A Tool for Model Generation and Knowledge Acquisition

Sally Jo Cunningham sallyjo@waikato.ac.n z
Paul Denize pdenize@waikato.ac.nz

Dept. of Computer Science
University of Waikato
Private Bag 3105
Hamilton, New Zealand
phone: (647) 856-2889
fax: (647) 838-4155

Abstract: Tools to automatically induce domain descriptions from examples are valuable aids for the knowledge acquisition stage of expert system construction. This paper presents a description of an algorithm that induces two domain descriptions: a conceptual model, which gives a broad understanding of variable interactions and their effect on the system, and a predictive model, which determines the system value associated with variable values input to the model. This induction algorithm is based on Entropy Data Analysis (EDA), which builds linear equations to approximate the system described by the training data.

keywords: knowledge acquisition, model generation, machine learning

1. Introduction

Systems that learn from sets of examples are valuable tools in the construction of expert systems. Knowledge acquisition has long been realized to be the bottleneck in expert system development. By automating the induction of general concepts or rules from available data, this bottleneck can be eased.

This paper describes the application of Entropy Data Analysis (EDA) to the problem of inducing rules from raw, empirical data. EDA, also known as k-systems analysis, is a relatively new technique for modeling multivariate numerical systems. Unlike more traditional statistical methods, which often must assume a model, EDA builds a model to fit the data. Moreover, nonlinearity does not adversely affect the construction of the model. EDA analysis of raw data produces a system model in the form of a set of equations that approximate the behavior of the system function implicit in the data. The "true" system function may be arbitrarily complex, but can still be effectively approximated by EDA models.

In turn, this model may be analyzed to provide both a high-level conceptual description of the domain as well as a predictive set of production rules:

> The ***conceptual domain description*** reveals the combinations of factors (variable value combinations) in the data that influence the dependent variable, and the effect that these factors have on the dependent variable; i.e., "Low levels of rain and low temperatures adversely affect shrimp catches".
>
> The ***predictive production rules*** are IF-THEN rules that predict the effect that specific independent variable values will have on the dependent variable; i.e., "IF the level of rain is between 0 and 0.7 inches and the temperature is less than 40 degrees Fahrenheit, then the shrimp catch is predicted to be about 1200 pounds".

Previous research

Entropy data analysis was developed as a modelling technique for general systems. It has been used as a technique for multivariate analysis (supplementing traditional statistical analysis) in several domains (see, for example, [Kn86], [Sh87]). The use of EDA as a method for obtaining both a semantic model and a "grammar" of rules

describing a system was first suggested by Gary Shaffer in [Sh86]; this paper examines the applicability of EDA to machine learning, in particular to function induction.

Quinlan's ID3 algorithm is perhaps the best known ([Qu79], [Qu83]) induction algorithm. ID3 produces a decision tree, which is commonly transformed into a set of rules for incorporation into an expert system. Cendrowska [Ce87] noted that this transformation generally introduces redundancy into the rule set, and developed an algorithm that produces rules directly from the data. Her PRISM algorithm is similar to EDA in that both use an entropic measure to determine the "important" attribute-value pairs in a system. Where PRISM uses this ranking of importance to create a minimal rule set, EDA instead calculates a set of linear equations that approximates the system behaviour. These equations may be used to create a rule set. Experimentation indicates that if the example set is complete (i.e., all possible combinations of all variable values are represented in the data instances), then PRISM and EDA will produce the same rule set. If the data is incomplete, then PRISM may produce a smaller rule set. PRISM is appropriate only for discrete classification tasks, while EDA-based induction is also appropriate for function induction. In addition, EDA also provides a conceptual model of the domain.

This paper is organized as follows: Section 2 describes the process of entropy data analysis; Section 3 presents a case study, and describes the production rules and conceptual description derived for the case; Section 4 discusses the effect of noise and sparse data on the quality of the model induced; and Section 5 presents conclusions about the use of EDA in induction. The "Anima Mundi" program (a trademark of the Seer Corporation, [Jo85]) is used to perform the computations of EDA.

2. Entropy Data Analysis

Complete discussions of the mathematical foundations of EDA, the details of the EDA algorithm, and issues in efficiently implementing EDA are found in [Jo85b-e]. The discussion below summarizes the EDA algorithm.

Consider a system with two or more independent variables and one system variable. Each combination of variable variable values is called a ***factor***. EDA attacks the problem of discerning the information contributed by each variable or variable combination (factor) in determining the system variable value by the following algorithm:

1) Begin with a "flat" system--a system in complete entropy, in which all variable or variable combinations are considered equally likely. This system is represented by a set of linear equations based on the marginal probabilities for the system.

2) Measure the information content of the factors (by the classic information theoretic entropy measure), and choose the factor with the greatest information content.

3) Add the factor chosen by (2) to the system equations. Adding this factor will bring the flat system closer to a description of the "true system" (the system that produced the original data).

4) Test for convergence between the equations in (3) and the marginal probabilities found in the data. If convergence has occurred, then the equations in (3) approximate the "true system"; otherwise, go to step (2).

Usually only a small subset of the set of all possible factors are necessary to approximate the true system. Knowing these factors permits the storage of system information in a highly compressed form for reconstructing the original system to a desired degree of approximation.

3. A Sample Application--rye yield prediction

The following example, taken from the domain of agriculture, will illustrate the procedures involved in applying the general EDA algorithm described in Section 2 to the induction of a conceptual and a predictive model of a data set.

Rye crop yields, like many other crop yields, are known to be sensitive to rainfall and temperature. The following data was taken from [Du74]:

Rainfall (inches)	Temperature (Fahrenheit)	Rye crop yield (bushels/acre)
45.0	54.1	21.0
47.0	61.6	20.0
33.0	50.8	21.0
39.0	52.1	24.0
30.0	50.2	20.0
28.0	57.1	12.5
41.0	55.7	19.0
44.0	57.6	23.0
31.0	50.1	23.0
29.0	38.0	19.0
34.0	56.2	21.0
27.0	51.5	12.0
42.0	54.1	21.0
35.0	46.7	27.0
43.0	60.8	17.5
39.0	56.9	26.0
31.0	60.3	11.0
42.0	54.6	24.0
43.0	53.5	26.0
47.0	64.0	18.5
25.0	45.7	15.5
50.0	61.5	16.5
45.0	59.7	18.0
34.0	53.2	20.5
29.0	45.1	22.0

Table 2: rye crop yield data

The independent variables rainfall and temperature are both continuous variables, with rainfall having the range [25,50] and temperature having the range [38.0,64.0]. The completeness of the data (the extent to which all variable value combinations are present) affects the effectiveness of the analysis. Since the two variables are continuous, the data therefore has a very low level of completeness. The solution to this problem is straightforward: to increase completeness, data values that are "close" to one another may be lumped together and represented by a single value. This "clustering" problem is well-studied in statistical analysis (see, for example, the discussion in [St80]).

We choose to group rainfall and temperature into clusterings representing low, medium, and high ranges for both rainfall and temperature:

Temperature:	[38.00, 41.92]	[41.93, 51.03]	[51.04, 64.00]
Rainfall:	[25.00, 36.31]	[36.32, 45.43]	[45.44, 50.00]

We now wish to produce two types of information about rye crop yields:

a) a *conceptual description* of the weather conditions that tend to either depress or increase crop yield, and

b) a set of *predictive rules* that will let us determine, for a given temperature and rainfall, what our rye crop is likely to be.

Conceptual description

As described in Section 2 above, EDA proceeds by first determining the *factors* (variable and value combinations) with the greatest information content. These factors are added to the set of "flat" system equations (equations which assume that all factors are equally likely), and the effect of the factor on the system value may then be observed (whether the factor will increase or decrease the system value). When the above data was analyzed, the following factors were found to possess the greatest amount of system information:

Factor	Effect on system value
rainfall e [25.00, 36.31] and temperature e [51.04, 64.00]	negative, -19.9%
rainfall e [36.32, 45.43] and temperature e [51.04, 64.00]	positive, 14.2%
rainfall e [25.00, 36.31] and temperature e [41.93, 51.03]	positive, 11.4%

These factors capture a semantic model of the domain, and may be translated as:

> If the rainfall is low and the temperature is high, then the crop yield will be lowered.
> If the rainfall is moderate and the temperature is high, then the crop yield will be increased.
> If the rainfall is low and the temperature is moderate, then the crop yield will be increased.

The degree to which the factor affects the system value can also be incorporated into the conceptual model by assigning a semantic interpretation to the percentage of increase/decrease of the system value. Based on personal experience, we assign the following semantic labels: a factor influencing the system value by more than 30% "strongly" increases or decreases the value, while an influence of less than 10% "slightly" affects the system value. The percentage levels and the labels are arbitrary, and may be modifed to incorporate domain-specific information.

Predictive rules

As described in Section 2 above, factors are added to the "flat" system equations until an arbitrary degree of closeness to the original system is reached. We then possess a set of linear equations that describe the system to the desired accuracy, for those variable range values present in the original data. To predict the crop yield for a given pair of rainfall and temperature values, we determine the predicted crop yield for that pair (i.e., to determine the yield for low rainfall and low temperature, we solve the equations for the clusters "rainfall e [25.00, 36.31]" and temperature e [38.00, 41.92], and calculate the predicted crop yield of 19.00 bushels/acre).

By solving these equations for all possible combinations of the rainfall and temperature clustered values, the following table of predictive rules are derived:

Rainfall	Temperature	Yield
Low	Low	19.00
	Moderate	21.42
	High	15.40
Moderate	Low	19.22
	Moderate	19.22
	High	21.95
High	Low	19.22
	Moderate	19.22
	High	18.33

But these predictive rules are of a coarse granularity. To obtain a finer grain for predictive power, we may re-cluster the variables rainfall and temperature into a larger number of clusters, each containing a smaller range of values.
We can then perform EDA analysis with this larger number of clusters, calculate new system equations, and use these new equations to calculate finer predictive rules. How fine a granularity is possible? A general rule is that the product of the number of clusters over all variables should not be more than 25% greater than the number of instances in the training set. If this threshold is exceeded, then the data will be too sparse to support meaningful induction.

4. Issues in Inducing Rules from Data

The accuracy and predictive power of the rules derived from data depends on the quality of the data. This data quality is determined by the completeness of the data and the amount of noise present in the data.

Completeness

A data set is *complete* if each possible combination of variable values is present in the training set. With 100% completeness (and an absence of noise), the system function can be perfectly reconstructed by EDA. In "real world" data, however, completeness is rarely possible. The effect of low levels of completeness on the accuracy of the conceptual model and predictive rules varies. If the "important" areas of the system function are present in the data (i.e., the local maxima, local minima, etc.), then meaningful analysis and rule induction is possible. Generally, we do not have this kind of knowledge about raw data. It is therefore usually necessary to increase completeness by clustering the data. However, there is a trade-off between completeness and predictive power; the greater the completeness achieved by coarse clustering, the less predictive the rule set derived from analysis. In addition, clustering tends to add to the noise present in the data, since a cluster will often be associated with several system function values. As discussed below, however, this problem of function value contradiction is easily resolved.

Additionally, the EDA algorithm builds a system model that is predictive and descriptive only in the ranges of values actually found in the data (i.e., for the data in Table 1, we cannot construct predictive or descriptive models to determine the effect of a temperature of 75 degrees on rye crop yield). If the data is incomplete in the sense that extrapolation beyond the given data ranges is required, then EDA is not appropriate for system modelling. The system model constructed by EDA is based only on the information actually contained in the training data, and does not assume the additional information that would be necessary to permit extrapolation.

Noise

Noise in a data set arises from two sources: from mis-measurement, and from measuring the wrong variables. Noise is common in "real world" data, and some forms of noise are successfully managed by EDA induction.

Mis-measurement occurs when a value of a variable is incorrectly measured or missing. This problem may occur because the value is incorrectly perceived or transcribed, the measuring instrument is faulty, or for many other reasons. If the noise is randomly distributed throughout the data and the data is relatively complete, then the noise may be "smoothed out" by clustering. Noise in the variable values is reduced as the variable values are clustered. Noise in the system function value may lead to contradictory states in the data, where a single set of variable values is associated with multiple system function values. As noted above, the number of contradictory states tends to increase with clustering. Fortunately, a number of classic methods are available to resolve this contradiction: averaging the function values, choosing the mean value, choosing the mode value, etc. A more pernicious problem occurs, however, if the noise is not random; if, for example, a measuring instrument is biased to give consistently high readings. In this case, the system model derived can only be as good as the data. EDA, like other induction methods, is not capable of detecting a systematic bias, and will incorporate this bias into the derived system model.

Noise may also occur in the form of "residual variation", when factors additional to those recorded affect the values of the system function. In complex domains, this form of noise is highly likely; however, a simplified (and therefore less accurate) model of a domain may be necessary, as some factors may be unknown or not measurable.

A final form of noise is that of extraneous information--the presence of variables that do not affect the system function. EDA-based induction is particularly successful at dealing with this problem. Since the system equations derived by EDA are formed by choosing only those factors that contribute the greatest degree of information to the system (by the information theory entropy measure), any extraneous variables are not incorporated into the system model. These extraneous variables are effectively ignored.

5. Conclusions

EDA-based induction is appropriate for domains in which the values of the dependent variable are derived from an unknown function. This type of domain is not the norm for machine learning algorithms; most research has concentrated on discrete classification rather than function induction (see, for example, Quinlan's ID3 and its descendants discussed in [Qu83], [Qu87], [Ce87]). If classification values for a domain can be linearly ordered, then EDA-based induction is possible for that domain. For example, if the classifications are "patient will die" and "patient will live", then we may represent the former possibility as a 0 and the latter as a 1. A cut-off in the range [0,1] determines which category a value generated by a predictive rule falls in--say, that any value greater than 0.5 indicates that the rule predicts that a patient will live.

Many classification values cannot be linearly ordered. Consider the classic Iris classification domain first referenced in [Fi36]. In this domain, sepal and petal lengths and widths are used to determine whether a given flower is an *Iris setosa*, *Iris verscolor*, or *Iris virginica*. There is no semantically meaningful ordering that can be applied so that, for example, *setosa* < *verscolor* < *virginica*. If an arbitrary ordering is imposed on the discrete system values, EDA-based induction is appropriate for this type of classification problem only if the training data set is 100% complete, contains no noise (either from mis-measurement or residual variation), and contains mutually exclusive classifications. Since these requirements are rarely met by "real world" data, EDA-based induction is not considered appropriate for unorderable classification domains. If the domain does not fulfil these requirements, then the descriptive domain model tends to contain a very large number of factors that influence the system values only slightly (and hence the model is not very descriptive!), and the classifications produced by the predictive model tend to be artifacts of the arbitrary ordering (and hence there are a large number of mis-classifications).

Of course, the conceptual and predictive models induced can only be as good as the data provided. However, the EDA-based induction model is relatively robust in the presence of noise and incomplete data.

References

[Ce87] Cendrowska, Jadzia. PRISM: An Algorithm for Inducing Modular Rules. International Journal of Man-Machine Studies, vol. 27, 1987, pp. 349-370.

[Du74] Dunn, Olive Jean, and Virginia A. Clark. Applied Statistics: Analysis of Variance and Regression. Sydney: John Wiley and Sons, 1974.

[Fi36] Fisher,R.A. The use of multiple measurements in taxonomic problems. Annual Eugenics, 7, Part II, 179-188 (1936); also in Contributions to Mathematical Statistics, John Wiley 1950.

[Jo85a] Jones, Bush, and Jim Brannon. Anima Mundi User's Manual. Seer Corporation, 1985.

[Jo85b] Jones, Bush. Determination of Unbiased Reconstructions. International Journal of General Systems, vol. 10, 1985, pp. 169-176.

[Jo85c] Jones, Bush. A Greedy Algorithm for a Generalization of the Reconstruction Problem. International Journal of General Systems, vol. 11, 1985, pp. 63-68.

[Jo85d] Jones, Bush. Reconstructability Analysis for General Functions. International Journal of General Systems, vol. 11, 1985, pp. 133-142.

[Jo85e] Jones, Bush. Reconstructability Considerations with Arbitrary Data. International Journal of General Systems, vol. 11, 1985, pp. 143-151.

[Kn86] Knudsen, P.A. Emigration Patterns and Size Characteristics of Brown Shrimp Leaving a Louisiana Coastal Marsh. M.S. Thesis, Louisiana State University, Baton Rouge, 1986.

[Qu79] Quinlan, J. R. Discovering Rules from Large Collections of Examples: a Case Study. in Michie, D., Ed., Expert Systems in the Micro-Electronic Age. Edinburgh: Edinburgh University Press, pp. 168-201.

[Qu83] Quinlan, J.R. Learning Efficient Classification Procedures and Their Application to Chess Endgames. In Michalski, R.S., J.G. Carbonell, and T.M. Mitchell, Eds., Machine Learning: An Artificial Intelligence Approach. Palo Alto: Tioga, 1983, pp. 463-482.

[Sh86] Shaffer, Gary. Benthic Microfloral Production on the West and Gulf Coasts of the United States: Techniques for Analyzing Dynamic Data. Ph.D. thesis, Louisiana State University, USA, 1986.

[Sh87] Shaffer, Gary, and Peter Cahoon. Extracting Information from Ecological Data Containing High Spatial and Temporal Variability: Benthic Microfloral Production. International Journal of General Systems, vol. 13, 1987, pp. 107-123.

[St80]. Steel, Robert G.D., and James H. Torrie. Principles and Procedures of Statistics: A Biometrical Approach, 2nd edition. McGraw-Hill Book Company, 1980.