Generating Better Decision Trees

Steven W. Norton*
Siemens Corporate Research, Inc.
755 College Road East, Princeton, NJ 08540
swn@demon.siemens.com

Abstract

A new decision tree learning algorithm called IDX is described. More general than existing algorithms, IDX addresses issues of decision tree quality largely overlooked in the artificial intelligence and machine learning literature. Decision tree size, error rate, and expected classification cost are just a few of the quality measures it can exploit. Furthermore, decision trees of varying quality can be induced simply by adjusting the complexity of the algorithm. Quality should be addressed during decision tree construction since retrospective pruning of the tree, or of a derived rule set, may be unable to compensate for inferior splitting decisions. The complexity of the algorithm, the basis for the heuristic it embodies, and the results of three different sets of experiments are described.

1 Introduction

Incomplete knowledge is characteristic of interesting classifier learning problems. Part of the reason for developing connectionist models [Rumelhart *et* a/., 1986], genetic classifier systems [Holland, 1986], and Bayesian classifiers fTou and Gonzalez, 1974] is to contend with this incompleteness. Analytical learners use strong background knowledge to reason about training instances. Their problem is often one of finding the right formulation of existing knowledge. One analytical learning method can even deduce classifiers from single training instances [Mitchell *et al,* 1986]. Empirical learners, such as ID3 [Quinlan, 1983], CART [Breiman *et* a/., 1983], and IDX, do without extensive background knowledge. Instead, they try to recognize and exploit regularities in larger training sets.

Decision trees are a popular representation for classifiers. The interior nodes of a decision tree are tests applied to instances during classification. Branches from an interior node correspond to the possible test outcomes. Classification begins with the application of the root node test, its outcome determining the branch to a succeeding node. The process is recursively applied until a leaf node is reached. Then the instance is labeled with the class of the leaf node, and the process halts. During construction the decision tree naturally directs training data downward to the leaves. In a given leaf node, one of the classes can often be identified as incorporating more of the training instances than any other class. This typically becomes the class for that leaf node. Another common strategy is to choose the class that minimizes the average cost of misclassification. In my experiments the majority classification of the training data at each leaf has guided class selection.

Successful decision tree construction depends on the choice of good tests for the interior nodes. (It also depends on the decisions to stop splitting the training data and form leaf nodes, but this issue is not addressed here.) Figure 1 illustrates the point. Four types of instances arc found in the training population each having the given probability of each occurrence. Three binary tests can be used. Each tree classifies the data accurately. If the tests are pre-computed, the expected depth of the decision tree is the expected classification cost. Then trees 2 and 3 are better than tree 1. In contrast, if tests correspond to pieces of special purpose hardware, minimizing the number of tests in the classifier is important, as in tree 2. Other preference schemes are possible.

The next section describes several other research projects in decision tree induction. Each overlooks important issues and opportunities in reducing the cost of classification. Section three describes two lookahead algorithms: GOTA [Hartmann *et al.,* 1982] and IDX. Lookahead and a different heuristic function should help them outperform other algorithms. The fourth section presents the results of three sets of experiments comparing ID3 [Quinlan, 1983], IDX, and GOTA. Section five discusses the complexity of IDX, plus the positive and negative results of this research. Section six summarizes and concludes the paper.

2 Background

Hunt, Marin, and Stone [1966) report a series of experiments in the induction of decision trees. Their incremental, failure driven learners induce small binary decision trees (no more than 6 tests) from noiseless data.

* And Rutgers University, Department of Computer Science, New Brunswick, NJ 08903.

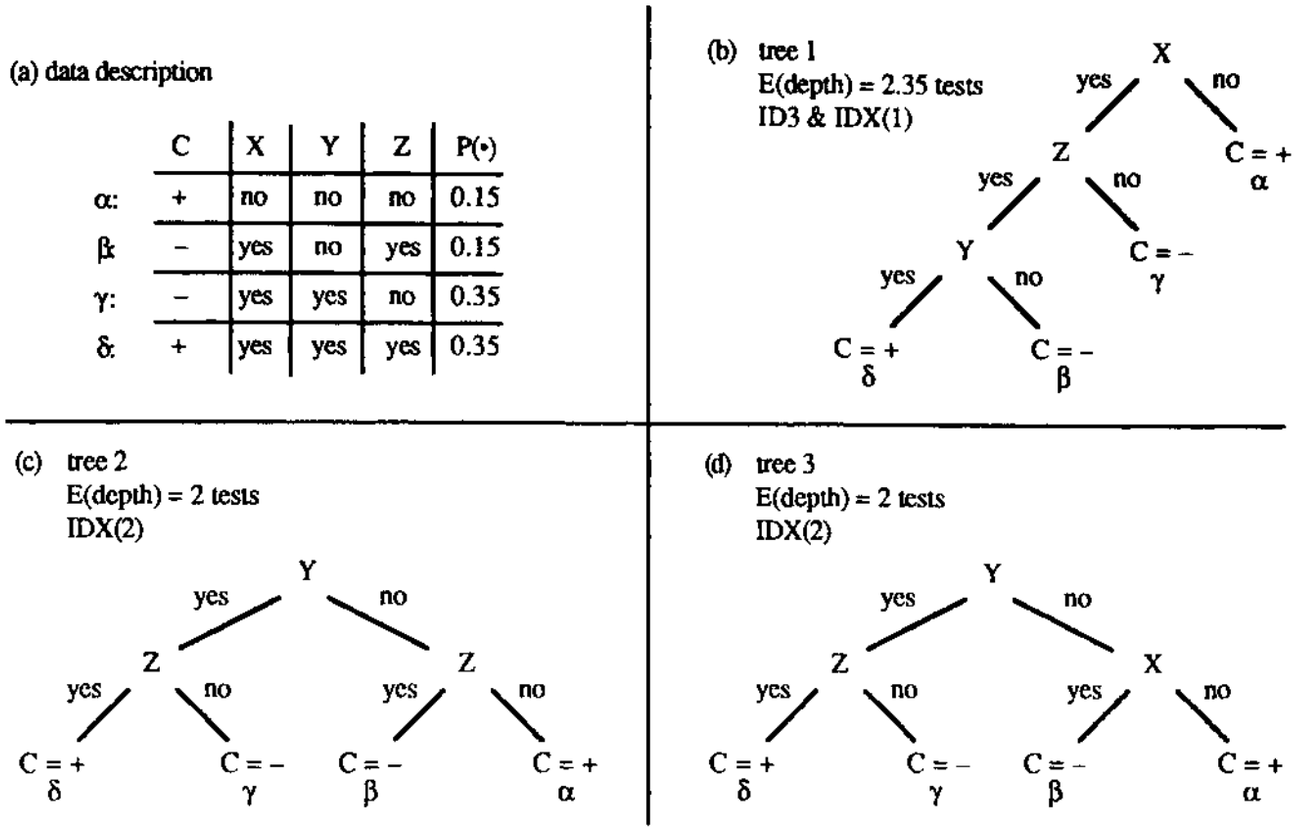

Figure 1. A Simple Example

They varied several components of their algorithm, but did not experiment with information theoretic heuristics, largely because of computational limitations of the day.

Quinlan uses an information theoretic test selection heuristic in ID3 [19831. A set of pre-classified training instances guides test selection, or splitting. Shannon's measure of information / [Gallager, 1968] measures the uncertainty in the classification of a sample given that it falls in a particular node. The entropy function *H* measures the average uncertainty in the classification given that an additional test is applied to split the node. At each step, ID3 chooses the test that maximizes the drop in classification uncertainty. This heuristic is meant to produce efficient decision trees as measured by average depth. ID3 generates decision tree 1 in Figure 1.

Breiman, Friedman, Olshen, and Stone [19831 present an extensive study on the induction of decision trees using CART. Classification costs are entailed in determining a classification; misclassification costs stem from incorrect classifications. A major concern of the CART team is reducing average misclassification cost. After generating large decision trees, they suggest pruning to reach a *right sized* tree. Quinlan's recent work on generating production rules from decision trees [1987] also uses pruning. Breiman *et al.* reach the tentative conclusion, "... that within a wide range of splitting criteria the properties of the final tree selected are surprisingly insensitive to the choice of splitting rule. The criterion used to prune or recombine upward is much more important." The degree to which this is valid depends on the setting for the finished application and on the nature of the costs involved.

3 Generating Better Decision Trees

Hartmann and his colleagues describe GOTA [1982], an algorithm that uses lookahead to maximize an information theoretic heuristic function. Control of splitting in GOTA is governed by branch levels, or horizontal slices of the decision tree. Within each branch level, GOTA does an exhaustive search to find the best test sequence as measured by the heuristic function. The key point for this discussion is that branch levels cover the decision tree completely and do not overlap. The depth of a test within a decision tree is the number of tests between it and the root. Consider the application of GOTA to a problem with four binary tests, using branch levels two tests deep. This would first entail an exhaustive search for the best combination of one depth-zero test and two depth-one tests. These would be fixed prior to processing in the next branch level. GOTA would then search for the best combination of four depth-two tests and eight depth-three tests.

GOTA's heuristic seeks to maximize the drop in uncertainty across the branch levels, much like ID3. But in addition, it seeks to minimize the cost of the decision tree. GOTA maximizes the function F given below. It is a function of the uncertainty H in the classification of the various branch levels BL, and the cost g of extending the decision tree from branch level i-1 to branch level i. Hartmann and his colleagues show that no disadvantage ensues from maximizing F over a single branch level rather than combining the results of maximizing over component branch levels. In fact better results are often obtained with the former approach. Their result is valid for cost functions satisfying g(a,c) < g(a,b) + g(b,c). Size, average cost, and performance all satisfy the inequality.

$$F(BL_{i-1}, BL_i) = \frac{H(BL_{i-1}) - H(BL_i)}{g(BL_{i-1}, BL_i)}$$

The F heuristic is frequently helpful in practice. The intuition behind the phenomenon involves the choice of tests at the bottom of the branch levels. Consider using GOTA in the four test case mentioned above. When using two branch levels two tests deep, the choice of the depth-one tests is based only on the choice of the depth-zero test. What if all four tests are chosen by processing a single branch level four tests deep? The choice of the depth-one tests is now based, in part, on the candidates for depth-two and depth-three tests. The information gained by the extra search can only help, not hurt.

But consider the eight-test case. If the largest branch level that can be processed is four tests deep, there will still be a problem at the border between levels. Whereas GOTA uses branch levels that cover the decision tree but do not overlap, IDX uses levels that do overlap. This allows IDX to be better informed at test selection time. There is a second difference. Processing of an entire branch level with GOTA results in the selection of tests throughout that branch level. In contrast, processing of a level in IDX results only in the selection of the shallowest tests in that level. Of course, nearly as many levels are required as the tree is deep. When using IDX and branch levels four tests deep on the present case, the choice of depth-zero test is made on the basis of an exhaustive search for the best configuration of tests from depth zero to depth three. Then the depth-one tests are chosen on the basis of an exhaustive search for the best configuration of tests from depth one to depth four. This continues until the lookahead reaches the leaves. The results of that last search are optimal, and therefore applied directly.

4 Experimental Results

This section describes decision tree building experiments using Jeffrey Schlimmer's Congressional voting record data (provided April 23, 1987) drawn from [Congressional Quarterly, 1985]. The data set contains 435 samples, one for each congressional representative. Each sample contains a representative's party affiliation (Republican or Democrat) and voting record on 16 key bills during 1984. For simplicity the categories of votes have been reduced to *Yes, No,* and *?* (other). 17 tests were considered during experimentation: party affiliation and votes. Each experiment excluded the test for which the decision tree was being built.

4.1 Decision Tree Size

The first experiment deals with decision tree size, measured as the expected value of the number of tests needed to classify the instances in the entire data set. Let IDX(n) stand for the use of IDX with n-test lookahead. ID3, IDX(l), IDX(2), and IDX(3) were used to generate decision trees for party affiliation, and for each tuple <vote, outcome>, using the entire data set. The cost function used in IDX measured the expected number of tests needed to extend the decision tree. Because of the volume of data, only the regression lines are plotted in Figure 2. Since each test was given unit cost, ID3 and IDX(l) performed identically. The performance of IDX(2) was slightly better on average, particularly as larger trees were needed to correctly classify the training data. IDX(3) did better still. Let GOTA(xi, x2, ...) stand for the use of a branch level including tests of depths 0 to x1 - 1, a branch level including tests of depths x1 to x2 - 1, and so on. GOTA(0,2,4, ...) and GOTA(l,3,5, ...), evens and odds, were applied using the same cost function as IDX. Figure 3 plots regression lines for the results of GOTA(evens) and GOTA(odds) with the results of IDX(2). The three algorithms were nearly equivalent. Each performed best, worst, or in between about as often as the others.

4.2 Robustness Tests

One effect of IDX is structural. It tends to balance decision trees. The second experiment was meant to determine whether generality and error rates of decision trees generated with IDX(2) differed systematically from those generated with ID3. For each vote, decision trees were generated for the *Yes* outcome using different fractions of the data for training: .01, .02, .04, .06, .08, .10, .15, .20, .30, .40, .50, .70, and .90 . The remaining data were used to compute false positive and false negative rates for the decision trees. For each vote and each training fraction, ten trials of this kind were performed and the results averaged. Error rates obtained using IDX(2) decision trees did not systematically vary from those obtained using ID3 decision trees.

4.3 Cost of Classification

The third set of experiments highlights the importance of recognizing cost measures other than size, and demonstrates the proficiency of IDX in doing so. For each tuple <vote, outcome> a decision tree was generated using ID3. For each tree, five experimental runs were conducted. In each run

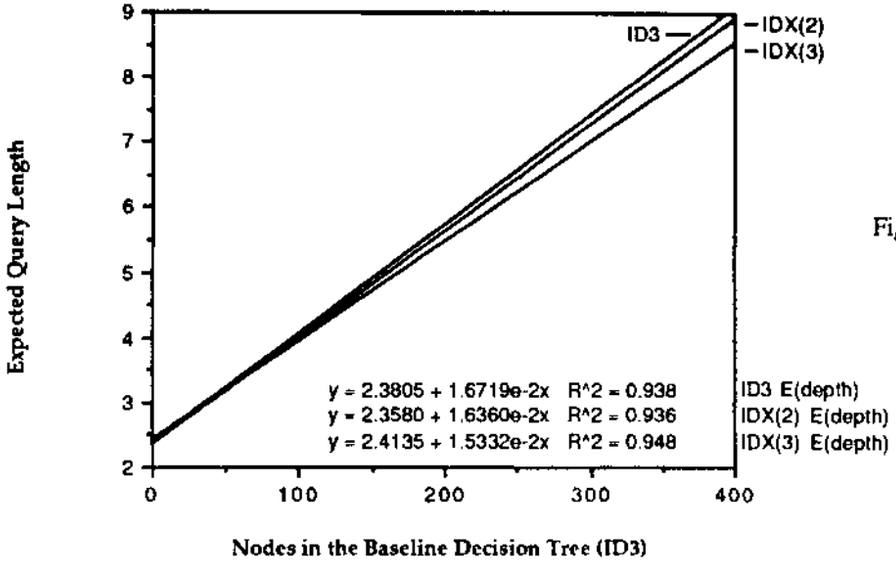

Figure 2. Decision Tree Size

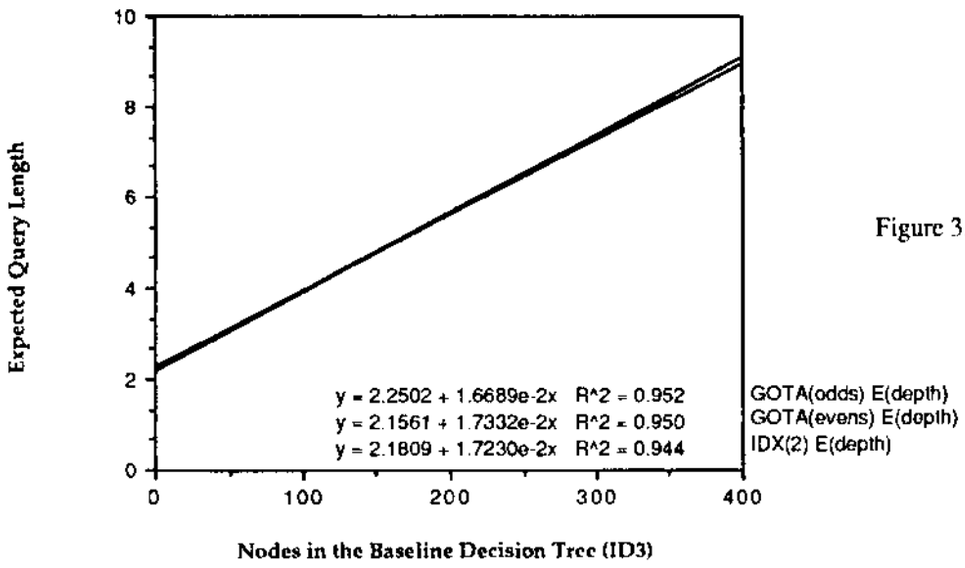

Figure 3. GOTA vs. IDX(2)

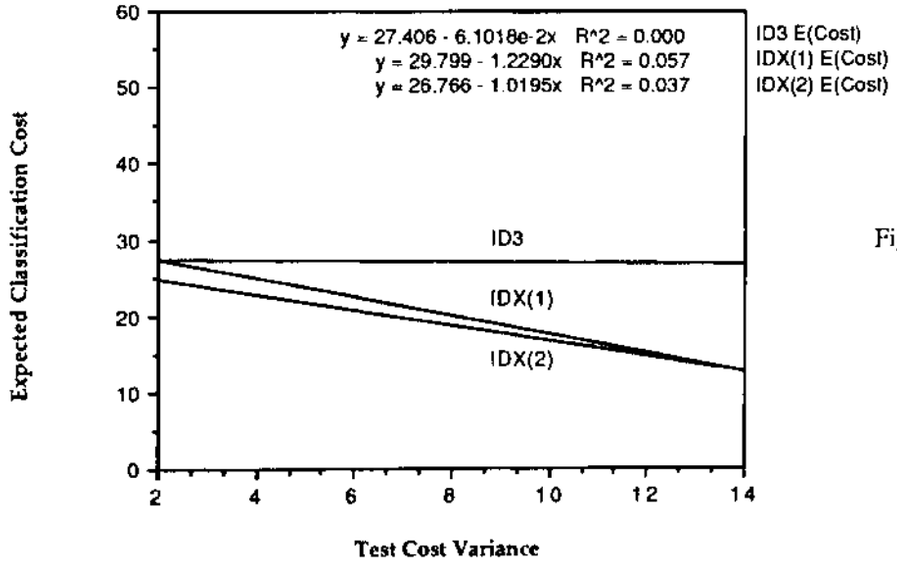

Figure 4. Decision Tree Cost

random real valued costs between 1 and 10 were assigned independently to each test For each set of costs, IDX(I) and IDX(2) were executed. The g function measured expected cost, not expected size. The expected classification costs for ID3, IDX(I), and IDX(2) are plotted versus test cost variance in Figure 4. Again because of the large volume of data, only the regression lines are shown.

5 Discussion

There was some uncertainty about how much of an effect lookahead would have on classification cost as measured by average decision tree depth. It is widely known that the heuristic used in ID3 is a good one for minimizing that measure. The experiments did, however, show a consistent improvement using lookahead. Regression suggests 1.44% and 1.76% improvements in decision tree depth on small (100 nodes) and large (300 nodes) decision trees when using two-level lookahead. Three-level lookahead demonstrated between 2.61% and 5.18% improvement. While these gains may not seem impressive, consider a high volume task such as optical character recognition for a postal application. Small improvements applied countless times can yield real benefits.

The results of the comparison of GOTA and IDX are somewhat disappointing, but they highlight something important about the heuristics in question. While enlarging levels is guaranteed not to reduce the heuristic measure, it does not imply the overall goal of minimizing classification cost. The limiting case is the obvious exception, when lookahead covers the entire tree. To its credit, IDX provides a more flexible tool in the search for efficient decision trees. Let IDX(xo, x1,...) mean that tests at depth zero are based on xo level lookahead, that tests at depth one are based on x1 level lookahead, and so on. GOTA(0, 3, 5) is certainly equivalent to IDX(1, 2, 1, 2, 1). What IDX does not provide is independent control of lookahead in disjoint subtrees within the same level. There may be circumstances where this will be important, and we will see that test cost variance-based heuristics are strong candidates.

In the third set of experiments, ID3 was expected to perform consistently despite test cost variance. After all, it does not recognize test costs in its heuristic. IDX was generally expected to outperform ID3 whenever test costs varied. In addition, the disparity between ID3 and the IDX algorithms was expected to increase as test cost variance increased. Variation in test costs gives IDX an opportunity to recognize when expensive tests can be delayed or avoided altogether. Of course, test cost variance is not the only factor influencing performance. Test relevance is at least as important. If too much of the contribution to the variance is from tests irrelevant to the classification, the variance effect will tend to be diminished. Unfortunately, test relevance is difficult to capture since the algorithm has some freedom in selecting the tests it uses. To compensate for this problem the experiment called for a larger number of trials. These expectations are born out in the regression lines of Figure 4.

The performance of IDX was also expected to improve in the presence of test cost variance as the lookahead parameter increased. This was bom out but with diminishing returns as shown in Figure 4. If an expensive test is essential for finding the correct classification, both algorithms will tend to push it to the bottom of the decision tree, trying to apply it less often. In the end, the necessary tests *will* be applied. Since IDX tends to balance decision trees, maximum depth does not change very much. Therefore, expensive but essential tests will be delayed but eventually applied at similar depth in either case. Non-essential tests are never used. This accounts for the effect of diminishing returns.

Computing optimal decision trees is known to be NP-complete fHyafil and Rivest, 1976]. Calculation of a loose upper bound on the number of times IDX partitions the entire data set is given below. Parameters: branch levels L tests deep; T available tests. The bound assumes that costs can be computed independently for disjoint subtrees, a favorable but typical case. The bound is linear in the size of the data set. If the number of tests is large, the cost of additional lookahead will be substantial. One area for future research involves significant reduction in the number of tests considered at a given stage during the exhaustive search. The good news is that as test cost variance increases the need for large L decreases.

$$\sum_{i=0}^{T-L} \prod_{j=0}^{L-1} (T-i-j) = O(T^{L+1})$$

The efficacy of lookahead with IDX has been established, but the question remains whether a less complex tree generation technique followed by some type of retrospective pruning could accomplish the same ends. By "retrospective pruning" I mean any technique in which a constructive stage is followed only by pruning and by no other tree modification. The answer depends on the criticality of cost in the final application. Breaking tree 1 of Figure 1 into rules in the obvious way yields the two candidate rule sets in Figure 5. Pruning does not eliminate the X tests in either case. Likewise, cost complexity pruning of the same tree, as done in [Breiman *et al.,* 1983], does not eliminate test X unless the cost parameter swamps the error rate term. Still, two-test lookahead suffices to eliminate the X lest as shown in tree 2. If reduction of classification costs has high priority in an application then lookahead will be important in the splitting criterion for decision tree construction. Such cases are easy to conceive. In building a medical expert system for diagnosis, blood work and imaging will generally be preferred to exploratory surgery although the surgery greatly reduces uncertainty. Invasive tests should be avoided as much as possible. Techniques such as IDX can be useful in this regard.

$$
\begin{array}{rcll}
(X = no) & \longrightarrow & (C = +) & ; \alpha \\
(X = yes) \,\&\, (Z = no) & \longrightarrow & (C = -) & ; \gamma \\
(X = yes) \,\&\, (Y = no) & \longrightarrow & (C = -) & ; \beta \\
(Z = yes) \,\&\, (Y = yes) & \longrightarrow & (C = +) & ; \delta
\end{array}
$$

$$
\begin{array}{rcll}
(X = no) & \longrightarrow & (C = +) & ; \alpha \\
(X = yes) \,\&\, (Z = no) & \longrightarrow & (C = -) & ; \gamma \\
(Z = yes) \,\&\, (Y = no) & \longrightarrow & (C = -) & ; \beta \\
(Z = yes) \,\&\, (Y = yes) & \longrightarrow & (C = +) & ; \delta
\end{array}
$$

Figure 5. Rule Sets Derived from Tree 1

6 Conclusions

A new decision tree learning algorithm is described that is more general than ID3 and GOTA. Using lookahead often permits IDX to make more informed decisions about test selection during decision tree construction. The complexity of the algorithm is largely controllable by the user. In practice, high quality decision trees are generated by using lookahead and by considering costs of classification. Depending on the demands of the classification problem, accepting higher cost during decision tree construction can yield better decision trees than schemes using simpler splitting criteria and retrospective pruning. An important negative result regarding variance of test costs describes circumstances under which additional lookahead becomes less and less warranted.

Acknowledgements

Thanks go to my colleagues Mike Hudak, Narendra Gupta, and Kangsuk Lee for reviewing earlier versions of this paper. Thanks also go to Kevin Kelly, and Neeraj Bhatnagar for acting as sounding boards for some of these ideas.

References

[Breiman *et al,* 1983] Breiman, L., Friedman, J.H., Olshen, R., & Stone, C. (1983). *Classification and Regression Trees.* Wadsworth & Brooks. Monterey, CA.

[Congressional Quarterly, 1985] *Congressional Quarterly Almanac, 98th Congress, 2nd Session 1984, Volume XL.* Congressional Quarterly Inc. Washington, D.C., 1985.

[Gallager, 1968] Gallager, R. (1968). *Information Theory and Reliable Communication.* John Wiley and Sons. New York, NY.

[Hartmann *et al,* 1982] Hartmann, C, Varshney, P., Mehrotra, K., & Gerberich, C. (1982). Application of information theory to the construction of efficient decision trees. IEEE Transactions on Decision Theory, IT-28,4: 565-577.

[Holland, 1986] Holland, J. (1986). Escaping Brittleness: The possibilities of general-purpose learning algorithms applied to parallel rule-based systems. In *Machine Learning: An Artificial Intelligence Approach, Volume* 2, edited by Michalski, Carbonell, & Mitchell. Morgan-Kaufmann. Los Altos, CA.

[Hunt *et al* 1966] Hunt, E., Marin, J., & Stone P. (1966). *Experiments in Induction.* Academic Press. New York, NY.

[Hyafil and Rivest, 1976] Hyafil, L., & Rivest, R. (1976). Constructing optimal binary decision trees is NP-complete. Information Processing Letters, Vol. 5, No. 1: 15-17.

[Mitchell *et al.,* 1986] Mitchell, T., Keller, R., & Kedar-Cabelli, S. (1986). Explanation-based generalization: A unifying view. Machine Learning, 1: 47-80.

[Quinlan, 1983] Quinlan, J.R. (1983). Learning efficient classification procedures and their application to chess end games. In *Machine Learning: An Artificial Intelligence Approach,* edited by Michalski, Carbonell, & Mitchell. Tioga Publishing. Palo Alto, CA.

IQuinlan, 1987] Quinlan, J.R. (1987). Generating production rules from decision trees. *Proceedings of the Tenth International Joint Conference on Artificial Intelligence.* Morgan Kaufmann Publishers.

[Rumelhart *et al.,* 1986] Rumelhart, D.E., McClelland, J.L., & the PDP Research Group (1986). *Parallel Distributed Processing: Explorations in the Microstructure of Cognition, Volume I.* MIT Press.

[Tou and Gonzalez, 1974] Tou, J.T., & Gonzalez, R.C. (1974). *Pattern Recognition Principles.* Addison-Wesley Publishing Company. Reading, MA.