DRAFT

Heuristic Search for Model Structure

John F. Elder IV, Rice University
Dept. Computational & Applied Mathematics, Box 1892, Houston Texas 77251
elder@rice.edu

Key Words: model selection, regression, decision trees, CART, TX2step, neural networks

Abstract[1]

Modern "machine learning" techniques can often discover useful patterns in high-dimensional data in a nearly automated fashion. These adaptive statistical algorithms -- including decision trees, regression networks, projection pursuit models, and additive network methods -- incrementally build up the model structure (inputs, interconnections, and terms) as well as set parameter values. That is, they sequentially add the component, from the set of candidates, which works best with the existing collection. This model expansion typically ceases when the structure is judged to optimally trade off 1) training accuracy and 2) simplicity (of model form or function surface). Complexity is regulated in order to protect against the serious danger of overfit, and thus lead to greater success on new data.

The heuristic search procedures employed by these methods are all "greedy" to some degree, as a consequence of making workable choices in an open or combinatorially huge domain of possible models. This procedure basically works well in practice, though it is known that, theoretically, greedy searches can result in arbitrarily worse models than optimal ones (when such are possible) on finite data sets. Some real and artificial examples of this will be shown for regression subset selection, decision trees, and artificial neural networks.[2]

It is useful then, to explore what restraining greed, or "lengthening the scoring horizon", can do for the methods. Toward this end, a recent decision tree algorithm, "Texas 2-Step", is described which looks two steps ahead to choose threshold variables and values, rather than one. (That is, it judges a split not by the purity of the resulting child nodes, but how the grandchildren turn out.) Preliminary results for some of these inductive methods are compared on a recent field application: identifying a bat's species by its chirps.

In practice, greedily-constructed models sometimes outperform more optimal ones when tested on new data. We conclude by making preliminary suggestions about when this inversion is most likely to occur, and outline ways to improve the robustness of inductive procedures.

[1]This work was partially supported by an NSF Research Associateship in Computational Science and Engineering.

[2]In the full paper (Lord willing)! The complete resultsis are not available at draft time.

1. Automated Induction

Inductive algorithms are, at one level, "black boxes" for developing classification, estimation, or control models from sample data. They automatically search a vast space of potential models for the best inputs, structure (terms and interconnections), and parameter values. The models are pieced together in a stepwise manner into a feed-forward network (e.g., tree) of simple nodes. The better methods also *prune* unnecessary terms or nodes from the model, thereby regulating complexity to reduce the chance of *overfit*. Overfit models are over-specialized to the training data and generalize poorly (fail on new data). This is widely held to be the chief danger of using inductive methods.

Complexity is regulated either through

1) *term penalties*, as with model selection criteria such as C_p (Mallows, 1973) and *Minimum Description Length*, MDL (Rissanen, 1978),
2) *roughness penalties* (integrated second derivatives of the estimation surface), or
3) *tests on withheld data* (e.g., *V*-fold cross-validation).

The penalties add to an error measure, and models having the lowest combined score are judged the best candidates for use.

Stepwise regression can be considered a low-level automated induction algorithm. Though the set of possible models (linear combinations of a subset of original candidate inputs) is quite constrained, the procedure does identify which variables to employ and can increase or reduce the size of the set under consideration.

In contrast, *Artificial Neural Networks* (ANNs) are not inductive methods by the definition used here, as their structure is fixed *a priori*.[3] They can more precisely be viewed as a class of nonlinear models whose parameters are typically set through a local gradient search called *back-propagation*.[4] (One suspects that ANNs, which can perform well even when they appear over-parameterized, may avoid overfit partly because of the weakness of this search algorithm! It is possible that improvement of the search procedure without simplification of the model structure may result in better training but worse out-of-sample performance.)[5]

Leading automated induction methods, using "building blocks" consisting of logistic functions, splines, polynomials, planes, non-parametric smoothes of weighted sums, etc. -- are briefly

[3]Removing small terms within ANN nodes does not address over-parameterization, where useless terms can appear significant though their coefficients collectively cancel. (The dangers of collinear variables in regression are analogous.)

[4]This iterative search converges relatively slowly to a local minimum in parameter space, and it has recently been shown (Mulier and Cherkassky, 1993) that the presentation order of the data affects the particular minimum found.

[5]If this danger is real, then the "greedy" nature of the gradient search may have benefits as well.

described in (Elder, 1993) along with their chief strengths and weaknesses. Here, we focus on one of the latter: greediness, and look briefly at its effect on regression and decision trees.

2. Subset Selection in Regression

Due to the combinatorial explosion of a trial-and-error search process (the methods are at least polynomial in the inputs and often exponential), a greedy heuristic is often employed: models are constructed in stages, and only the current step is optimized at a given time. *Forward selection* finds the single best term, then adds to it the term which works best with the first, then the one which best assists the pair, and so on. (Note that this is very much more useful than a "first impression" model, which ranks the candidate terms according to their individual performance and employs the top *K*.) *Reverse elimination* begins with a "full" model and sequentially removes the least useful term.

A combined method, *stepwise selection* (e.g., Draper and Smith, 1966) considers removing variables after each new variable is introduced. The standard selection mechanism, checking "F-to-enter" and "F-to-exit" significance values, is a kind of heuristic term penalty method, but not a correct use of F-tests. (The static significance measure is invalid in the dynamic modeling situation and can lead to highly inflated confidences in the resulting parameter values (Miller, 1990).

This greedy growth strategy makes the search feasible and often discovers useful features, but can miss "reachable" structure in the data; that is, within the form of the basis functions employed. For example, given $Y= \{1,1,1,1\}$, $X_1=\{1,1,1,0\}$, $X_2=\{1,1,0,0\}$, $X_3=\{0,0,1,1\}$, a stepwise procedure would first choose x_1 with which to estimate Y, and then seek to add another x. However, an exact model, $Y = x_2 + x_3$, would not include that single best input. Surprisingly, even if there is agreement between the forward and backward procedures on the best model of each size, they can differ by an arbitrarily large amount from some of the best subsets (Berk, 1978).

For example, Desroachers and Mohseni (1984) presented a purportedly optimal algorithm for model selection, and demonstrated it on a problem of estimating rocket engine temperature (from Lloyd and Lipow, 1962), where their small set results agreed with earlier analyses by Draper and Smith (1966). However, the approach turned out to be a version of forward selection. To compare these models with optimal subsets (of the candidate set defined by Desroachers and Mohseni), a new technique for term elimination had to be developed (Elder, 1990). Figure 1 shows the SSE of the greedy and optimal models of each size. The former leveled off at a limit of 40, while the latter were able to reach nearly the minimum error possible for the data (approximated by the *Y* axis base). Clearly, greedy methods can be improved upon significantly, in training, on real applications.

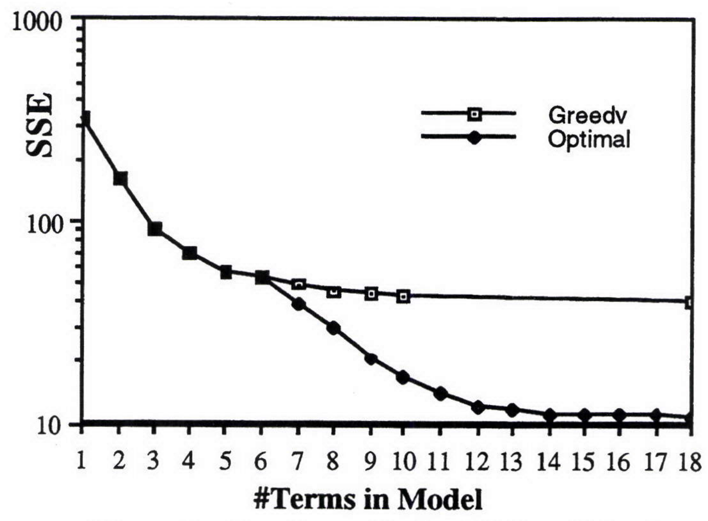

Figure 1: Greedy vs. Optimal Subset Selection

For regression model building, a logical extension of the greedy growth strategy (while stopping short of the hope of "optimal" models) is to add *chunks* of terms at a time, rather than just one. This is the heart the approach taken in GMDH-like techniques, such as ASPN (Algorithm for the Synthesis of Polynomial Networks, Elder, 1985). There, sets of several terms, employing a few independent variables, are considered for inclusion simultaneously, then pared down by reverse elimination. Nodes of such equations are built up until the added complexity cannot be justified, according to a penalty criterion -- either Predicted Squared Error (A. Barron, 1984) or MDL. An ASPN regression network, such as that shown in Figure 2, can have multiple layers of diverse nodes, each with several terms, resulting in a flexible compound function form.

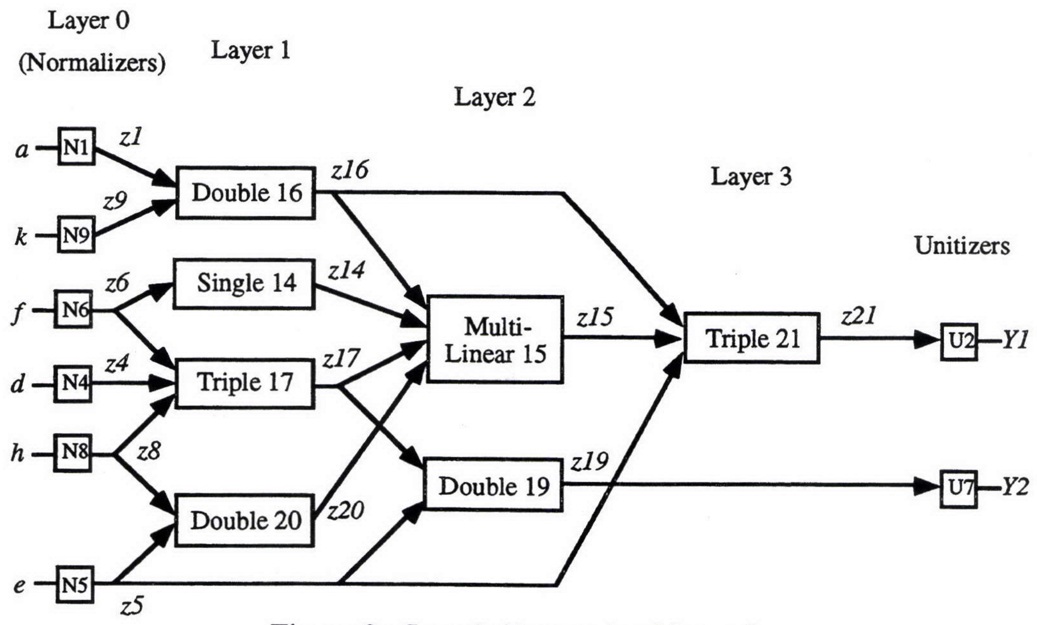

Figure 2: Sample Regression Network

Extensive comparison with more greedy algorithms has yet to be performed, but several researchers have successfully employed such regression networks on applications which had proven very difficult by other methods, including automatic pipe inspection (Mucciardi, 1982), fish stock classification (Prager, 1988), reconfigurable flight control (Elder and Barron, 1988), tactical weapon guidance (Barron and Abbott, 1988), and temperature distribution forecasting (Fulcher and Brown, 1991). Though several areas of possible improvement have been identified (Elder and Brown, 1994), its success suggests that taking complex, rather than simple, steps might improve other constructive algorithms for induction, such as those used to build decision trees.

3. Constructing Decision Trees

Though there are other and earlier decision tree algorithms (e.g., ID3 and CHAID), CART (Classification and Regression Trees, Breiman, Friedman, Olshen and Stone, 1984) is perhaps the best known and, arguably, most powerful. Some of its nicer features include built-in cross-validation, the ability to handle categorical variables and missing data, and a good presentation of the output. (Versions are also appearing which tie into commercial statistical packages and improve the interface.) Still, the basic classification algorithm is very simple: try to discriminate between classes by recursively bifurcating the data until the resulting groups are as pure as can be sustained. That is, start with all the training data and choose the univariate threshold split (e.g., $x3 < 1.14$) which divides the sample into two maximally pure parts (i.e., minimizes the sample variance of the sum). (Multi-linear splits (e.g., $x1 + 2x2 < 3$) are possible, but do not seem to work well in practice, perhaps because of a poor internal search algorithm.) Then, continue with each of the parts (child nodes) until either no splits are possible, or the leaves (terminal nodes of the tree) are pure (represent only one class) or have some minimum size. Then, CART prunes back (simplifies) the tree, typically using cross-validation, to avoid overfit. This over-training followed by pruning was found by CART's authors to lead to better trees than under the competing method of trying to select the growth stopping point.

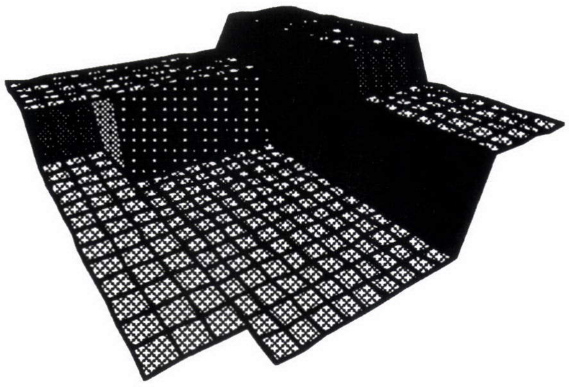
Figure 3: Example Decision Tree Surface

Table 1: Greedy Counter-Example for CART

Y	*a*	*b*	*c*
0	0	0	0
0	1	0	0
1	0	0	1
1	1	0	1
1	0	1	0
1	1	1	0
0	1	1	1
0	1	1	1

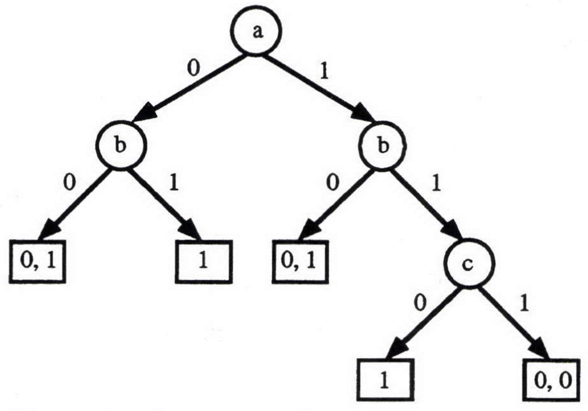

Figure 4a: Inaccurate Greedy Decision Tree

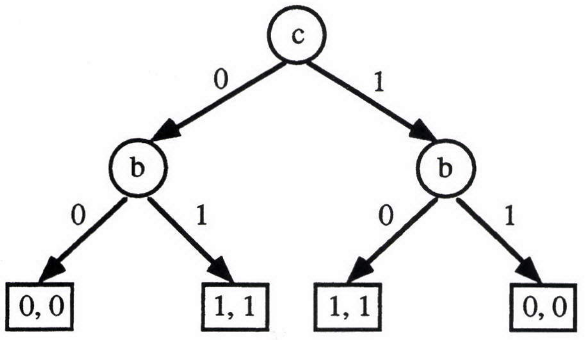

Figure 4b: Correct Decision Tree

For estimation, the leaves are set to the mean or median value of the cases contained, forming a piecewise-constant surface, as shown in Figure 3 for a 4-node tree.

This simple splitting approach is nevertheless powerful, as a sequence of threshold questions quickly conditions an individual case. Each path down the tree can have its own important variables and outliers have no special influence. Also, as with other methods which implicitly select variables, a user can feel free to try more candidates than otherwise, since CART will sift through them unfettered by concerns about multicollinearity, which can hurt regression methods. However, if the candidate variables are jointly useful, relatively independent, and not beset by many outliers, other methods of discrimination can outperform CART.

Here, we wonder simply if CART's strategy of choosing the greedy split cannot be improved. As a motivating example, consider the XOR-like data of Table 1. CART forms the approximation tree of Figure 4a (using a leaf size limit of ≤ 2 cases). Its greedy search does not find the simpler, exact tree of Figure 4b.

To explore whether an extension of the horizon to two steps ahead would be beneficial, a decision tree algorithm called "Texas Two-Step" was written.

4 Texas Two-Step (TX2step)

The algorithm TX2step is a slimmed-down version of CART for classification which is not able to handle missing data, perform internal cross-validation, set misclassification costs, or adjust priors, and so on. Yet it can look two steps ahead to choose the current split, and thereby finds the tree of Figure 4b given the data of Table 1. TX2step has one other new feature:: given more than one split which results in the same score, it uses the split with the *largest relative gap* between border training cases. That is, the tie-breaker to choose the dimension *d* of the split depends on

$$\text{gap}[d] = 0.5\frac{\min \text{Right } Xd - \max \text{Left } Xd}{\max Xd - \min Xd}$$

The algorithm can optionally be greedy as well; in that mode, and ignoring gaps, it was validated on several test problems to reproduce the same tree as CART without cross-validation. Therefore, to focus solely on the greediness issue, TX2step-1 (with gap measurement) was actually run in place of CART on the example application shown next. Training was performed until all nodes were pure, but those leaves with a majority class having <3 cases were pruned back (i.e., re-absorbed into their parent node).

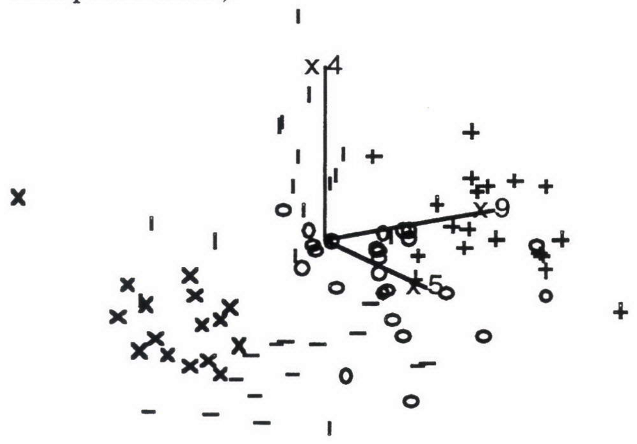

Figure 5: Example Projection of Bat Classes

5 Example: Identifying Bat Species

Researchers from the University of Illinois, Urbana/ Champaign[6] have measured bat echolocation calls and extracted time-frequency features from the signals, toward developing an automated classifying system to track species of bats -- especially those considered endangered. After visualization of projections of the data by the author, and analysis of correlations, multicollinearity, redundancy, and outliers (for suggested techniques see e.g., Elder, 1993), some variables were eliminated and other new ones tried at UIUC, resulting in a database of 93 cases, each with 15 candidate input features, representing 5 different species (classes) of bats.[7,8] One of the better projections of the data is shown in Figure 5, where the classes are noted by different symbols. Note that the groups do tend to cluster but that a fair amount of overlap is evident in this (and all low-d) views.

[6]Biologists Ken White, Curtis Condon, and Al Feng, and Electrical Engineers Oliver Kaefer and Doug Jones.

[7]A single bat from a sixth "Long-Eared" species contributed 5 signals originally, but was removed since it could be easily distinguished by its low-frequency signals and since having only one representative did not allow proper evaluation testing.

[8]It takes less than a second on a SPARC-2 to run the 1-step algorithm on this problem, but about 75 seconds for 2 steps.

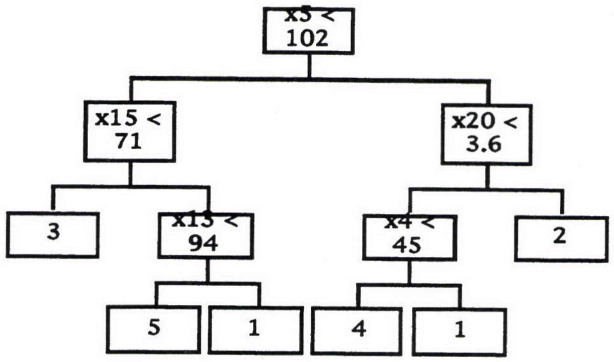

Figure 6: CART (1-step) Tree (using all data)

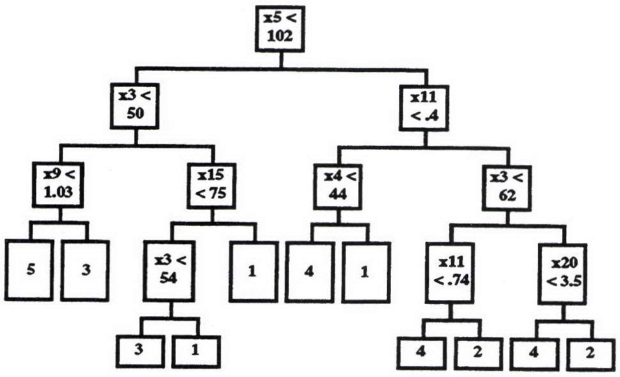

Figure 7: TX2step Tree (using all data)

Trained on all the data, the 1-step tree, shown in Figure 6, had 5 splits (17 prior to pruning) and made 13 training errors. (In the trees, "Yes" answers travel to the left child; "No" to the right.) The 2-step tree of Figure 7 started out simpler, with 14 splits, but pruned less, ending with 10 splits and only 5 training errors. The best root node split happened to be greedy but several other splits were not. For example, the data in the right child node of the root, shown in Figures 8 and 9, are those 58 of 93 cases where $x5 > 101.5$. The greedy tree was drawn to split first on $x20 < 3.59$, then on $x4 < 44.5$, and it missed 6 cases on that branch. The 2-step tree instead first chose $x11 < 0.39$ -- a seemingly worse split, but when followed by $x4 < 43.5$ on one branch, one which allowed it to correctly classify 4 more cases. (The difficulties the split caused its sibling branch were cleared up by subsequent splits.) The 2-step cuts were often more appealing visually; that is, they accorded more with what an analyst would do when viewing two dimensions of data simultaneously, rather than one.

As expected, the less greedy algorithm performed better on training data. The best test, of course, involves new data. Since there were not many cases, a cross-validation evaluation was performed, where all 3-8 signals for each bat, in turn, were held out of training and independently run down the tree for testing (18 runs for each method). Tables 2-4 show the resulting *confusion matrices* for CART, TX2step, and a neural network (courtesy of Oliver Kaefer and Doug Jones of

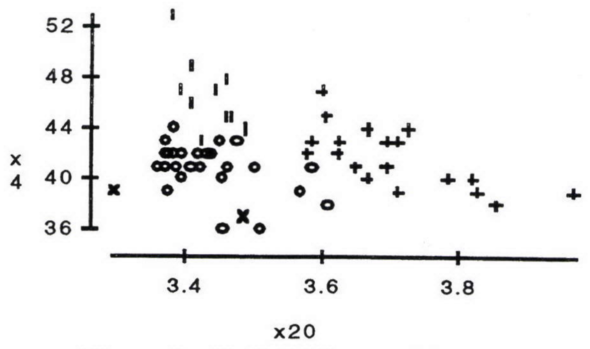

Figure 8: CART View at Right Node

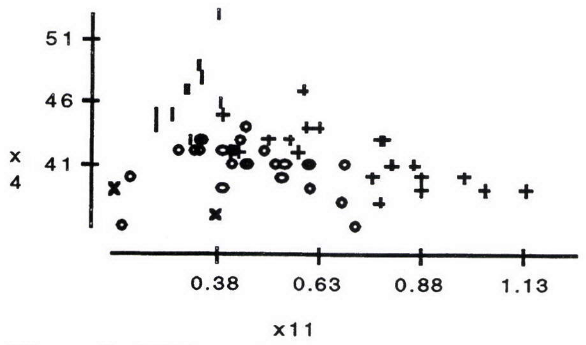

Figure 9: TX2step View at Right Node

UIUC) trained on the variables selected by the two tree methods. Correct classifications are along the diagonal and the hit percentage is shown in the corner.

CART gets 43 of 93 signals correct (46%), TX2step 54 (58%), and the ANN performs best with 64 (69%). The difference in accuracy for the tree methods appears more critical when using a *voting scheme*, where several different signals from a single bat are classified and the majority class is assigned. Then, CART misses 11 of the 18 bats but TX2step only 6. (The voting ANN misses just 4.)

In this experiment (counter to our usual experience), the tree methods were outperformed by an ANN. However, the variable selection performed by CART and TX2step proved helpful to the ANN; one trained on all 35 original data features got only 52% correct in bat-wise cross-validation, and one trained on 17 variables (those given as candidates to the tree methods) was 63% correct. Here, simpler ANNs performed better on new data. Clearly, an inductive ANN algorithm, which adapts the network structure to the data, would be a useful tool. The data characteristics -- filtered features, lack of outliers, clustered classes -- which helped the neural network perform well, should also be agreeable to exemplar-based statistical techniques, such as *kernels* and *nearest neighbors*. (We hope to soon try them, as well as regression networks and other inductive methods.)

Confusion Matrices

Table 2: CART

		True Class					
		1	*2*	*3*	*4*	*5*	*Tot*
P C	*1*	**7**	2	2	4	3	18
R L	*2*		**16**	4	2		22
E A	*3*	2		**2**		6	10
D S	*4*	5	1		**13**	2	21
. S	*5*	4		6	7	**5**	22
	Tot	18	19	14	26	16	**46%**

Table 3: TX2step

		True Class					
		1	*2*	*3*	*4*	*5*	*Tot*
P C	*1*	**7**		1	7	2	17
R L	*2*		**15**		3		18
E A	*3*	3		**9**		5	17
D S	*4*	6	4		**16**	2	28
. S	*5*	2		4		**7**	13
	Tot	18	19	14	26	16	**58%**

Table 4: 8-Input Neural Network

		True Class					
		1	*2*	*3*	*4*	*5*	*Tot*
P C	*1*	**6**	1	1	5	1	14
R L	*2*	2	**16**	1	3		21
E A	*3*	2		**11**		2	19
D S	*4*	5	2		**18**		25
. S	*5*	3		1		**13**	14
	Tot	18	19	14	26	16	**69%**

6 Performance on New Data: Remarks

We have seen that regression subsets and decision trees can be sub-optimal if the single best step is always taken. This is true in other venues as well. Cover (1974) showed an investigation in which greed hurts, where: If only one experiment is allowed, E_1 provides the most information, but if two are possible, then independent versions of the "worse" experiment E_2 are better.

But the degree to which greediness generally hurts performance in practice, on new data, is an open question. Berk (1978) sounded a slightly cautionary note in the case of regression subset selection. Using nine well-studied data sets (having from 4 to 15 predictors, 13 to 541 cases, and often more analysts!), he noted the *maximum* training error difference between all-subsets (optimal) models and both 1) forward selection and 2) reverse elimination models. An improvement of up to 29% in SSE was observed. Then, the sample distributions of each data set were employed to generate synthetic data with known population characteristics, and the study again performed for this new evaluation data. Figure 10 plots the training vs. evaluation data differences for the forward and reverse models from the (Berk, 1978) study. Most evaluation differences were smaller and in a tighter range (-2 to 7%, with one exception). In two cases, a greedy method won on the evaluation data by a slight margin.

Note that the differences are somewhat exaggerated, as the maximum disagreement between methods is shown, not that at some automated stopping point. For instance, the two worst reverse values (one training, one evaluation), are for models of size 1 and 2 -- where the forward method would clearly be preferable. Still, the greedy training and evaluation under-performances are correlated, and it can tentatively be concluded that regression differences on new data, while usually less dramatic than on training data, are still likely to be significant.

This was also shown to be the case for decision trees, where a version of CART was out-performed on an example problem by TX2step, which looks ahead an additional step when selecting a threshold for the current node. Further research is planned to examine the effects of greedy model construction strategies in these and other inductive methods, with the hope of understanding better the trade-offs between complexity (in the algorithm as well as model) and accuracy (training and evaluation).

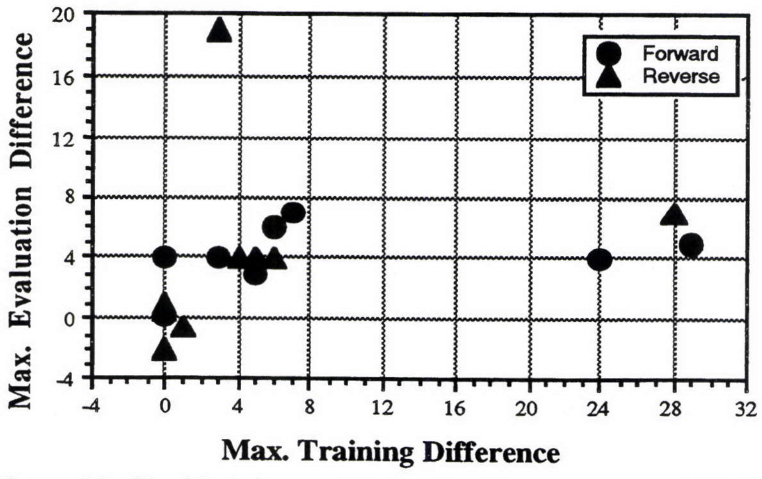

Figure 10: Ex. Training vs. Evaluation Improvement of Optimal over both Forward and Reverse Greedy Methods

References

Barron, A.R. (1984). Predicted Squared Error: A Criterion for Automatic Model Selection. Ch. 4 (Farlow, 1984)

Barron, R.L. and D. Abbott (1988). User of Polynomial Networks in Optimum, Real-time, Two-Point Boundary Value Guidance of Tactical Weapons, *Proc. Military Comp. Conf.*, Anaheim, CA, May 3-5.

Berk, K.N. (1978). Comparing Subset Regression Procedures, *Technometrics* **20**, no. 1: 1-6.

Breiman, L., J.H. Friedman, R.A. Olshen, and C.J. Stone (1984). *Classification and Regression Trees*. Wadsworth & Brooks, Pacific Grove, CA.

Cover, T.M. (1974). The Best Two Independent Measurements Are Not the Two Best. *IEEE Trans. Systems, Man & Cybernetics* **4**.

Desroachers, A. and S. Mohseni (1984). On Determining the Structure of a Non-Linear System, *International Journal of Control* **40**: 923-938.

Draper, N.R. and H. Smith (1966). *Applied Regression Analysis*. Wiley, New York.

Elder, J.F. IV (1985). *User's Manual: ASPN: Algori-thm for Synthesis of Polynomial Networks* (4th Ed., 1988). Barron Assoc. Inc., Stanardsville, VA.

Elder, J.F. IV (1990). Feature Elimination Using High-Order Correlation, *Proc. Aerospace Applications of Artificial Intelligence,* Dayton, OH, Oct 29-31:65-72.

Elder, J.F. IV (1993). Assisting Inductive Modeling through Visualization, *Proc. Joint Statistical Mtg.*, San Francisco, CA, Aug. 7-11.

Elder, J.F. IV and R.L. Barron (1988). Automated Design of Continuously-Adaptive Control: The "Super-Con-troller" Strategy for Reconfigurable Systems, *Proc. American Control Conf.*, Atlanta, GA, June 15-17.

Elder, J.F. IV, D.E. Brown (1994, to appear). Induction and Polynomial Networks, in *Advances in Control Networks and Large Scale Parallel Distributed Processing Models* Vol. 2. Ablex, Norwood, NJ (avail. from Univ. VA, Charlottesville, as IPC-TR-92-9).

Farlow, S.J. (1984), Ed. *Self-Organizing Methods in Modeling: GMDH Type Algorithms*. Marcel Dekker.

Fulcher, G.E. and D.E. Brown (1991). A Polynomial Network for Predicting Temperature Distributions, Institute for Parallel Computation Tech. Rpt. 91-008, Univ. VA, June.

Ivakhnenko, A.G. (1968). The Group Method of Data Handling -- A Rival of the Method of Stochastic Approximation, *Soviet Automatic Control* **3**.

Lloyd, D.K., and M. Lipow (1962). *Reliability: Manangement, Methods, and Mathematics*. Prentice Hall, Englewood Cliffs: 360.

Mallows, C.L. (1973). Some Comments on C_p, *Technometrics* **15**: 661-675.

Miller, A.J. (1990). *Subset Selection in Regression.* Chapman and Hall, NY.

Mucciardi, A.N. (1982). ALN 4000 Ultrasonic Pipe In-spection System. *Nondestructive Evaluation Program: Progress in 1981*, EPRI Rpt. NP-2088-SR, Jan.

Mulier, F., V. Cherkassky (1993). *Statistical Analysis of Self-Organization*, Dept. EE, Univ. Minnesota, Minneapolis, MN 55455.

Prager, M.H. (1988). Group Method of Data Handling: A New Method for Stock Identification. *Trans. American Fisheries Society* **117**: 290-296.

Rissanen, J. (1978). Modeling by Shortest Data Description, *Automatica* **14**: 465-471.