MOC model was generated from the sequence of choices, A20 to A31. This left the MOC, R_W and RESSQ models quite distinct. This raises the question whether this was the intended behavior from the algorithm: rarely or almost never select MOC when arriving at choice A20.

As it turns out, this was not the intended behavior of CHOICE: a choice between deterministic or Monte Carlo simulation (i.e., node A20) should only be made after the appropriate mathematical model has been selected. In other words, it refers to the mode under which that model is to be executed. Thus choice A20 was wrongly placed in the sequence of choices. This is an excellent example of how the ML technique focused us on a semantic flaw in the algorithm that might not have been otherwise found. Second, the elimination of the user selection alleviated some of the problem as well, since choices emanating from branch I are arbitrary and may act as noise to the other examples. This points to the benefit from separately modeling different parts of a DP to gain better local insight.

3.5 Modeling with IND

IND can create decision trees by using different algorithms such as BAYES, C4, or CART. Two decision trees created by the CART and C4 algorithms on data generated from CHOICE without branch I and with flowchart (b) are presented here.

The first tree (Figure 2) was created with the CART algorithm. The bold lines in Figure 3 in the area enclosing the word 'CART' approximately display the choices in the tree graphically. The accuracy of the model estimated by a 10-fold CV test is about 94.0%. Compared to Figure 1 or Figure 3, the first branch in the tree which starts with A3=1 represents a small portion of the analytical branch II that deals with the semi-analytical model RESSQ, combined with the corresponding part from branch III. Note that according to Figure 3, A3 is immaterial to this branch. The second branch, which starts with A3=2, deals with the numerical models. It is a perfect reflection of the data from branch III since, as noted before, no MOC example was generated randomly from the path A26 to A31. No analytical model was incorporated into the tree.

The second tree is created based on the C4 algorithm (see Figure 4 and the area enclosing the word C4 in Figure 3) and its accuracy estimated by a 10-fold CV test is about 95.8%. This tree is significantly more elaborate than the one created by CART: it is deeper and contains more branches and decision nodes. The R/b `ratio` is again induced as a node in both branches of A3 and is thus considered more important by C4 than as reflected in the CHOICE flowchart.

The placement of the `ratio` attribute in the trees induced may provide some insight about its importance for selecting among groundwater flow models. Its role will be re-examined once more experience in the use of CHOICE is acquired. Equally probable, the importance of the `ratio` attribute may point to the common difficulty of combining numeric (i.e., the `ratio` attribute) with nominal attributes (i.e., all the remaining attributes describing the training examples: A1, ..., A31) in learning programs. The mix between the different branches, including placing attributes in branches not applicable is apparent. For example, A4 and A5 should not be in the branch of A3=2; A4 can lead to the use of the EPAGW model directly, and A5 needs more information to indicate the specific model recommendation. These two attributes may not have a strong direct relationship with RESSQ, MOC and R_W.

It seems from the modeling results obtained by applying both the CART and C4 algorithms to CHOICE, that CHOICE can be streamlined without significantly affecting the final model selection decision. The C4 algorithm appears to have eliminated several choices (PLUM2D, TDAST, DUPVG, LTIRD) as shown in Figure 4, while CART eliminated more.

It appears appropriate to re-design CHOICE, and subject it to further testing, particularly since other numerical models are currently being added to the advisory system: a bioremediation model, and a finite element code that will allow for more sophisticated boundary conditions. We are currently in the process of incorpoating selection criteria between these and the other grounwater flow models into CHOICE and testing CHOICE with ML systems using the same procedure described in this paper.

3.6 Summary of experimental results

The results of learning can be subdivided into three categories: (1) *errors*, (2) *theoretical ambiguities*, and (3) *practical implications*. Several errors have been detected in our study. First, several inconsistencies between the algorithm flowchart and the computer code were detected and corrected. The flowchart in Figure 1 and the DP used to generate the examples for the modeling activities reported reflect these corrections. Second, one attribute (A20) was detected in a wrong location in CHOICE. Third, two attributes, A23 and A25, were observed as immaterial to the algorithm recommendations, thus could be removed from the analysis (and maybe from the algorithm, at least for presentation and training purposes).

There were several theoretical ambiguities detected. First, some of the attributes that were modeled as irrelevant to the algorithm recommendation are not considered as such. These findings require further investigation. Second, while developing CHOICE the developers did both implicitly and in some cases explicitly bias the selection of the more complex codes since groundwater modeling is a relatively complex undertaking. It is also important to point out that experience is also important in applying the simpler codes. These algorithm biases were made very explicit through simple frequency modeling. Other biases detected, for example, favoring R_W over MOC were not intended: Although R_W is able to handle more complex flow circumstances than MOC, MOC is easier to apply for inexperienced users. This may require elaborating CHOICE. Third, some attributes (`ratio`) that were often used in the knowledge structure generated (mainly decision trees) may prove important beyond their explicit role in the algorithm (or in the way they are understood). These theoretical ambiguities must be studied further to guide the action needed in relation to the algorithm.

There are also some practical implications that were detected in the modeling activities. The simplicity of the models created by ML techniques focus on potential simplifications of the algorithm. That is, simple models (rules of decision trees) could replicate the algorithm recommendations with accuracy of close to 97%. One can use these simplifications, while losing slightly accuracy, or one can use them while augmenting/correcting them locally to almost perfectly simulate the original algorithm by studying the discrepancies between the algorithm and its ML model's recommendations. In doing so, pragmatic issues such as, how is the algorithm comprehensible to users versus its sophistication, are critical.

4 Summary and Conclusions

In this study, we demonstrated a practical application of ML methods. We used available ML systems for the *modeling and evaluation* of DPs. We illustrated how modeling a particular DP—CHOICE—with ML systems has led to insight about it and described how this insight has already resulted in modifications and improvements of CHOICE. These modifications corrected some errors in the original algorithm. Additional re-ordering of choices, elimination of redundancies, and incorporation of other models will necessitate further testing by users and will benefit from ML modeling. We intend to monitor the algorithm and perform such tests (as part of the groundwater management advisory system) after its deployment in the U.S. Air Force.

We also discussed some of the issues involved in using ML systems to model DPs for the purpose of evaluation. These issues also contain guidelines for using ML programs for modeling DPs.

1. Evaluation benefits from modeling with multiple tools; therefore, the use of several ML systems, with different inputs, and by varying the systems' operational parameters is advantageous. The systems should match the modeling task at hand. *For modeling a DP, the appropriate ML technique is supervised concept learning. It is recommended to use at least one system that creates decision rules (e.g., CN2) and one that creates decision trees (e.g., IND).*

2. The comprehensibility of models created by ML is critical rather than their prediction performance accuracy as compared to the actual DP. *Therefore, programs that are geared toward comprehensible knowledge structures are preferred for modeling DPs.*

3. Nevertheless, since the combination of several ML techniques, their inputs and operational parameters, can lead to creating many models, *a good heuristic for focusing on a subset of them is to select parameters that lead to better results in a 10-fold CV performance test.*

4. *Different insight about the DP emerges from modeling it as a whole and also its parts separately.* Similarly, *when modeling a complex DP, several interpretations of the DP followed by generating different sets of input to the ML programs can display different properties of the DP. It is recommended to generate one data set by simulating the DP with the default equal probability strategy used in this paper.* In subsequent modeling activities *other data sets can be generated for focusing on specific particular branches of the DP.*

5. Some properties of DPs may lead to creating input that is inappropriate for use by most ML systems. While some issues can be addressed by facilities available in some ML systems, such as the solution of the don't-care problem by using prior knowledge, additional work on ML is required to provide adequate support for this modeling task.

The modeling of a complex DP with ML is not only useful for debugging. A series of simplified models can improve the presentation of the DP or the training of engineers in using it. Such models can also guide future research by focusing on branches that were less developed compared to others. Finally, ML modeling can help evaluate several competing DPs for selecting among them.

Future work involves testing the modeling ability of ML techniques, those used in this study and others, on other DPs. We plan to start with a DP that recommends remediation activities for contaminated sites. This DP is more complex than CHOICE in that it deals not only with simulations but also with collection of actual field measurements. Consequently, the cost of using the algorithm modeled becomes critical as a modeling parameter for the ML programs. This, in turn, increases the potential benefits from using ML as modeling and feedback tools in the design of the algorithm.

The future goal of this study is to make ML modeling an integral part of DPs development. This entails addressing the following issues:

1. Understand the usefulness of different ways of modeling: A critical issue in light of the variety of techniques available.

2. Understand the tradeoffs involved in modeling with ML techniques such as generality vs. accuracy.

3. Expand this understanding by accumulating case studies (failures and successes) dealing with ML modeling.

Appendix I. References

Arciszewski, T. and Rossman, L. A., editors (1992). *Knowledge Acquisition in Civil Engineering*, American Society of Civil Engineers, New York, NY.

Arciszewski, T., Mustafa, M., and Ziarko, W. (1987). "A methodology of design knowledge acquisition for use in learning expert systems." *International Journal of Man-Machine Studies*, 27(1):23–32.

Arciszewski, T., Khasnabis, S., Hoda, S. K., and Ziarko, W. (1994). "Machine learning in transportation engineering: A feasibility study." *Applied Artificial Intelligence*, 8(1):109–124.

Boswell, R. (1990). "Manual for CN2 version 2.0." Technical Report TI/P2154/RAB/4/1, The Turing Institute, Glasgow.

Buntine, W. and Caruana, R. (1992). *Introduction to IND Version 2.1 and Recursive Partitioning*, NASA Ames Research Center, Moffet Field, CA.

Clark, T. and Niblett, P. (1989). "The CN2 induction algorithm." *Machine Learning*, 3(4):261–283.

Garrett, J. and Ivezic, N. (1992). "A neural network approach for acquiring and using synthesis knowledge." In Dagli, C. H., Burke, L. I., and Chin, Y. C., editors, *Intelligent Engineering Systems Through Artificial Neural Networks, Vol. 2*, pages 767–772, New York, NY, ASME Press.

Javandel, I., Doughty, C., and Tsang, C. F. (1984). *Groundwater Transport: Handbook of Mathematical Models*, American Geophysical Union, Washington DC.

Julien, B., Fenves, S. J., and Small, M. J. (1992). "Knowledge acquisition methods for environmental evaluation." *AI Applications*, 6(1):1–20.

Kinzelbach, W. (1987). "Methods for the simulation of pollutant transport in ground water." In *Proceedings of Solving Ground Water Problems with Models*, pages 656–675, Denver CO.

Kinzelbach, W. (1988). "The Random Walk method on pollutant transport simulation." In Custodio, E., editor, *Groundwater Flow and Quality Modeling, NATO ASI Series C(224)*, pages 227–245, Boston, MA, D. Reidel Publishing.

Konikow, L. E. and Bredehoeft, J. D. (1978). *Computer Model of Two-dimensional Solute Transport and Dispersion in Groundwater*, U.S.G.P.O., Washington D. C. Techniques of Water-Resources Investigations of the U.S.G.S, chapter c2, Book 7.

Lobo Ferreira, J. P. (1988). "A comparative analysis of mathematical mass transport codes for groundwater pollution studies simulation." In Custodio, E., editor, *Groundwater Flow and Quality Modeling, NATO ASI Series C(224)*, pages 699–716, Boston, MA, D. Reidel Publishing.

Marin, C. M., Medina, M. A., and Butcher, J. B. (1989). "Monte carlo analysis and bayesian decision theory for assessing the effects of waste sites on groundwater: Theory." *Journal of Contaminant Hydrology*, 5(1):1–15.

Medina, M. A., Butcher, J. B., and Marin, C. M. (1988). "An advisory system for groundwater quality modeling and management." *Hydrosoft*, 1(4):197–203.

Medina, M. A., Butcher, J. B., and Marin, C. M. (1988). "An advisory system for North Carolina groundwater quality management and modeling needs." Technical Report 236, Water Resources Research Institute of the University of North Carolina, Raleigh, NC.

Medina, M. A., Butcher, J. B., and Marin, C. M. (1989). "Monte carlo analysis and bayesian decision theory for assessing the effects of waste sites on groundwater: Applications." *Journal of Contaminant Hydrology*, 5(1):15–31.

Mulkey, L. and Brown, L. (1985). "Development of land disposal banning decisions under uncertainty.". U.S. EPA, Environmental Research Laboratory, Athens, GA.

Oliver, J. J. (1993). "Inferring decision graphs using the minimum message length principle." In *Artificial Intelligence and Statistics*.

Prickett, T. A., Naymik, T. G., and Lonnquist, C. G. (1981). "A 'random walk' solute transport model for selected groundwater quality evaluations.". Illinois State Water Survey, Bulletin 65, Champaign, IL.

Reich, Y. and Fenves, S. J. (1991). "The formation and use of abstract concepts in design." In Fisher, D. H. J., Pazzani, M. J., and Langley, P., editors, *Concept Formation: Knowledge and Experience in Unsupervised Learning*, pages 323–353, Los Altos, CA, Morgan Kaufmann.

Reich, Y. and Fenves, S. J. (1992). "Inductive learning of synthesis knowledge." *International Journal of Expert Systems: Research and Applications*, 5(4):275–297.

Reich, Y., Konda, S., Levy, S. N., Monarch, I., and Subrahmanian, E. (1993). "New roles for machine learning in design." *Artificial Intelligence in Engineering*, 8(3):165–181.

Shieh, T.-Y. (1993). "The potential of machine learning techniques for improving decision algorithms in groundwater contaminant transport modeling." Master's thesis, Graduate School of Duke University, Durham, NC.

U.S. EPA (1988). "Selection criteria for mathematical models used in exposure assessments: Ground-water models.". EPA 600/8-88/075, Washington, DC.

van der Heijde, P. K. M. (1985). "Analytical solution for transport of a conservativer or non-conservative tracer in groundwater, 'PLUME 2D'." Technical report, International Ground-water Modeling Center, Bulter University, Indianapolis, IN.

Volker, R. E. and Guvanasen, V. (1987). *Mass Transport in Unconfined Aquifers*, Gulf Publishing Co., Huston, TX. Encyclopedia of Fluid Mechanics, chapter 20.

Weiss, S. M. and Kapouleas, I. (1989). "An empirical comparison of pattern recognition, neural nets, and machine learning classification methods." In *Proceedings of The Eleventh International Joint Conference on Artificial Intelligence, Detroit, MI*, pages 781–787, San Mateo, CA, Morgan Kaufmann.

Appendix II. The ML programs CN2 and IND

CN2 is a supervised concept learning program that creates classification rules from training examples represented by a list of attribute-value pairs. CN2 creates rules by searching for conjunction of attribute tests that *cover* a subset of the training examples. Such a conjunct forms the condition part of a classification rule and the classification of the rule is the most common class in the training examples which satisfy the condition. In its search for rule conditions, CN2 specializes conjuncts until one that covers a large number of examples of a single class and few of other classes is located by passing a predictive and reliability evaluations that are governed by operational parameters that the user controls. Once a rule it created, the training examples it covers are removed from further consideration and another rule is sought. This process continues until the training set is exhausted. The product of this process is an ordered set of rules; that is, in classification, the rule set is used in the order in which is was created.

CN2 can also create an unordered rule set in which all rules are matched in classification. In this case, conflicting classification can arise from different rules. This is resolved by preferring the rule that covers more positive training examples.

IND is a suite of reimplementations of several decision tree and decision graph ML programs. IND integrates the CART, C4.5, Bayesian, and minimum encoding techniques. While CN2 uses a bottom-up strategy, IND implement methods that work top-down in a divide and conquer strategy. There are several ways to select how to devide the training examples and how to grow the tree (e.g., minimum entropy or gain using single or multi-ply look ahead search) as well as different methods for stopping the growth and trimming the tree to account for noisy data. All these are goverened by operational parameters.

Appendix III. A subset of the attributes used by CHOICE

Bracketed terms indicate branching sequence or final model selection.

A1: Site can best be characterized as: (1) Wastewater Lagoon [A2 if A24=7]; (2) Spray irrigation of wastewater [Ratio if A24=7]; (3) Land application of sludge [A4 if A24=7]; (4) Individual rotary distributor [Ratio if A24=7]; (5) Landfill [A4 if A24=7]; (6) Injection of waste into confined aquifer [Ratio if A24=7]; (7) Septic tanks [A5 if A24=7]; (8) Direct user selection of model choice [A22 if A24=7].

A2: Wastewater lagoon is: (1) effectively lined, effect of infiltration on regional flow is slight [A4]; (2) not effectively lined, significant infiltration may occur [Ratio].

A3: Utility of semi-analytical solutions: (1) Aquifer can be characterized as not drastically inhomogeneous in the region of interest, but may be affected by constant head boundaries or pumping wells [RESSQ]; (2) System other than (1) [A20].

Ratio (R/b): Estimate long term head rise (R) due to recharge using Hantush method to estimate approximate degree of deformation of regional flow. Calculate ratio of head rise to aquifer thickness (R/b). If: (1) $R/b < 5$ methods [A5]; (2) 530

A4: Confined aquifer, or situation approximately modeled by confined aquifer solutions, or characteristics unknown. If: (1) Aquifer characteristics are ill-defined [EPAGW]; (2) Aquifer can be characterized as approximately homogeneous and infinite [A5].

A5: Approximations with confined solutions. Select between: (1) Perimeter of interest is located close to the edge of the source, so that source can be treated as approximately horizontally infinite [A15]; (2) perimeter of interest is far, source lateral boundaries must be considered [A7].

A6: Dupuit approximation methods for unconfined flow, with radial component induced by recharge. Applicable solutions depend on distance from source: (1) perimeter of compliance is located near edge of the source, the source can be treated as infinite in the lateral direction in relation to the observation point of interest. Initial gradient of water table < 1% [A9]; (2) perimeter of compliance is located further from the source, source approximated as a point, distance great enough so that full penetration is reasonable approximation [A10]; (3) neither of the above is adequate [RESSQ].

A7: Confined aquifer solutions. Select between: (1) Single areal, finite line or point sources [A8]; (2) multiple point sources of varying strength [A13]; (3) other situations [A20].

A8: Single point, line or areal sources: (1) source strength relatively constant [A11]; (2) source strength represented by an exponential decay [A14]; (3) other [A20].

A9: Perimeter of compliance near enough to source so that source can be treated as infinite in the y direction. If: (1) regional flow regime is unimportant in relation to recharge-induced radial flow, no consideration of decay [DUPVG]; (2) regional flow regime is important to the transport of contaminant(s) [RESSQ]; (3) decay must be considered [RESSQ].

A10: Perimeter of compliance far enough away that source is treated as a point. If: (1) regional flow regime is unimportant in relation to recharge-induced radial flow, no consideration of decay [LTIRD]; (2) regional flow regime is important to the transport of contaminant(s) [RESSQ]; (3) Analysis requires consideration of decay [RESSQ].

A11: Source of constant strength. If: (1) source can be characterized as a point, or collection of points [A12]; (2) source with significant width in relation to perimeter of interest [A19].

A12: Point source(s) in confined aquifer, source strength not decaying. If: (1) effects of partial penetration of contaminant into aquifer must be considered [A16]; (2) full penetration analysis adequate [PLUM2D].

A13: Multiple point sources in confined aquifer source strength not decaying. If: (1) full penetration analysis adequate [PLUM2D]; (2) partial penetration effect important [A20].

A14: Single source in confined aquifer, source strength may be decaying. If: (1) full penetration analysis adequate [TDAST]; (2) partial penetration effect is important, analytical solution is not available, examine plume with semi-analytical method. [A20].

A15: Semi-infinite line source in confined aquifer, source strength may be decaying. If: (1) full penetration analysis adequate [ODAST]; (2) partial penetration effect important [A9].

A16: Areal source of non-decaying strength, partial penetration evaluation. If: (1) expected maximum impact is along main flow axis, single source [EPAGW]; (2) expected impact not along main flow, use semi-analytical methods [RESSQ].

A17 and A18 provide general information on status of preliminary analysis.

A19: Source presents significant width. If: (1) source concentration is focused in one area [A16]; (2) source is a large area of approximately equal source strength, as in land applications of waste [TDAST].

A20: Select between: Monte Carlo [MOC] or Deterministic [A21].

A21: Selection between numerical models R_W and MOC. Select between: (1) variable grid increments desirable [R_W]; (2) grid increments in the X or Y direction must be larger by an order of magnitude than the transverse dispersivity [R_W]; (3) the ratio of longitudinal to transverse dispersivity is greater than 10.0 [R_W]; (4) flow patterns at the site are likely to be non-parallel to the grid axes [R_W]; (5) the storage coefficient varies significantly over the area to be modeled [R_W]; (6) conditions in all or part of the aquifer are likely to change from confined to water table during the simulation, or vice versa [R_W]; (7) None of the above apply [MOC].

A22: Select between: PLUM2D, EPAGW, TDAST, ODAST, LTIRD, DUPVG, RESSQ, MOC, R_W

A23: Select between: Monte Carlo or Deterministic.

A24: Do presently available data preclude use of an analytical solution. If: (1) the horizontal extent of the region of permit interest in the aquifer is known to be within 250 ft of a distinct hydrogeologic boundary [A3]; (2) the regulatory perimeter is in an aquifer that is distinct from the aquifer directly affected by the site [A3]; (3) contaminant of interest is immiscible in water, or has a specific gravity significantly different from water [A3]; (4) flow in region of site is suspected to be strongly affected by pumping wells [A3]; (5) impacted aquifer is a fractured rock system [A3]; (6) source input must be treated as slug injection [A3]; (7) none of the above [follow A1 selection].

A25: Type of impact standard is one of: Environmental standard or Human exposure.

A26: Variable grid size not needed: False or True.

A27: DX and DY $\leq$ 10 x longitudinal dispersion: False or True.

A28: Ratio of longitudinal to transverse dispersion s 10: False or True.

A29: Flow parallel to grid axes: False or True.

A30: Storativity does not vary significantly: False or True.

A31: No part of aquifer will change from confined to unconfined or vice versa: False or True.

Table 1: Models incorporated in CHOICE

Model	**Description**	**Source**
Analytical Models		
PLUM2D	Two-Dimensional Transport Model	van der Heijde, 1985
EPAGW	Three-Dimensional EPA Monte Carlo Transport Model	Mulkey and Brown, 1985
TDAST	Two-Dimensional Transport Model	Javandel et al, 1984
ODAST	One-Dimensional Transport Model	Javandel et al, 1984
LTIRD	Two-Dimensional Radial Transport Model	Javandel et al, 1984
DUPVG	Two-Dimensional (x,z) Transport Model	Volker and Guvanasen, 1987
Semi-analytical Model		
RESSQ	Two-Dimensional Complex Velocity Potential Model	Javandel et al, 1984
Numerical Models		
MOC	Method of Characteristics Model	Konikow and Bredehoeft, 1978
R_W	Random Walk Solute Transport Model	Prickett et al, 1981

Table 2: Appearance frequency of each model recommendation

Analytical Models		Semi-Analytical/Numerical Models	
Model	frequency	Model	frequency
DUPVG	13	MOC	115
EPAGW	20	R_W	101
LTIRD	6	RESSQ	225
ODAST	8		
PLUM2D	6		
TDAST	6		

Table 3: Appearance frequency of each attribute for flowchart (a) in Figure 1

Attribute	frequency	Attribute	frequency	Attribute	frequency	Attribute	frequency
A1	500	A14	0	A2	9	A11	1
A3	406	A17	0	A4	22	A12	1
A20	209	A18	0	A5	19	A13	1
A21	113	A19	0	A6	7	A15	11
A24	455			A7	8	A16	1
A25	275			A8	2	A22	45
				A9	11	A23	34
				A10	1	R/b `ratio`	21

Table 4: Prediction of models: distributions and accuracies

	PREDICTED									
ACTUAL	PLUM2D	EPAGW	TDAST	ODAST	LTIRD	DUPVG	RESSQ	MOC	R_W	Accuracy
PLUM2D	5	0	0	0	0	1	0	0	0	83.3 %
EPAGW	0	15	0	0	0	0	2	3	0	75.0 %
TDAST	0	0	4	0	2	0	0	0	0	66.7 %
ODAST	0	0	0	7	0	0	0	1	0	87.5 %
LTIRD	0	0	0	0	6	0	0	0	0	100.0 %
DUPVG	0	0	0	0	0	13	0	0	0	100.0 %
RESSQ	0	0	0	0	0	0	225	0	0	100.0 %
MOC	0	0	0	0	0	2	41	38	34	33.0 %
R_W	0	0	0	0	0	0	35	8	58	57.4 %

Table 5: Prediction of models with modified input: distributions and accuracies

	PREDICTED									
ACTUAL	PLUM2D	EPAGW	TDAST	ODAST	LTIRD	DUPVG	RESSQ	MOC	R_W	Accuracy
PLUM2D	1	1	0	0	0	0	0	0	0	50.0 %
EPAGW	0	8	0	0	0	0	0	0	1	88.9 %
TDAST	0	0	0	0	0	0	0	0	0	—
ODAST	0	2	0	4	0	0	0	1	0	57.1 %
LTIRD	0	0	0	0	0	0	0	0	0	—
DUPVG	0	2	0	0	0	4	0	0	0	57.1 %
RESSQ	0	4	0	0	0	0	209	0	0	98.1 %
MOC	0	0	0	0	0	0	2	133	2	97.1 %
R_W	0	0	0	0	0	0	2	2	121	96.8 %

Figure captions

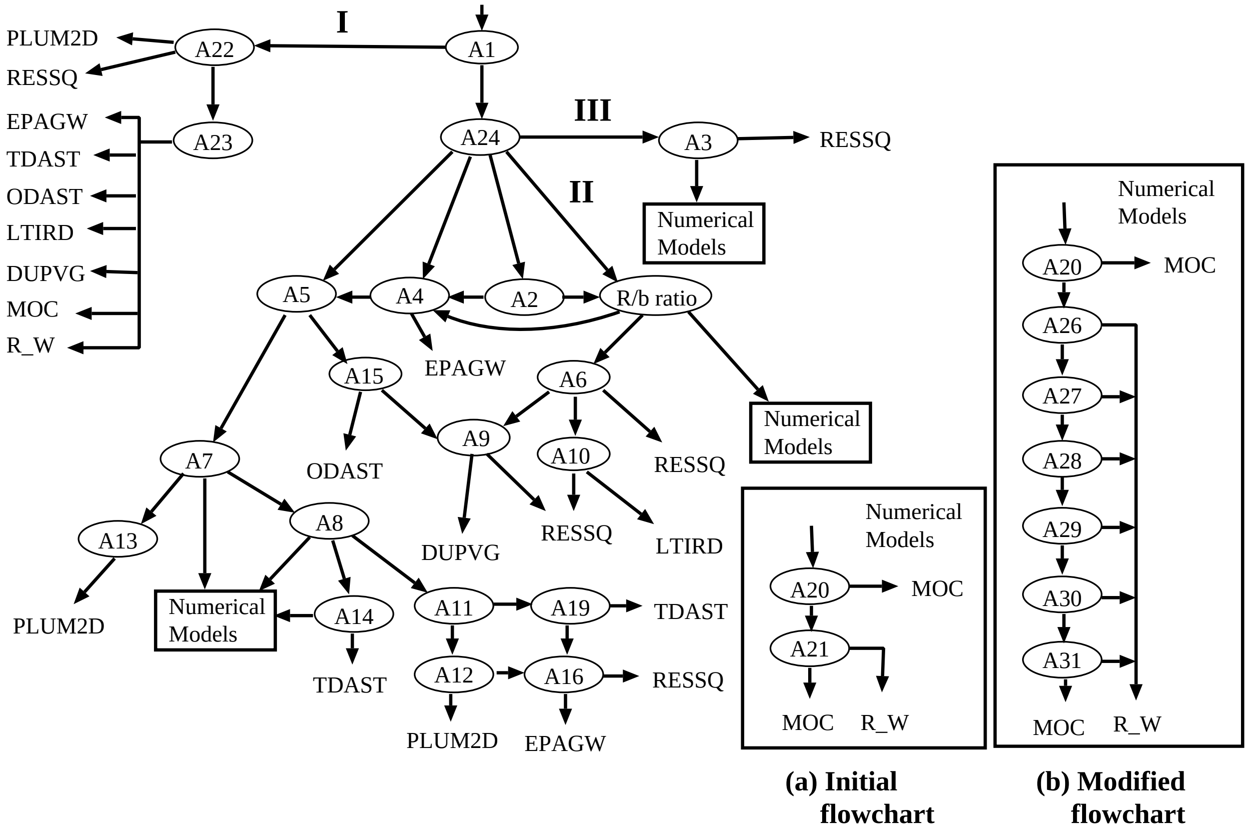

Figure 1: Flow chart of CHOICE

```
A3 = 1:      [approximates the semi-analytical branch]
|   A24 in 1,2,3,4,5,6, :  RESSQ
|   A24 in 7, :
|   |   ratio < 0.2845:  RESSQ
|   |   ratio >= 0.2845:
|   |   |   A20 = 1:  MOC
|   |   |   A20 = 2:  R_W
A3 = 2:      [approximates the numerical branch]
|   A20 = 1:  MOC
|   A20 = 2:  R_W
```

Figure 2: Modeling with the CART algorithm

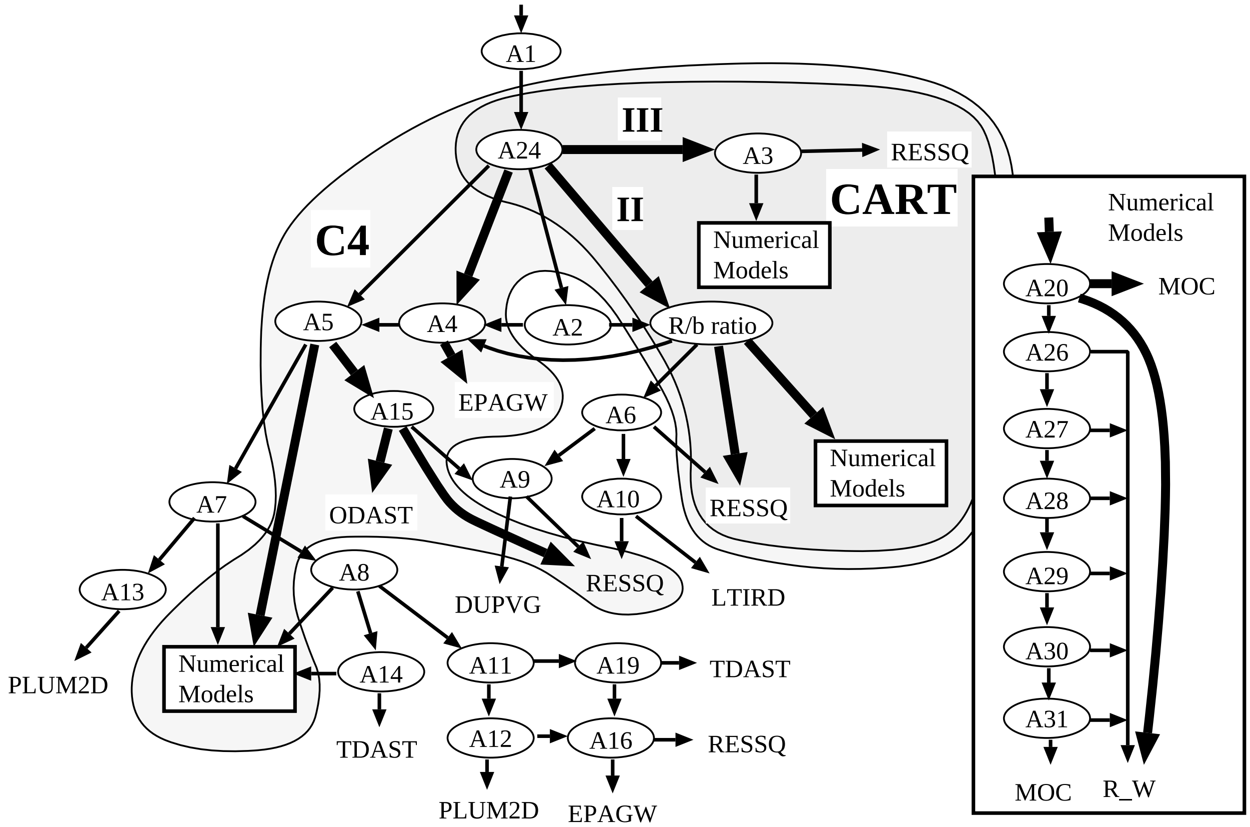

Figure 3: Part of CHOICE modeled by the CART and C4 algorithms

```
A3 = 1:
|   ratio < 0.2845:
|   |   ratio < 0.034:
|   |   |   A24 in 1,2,3,4,5,6, :  RESSQ
|   |   |   A24 in 7, :
|   |   |   |   A4 = 1:  EPAGW
|   |   |   |   A4 = 2:
|   |   |   |   |   A5 = 1:
|   |   ratio >= 0.034:  RESSQ
|   ratio >= 0.2845:
|   |   A20 = 1:  MOC
|   |   A20 = 2:  R_W
A3 = 2:
|   A20 = 1:
|   |   ratio < 0.0145:
|   |   |   A4 = 1:
|   |   |   |   A24 in 1,2,3,4,5,6, :  MOC
|   |   |   |   A24 in 7, :  EPAGW
|   |   |   A4 = 2:  MOC
|   |   ratio >= 0.0145:
|   |   |   ratio < 0.2855:  RESSQ
|   |   |   ratio >= 0.2855:  MOC
|   A20 = 2:
|   |   ratio < 0.0145:
|   |   |   A5 = 1:  R_W
|   |   |   A5 = 2:  R_W
|   |   ratio >= 0.0145:
|   |   |   ratio < 0.2845:  RESSQ
|   |   |   ratio >= 0.2845:  R_W
```

Figure 4: Modeling with the C4 algorithm