

Using Reconstructability Analysis for Input Variable Reduction: A Business Example

Stephen Shervais
Eastern Washington University,
Cheney, WA 99004, USA
sshervais@ewu.edu

Martin Zwick
Portland State University,
Portland, OR 97207, USA
zwickm@pdx.edu

Abstract

We demonstrate the use of Reconstructability Analysis (RA) on the UCI Australian Credit dataset to reduce the number of input variables for two different analysis tools. Using 14 variables, an artificial neural net (NN) is able to predict whether or not credit was granted, with a 79.1% success rate. RA preprocessing allows us to reduce the number of independent variables from 14 to two different sets of three, which have success rates of 77.2% and 76.9% respectively. The difference between these rates and that of the 14-variable NN is not statistically significant. The three-variable rulesets given by RA achieve success rates of 77.8% and 79.7%. Again, the difference between those values and the 14-variable NN is not statistically significant, that is, our approach provides a three-variable model that is competitive with the 14-variable equivalent.

1. Introduction

This paper uses a method called reconstructability analysis (RA) to reduce the number of variables used in an industry-standard classification problem. Although the RA technique is over twenty years old, it is underutilized and this paper illustrates its capacities. RA is used here to develop models which are simpler, i.e., have fewer variables, than the original problem, yet still capture most of the predictive information in the data. We then use these simpler models to analyze training and testing datasets for an artificial neural net, as well as to construct lookup tables specifying rules derived from the models. Related work on feature selection for artificial neural networks by RA methods has been reported by [1], [2], and [12], and for data mining by [3]. Note that RA is not merely a feature selection method; it can be used also to produce the predictive model based on the selected features, as this paper shows. This study is part of a continuing exploration of the robustness of RA for

both feature selection and predictive modeling in different application contexts.

The rest of the paper is in five parts. First, we provide a brief introduction to reconstructability analysis. Next, we describe the Australian Credit Card dataset. We then describe the procedures we used to build our training and testing datasets, and present our results for both the neural nets and the lookup tables. We finish with a discussion of the results.

2. Reconstructability Analysis

Reconstructability analysis (RA) derives from [4], and was developed by Broekstra, Cavallo, Cellier, Conant, Jones, Klir, Krippendorff, and others; an extensive bibliography is available in [5], and a compact summary of RA may be found in [6] and [7]. RA resembles log-linear methods [8], used widely in the social sciences [9], and where RA and log-linear methodologies overlap they are equivalent [10]. In RA [11], a probability or frequency distribution or a set-theoretic relation is decomposed (compressed, simplified) into component distributions or relations. The most common application is the decomposition of frequency distributions, where RA does statistical analysis.

RA can model problems both where “independent variables” (inputs) and “dependent variables” (outputs) are distinguished (directed systems) and where this distinction is not made (neutral systems). In the present case, we have a directed system, with up to 15 independent variables A-O as inputs, and a single dependent variable, Z as the output. The goal, in our analysis, is to find some subset of the inputs that provides an acceptable level of prediction of the output. Since the information contained in a model is *not* the same as the classification rate, nor even a covariance measure, it is possible to obtain high classification rates with models that provide only limited information.

Consider a frequency distribution $f(A, B, C, Z)$ for a directed system, where A, B, C are inputs and Z is an output. RA decomposes such distributions into models consisting of sets of projections, for example into $f_1(A,B,C)$, $f_2(A,B,Z)$ and $f_3(B,C,Z)$, written as the (cyclic) model $ABC:ABZ:BCZ$. This model has three 'components,' each of which corresponds to a projection of the frequency distribution of the data. Models are acyclic if recursively eliminating variables unique to any single component and eliminating components that are projections of other components (and are thus redundant) arrives at the null structure [10]. The $ABC:ABZ:BCZ$ model cannot be reduced by either of these operations and is thus cyclic.

Taken together, these three projections, two of which predict the output from the inputs, constitute a model of the data that is less complex (has fewer degrees of freedom) than the data. By maximum-entropy (uncertainty) composition of these projections, the model yields a *calculated* trivariate $ABC_{ABC:ABZ:BCZ}$ frequency distribution (the subscripts show the model used), which may differ from the *observed* $ABCZ$ data. Dividing by the sample size gives the calculated probability distribution for the model.

Such a model may be assessed by its %Uncertainty Reduction, 100. $[H(Z) - H_m(Z|ABC)] / H(Z)$, where H is Shannon entropy, and $H_m(Z|ABC)$ is the conditional entropy of the output, knowing the inputs, for model m . The model can be used for prediction by generating the conditional probabilities of the output (Z) states, given the input (ABC) state.

Because RA models for directed systems always include a component that has all the input variables (to allow for the existence of relationships between these variables), any model with more than one "predicting component" (a component including the output) has a loop in it, e.g., the model previously discussed, $ABC:ABZ:BCZ$. By contrast a model with only one component, e.g., $ABC:ABZ$ which says that Z is predicted by A and B , has no loops (is acyclic). (Z is unique to the 2nd component and can be removed; the remaining AB is now redundant; this leaves only the 1st component, which can also be deleted since all of its variables are unique to it.) The presence of loops causes RA to require an iterative rather than a single-step algebraic algorithm for the calculation of model probabilities. If the inputs are independent of one another, however, $ABC:ABZ:ACZ$ becomes simply $ABZ:ACZ$, and has no loop; in such cases, loops can occur if there are three or more predicting components, e.g., $ABZ:BCZ:ACZ$. In the present paper, models with loops involving an input component were used for variable reduction (feature selection). These complex models can also be used to

predict the output, as discussed briefly at the end of Section V, or to prestructure a neural net with less than full connectivity, see [12]. and papers cited therein.

Calculations for this paper were made using the RA software programs developed at Portland State University, now integrated into the package OCCAM (for the principle of parsimony and as an acronym for "Organizational Complexity Computation And Modeling"). The earliest of these programs was developed by Zwick and Hosseini [13]; a list of recent RA papers of the PSU group is given in [14] and [15].

Models are selected from the one of the measures that OCCAM outputs for different models applied to the training set data, namely the Bayesian Information Criterion (BIC) also known as the Schwartz Criterion [16]. BIC is a way of linearly integrating the error of a model and its complexity (DF) which differs from the Akaike Information Criterion (AIC) [17] by its inclusion of a factor which depends on the sample size, N :

$$AIC = -2 N \sum p \ln q + 2 DF.$$

$$BIC = -2 N \sum p \ln q + \ln(N) DF$$

These measure are unaffected by adding the constant $2N \sum p \ln p$, which gives

$$AIC' = 2 N \sum p \ln (p/q) + 2 DF.$$

$$BIC' = 2 N \sum p \ln (p/q) + \ln(N) DF$$

The first term of AIC and BIC is now the familiar likelihood-ratio (LR) Chi-square measure of a model. In OCCAM, AIC and BIC are given relative to a reference model. When the reference model is the the top of the lattice of structures, namely the data, good models have low values of these measures, since LR, the model error, is ideally small and so is DF, the model complexity. When the reference model is the bottom of the lattice of structures, namely the independence model, which is the convention used in this study,

$$\Delta AIC = AIC(ind) - AIC(model) = \Delta LR + 2 * \Delta DF$$

$$\Delta BIC = BIC(ind) - BIC(model) = \Delta LR + \ln(N) * \Delta DF$$

In this case ΔAIC and ΔBIC have *high* values for good models, since ΔLR is the information *captured* in the model, and ΔDF , which is always negative, diminishes the measure the more complex the model is. Including the $\ln(N)$ factor in ΔBIC penalizes more complex models. BIC is thus more conservative than AIC in recommending departures from the reference independence model. In our experience, models picked by ΔBIC do better on generalization (test or recall data) than the more complex models picked by ΔAIC .

3. The Australian Credit Dataset

The University of California at Irvine maintains a repository of machine learning databases, including the Australian Credit dataset [18].

The dataset has 690 records, of which 37 were discarded due to missing data, leaving 653 records. There are 15 independent variables and one dependent variable. The 15 independent variables include 6 continuous variables (B,C,H,K,N,O), four binaries (A,I,J,L), and five multi-value nominal (D,E,F,G,M). The binary dependent variable, Z, codes for whether or not credit was granted. The data is encoded, and there is no information on the meanings of any of the variables or their values.

To allow appropriate processing by the NN, the multi-valued nominal variables were recoded into appropriate numbers of 0/1 binaries, giving a total of 41 input nodes.

Looking ahead slightly, the tenth variable (I) proved to be such a strong predictor of the outcome that it dominated all the others. In order to make the test more difficult, this variable was dropped from the analysis.

The data was split into five independent Learning (training) set (588 records) / Recall (test) set (65 records) partitions. Only data from the Learning sets was used to select and fit RA models.

4. Procedure

The first step in reconstructability analysis is to bin any continuous variables in the dataset. The choice of the number of bins requires balancing the desire for maximum predictive power, which calls for high granularity, and also for statistical significance which, for complex models and low sample size, calls for low granularity; also issues of computational load may be involved. In this case, the six continuous variables were each binned into five bins of approximately equal frequency. Next, the OCCAM software was used to process this version of the dataset. Table 1. shows ten typical models (of almost 3,000) generated from one of the Learn datasets.

The IV component in the models contains all the inputs, i.e., ABCDEFGHJKLMNO. The two highlighted models (rows 4 and 5) were used to determine the variables for the NN. The other models show the range of possibilities and their usefulness. Row 10 is the independence (bottom) model, where nothing predicts Z. Rows 3, 8, 9, and 10 are single predictive component models, which have no loops. Rows 2, 4, 5, 6, and 7 contains models with loops, when relations exist among the inputs. Row 1 contains a model with loops (involving

F, N, and H) even if the IV component is omitted. Only rows 4 and 5 have positive ΔAIC and ΔBIC values.

Table 1. Examples of OCCAM models.

Model refers to the RA model under consideration, **dDF**, the degrees of freedom used up by that model, **Inf**, the information content of the model, and **dAIC** and **dBIC** the Akaike and Bayesian Information Criteria

MODEL	ΔDF	Inf	ΔAIC	ΔBIC
IV:BFNZ:FHZ:HNZ:				
KZ	425	0.76	-238	-2098
IV:CZ:FHNZ:OZ	357	0.64	-195	-1758
IV:FKOZ	349	0.49	-302	-1830
IV:HZ:KZ:OZ	12	0.31	226	173
IV:HZ:JZ:OZ	9	0.30	225	186
IV:AZ:MZ	3	0.01	-0.33	-13.5
IV:AZ:LZ	2	0	-0.73	-9.48
IV:LZ	1	0	0.62	-3.76
IV:AZ	1	0	-1.45	-5.82
IV:Z	0	0	0	0

Across the five data partitions, model IV:HZ:KZ:OZ was selected three times by Occam as having the highest dBIC values. Model IV:HZ:JZ:OZ was selected twice. For this reason, we tested the NN with inputs HJO and HKO.

4.1. The Neural Net

The NN was tested against two versions of the data – binned and continuous. Also, NNs were examined with all fourteen inputs, and with the two sets of three inputs that were selected by RA preprocessing. In a standard NN approach, the original (unbinned) versions of the continuous variables were used, with their values normalized so they all lay between one and zero. In order to see how much the NN was dependent upon the continuous nature of the variables, we also used the binned version. We would expect that the use of continuous variables would improve the NN accuracy. The NNs used had one input node for each variable or coded value thereof, one bias node, and one output node. The number of nodes in the hidden layer was the sum of the input and output nodes. The hidden and output nodes used a log-sigmoid transfer function with continuous outputs that range from 0 to 1. The input nodes connected only to the hidden layer. One of the three-input NNs is shown in Figure 1.

The NN was built and tested using the NeuralWorks tool from Neuralware, Inc. During training, the errors were computed based on the continuous outputs. For testing purposes, since the object was classification, the

softmax function of the NN tool was used to force outputs to 1 or 2. Each initialization of the NNs was trained and tested on the 588-record Learn set using shuffle and deal

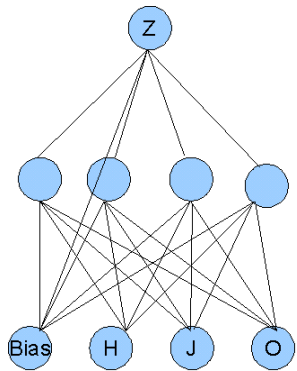


Figure 1. Neural Net Structure.

randomization and a 16-record update cycle. Learning rate was 0.8 and momentum was 0.6 throughout the process. Every 600 training events the net was tested against the full Learn set, and saved if it produced improved results. When no improvement was seen for ten of these learn/test cycles, the last saved net was tested on the Recall set. For each data partition and each set of variables, the NN was initialized 100 times and the results saved and averaged.

4.2. The Rule Set

In addition to using the RA technique to decide on the best inputs for the NN, we also used the variables from the three variable sets to define an array of decision rules that could be basis for a lookup table (Table 2). The decision rule is derived by comparing for every input state (HJO) the two conditional probabilities, $p(Z=1|input)$ and $p(Z=2|input)$. The rule is: predict $Z=1$ or $Z=2$, whichever conditional probability is greater.

As an example, a total of 60 rules captured all the information available in the three variables HJO. Table 2. shows the process and resulting rule set for an example twenty rules using the HKOZ model, based on data from the Learn set. The rules were constructed by counting the instances of each outcome (1 or 2) in the output variable for a given set of values in the input variables and assigning a rule based on the majority of the outcomes. In the learn dataset for HKOZ there were 51 instances *observed* where $H = 1, J = 1$ and $O = 1$. The *calculated* probability for model IV:HZ:KZ:OZ, that Z would have value of 1 (credit approved) was 16.78%, and the *calculated* probability that Z would have a value of 2 (credit denied) was 83.22%. The rule therefore assigns all

Table 2. Twenty sample rules.

Rules are taken from the sixty rule set for model HJOZ, based on the learning data. Column HJO shows the values possible for those variables. Column Freq is the number of cases observed in the Learn set. Columns Z1 and Z2 show the calculated percentage of outcomes for that HJO combination. If Z1 is higher, then the Rule for that combination of input values is set to one. Tied rules were assigned a value of 2, since that was the majority outcome in the Learn set. The Score column counts what proportion of the Z-values each rule correctly captures.

HJO	Freq	Z=1	Z=2	Rule	Score
111	51	16.78	83.22	2	76.47
112	23	2.91	97.1	2	91.3
113	11	13.59	86.41	2	90.91
114	16	39.95	60.05	2	56.25
222	8	8.52	91.48	2	75
223	11	32.85	67.15	2	81.82
224	4	67.42	32.58	1	50
231	2	76.78	23.22	1	100
232	2	32.92	67.08	2	50
313	11	25.63	74.37	2	54.55
314	6	59.32	40.68	1	66.67
321	9	64.27	35.73	1	66.67
423	9	67.76	32.25	1	66.67
424	6	89.89	10.11	1	100
433	16	91.72	8.28	1	93.75
434	14	97.91	2.09	1	92.86
531	15	95.89	4.11	1	100
532	3	77.58	22.42	1	66.67
533	3	94.79	5.21	1	100
534	24	98.72	1.28	1	100

future (Recall) instances of $H = 1, K = 1, O = 1$ to the *credit denied* category. Since the Learn set had a larger number of instances where $Z = 2$, ties ($Z1 = Z2$) were broken by assigning an outcome of 2 for each.

5. Results

The results are shown in Tables 3 and 4. Classification performance of the model-based NNs is shown in Table 3, in the sense of percentage of Recall records correctly classified. As expected, the 14-variable NNs using continuous variables outperformed those using binned variables, and a single tail paired t-test shows this significant at the 0.069 level.

Table 3. NN-based classification performance.

Performance of OCCAM models on the 65-record Recall set. The first column (**# of Vars**) is the number of variables used in the model. Column **Data Format** reports if the continuous variables were binned or not. Column **Vars** lists the variables used. The average **Score** is reported in the next column, along with the **Standard Deviation**.

# of Vars	Data Format	Vars	Score	Std Dev
14	Bin	A-H, J-O	75.0%	3.0%
14	Cont	A-H, J-O	79.1%	5.0%
3	Bin	HJO	76.6%	6.0%
3	Cont	HJO	77.2%	7.0%
3	Bin	HKO	69.9%	8.0%
3	Cont	HKO	76.9%	6.0%

Dropping the number of variables to three, either HJO or HKO, and still using continuous versions of the continuous variables produced results that were not significantly different from the 14-variable models, at the 0.41 and 0.29 levels, respectively.

Looking now at the performance of the ruleset, obtained directly from the data for the variables selected by RA (Table 4), we see that the results for the HJO and HKO rulesets are close, and in fact the difference is non-significant at the 0.26 level. In addition, we see that the ruleset is competitive with the 3-variable NNs, and the differences are non-significant at the 0.35 and 0.13 levels, respectively for HJO and HKO.

Table 4. Rule table classification performance.

Performance of OCCAM-model-based rule tables. The first column (**# of Vars**) is the number of variables used in the model. Column **Vars** lists the variables used. The average **Score** is reported in the next column, along with the **Standard Deviation**.

# of Vars	Vars	Score	Std Dev
3	HJO	77.8%	5.3%
3	HKO	79.7%	6.0%

6. Discussion

We have shown that applying reconstructability analysis allows us to reduce the number of variables in a standard problem to a small subset of the original, and that this reduction allows the creation of simple NN

architectures that have most of the predictive power of maximally complex NNs. Since a simpler NN that can learn the training set is, in theory, more likely to generalize well compared to a more complex NN of equal performance, it is to be preferred. In the present study, the simpler NN did not actually do better than the full input set NN, but simpler NNs are also to be preferred because they are easier to interpret and are trained faster. We also find that predicting the output with a simple and transparent look-up table obtained directly by RA modeling performs as well as NNs trained on the same data subsets.

7. References

- [1] Lendaris, G., Shannon, M., and Zwick, M. 1999. "Prestructuring Neural Networks for Pattern Recognition Using Extended Dependency Analysis", invited paper, *Applications and Science of Computational Intelligence II AeroSense'99*, Orlando, FL, SPIE. <http://www.sysc.pdx.edu/download/papers/99spie.pdf>
- [2] Chambless, B., and Scarborough, D. (2001). "Information-Theoretic Feature Selection for a Neural Behavioral Model", *International Joint Conference on Neural Networks (IJCNN)*, Washington D.C. <http://www.sysc.pdx.edu/download/papers/feature.pdf>
- [3.] Shannon, T., and Zwick, M. (2004). "Directed Extended Dependency Analysis for Data Mining." *Kybernetes*, vol. 33, No. 5/6, pp. 973-983.. <http://www.sysc.pdx.edu/download/papers/99spie.pdf>
- [4] Ashby, W. R. 1964. Constraint Analysis of Many-Dimensional Relations. *General Systems Yearbook*, 9, pp. 99-105.
- [5] Klir, G. 1986. "Reconstructability Analysis: An Offspring of Ashby's Constraint Theory." *Systems Research*, 3 (4), pp. 267-271.
- [6] Zwick, M. (2001a). "Wholes and Parts in General Systems Methodology." In: *The Character Concept in Evolutionary Biology*, edited by Gunter Wagner. Academic Press, New York, 2001a, pp. 237-256. <http://www.sysc.pdx.edu/download/papers/wholesg.pdf>
- [7] Zwick, M. (2004). "An Overview of Reconstructability Analysis." *Kybernetes*, vol. 33, No. 5/6, pp. 877-905. <http://www.sysc.pdx.edu/download/papers/ldlpitf.pdf>
- [8] Bishop, Y., S. Feinberg, and P. Holland, 1978. *Discrete Multivariate Analysis*. MIT Press, Cambridge.

- [9] Knoke, D. and P.J. Burke 1980. *Log-Linear Models*. (Quantitative Applications in the Social Sciences Monograph # 20). Sage, Beverly Hills.
- [10] Krippendorff, K. 1986. *Information Theory: Structural Models for Qualitative Data*. (Quantitative Applications in the Social Sciences #62). Sage, Beverly Hills.
- [11] Klir, G. 1985. *The Architecture of Systems Problem Solving*. Plenum Press, New York.
- [12] Chambless, B., Lendaris, G., and Zwick, M. (2001). "An Information Theoretic Methodology for Prestructuring Neural Networks", *International Joint Conference on Neural Networks (IJCNN)*, Washington D.C. <http://www.sysc.pdx.edu/download/papers/nnra.pdf>
- [13] Hosseini, J.C., R. R. Harmon, and M. Zwick, "Segment Congruence Analysis Via Information Theory." *Proceedings, International Society for General Systems Research*, Philadelphia, PA, May 1986, pp. G62 - G77. <http://www.sysc.pdx.edu/download/papers/inftheoretic.pdf>
- [14] Zwick, M. (2001b). "Discrete Multivariate Modeling": <http://www.sysc.pdx.edu/dmm.html>
- [15] Willett, K., and Zwick, M. (2004). "A Software Architecture for Reconstructability Analysis." *Kybernetes*, vol. 33, No. 5/6, pp. 997-1008. <http://www.sysc.pdx.edu/download/papers/kenpitf.pdf>
- [16] Schwartz, G. (1978). "Estimating the dimension of a model." *Annals of Statistics* 6, pp. 461-464.
- [17] Akaike, H. 1994. Implications of Informational Point of View on the Development of Statistical Science. In *Proceedings of the First US/Japan Conference on the Frontiers of Statistical Modeling: An Informational Approach*, H. Bozdogan, ed., pp. 27-38, Kluwer Academic Publishers, the Netherlands.
- [18] Newman, D.J. & Hettich, S. & Blake, C.L. & Merz, C.J. (1998). UCI Repository of machine learning databases [<http://www.ics.uci.edu/~mllearn/MLRepository.html>]. Irvine, CA: University of California, Department of Information and Computer Science.