

# Knowledge Discovery Through Induction with Randomization Testing

David Jensen  
Campus Box 1106  
(314) 889-5456

Department of Engineering & Policy  
Washington University  
St. Louis, Missouri 63130

## Introduction

This paper describes an approach that combines elements of both machine learning and statistics. The approach — Induction with Randomization Testing (IRT) — provides an environment for discovery of new knowledge through flexible interaction with data and models. The approach allows investigators to use their own skills where they are strong and provides automated assistance where those skills are weak.

Of particular importance is how IRT tests models<sup>1</sup>. The approach estimates the probability that apparent improvement in the accuracy of a given model is due to chance alone. These estimates, provided by *randomization testing*, protect against constructing models of inappropriate complexity.

Below, I discuss the inadequacies of existing machine learning and statistical tools. Then I describe the IRT approach — outline its basic form, detail the underlying theory, and discuss the implementation. Finally, I provide an example of using the system and briefly discuss experience with applying the system in psychiatric diagnosis.

## Existing tools are inadequate

Two types of tools currently exist that help investigators<sup>2</sup> examine databases. *Machine learning tools* automate the search for useful models. Humans are quite skilled at this task, and machine learning tools attempt to duplicate this strength. *Statistical tools* facilitate the testing of single models. Humans are notoriously poor at this task, often mistaking chance variations for useful relationships. Statistical tools help compensate for this weakness.

Neither tool is completely satisfactory for knowledge discovery. Machine learning tools totally automate the search for useful models, not allowing investigators to guide the process using their own domain knowledge. Statistical tools are intended to test single models in isolation and don't allow investigators to accurately test multiple models while searching for useful ones.

New tools are needed. They should allow human investigators to use their own skills where they are strong and assist them where they are weak. Specifically, they should allow explicit control of the search process, but should automate the testing of the models constructed during search.

## Generating Models

Generating plausible models is the focus of machine learning tools. For example, Michalski states:

---

<sup>1</sup>*Model* refers to any method of relating two or more variables. Examples include decision trees and classification rules.

<sup>2</sup>*Investigator* refers to anyone seeking to discover new knowledge in a database. It is assumed that they often possess relevant domain knowledge.

"...of the two aspects of inductive inference — the generation of plausible hypotheses and their validation (establishment of their truth status) — only the first is of primary interest to inductive learning research. The problem of hypothesis validation...is considered of lesser importance, because it is assumed that the generated hypotheses are judged by human experts, and tested by known methods of deductive inference and statistics." (1983, p. 88)

Despite substantial attention, however, generating plausible models is still more art than science. No strong theory exists that produces general methods for constructing an effective model<sup>3</sup>. Indeed, nearly all methods employ simple hillclimbing heuristics — checking models closely related to the current model. Humans appear to use hillclimbing with some success, particularly when they lack domain knowledge. It is interesting, however, that even with substantial research, improved general techniques have not emerged. Automation has increased the speed of model generation, but has not altered its essential character.

However, complete automation of model generation imposes a considerable cost. It eliminates the guidance of human investigators who could use their own knowledge to guide the search for good models. Human knowledge can influence model generation in a variety of ways. Based on knowledge of the phenomena being modeled, investigators can direct the search for good models toward particularly profitable areas. Also, investigators know to ignore *formal* correlations in data that result from variable definition (e.g. new-balance = previous-balance + credits - debits) or *joint* correlations that result from a third intervening variable (e.g. the length of the right and left legs are correlated).

### *Testing Models*

In contrast to model generation, humans are quite poor at model testing. *Testing* refers to more than simply evaluating how well a particular model matches the data; It also involves answering the question: "What is the probability that the apparent accuracy of this model is only due to chance?" Humans have been shown to possess remarkably poor conceptions of randomness (Fischhoff, Slovic & Lichtenstein 1981), particularly when they are asked to assess whether relationships are non-random. This is why statistical significance testing is so useful — it effectively complements human abilities.

Statistical significance testing is particularly important when searching through large numbers of possible models. The number of models examined strongly affects the probability that the apparent accuracy of the best model is due to chance alone (Pearl 1978; Jensen 1990). If many models are examined without accurate significance testing, models can be constructed that work quite well on one set of data, but that perform poorly when tested on new data. This phenomenon, called *overfitting*, has been noted in both machine learning (Hart 1987; Quinlan 1988) and statistics (Einhorn 1972; Payne & Dyer 1975).

Unfortunately, conventional statistical tests are not designed for testing multiple models. Conventional tests (e.g. chi-square) are intended for testing a single model and assume that it is the only model tested. When used to test multiple models, conventional significance tests greatly underestimate the probability that the apparent accuracy of the best model is due to chance alone. Corrections for multiple tests (e.g. Bonferroni inequality) assume that the models are statistically

---

<sup>3</sup>There are exceptions in the field of computational learning theory (Haussler 1987). However, the assumptions of these techniques are extremely confining.

independent, a poor assumption when models are generated by making small alterations in an existing model. When models are not independent, corrections for multiple tests can greatly overestimate the probability that the apparent accuracy is due to chance alone.

Two solutions to overfitting have been proposed in machine learning, but both have substantial problems. Some systems employ *complexity correction factors* that penalize more complex models because of their greater propensity to overfit data. This approach confuses complexity with testing multiple models. Complexity measures are also dependent on the language used to represent models and will change depending on how models are expressed.

Other systems employ *pruning* to avoid overfitting. Pruning removes useless components of models by testing simplified versions of the model, often on fresh data. This is only done after the model is constructed, and thus offers no guidance during search. Pruning can also be wasteful of data, particularly if other methods exist. Neither pruning nor complexity corrections offers a strong theoretical foundation or aids intuitive understanding of overfitting.

### *The Interaction of Model Generation and Model Testing*

Besides influencing how model should be tested, the number of models examined has another subtle influence on the capabilities of knowledge discovery tools. As more models are examined, the classification accuracy that would be expected by chance alone increases. For a model to be considered significant, it must exceed this chance level of classification accuracy. Therefore, examining many models lowers the capability of any system to detect weak (but useful) relationships. Statisticians refer to this capability as *statistical power*.

Statistical power underscores the importance of careful generation of models. In some problems, lack of domain knowledge can be made up for by intensive search. However, this does not appear true in knowledge discovery. Statistical power goes down as the number of models examined increases. Thus, it is important not to test models that are obviously useless. Human guidance can eliminate such models before they are tested and thereby increase the statistical power of the discovery process.

### *An Alternative to Existing Tools*

To summarize, human investigators are good at generating plausible models but poor at testing them. Existing knowledge machine learning tools share these strengths and weaknesses, and existing statistical tools are ill-suited to the needs of knowledge discovery.

New tools are needed: tools that effectively complements human skills. Such tools:

- would *not* automate model generation, but instead would allow investigators to generate the basic structure of new models themselves; and
- *would* automate the testing of new models in order to compensate for human weakness in assessing statistical significance.

This approach differs from much current AI research, but it has been persuasively supported. Woods (1986) argues generally that we should build machines that compensate for human weaknesses, not that duplicate human strengths. Others have argued this point specifically regarding statistical techniques:

"Rather than attempting to match human performance in tasks at which we excel, AI researchers ought to concentrate on *surpassing* human performance in tasks to which we are ill-adapted. Research should focus, not on imitating human cognition, but on exploring alternative styles of intelligence...statistical intelligence being a particularly promising candidate." (Schaffer 1988, p. 346, emphasis in original)

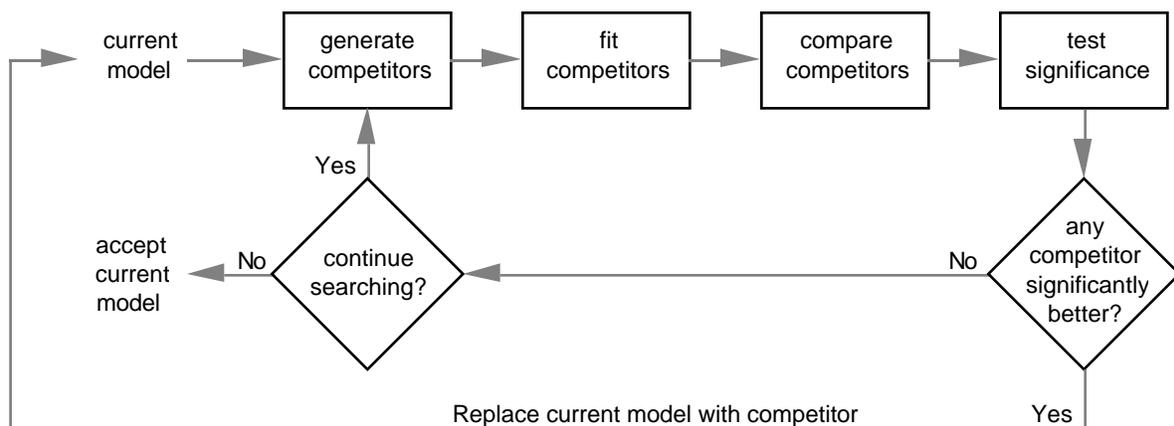
## A new tool

Below is a description of a new technique — Induction with Randomization Testing (IRT). It is the result of two basic design principles: 1) complement human skills by allowing manual control where human skills are strong and automating where human skills are weak; 2) provide an environment that supports the entire process of knowledge discovery.

The first principle addresses the major problem with machine learning tools — they duplicate both the strengths and weaknesses of human investigators. The second principle addresses the major problem with statistical tools — they provide support that is too limited to be of use in the broader context of knowledge discovery.

### Abstract design

IRT embodies a view of induction as a four-phase process (shown in Figure 1). The process alters a current model by generating a group of new competitor models, fitting those competitor models to data, comparing the competitors to each other, and then testing the statistical significance of the competitors. The process is iterative — it can be repeated until no competitor can be found that is significantly better than the current model.



**Figure 1: IRT's inductive process**

*Generating* competitors creates one or more models with a different structure than the current model. Examples include creating decision trees with different attributes and structure or classification rules with different conditions and operators. Each of these competitors is a candidate to replace the current model.

*Fitting* sets the free parameters internal to each competitor model. Examples include setting the split points of decision trees or the condition values of classification rules.

*Comparing* competitors evaluates the accuracy of each competitor in relation to the other competitor models. This is a relative comparison among competitor models, such as comparing two or more decision trees or classification rules that have different structure.

*Testing* models evaluates the statistical significance of the competitors, asking "Does any of them perform substantially better than would be expected by chance?" This compares the competitors to external, absolute references.

The statistical significance and raw accuracy of the competitors provide information to the investigator on whether a competitor should replace the current rule. Based on this information, and prior beliefs, the investigator can choose to replace the current rule, to generate additional competitors, or to accept the current rule.

The process uses two disjoint samples of data. One sample is used for fitting models; The other sample is used for comparing and testing models. Two samples are used to assure accuracy — if models were fitted and compared based on the same sample, the comparison phase would favor models with more free parameters, because of their greater ability to conform to any particular sample.

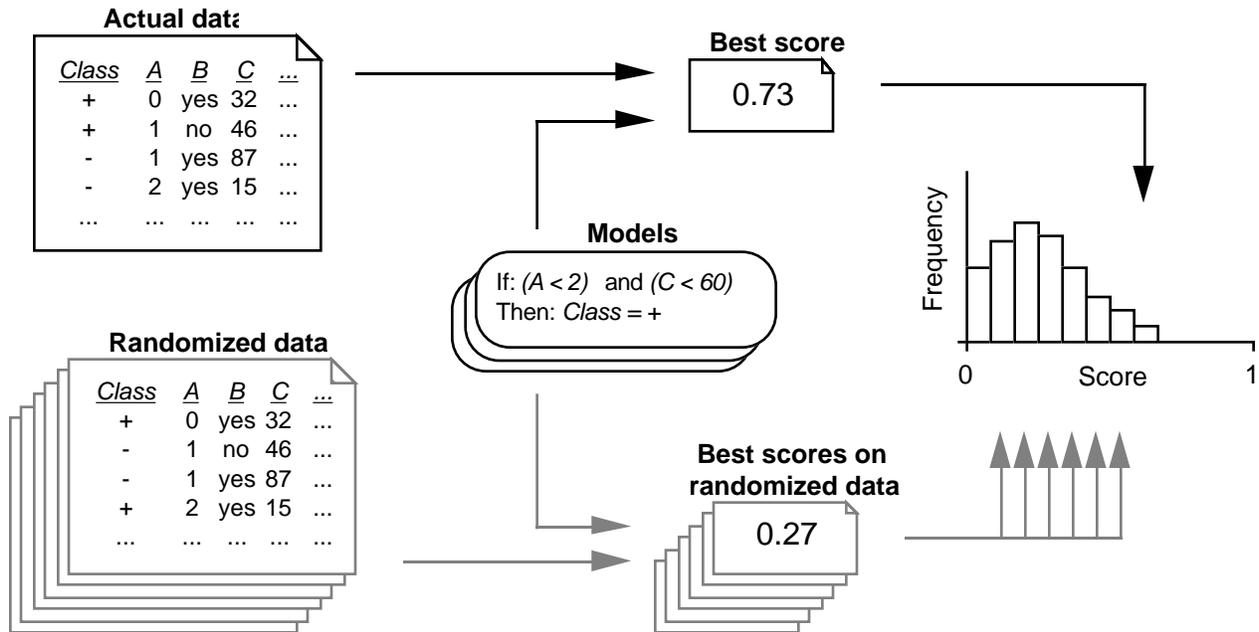
#### *Randomization testing*

IRT tests models using randomization testing (Edgington 1980; Jensen 1991). Randomization testing generates a distribution that can be used to determine whether the score of the best of the competitors is significantly better than that of the current model. The distribution is generated by creating and testing large numbers of randomized data sets. The distribution reflects the scores that could be expected by chance alone.

For instance, consider applying randomization testing in the problem of binary classification (Figure 2). The data consist of a set of examples, each with a label indicating its class (*Class*) and values for one or more attributes (*A, B, C, etc.*). Several competitors are tested on the data and the classification ability of each competitor is scored using an evaluation function. To test whether the the best competitor performs significantly better than the current model, we need to compare the score of the best competitor to the distribution of scores that would be expected by chance alone. As noted earlier, conventional statistical approaches to deriving this distribution are inappropriate because multiple, correlated models are tested.

However, randomization testing can be used to create an accurate distribution. Randomization testing creates randomized data sets; Each randomized set is a copy of the actual data, but with class labels that have been reassigned. This reassignment is done in such a way that the current model retains its classification accuracy, but so that any remaining consistent relationship between

the attributes and the class is destroyed. On randomized data, the current model will misclassify different cases and yet still maintain its overall classification accuracy. If any of the competitor models classifies randomized data more accurately than the current model, it will be due to chance alone.



**Figure 2: Randomization testing process**

Creating randomized data is best explained with an example. Suppose we have a data set consisting of eight examples. For each example, the current model predicts a class of either + or - and the actual class of the example is either + or -. The relationship between the actual and predicted class labels can be characterized by a four-cell contingency table. Given the example contingency table (Table 1), there are 15 possible class labellings (besides the actual class labelling) that will produce exactly the same table (shown in Table 2). Each randomized data set consists of the actual data with a randomized class labelling substituted for the actual class labelling.

**Table 1: Model prediction vs. actual classification**

Predicted class	Actual class	
	+	-
+	3	1
-	1	3

**Table 2: Randomized classifications**

Predicted labelling	Actual class labelling	Possible randomized class labellings															
+	+	+	+	+	+	+	+	+	+	+	+	+	-	-	-	-	
+	+	+	+	+	+	+	+	+	-	-	-	-	+	+	+	+	
+	+	+	+	+	-	-	-	-	+	+	+	+	+	+	+	+	
+	-	-	-	-	+	+	+	+	+	+	+	+	+	+	+	+	
-	+	-	-	-	+	-	-	-	+	-	-	-	+	-	-	-	
-	-	+	-	-	+	-	-	-	+	-	-	-	-	+	-	-	
-	-	-	+	-	-	-	+	-	-	-	+	-	-	-	+	-	
-	-	-	-	+	-	-	-	+	-	-	-	+	-	-	-	+	

Randomization testing creates a large number of randomized data sets (e.g. 100). For each one, the procedure substitutes it for the actual data, evaluates all of the competitor models, and chooses the model with the best score. The set of best scores on the randomized data forms a distribution reflecting the scores that could be expected by chance alone. The score of the competitor most accurately predicting the actual data can be compared to this distribution; The percentage of the distribution falling below the actual score estimates the probability that the actual score is better than would be expected by chance alone.

Randomization testing is not a new idea in statistics. Edgington (1980) argues for randomization testing as a way to perform conventional statistical tests while avoiding assumptions about the population from which the data were drawn. The basic idea of randomization testing is at least 50 years old, and forms the foundation of Fisher's exact test (Fisher 1935). However, randomization testing has not been used to test multiple models, nor to test whether a new model represents significant improvement over an existing model. Finally, the technique is unused in machine learning.

When combined with mechanisms for generating, fitting, and comparing models, randomization testing assists the iterative construction of models by preventing the overfitting problems that can plague existing machine learning tools. To accomplish this, randomization testing accurately tests statistical significance in the face of extensive search — something conventional statistical tools cannot do.

*Implementation*

IRT has been implemented in Scheme running on Macintosh II microcomputers. The particular implementation discussed here aids the creation of simple binary classification rules. IRT appears to extend easily to a variety of other tasks such as non-binary classification and function fitting, but binary classification was chosen as an initial demonstration.

The classification rules employed by this implementation define *polymorphic* concepts (Hanson and Bauer 1989). They specify that at least  $m$  of  $n$  conditions must be true, where  $n = (total\ number\ of\ conditions)$  and  $0 < m \leq n$ . Thus, a rule might specify:

If  $m$  or more of the following are true:

(*condition 1*)

(*condition 2*)

(*condition 3*)

...

(*condition n*)

Then  $class = label$

In practice, the specification of the class label is not necessary, given that the intent is to construct *binary* classification rules. The rules are displayed in a spreadsheet-like format and can be selected, deleted, and sorted by various criteria. Groups of competitor rules can be generated by adding, substituting, or removing conditions; individual rules can also be edited. Several options are provided for rule fitting. Finally, competitor rules can be tested with randomization testing.

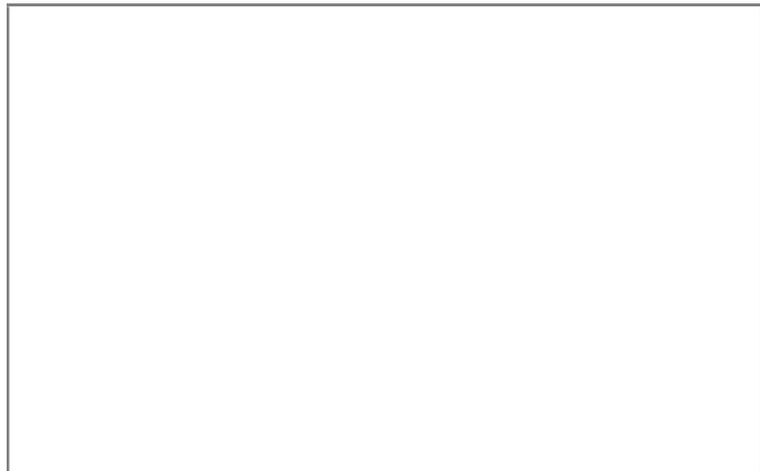
### Applying IRT

Perhaps the best way to explain the operation of the program is to provide an example of IRT in action. Below is a portion of a session analyzing Fisher's iris data. The data are a classic in the classification literature and consist of 150 examples, each with four attributes and a tertiary classification. For demonstration purposes, two of the class labels have been combined to create a binary classification.

#### Example

Analysis begins with a current rule. The current rule can be a *null* rule that indicates no prior knowledge (always predict the most prevalent class) or a rule that is thought, *a priori*, to be useful.

Figure 3 shows a one-condition current rule. The rule has already been evaluated on the data samples for fitting and testing. Column headings at the top of the window identify the various elements displayed for the rule. The scores on the fitting and testing data vary between zero and one with higher scores indicating more accurate classification (the scoring function is the chi-square statistic divided by the number of examples). A thin line separates the current rule and the competitor rules (no competitors currently exist for the window shown in Figure 3).



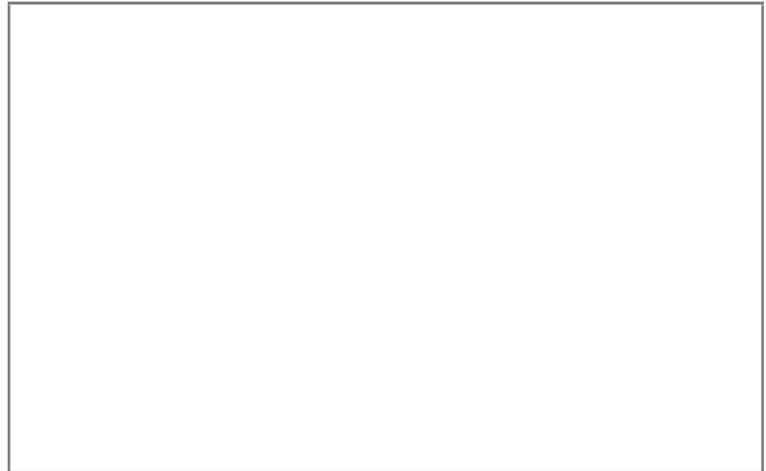
**Figure 3: Current rule**

Next, competitor rules are generated. The current rule is selected and serves as the base from which competitors will be created. As shown in Figure 4, conditions can be added, substituted, or removed. Specific attributes, lists of attributes, and conditions can be selected as is appropriate to the method of generating competitors. By repeated execution of this command, the investigator can completely control which competitors are generated.



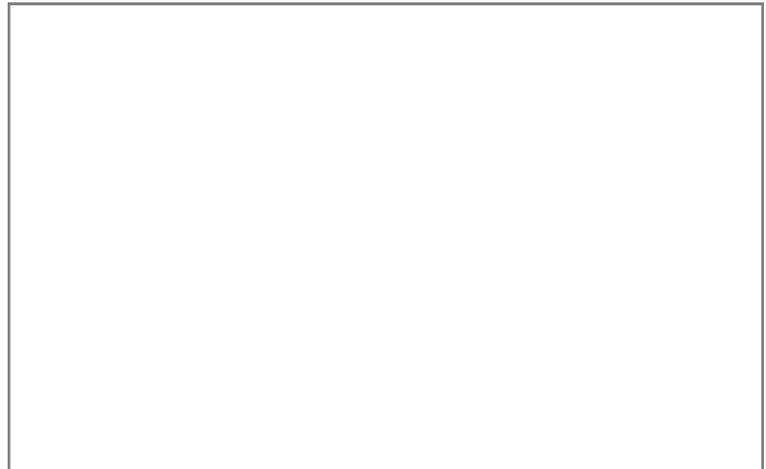
**Figure 4: Competitor generation**

In this case, conditions were added to the current rule. All possible second conditions were added, resulting in three competitors (a condition cannot appear twice in the same rule). Figure 5 shows the competitors. Each competitor is both unfitted and untested. The parameter values of the newly added conditions are set to default values. At any time, these values can be set by the investigator.



**Figure 5: New competitors**

Next, the competitors are fitted. All the competitors are selected and a fitting method is chosen as shown in Figure 6. Fitting methods differ in how many different parameter settings are evaluated. If desired, different sets of competitors can be fitted with different methods. An investigator can also set parameter values manually, based on domain knowledge.



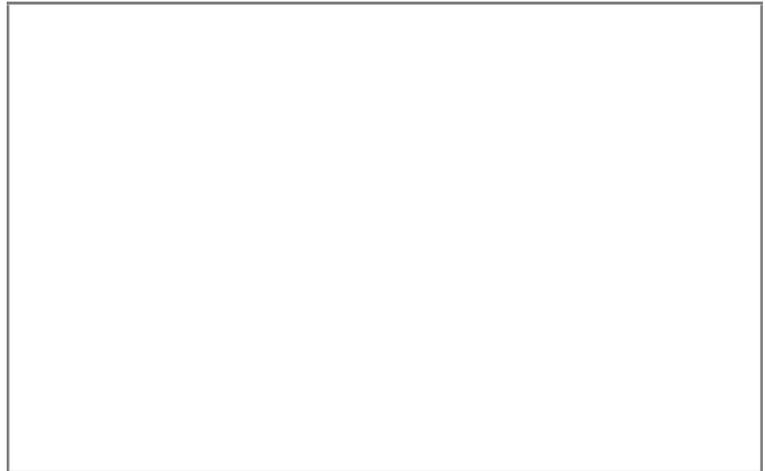
**Figure 6: Competitor fitting**

In this case, exhaustive search was used. Figure 7 shows the results of fitting each competitor. Fitting has affected the cutoff values of continuous attributes (e.g. 17.5 and 30.5), the operators (e.g.  $>$  and  $\leq$ ), and the junctions (e.g. "one of" and "all of"). Based on the fitting data, all the competitors appear to be improvements on the current rule. However, these results could just be capitalizing on chance variation.



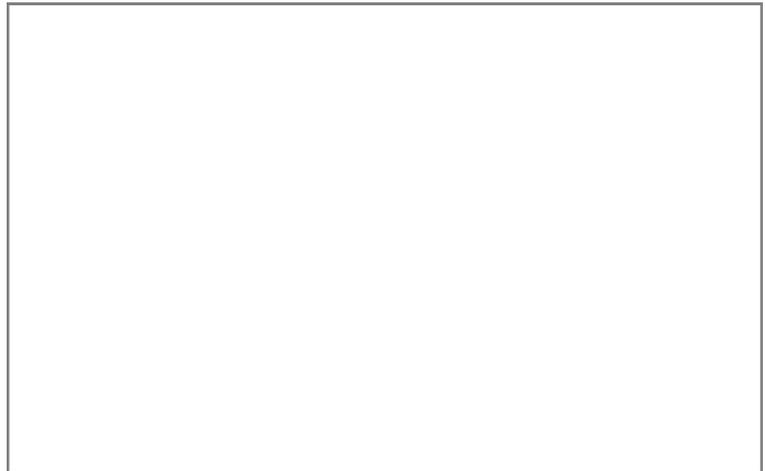
**Figure 7: Fitted competitors**

Next, the competitors are tested. Testing produces both a score on the test data (analogous to the score on the fitting data) and a probability value. The probability value estimates the probability that the competitor is significantly more accurate than the current rule on the testing data. All three of the competitors are highly significant and are candidates for becoming the new current rule. The final decision regarding replacement of the current rule is left to the investigator.



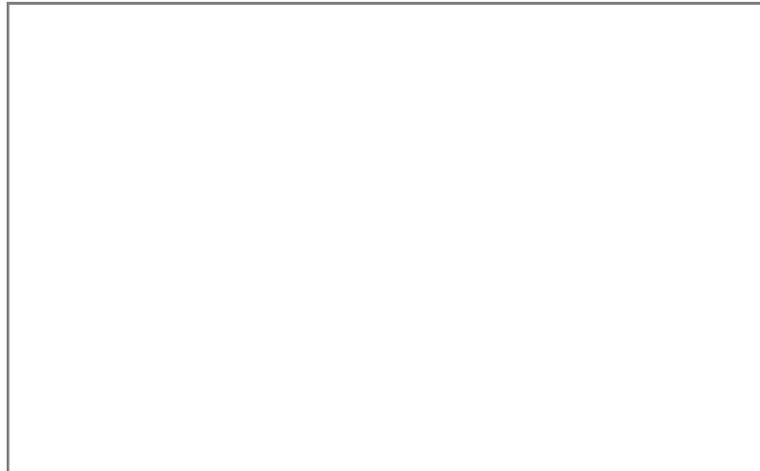
**Figure 8: Tested competitors**

The first competitor might be preferred because it has a higher score on the testing data. It can become the new current rule, as shown in Figure 8. Upon becoming the current rule, a rule must be retested so that a set of randomized data sets can be created that are appropriate. Figure 9 represents a similar situation to Figure 3, and the entire process can now be repeated — this time with a new current rule and the additional knowledge that it represents.



**Figure 9: New current rule**

Figure 10 shows the results of resampling the data and testing both possible three-condition competitors. While the competitors more accurately classify the testing data, the probability value of 0.71 indicates that an improvement this large would be expected by chance nearly 30% of the time. Given no other information, it would appear that a third condition is not warranted.



**Figure 10: Next iteration**

In addition to testing whether additional complexity is warranted (i.e. is statistically significant), the IRT can be used to test whether simpler rules are more accurate than would be expected if all the conditions were truly important. Thus, IRT can be used to explore a space of rules that is both more and less complex than the current rule. Throughout this exploration, automated assistance is provided to compensate for human weaknesses in significance testing.

#### *Application experience*

To date, IRT has been applied primarily to support a project on Alzheimer's screening at the Washington University School of Medicine. The implementation described here resulted from an extensive revision of an initial prototype. The prototype was substantially more automated, and proved to be too limiting for the type of analysis desired.

The current implementation produces classification rules with three to four conditions that perform quite well. The rules appear to reproduce a clinician's diagnosis with the same frequency as do other clinicians. The rules also outperform conventional multivariate statistical methods that were tried previously. Tests are currently underway to apply IRT to standard machine learning data sets. This will allow a broader comparison of IRT and conventional statistical and machine learning methods.

#### **Acknowledgements**

I wish to thank Dr. William Darby of the Department of Engineering & Policy for his continuing support and encouragement, Dr. Lee Robins of the Department of Psychiatry for her provision of interesting problems and demanding questions, Dr. Edward Spitznagel of the Department of Mathematics for originally alerting me to the problems of statistical significance testing in the face of extensive search, and Dr. William Ball of the Department of Computer Science for providing a stimulating and collegial environment to discuss my research.

## References

- Edgington, E. (1980). *Randomization Tests*. New York: Marcel Dekker, Inc.
- Einhorn, H. (1972). Alchemy in the behavioral sciences. *Public Opinion Quarterly*. 36:367-378.
- Fischhoff, B., Slovic, P., and Lichtenstein, S. (1981). Lay foibles and expert fables in judgments about risks. *Progress in Resource Management and Environmental Planning*. 3:161-202.
- Fisher, R. (1935). The logic of inductive inference. *Journal of the Royal Statistical Society, A*. 98:93-154.
- Hanson, S. and Bauer, M. (1989). Conceptual clustering, categorization, and polymorphy. *Machine Learning*. 3(4):343-372.
- Hart, A. (1987). The role of induction in knowledge elicitation. *Knowledge acquisition for Expert Systems*. A. Kidd (Ed.). New York: Plenum. 165-189.
- Haussler, D. (1987). Bias, version spaces and Valiant's learning framework. *Proceedings of the Fourth International Workshop on Machine Learning*. 324-336.
- Jensen, D. (1990). Concept reliability in machine learning. *Proceedings of the Second Midwest Artificial Intelligence and Cognitive Science Society Conference*. J. Dinsmore and T. Koschmann (Eds.). 132-136.
- Jensen, D. (1991). Randomization testing: A significance testing method for inductive learning. Poster presented at the Third International Workshop on Artificial Intelligence and Statistics. Ft. Lauderdale, FL. January 3-5. Submitted for publication in the proceedings of the Workshop.
- Michalski, R. (1983). A theory and methodology of inductive learning. In Michalski, R., Carbonell, J. and Mitchell, T. (Eds.) (1983). *Machine Learning: An Artificial Intelligence Approach*. Palo Alto, CA: Tioga. 83-134.
- Payne, J. and Dyer, J. (1975). Betting after the race is over: The perils of post hoc hypothesizing. *American Journal of Political Science*. 19(3):559-564.
- Pearl, J. (1978). On the connection between the complexity and credibility of inferred models. *International Journal of General Systems*. 4:255-264.
- Quinlan, J. (1988). Decision trees and multi-valued attributes. *Machine Intelligence 11*. J. Hayes, D. Michie, & J. Richards (Eds.). Oxford: Clarendon Press. 305-318.
- Schaffer, C. (1988). Statistical intelligence: A manifesto. *Proceedings of the First International Symposium on Artificial Intelligence*. 345-351.
- Woods, D. (1986). Cognitive technologies: The design of joint human-machine cognitive systems. *AI Magazine*. 6(4):86-92.