

Data Strip Mining for the Virtual Design of Pharmaceuticals with Neural Networks

Robert H. Kewley, Mark J. Embrechts, *Member, IEEE*, and Curt Breneman

Abstract—A novel neural network based technique, called “data strip mining” extracts predictive models from data sets which have a large number of potential inputs and comparatively few data points. This methodology uses neural network sensitivity analysis to determine which predictors are most significant in the problem. Neural network sensitivity analysis holds all but one input to a trained neural network constant while varying each input over its entire range to determine its effect on the output. The least sensitive variables are iteratively removed from the input set. For each iteration, model cross-validation uses multiple splits of training and validation data to determine an estimate of the model’s ability to predict the output for data points not used during training. Elimination of variables through neural network sensitivity analysis and predicting performance through model cross-validation allows the analyst to reduce the number of inputs and improve the model’s predictive ability at the same time. This paper illustrates this technique using a cartoon problem from classical physics. It then demonstrates its effectiveness on a pair of challenging problems from combinatorial chemistry with over 400 potential inputs each. For these data sets, model selection by neural sensitivity analysis outperformed other variable selection methods including forward selection and a genetic algorithm.

Index Terms—Computational chemistry, data mining, pharmaceuticals.

I. INTRODUCTION

THE PURPOSE of this paper is to introduce and illustrate a neural network methodology for solving problems with many more free parameters than data points to support them—“data strip mining problems.” Section II further defines this class of problems. Section III introduces an illustrative cartoon problem from classical physics as a sample data strip mining problem for this methodology to solve. Since a theoretical model for this problem already exists, it will serve as a benchmark to demonstrate how closely a neural model approximates the theoretical model. The solution depends on two subordinate methodologies. The first is neural network sensitivity analysis, described in Section IV. This process determines the most important variables in a system by successively holding all but one input constant and varying the other over its range of values to observe its effect on the system. It then eliminates a fraction of the least sensitive inputs from the system. At each step, cross-validation is used to estimate

the performance of the model with the remaining inputs. The model cross-validation technique explained in Section V first produces an estimate of the ideal training error at which to stop training the network before using it for prediction on data outside the training set. This ideal training error stop point, further described in Section V-C, is a new extension of the early stopping procedure to cross-validated models, which allows a network with many free parameters to train on a small data set and still give good prediction. The cross-validation methodology then divides the data into multiple training and validation sets in order to generate more observations of prediction error than one split would produce. These individual error predictions produce a correlation coefficient r between the predicted values and the actual values in the test set. Model comparison uses a $Q^2 = 1 - r^2$ statistic explained in Section V-B. Section VI proposes an automatic and repetitive process of sensitivity analysis, variable reduction, cross-validation, and model comparison which produces, using fully connected feed-forward neural networks trained by backpropagation [1], an estimation of the highest performing input set for the problem. This methodology produced good results on the sample problem from classical physics and on two challenging problems, described in Section VII, from combinatorial chemistry with over 400 inputs each and as few as 66 data points. It reduced the input variable sets to smaller subsets and produced neural network models with good predictive ability. In each case, the neural model produced a significantly smaller mean prediction error than a linear model for which forward selection [2] generated the input variables. In addition, the neural models selected using neural sensitivity analysis outperformed neural models whose inputs were selected using both linear forward selection and a genetic algorithm.

II. THE STANDARD DATA MINING PROBLEM AND “DATA STRIP MINING”

Data mining is the exploration and analysis, by automatic or semiautomatic means, of large quantities of data in order to discover meaningful patterns and rules [3]. This paper’s authors define the **standard data mining problem** as a multivariate regression or classification problem for which the analyst has many candidate inputs from which to choose. Furthermore, the analyst should not only build a model which predicts or classifies well, he or she should also be able to explain to some degree how and why the model works.

- 1) First, an input variable analysis should be performed to evaluate which inputs contribute most to the desired regression or classification model. The goal is to narrow

Manuscript received November 15, 1999; revised February 24, 2000.

R. H. Kewley is with the Department of Systems Engineering, United States Military Academy, West Point, NY 10996 USA (e-mail: fr6686@usma.edu).

M. J. Embrechts is with the Department of Decision Sciences and Engineering Systems, Rensselaer Polytechnic Institute, Troy, NY 12181 USA.

C. Breneman is with the Chemistry Department, Rensselaer Polytechnic Institute, Troy, NY 12181 USA.

Publisher Item Identifier S 1045-9227(00)03709-7.

a large number of potential inputs into a smaller subset which offers good prediction. This step is very important because excess variables in the model cause two problems. In many models, excess variables cause the parameter estimation methodology to fit the model to excess data which actually has no predictive value (over-fitting). If one were attempting to build a model which predicted a person's heart attack risk, the day of the week on which a person was born would not enhance the model's predictive capabilities. If the model builder did include birth day of the week in the model, the parameter estimation methodology, given a small number of data points, could build a coincidental correlation between days of the week and coronary risk into the predictive model. If the modeling paradigm had enough freedom, it may even attempt to force the model to overfit this day of the week data for this data set. The second problem of excess variables is that they make the model more difficult to explain. Some practices to reduce predictor variables include correlation analysis, all possible regressions [4], stepwise regression [2], factor analysis such as principal components [5], and intuition. However, these methods break down as the number of candidate variables becomes very large or if the problem is highly nonlinear.

- 2) Second, a predictive model should be built. There are many methods for this, all with some advantages and disadvantages. Linear models, if applicable, are easily explainable. As predicted properties become more complex, nonlinear models such as neural networks offer better predictive power. This step should also include an evaluation of how well the model predicts or classifies on data it has not seen before.
- 3) Finally, the model should have explanatory facilities. End users will be less likely to accept and use a model that cannot be understood, especially if the results are counter-intuitive. This is a very difficult challenge for complex nonlinear models. The methods introduced in this paper reveal the most important variables and their effects on the system. This is a first step to formulating univariate and multivariate rules to govern its behavior.

As the number of potential inputs to the model becomes large, and as the number of data observations becomes small, the problem takes on an additional challenge. For most modeling methodologies, there are no longer enough data points to support the number of inputs. Models become difficult to manage and maintain in many dimensions. If the model must also explain, the sheer number of possible interactions bedevils human intuition. The most obvious solution would be to get more data, but this is not always possible. Data points may be very expensive to obtain, or the conditions under which the data was taken may no longer be available. In this case, the modeler is challenged to obtain as much information out of the data and available variables as possible. A data strip mining problem is pivoted in the sense that there are more variables than data points to support them.

The authors of this paper term modeling this type of pivoted problem explicitly as **data strip mining**. Typical data mining problems are much like a ground mine. The prospectors do not

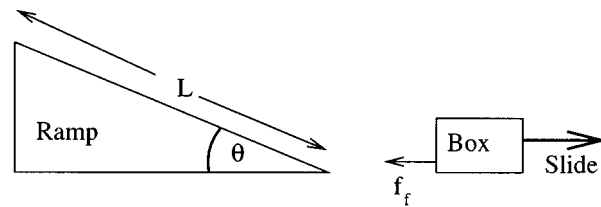


Fig. 1. Cartoon physics experiment for a neural network benchmark model: The model attempts to predict for how many seconds the box will slide along the ground once it reaches the bottom of the ramp.

need to figure out where to look. The inputs to the model have already been identified. They go to the same hole and dig deeper, gathering vast numbers of data points to produce a better model. Strip miners must constantly search the space of potential inputs to find combinations which provide good prediction. Furthermore, the data points they use to do this are limited. They cannot dig deeper to get more data. They must use what is available near the surface.

This paper presents a neural network methodology which iteratively performs the steps of this data mining cycle. It builds a complex model of the system with all possible inputs. It then performs sensitivity analysis on that model to eliminate some of the inputs which seem to have little effect. A new model is built with this subset, and the process repeats itself. It also evaluates the predictive ability of each model along the way so that the model builder knows when further simplification eliminates needed variables and reduces predictive ability. Finally, this variable reduction scheme, along with the sensitivity analysis, is an initial step in explaining the model.

III. CARTOON EXAMPLE—WHAT IF SIR ISAAC NEWTON USED NEURAL NETWORKS?

The physics problem shown in Fig. 1 illustrates the strip data mining approach to a problem for which a theoretical model already exists and may serve as a benchmark. Suppose Sir Isaac Newton had access to a personal computer and neural network software. He may have chosen to build data-driven models for his laws of classical physics. Suppose further that after conducting experiments and gathering ten data points, his watch broke, and he was forced to develop a model using only those data points. In his ten experiments, he released a box on a ramp equipped with rollers such that friction on the ramp was negligible. He wanted to build a model to predict how long the box would slide once it reached the ground at the bottom of the ramp before coming to rest. Since his laws were undeveloped, he took data for several variables which he hypothesized to be important in this problem. These included L —the length of the ramp, θ —the angle formed by the ramp and the ground, μ_k —the coefficient of kinetic friction between the box and the ground, h —the hour at which the measurements were taken, T —the temperature when the measurements were taken, and m —the mass of the box. He also recorded t_s —the length of time in seconds the box slid along the ground before coming to rest. These data, simulated by the authors, are shown in Table I. Note that,

TABLE I
EXPERIMENTAL DATA FOR SLIDING BOX

PROBLEM SHOWN IN FIG. 1. NOTE IN (5) THAT h , T , rv , AND m ARE NOT NECESSARY TO CALCULATE t_s . THEY ARE ADDED TO ALLOW THE METHODOLOGY OF SECTION VI TO REMOVE THEM FROM THE MODEL

h	T	m	μ_k	L	θ	rv	t_s
815	52	1	.10	5	.09	0.91	3.03
930	40	2	.14	5	.11	0.36	2.39
1215	72	3	.16	5	.10	0.03	1.99
1330	85	4	.17	5	.20	0.63	2.65
1440	68	3	.16	4	.15	1.00	2.18
1300	71	4	.17	4	.15	0.74	2.05
1330	75	5	.13	4	.32	0.54	3.90
1045	66	5	.13	3	.19	0.09	2.61
1150	70	1	.10	3	.22	0.00	3.65
1155	70	2	.14	3	.44	0.83	3.65

as (5) will show, t_s only depends on L , θ , and μ_k . The other variables are potential inputs which may be correlated with the output but are actually not important in our problem. For reference purposes, a random variable (rv) was also added to the data. The model development methodology of Section IV will filter the unnecessary variables from the model.

This system is actually governed by a few laws of classical physics. We must determine the time (t) at which the velocity of the box becomes zero. This is a function of its velocity at the bottom of the ramp (v_0) and its rate of deceleration ($-a$)

$$v = v_0 + at. \quad (1)$$

Solving for t at $v = 0$

$$t = \frac{v_0}{a}. \quad (2)$$

The velocity of the box at the bottom of the ramp is governed by the following equation:

$$v_0 = \sqrt{2gL \sin \theta} \quad (3)$$

where g is the acceleration of due to gravity (9.8 m/sec²). To determine the box's deceleration, we begin with the fundamental relationship $F = ma$. In this case, the force F is the force due to friction $f_f = \mu_k mg$. Therefore

$$a = \frac{F}{m} = \frac{\mu_k mg}{m} = \mu_k g. \quad (4)$$

Substituting (3) for v_0 and (4) for a into (2), the desired model for how long the box slides becomes

$$t_s = \frac{\sqrt{2L \sin \theta}}{\mu_k}. \quad (5)$$

This is the relationship which Newton's neural model will attempt to find using his ten experimental data points.

IV. VARIABLE REDUCTION THROUGH NEURAL NETWORK SENSITIVITY ANALYSIS

Neural network sensitivity analysis is the analysis of the system responses generated in the output layer of a neural network by changing the inputs going into the input layer [6]. It is hypothesized that a neural network, given a large set of input variables, will adjust its free parameters during training such that the most important variables in the problem will be the most sensitive.

Three different measures for sensitivity are proposed. The first is simply the sensitivity according to range S_r . Consider the sensitivity results for the fictitious system in Fig. 2. The output of the system y_i is shown for n levels of a particular input x_i varied within its allowable range. For this system, the sensitivity according to range is determined by $S_r = y_{\max} - y_{\min}$. A second possible measure is the variance produced in the output when the input is moved through its entire range. This is the sensitivity according to variance $S_v = \sum (y_i - \bar{y}) / (n - 1)$. A third possible measure for sensitivity is the average gradient over all the intervals. This is the sensitivity according to gradient $S_g = \sum_{i=1}^{n-1} y_i - y_{i+1} / (n - 1)$. The results in Fig. 2 show that all three measures capture the higher sensitivity of variable 2 compared to variable 1. However, only S_v and S_g capture the higher sensitivity variable 3 compared to variable 2. The analysis in this paper uses S_v .

This paper proposes the following simple methodology for performing neural sensitivity analysis. Once a network has been trained on a large set of input variables, calculate an average value for each input variable. Then, holding all variables but one at a time at their average levels, vary the one input over its entire range and compute the variability produced in the net outputs. Analysis of this variability may be done for several different networks, each trained from a different weight initialization. The algorithm will then rank the variables from highest to lowest according to the mean variability produced in the output. Although the use of average values by all but one input does not capture all of the complex interactions in the input space, it does produce a rough estimate of the model's univariate sensitivity to each input variable.

This sensitivity analysis procedure has two parameters. The first is the number of levels at which to evaluate each input during the analysis. All inputs but one will be held at their average values, and the net will be evaluated with this input at l values. If five values are selected, these values will be 0.0, 0.25, 0.5, 0.75, and 1.0. The data in Fig. 3 shows the effect this parameter has on S_v for the first chemistry data set (see Section VII). The addition of levels beyond the first few produces little change in S_v . The analysis in this paper is done with $l = 6$.

The second parameter for this procedure is the number of sensitivity runs r . This is the number of differently initialized trained nets on which to perform sensitivity analysis. Different random initializations of a network will yield different sensitivity levels for the variables. As the number of networks used increases, so does the accuracy of the rankings. If these rankings are to be used to eliminate variables from the model, a higher significance corresponds to a higher probability that those variables with the smallest effect on the output are eliminated. For

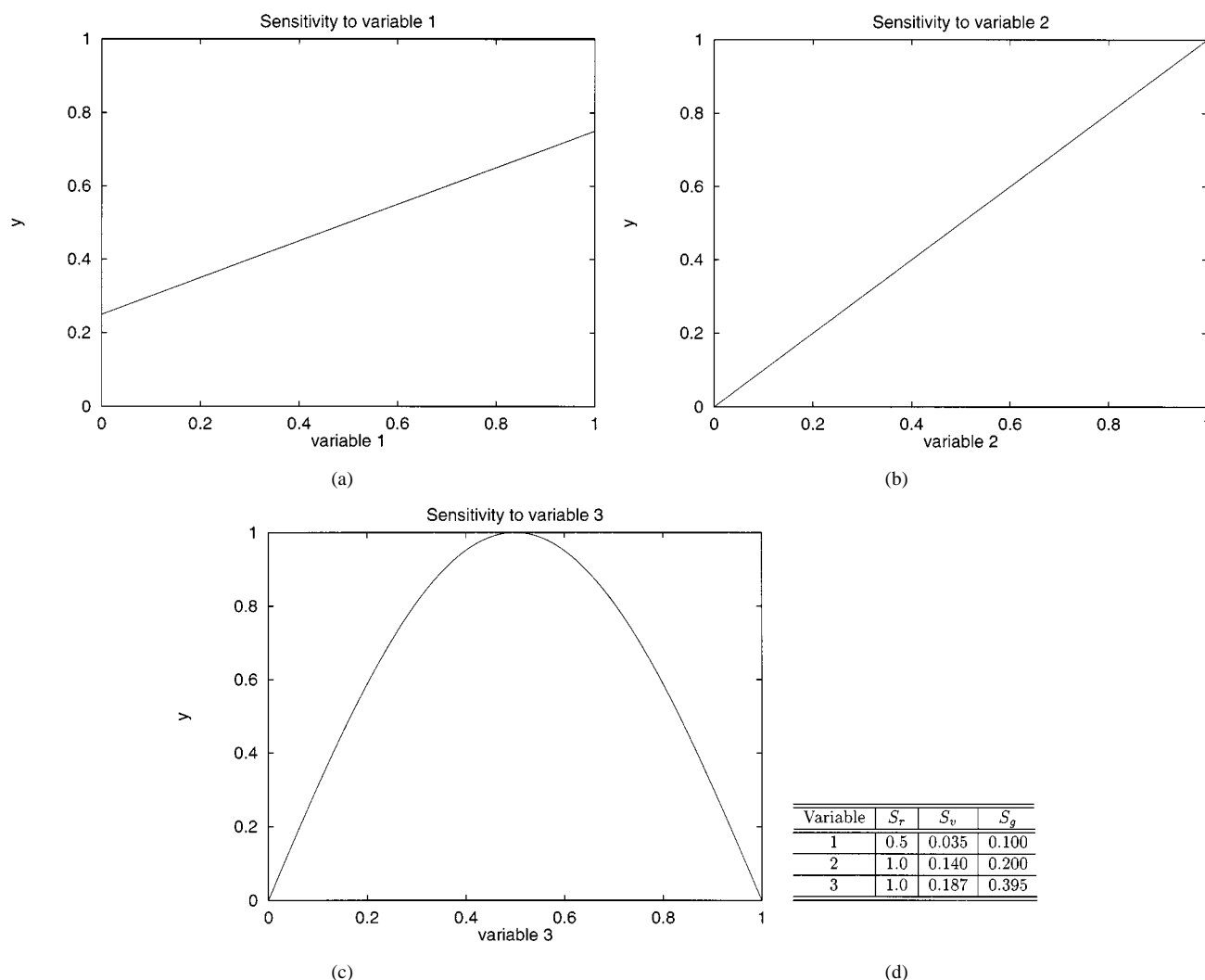


Fig. 2. The theoretical sensitivity of a fictitious system to three different variables and the different sensitivity measures for these variables calculated using $l = 6$ discrete points. S_v , sensitivity according to variance, and S_g , sensitivity according to gradient, capture the system's greater sensitivity to variable 3 than to variable 2. S_r , sensitivity according to range, does not capture this difference.

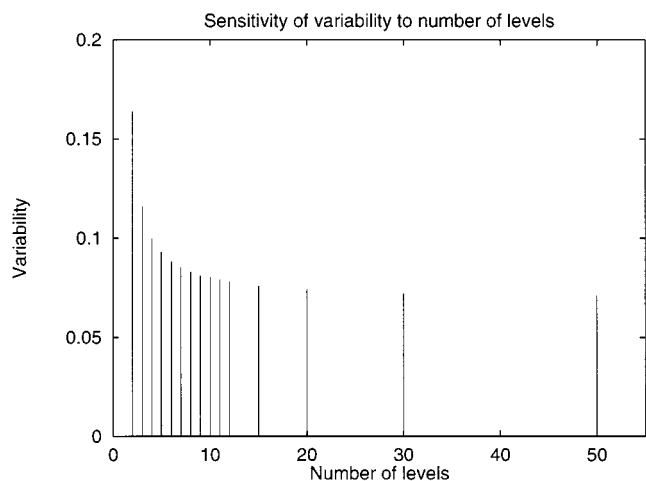


Fig. 3. The variability produced in a sample system using different numbers of levels for a given variable. Note that the variability changes very little after the addition of the first few levels.

example, the sensitivity analysis for one of the chemistry data sets called for the reduction of 15 variables to 11. Table II shows

TABLE II

THESE VALUES WERE TAKEN FROM SENSITIVITY ANALYSIS FOR A MODEL USED TO PREDICT A CHEMICAL PROPERTY. USING A TWO-TAIL PAIRED t -TEST, INCREASING THE NUMBER OF NETS USED FROM 3-5 INCREASES THE SIGNIFICANCE OF THE CONCLUSION THAT THE MEAN SENSITIVITY TO MUC IS NOT EQUAL TO THE MEAN SENSITIVITY TO K3 FROM 0.049 TO 0.034

r	\bar{S}_{vMUC}	\bar{S}_{vK3}	\bar{d}	s_d	t-value	p-value
5	0.00342	0.00276	0.000658	0.000463	3.17	0.034
3	0.00337	0.00260	0.000770	0.000307	4.34	0.049

the results of sensitivity analysis on the 11th variable (MUC) and 12th variable (K3) with 3 runs ($r = 3$). Assuming system variability produced by multiple runs is normally distributed, we may test the hypothesis that the mean sensitivity to MUC is equal to the mean sensitivity to time of day K3, with the alternate hypothesis that mean sensitivity to MUC \bar{S}_{vMUC} does not equal the mean sensitivity to time of day K3 \bar{S}_{vK3} . With only 3 runs, a p value of 0.049 on a two-tail paired t -test barely allows us to conclude inequality at $\alpha = 0.05$. However, more sensitivity runs ($r = 5$) gives a p value of 0.033, allowing us to reject the null hypothesis and conclude that $\bar{S}_{vMUC} > \bar{S}_{vK3}$ at the 0.034 level

TABLE III
INPUT PARAMETERS AND VARIABLES FOR "DATA STRIP MINING"

Input Parameters Determined in Advance

N_m – Number of models. The number of neural models to build for each candidate variable set. Each model will use a different random sample of training and test points.
 t – Number of validation points. The number of validation points to be used for each model. All remaining points will be used for training.
 N_i – Number of models for selection of exogenous ideal training error. This is the number of models used to determine the ideal training error stop point i .
 t_i – Number of validation points for selection of exogenous ideal training error. This is the number of validation points used to determine when test error is minimized for each model.
 l – Number of levels. The number of levels at which to evaluate each variable in sensitivity analysis.
 r – Number of sensitivity runs. The number of sensitivity runs to perform with each subset of candidate variables.
 f – Cut fraction. This is the fraction of variables eliminated from the model at each step.

Other Symbols Used

n – The number of variables used in the model
 n_{tot} – The total number of variables in the system
 e_i – Ideal training error corresponding to smallest pre-validation error observed during training of one of N_i networks
 i – Exogenous ideal training error average for all N_i networks
 x – Target value for a model prediction
 y – Actual value for a model prediction
 S_v – Sensitivity according to variance produced in model output by varying an input variable over l levels

of significance or higher. Increased sensitivity runs r allows for better decisions during variable reduction. The analysis in this paper is done with $r = 5$.

V. MODEL EVALUATION THROUGH CROSS-VALIDATION

A. Cross-Validation

As long as the net structure has enough complexity, a neural net can be eventually trained to produce a very small error level on the training set. Therefore, any measure of this error is a poor criterion for the ability of the net to predict outputs for data points it has not seen before. In order to determine a net's ability to generalize, it must be evaluated on a validation data set which was not used during the training. If the number of data points is large, it is possible to have a large number of data points in the validation set and a good estimate of the distribution of prediction errors. However, if the number of data points is small, the estimate of the mean prediction error produced by evaluation on the validation set will be highly variable. The presence of one outlier in the validation data set will significantly alter the estimate of mean prediction error. Cross-validation allows better estimates of prediction error.

Cross-validation segments the available data into k subsets (k -fold cross-validation). It successively sets each of the k subsets aside and trains the network using the remaining data. It then validates the network using the subset which was set aside.

The training and validation of k networks allows all of the available data to be used to estimate a model's ability to generalize [8], [9]. As k increases, the training set excludes fewer data points, and the model's predictive ability improves. If data points are extremely scarce, k may be increased until it reaches n , the number of data points. This is known as "leave one out" cross-validation. Cross validation generates one prediction error observation for each data point. The modeler uses prediction error estimates developed during cross-validation to evaluate alternative models during model selection. Cross-validation estimates of prediction error for the final model will be optimistically biased. For this reason, the modeler should leave aside a portion of the data, not used at all during model development, to test the final model's ability to generalize.

B. Network Evaluation Criteria

Multiple regression models typically evaluate model adequacy by the adjusted coefficient of determination or the mean square error in the model [4]. These statistics may not be calculated for strip mining models where the number of variables exceeds the number of data points. The coefficient of determination (without adjustment), R^2 , corresponds to the square of the correlation coefficient between the actual and predicted values. This may be calculated regardless of the number of variables in the system. However, it must be estimated for a validation set, not used in parameter estimation, to determine a model's ability to predict. This can often lead to confusion

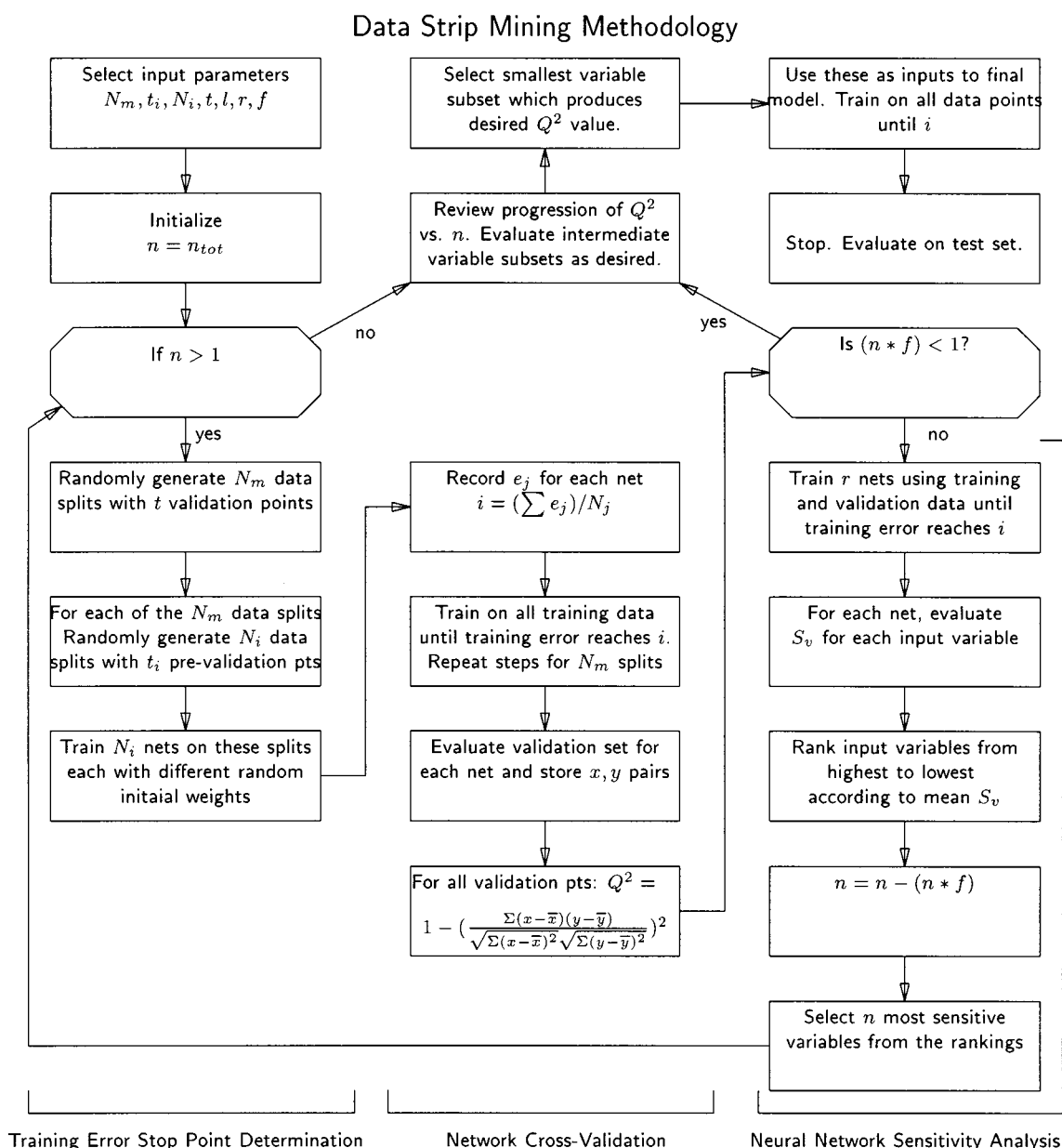


Fig. 4. Repeatable methodology for data strip mining which may also be automated. See Table III for an explanation of symbols used.

since many model building methodologies evaluate R^2 for the training set along the way. To ease this confusion, this paper will refer to a Q^2 statistic, often used in computational chemistry, which is simply one minus the correlation coefficient squared evaluated on the *validation* (during model development and selection) or *test* (during final model evaluation) set

$$Q^2 = 1 - R^2 = 1 - \left(\frac{\sum(x - \bar{x})(y - \bar{y})}{\sqrt{\sum(x - \bar{x})^2} \sqrt{\sum(y - \bar{y})^2}} \right)^2 \quad (6)$$

where x is the target value and y is the prediction given by the model.

This statistic will be used in conjunction with cross-validation to evaluate models. Its advantages are that it eliminates the confusion between analysis on the validation and training sets

and that it is not specific to one particular model building paradigm. It may be applied to linear regression, neural networks, or any other methodology.

C. Exogenous Training Error Stop Point Determination

In the conduct of cross-validation sessions, the analyst must determine at what error level to stop training the net on the training set and evaluate it on the validation set. The authors call this value the **ideal training error stop point**. This stop point allows a network with many free parameters to train on a small data set and still give good prediction. Stopping training too early will not allow the network to fully learn the the complexity required for prediction. Stopping training too late will allow the network to fit the output to the data so well that it no longer models the general trend in the output but fits the output curve to the individual data points exactly, even in the presence

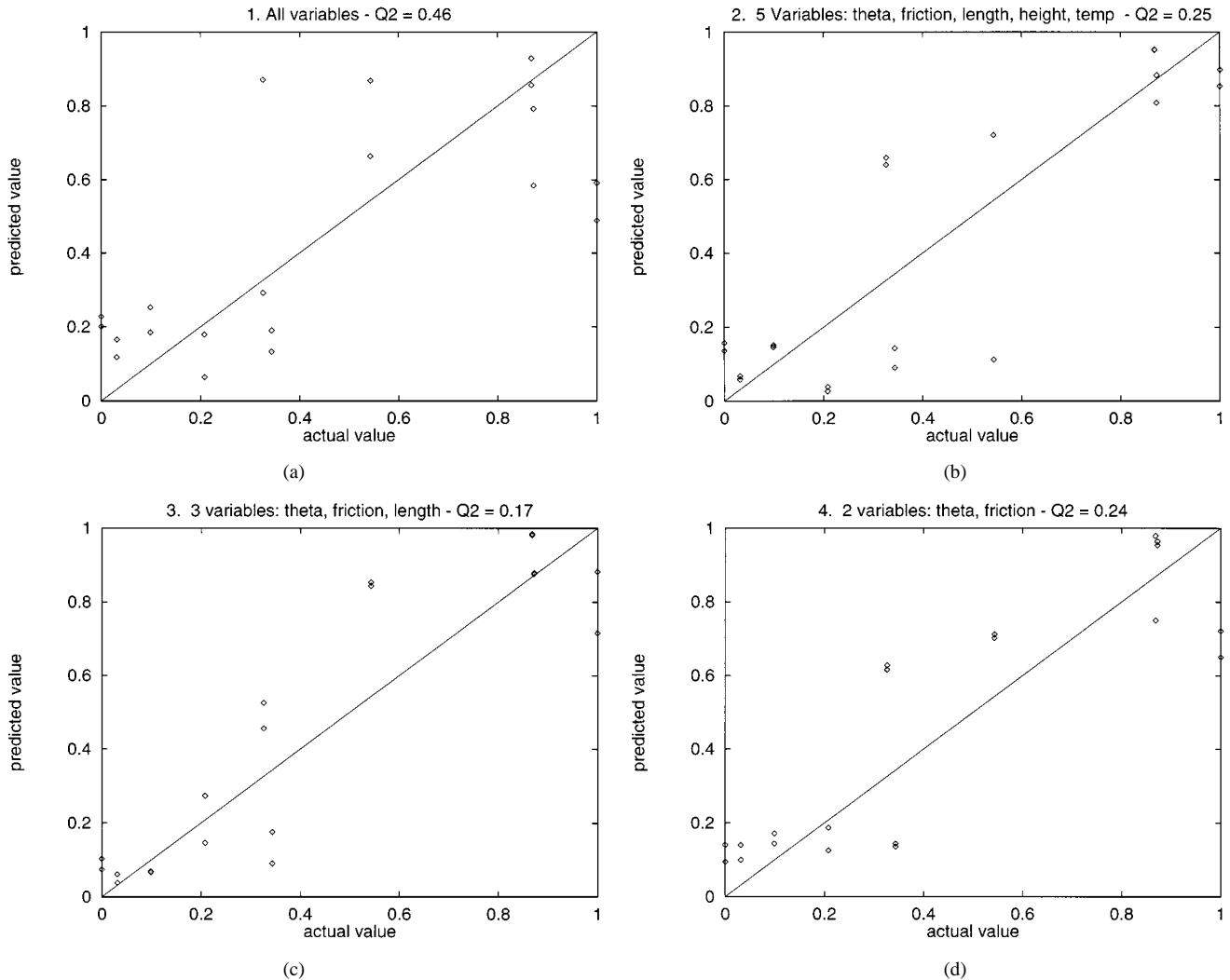


Fig. 5. Plots of actual versus predicted values for four iterations of Newton's benchmark model. The best Q^2 value is given by the model which corresponds to the theoretical 3-input model of (5). Though these Q^2 differences are not statistically significant for the small sample size in this illustrative example, the network is no longer fitting unnecessary data to the problem.

of data error. On-line observation of network errors on the validation and training sets confirms that test error decreases continually to a certain point, at which it begins to increase. The training error level at which the test error is minimized is the optimal training error stop point for this network [9]. This point will vary for different splits of validation and training sets.

One could simply monitor test error on line, record the optimal point, and evaluate the network using that error level. However, if the goal of model evaluation is to estimate a network's ability to predict outputs for data it has not seen before, this scheme will bias that estimate toward an optimal validation error for each split of the training and validation sets. A better technique would be to further subdivide each training set into a number of ideal error determination pre-training and pre-validation sets. Each pre-training set is used to train the network while observing the error on the pre-validation set. The modeler records the pre-training error which corresponds to the minimum pre-validation error for each split of pre-training and pre-validation data. The mean of the pre-training errors is the exogenous ideal training error stop point. The modeler trains the network using all of the training data until it reaches the

ideal training error stop point. He or she may then evaluate the model's predictive ability on the validation set. This produces an exogenous estimate of the net's ability to generalize.

D. Determining How Many Cross-Validation Sessions to Run

Another important consideration is how many data splits to use in cross-validation. This is driven by two factors.

- 1) As the validation set grows smaller in comparison to the training set, the model improves and Q^2 increases.
- 2) Each network training and evaluation session consumes processor time. The use of an exogenous ideal training error stop point increases the number of network training sessions by a multiplicative factor as k increases.

The goal of cross-validation is to efficiently get an acceptable and accurate Q^2 . Empirical analysis of Q^2 improvements as k increases suggest only marginal reduction in Q^2 with $k > 4$ for the pharmaceutical data sets used in this paper. In order to generate acceptable estimates of generalization capability without excessive computation, the analysis in this paper was performed with $k = 4$.

VI. DATA STRIP MINING METHODOLOGY USING NEURAL NETWORK SENSITIVITY AND CROSS-VALIDATION

The sensitivity analysis and cross-validation procedures may be combined into an automated methodology which efficiently builds a very good predictive model with a minimum number of input variables. The algorithm requires the parameters and variables shown in Table III. The flowchart in Fig. 4 illustrates the methodology. It generates N_m combinations of training and validation data. For each variable subset, it computes an exogenous ideal training error stop point then evaluates the model on the validation set. The combined errors for the N_m validation sets generate a Q^2 for the input variable subset. It then trains r networks using all data points and performs sensitivity analysis for each variable using each network. It ranks the variables according to mean S_v . A fraction f of the least sensitive variables are removed from the variable subset, and the algorithm repeats itself for the new subset. Upon completion, the variable subset with the lowest Q^2 should be selected for final model development. Currently, no specific rules of thumb exist to determine the parameters for this methodology. The experimenter considers the available computation time, problem complexity, network structure, and his or her own intuition to select them. The parameters used in Section VII represent a successful parameter set for this problem.

This methodology was applied to Newton's problem with the following parameters: $N_m = 10$, $t = 2$, $N_i = 5$, $t_i = 2$, $l = 6$, $r = 5$, and $f = 0.4$. All of the neural networks used had two hidden layers with 10 and 5 neurons, respectively. They were trained using backpropagation with a learning parameter of 0.01 and a momentum parameter of 0.5 [10]. It obtained very good results in four iterations, summarized in Fig. 5. These charts plot the actual values in the test set as compared to the predicted values obtained using the strip mining methodology. Better predictions are closer to the line. Visual inspection of the graphs shows the improvement in predictions as unnecessary variables are removed from the model. In the first iteration, the variables mass (m) and random noise (rv) were eliminated from the model. On the next iteration, time of day (h) and temperature (T) were eliminated leaving the angle (θ), the length of the ramp (L), and the coefficient of kinetic friction (μ_k) as the three variables in the model. This corresponds to the three variables in the theoretical model, and these variables produced the lowest Q^2 , 0.17, for any model, as shown in Fig. 5. Though these Q^2 differences are not statistically significant for the small sample size in this illustrative example, the network is no longer fitting unnecessary data to the problem. The final iteration eliminated L , a necessary input, and resulted in a higher Q^2 . Sensitivity analysis on the three-variable subset (Fig. 6) shows the accuracy and variability of this technique with respect to each variable. This sensitivity analysis may also be used as a way to explain the model. One can clearly see the different positive relationships of θ and ramp length with slide time. The inverse relationship between coefficient of kinetic friction and slide time may also be observed. These relationships make good sense if one uses intuition about the expected relationships. Although this model is no substitute for the theoretical model of (5), it has eliminated the unnecessary system inputs and captured the approximate mag-

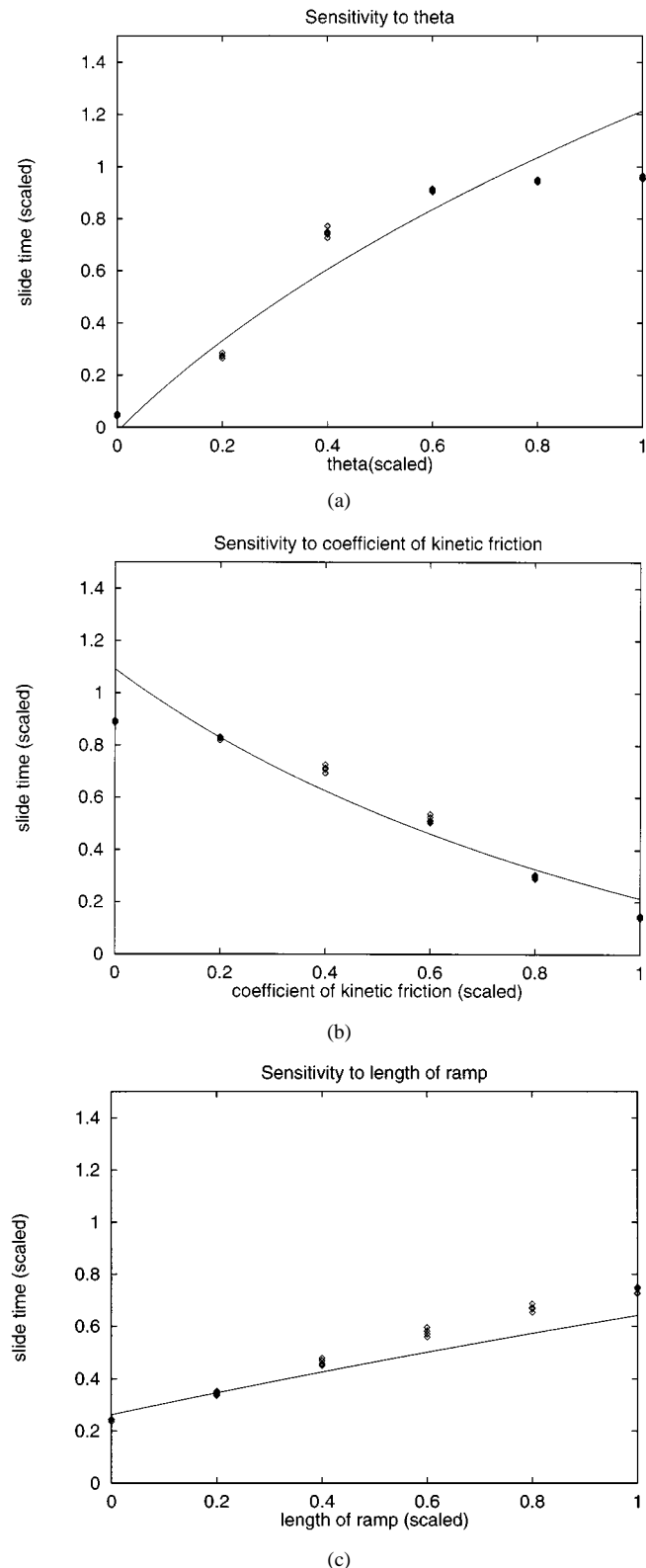


Fig. 6. System response for Newton's model with three variables. The model's response to changes in the inputs (dots) closely matches theoretical results (lines).

nitude and direction of the relationships between t_0 and L , θ , and μ_k . The availability of more experimental data would allow these methods to much more accurately and significantly portray the relationships seen in classical physics equations. In this

TABLE IV

RESULTS OF PAIRED ONE-TAILED t -TEST USED TO TEST THE HYPOTHESIS THAT THE MEAN PREDICTION ERROR PRODUCED BY DATA STRIP MINING MODELS IS EQUAL TO THE MEAN PREDICTION ERROR OF OTHER METHODS WITH $\alpha = 0.05$. THE ALTERNATIVE HYPOTHESIS WAS THAT THE MEAN PREDICTION ERROR FOR DATA STRIP MINING WAS LOWER THAN THE ALTERNATIVES. THE Q^2 VALUE FOR EACH METHODOLOGY IS ALSO SHOWN

Data Set	n	Methodology	Q^2	Mean Error	Error Variance	t -value	p -value	Sig?
CCK	66	Strip Mining	0.307	0.105	0.0093	-	-	-
CCK	66	Linear Fwd Sel	0.519	0.147	0.0269	1.99	0.025	Y
CCK	66	Neural Fwd Sel	0.428	0.124	0.0147	1.78	0.040	Y
CCK	66	Neural GA	0.429	0.131	0.0112	2.29	0.012	Y
Cancer	261	Strip Mining	0.614	0.137	0.0140	-	-	-
Cancer	261	Linear Fwd Sel	0.579	0.163	0.0174	3.84	0.00008	Y
Cancer	261	Neural Fwd Sel	0.740	0.153	0.0193	2.43	0.008	Y
Cancer	261	Neural GA	0.814	0.155	0.0207	2.43	0.008	Y

example the methodology obtained these promising results despite the fact that the initial system had seven potential inputs, 141 free parameters, and only ten data points.

VII. TWO TEST CASES—PREDICTING PHARMACEUTICAL PROPERTIES

A pharmaceutical manufacturer has a difficult problem. This person is attempting to design a drug which has a few desirable properties to exploit and many undesirable properties to avoid. There are too many candidate drugs to even consider, let alone test. The researcher must carefully choose a very small subset of candidate drugs to test in the laboratory for the target properties. Unfortunately, these lab tests are expensive and time consuming, ultimately concluding with human subjects. The researcher is armed with over 600 molecular descriptors, both explicit and derived, to guide his or her selection of good candidate drugs [11]. But which of these descriptors predict which properties? There are too many interactions to consider explicitly. Expertise and intuition are good guides, but they vary from individual to individual and must be learned over time at great expense.

The data strip mining methodology was compared to other model selection methodologies using two data sets from computational chemistry. The first set included 66 data points, each of them molecules. Each molecule had 480 descriptors as potential inputs to the model [12]. It also had a target predicted value for concentration required to inhibit binding with the Cholecystokinin (CCK) molecule in the human blood stream. The second data set had 400 predictors and 261 molecules along with observations of survival rates of cancer cells.

In conducting the tests, each molecular data set was divided into thirds. Each third was set aside as a test set, while the other two thirds composed the data available for training and validation. This created three different trial sets for each of the two molecular data sets, a total of six trial sets. Each trial set was used to evaluate each of the four modeling methodologies, generating a total of 24 modeling trials. In each trial, no methodology used the test data set until the final model had been selected. This evaluation gave a truly blind estimate of the generalization capabilities of each modeling methodology. All of the trials together generated a single observation of “blind” prediction error for each data point in both the CCK and the cancer prediction data sets. These prediction error observations generated a Q^2 and an estimate of mean prediction error which could

be used to compare the modeling methodologies for each data set.

Data strip mining built predictive models for each data set. Because this methodology does not yet have any parameter selection guidelines, the modelers used intuition to select parameters which were expected to provide good performance in the time available. For each case, $N_m = 4$, $N_i = 4$, $l = 6$, $r = 5$, $f = 0.33$. For the CCK data set, $t_i = 11$ and $t = 11$. For the cancer data set, $t_i = 43$ and $t = 43$. The final neural network had sigmoid neurons with 20 neurons in one hidden layer. An exogenous ideal training error stop point for this network was determined in accordance with Section V-C averaging eight observations of ideal pre-training error, each taken while using 11 pre-validation points for the CCK data set and 43 pre-validation points for the cancer data set. Note that for this final data set, the modelers increased N_m from 4 to 8 to gain a less variable estimate of the ideal training error. The network trained using back-propagation on all training data with a learning rate of 0.02 for the CCK data set and 0.01 for the cancer data set. A momentum parameter of 0.5 was used for both data sets. This trained network evaluated all points in the test data set to get observations of absolute prediction error.

Three modeling methodologies were compared to data strip mining.

- 1) **Linear Multiple Regression.** The first methodology was linear multiple regression using forward selection to determine the variables in the system. Forward selection was limited to at most five variables in the CCK data set and at most 16 variables in the cancer data set. This allowed least squares estimation to use at least ten data points for each free parameter in the model [2].
- 2) **Linear-Neural Hybrid.** In the second methodology, a neural network used the same variables selected during linear forward selection [9]. The neural network had sigmoid neurons with 20 neurons in one hidden layer. This network was trained and evaluated using the same procedures and parameters that were used to train and evaluate the final network for data strip mining. The only difference was the input variable set for the model.
- 3) **Neural-Genetic Hybrid.** The third methodology used a neural network to evaluate the adequacy of each variable subset, and a genetic algorithm to search for the best-performing subset. It proceeded as follows. The modeler selected a pre-determined number of variables to include

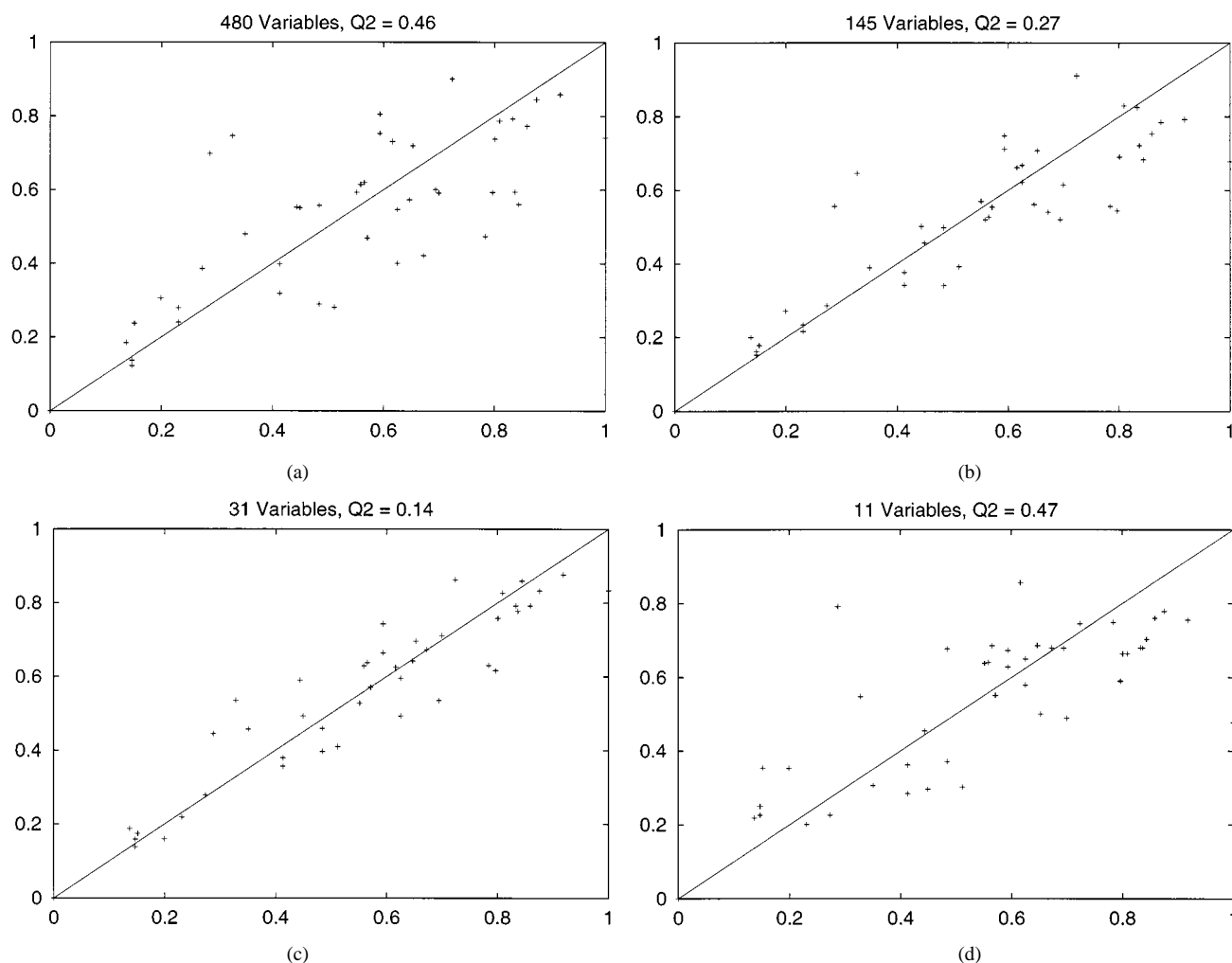


Fig. 7. Plots of actual versus predicted values as variables are removed from the CCK data set by the “strip data mining” methodology. The best Q^2 value occurs when $n = 31$. This best Q^2 is statistically significant with $\alpha < 0.001$.

in the input set. In each case, the genetic algorithm was asked to select the same number of input variables determined to be ideal during data strip mining. This was 31 variables for all splits of the CCK data set, and 37, 12, and 17 variables for each respective split of the cancer data set. He or she also selected the population size (30), number of generations (20), an evaluative network structure (20 Gaussian neurons in the hidden layer), and the number of epochs to train each evaluative network (600 for the CCK data set and 350 for the cancer data set). Each individual member of a population represented a possible set of predictor variables for the neural network. The algorithm determined the fitness of each member by first subdividing the data into a training and a validation set. It trained the network a pre-determined number of epochs and calculated the mean prediction error for both the training and validation sets. The member’s fitness was the sum of these two errors [13]. The genetic algorithm searched for 20 generations for the most fit variable subset. This network was trained and evaluated using the same procedures and parameters that were used to train and evaluate the final network for data strip mining and for the linear-neural hy-

brid. The only difference was the input variable set for the model.

For each of the three alternative methodologies, the authors tested the hypothesis that the mean prediction error generated by data strip mining was equal to that generated by the alternative methodology, with the alternative hypothesis that the mean prediction error generated by data strip mining was lower than the alternative methodology. The data was paired by observation, and a paired t -test was used to test for a significant difference in means [7]. In each case, the difference was significant at the $\alpha = 0.05$ level with p -values ranging from 0.00008–0.04 (see Table IV).

A graphic interpretation of these results is useful. Fig. 7 shows the progression of models versus n for one of the CCK data splits. This progression indicates that the best model occurs where $n = 31$, so the 31 variable subset was selected for the final model to be evaluated on the test set. This best Q^2 is statistically significant with $\alpha < 0.001$. Note that the Q^2 value in Fig. 7 is an optimistic one (as compared to the results in Table IV) because the validation data is also used in model selection. Fig. 8 shows actual versus predicted values on the CCK test data for all four of the competing methodologies. One can see

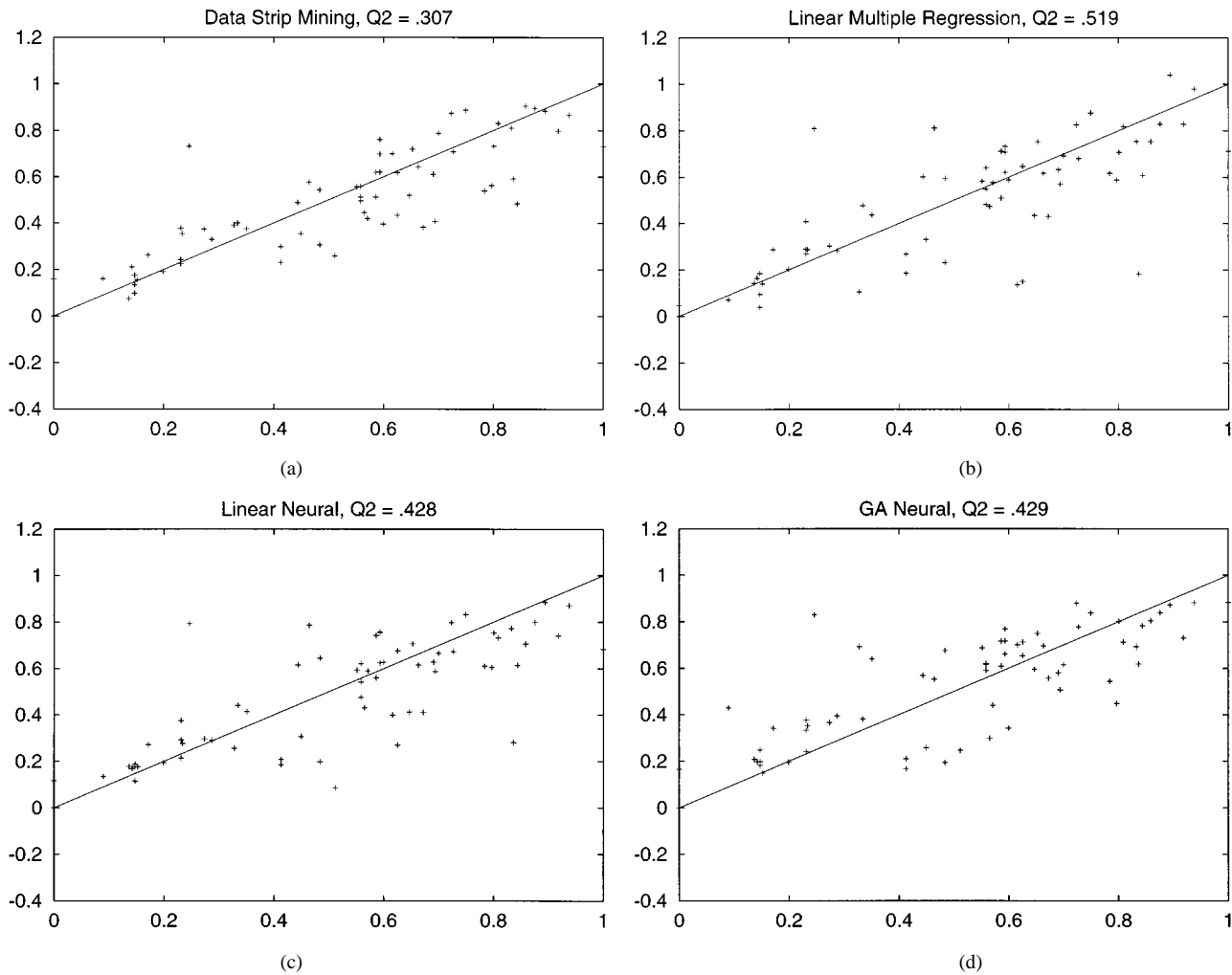


Fig. 8. Plots of actual versus predicted values for the competing modeling methodologies. Data strip mining has the lowest Q^2 value for the test data. The data points are collected more closely to the actual value = predicted value line. This best Q^2 is statistically significant with $\alpha < 0.001$.

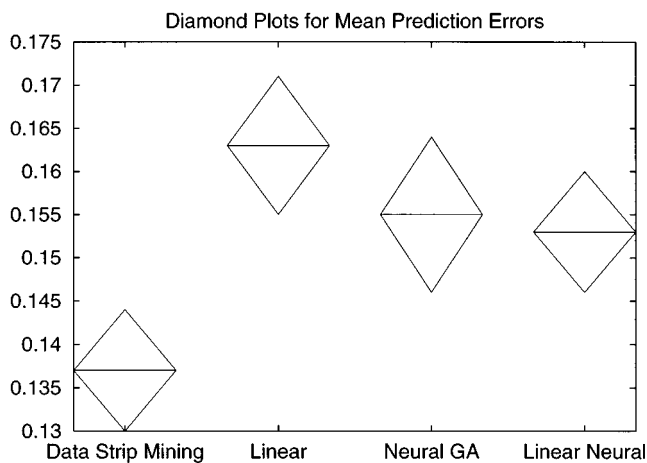


Fig. 9. Diamond plots for the mean prediction errors produced by the competing modeling methodologies. The centers of the diamonds represent the mean error, and the tips show one standard deviation from that error. Data strip mining shows the smallest error. This difference is statistically significant in all cases as shown in Table IV.

that the data points for the neural sensitivity model are packed more closely to the actual value = predicted value line than they

are for the other methodologies. Fig. 9 shows a diamond plot for the mean prediction errors for the cancer data set. This plot shows the lower mean error value for predictions determined using data strip mining. These plots serve as graphic confirmation of the statistical analysis.

VIII. CONCLUSIONS AND FUTURE ANALYSIS

This paper has introduced the standard data mining problem and a subclass of those problems, called data strip mining problems, which calls for the development and explanation of a model from a data set containing too few observations to support the number of potential input variables. Despite a larger number of free parameters than data points, calculation of an exogenous ideal training error stop point during backpropagation allowed the model to give good prediction. Neural network sensitivity analysis was used as a method to determine the most important inputs to the system, allowing some of the least sensitive inputs to be eliminated. The authors used model cross-validation as a method to accurately evaluate different neural models of the same system when the number of data points from which to draw training set and validation set samples is very small. All of these methods are combined and

automated in a well defined data strip mining methodology. This methodology works well for a benchmark problem from classical physics and for two very difficult problems from combinatorial chemistry for which other methodologies have yielded worse performance.

This methodology offers several benefits and a few weaknesses. It produces good predictive models for cases where the number of candidate inputs exceeds the number of data points. It simplifies the model to include only the most predictive inputs. The procedure may be automated and does not require extensive neural network or problem domain expertise to implement. To gain these benefits, the user must be able to tolerate stochastic results. A different, and perhaps poorer, ideal variable subset may be obtained with a subsequent iteration using different initial randomizations. A conservative choice of f and r may reduce this variability, but at the cost of processor time. In some cases, inspection of the predictors in the best subset reveals that the methodology has included a few variables which seem to add little valuable data to the model.

Based on these results, there are many directions for future research in this area. These include further sensitivity analysis on the algorithm parameters and comparison of this variable selection methodology to other methodologies with a more thorough investigation of parameters for those methodologies. For example, one could seek to determine how many generations it would take for the genetic algorithm to produce a subset which fared as well as data strip mining. Application to other difficult problems will verify its performance in other domains. It would also be very helpful to devise other ways to explain the model.

REFERENCES

- [1] J.-S. R. Jang, E. Mizutani, and C.-T. Sun, *Neuro-Fuzzy and Soft Computing: A Computational Approach to Learning and Machine Intelligence*. Upper Saddle River, NJ: Prentice-Hall, 1997.
- [2] J. Neter, M. H. Kutner, C. J. Nachtsheim, and W. Wasserman, *Applied Linear Regression Models*, 3rd ed. Chicago, IL: Irwin, 1996.
- [3] M. J. A. Berry and G. Linoff, *Data Mining Techniques for Marketing, Sales, and Customer Support*. New York: Wiley, 1994.
- [4] W. W. Hines and D. C. Montgomery, *Probability and Statistics in Engineering and Management Science*, 2nd ed. New York: Wiley, 1980.
- [5] R. A. Johnson and D. W. Wichern, *Applied Multivariate Analysis*. Upper Saddle River, NJ: Prentice Hall, 1992.
- [6] J. P. Bigus, *Data Mining with Neural Networks*. New York: Clarendon, 1996.
- [7] D. C. Montgomery, *Design and Analysis of Experiments*, 4th ed. New York: Wiley, 1997.
- [8] M. Stone, "Cross-validatory choice and assessment of statistical predictions," *J. Royal Stat. Soc. B*, vol. 36, no. 1, pp. 111–147, 1974.
- [9] C. M. Bishop, *Neural Networks for Pattern Recognition*. Oxford, U.K.: McGraw Hill, 1995.
- [10] S. Haykin, *Neural Networks—A Comprehensive Foundation*. New York: MacMillan, 1994.
- [11] C. M. Breneman, T. R. Thompson, M. Rhem, and M. Dung, "Electron density modeling of large systems using the transferable atom equivalent method," *Comput. Chem.*, vol. 19, p. 161, 1995.

- [12] R. Kewley, M. Embrechts, and C. Breneman, "Neural network analysis for data strip mining problems," in *Intelligent Engineering Systems through Artificial Neural Networks*. St. Louis, MO: ASME Press, vol. 8, pp. 391–396.

- [13] *Partek Pro 2000—On-line Help Manual*, Oct. 1999.



Robert H. Kewley received the B.S. degree and commission in the United States Army from the United States Military Academy, West Point, NY in 1988 and the M.S. degree in industrial and managerial engineering from Rensselaer Polytechnic Institute, Troy, NY, in 1998.

After serving two tours of duty with United States Army active duty units, he currently serves in the Department of Systems Engineering, United States Military Academy. After performing applications-oriented research in the department's Operations Research Center of Excellence, he currently teaches computer aided systems engineering and decision support systems courses to undergraduate cadets. His current areas of interest relate to neural networks, evolutionary computing, fuzzy logic, data mining, friend or foe combat identification, and decision support systems.



Mark J. Embrechts (S'74–M'81) received the M.S. degree in electrical engineering from the University of Leuven, Belgium, and the M.S. and Ph.D. degrees in nuclear engineering from Virginia Polytechnic Institute and State University, Blacksburg, in 1976, 1977, and 1981, respectively.

After working as a Postdoctoral Fellow at Los Alamos National Laboratory, Los Alamos, CA, as a student and postdoctoral staff member (1980 to 1983) he joined the Department of Nuclear Engineering, Rensselaer Polytechnic Institute, Troy, NY, in 1983. He is currently an Associate Professor in the Department of Decision Sciences and Engineering Systems and on the Faculty of the Information Technology Program, Rensselaer Polytechnic Institute. He has published more than 150 conference and journal papers and coauthored *Exchange Rate Theory* (Oxford, U.K.: Basil Blackwell, 1993). His current areas of interest relate to neural networks, evolutionary computing, fuzzy logic, data mining and applications of soft computing to data mining, biotechnology, and molecular chemistry.

Curt Breneman was born in Santa Monica, CA in 1956. He received the B.S. degree in chemistry from the University of California, Los Angeles, in 1980 and the Ph.D. degree in physical organic and computational chemistry from the University of California, Santa Barbara, in 1987.

During a post-doctoral appointment at Yale University, New Haven, CT, in 1987–1989, he worked in the field of theoretical electron density analysis. Since joining the faculty of Rensselaer Polytechnic Institute (RPI), Troy, NY, in the Summer of 1989, he developed the transferable atom equivalent (TAE) method of electron density modeling and has been active in the field of quantitative structure-activity relationships (QSAR) and quantitative structure-property relationships (QSPR). He is presently an Associate Professor of Chemistry and was elected Treasurer of the ACS Division of Computers in Chemistry in 1999. His research involves the use of new TAE-derived molecular property descriptors and semi-supervised machine learning techniques to create predictive QSAR and QSPR models.