EXPERIENCES OF NEURAL NETWORKS IN PRACTICE

John Murray, Murray Computing

Journal of Targeting, Measurement and Analysis for Marketing Vol 2, No 1 Summer 1993

Received 9th June, 1993

ABSTRACT

Neural networks are increasingly replacing statistical techniques, such as regression, discriminant analysis and CHAID, to build behavioural models to predict response to direct marketing campaigns.

The paper describes my own induction to the practical application of the technique and, coming from a statistical background, how the neural network models was applied to an existing system with the minimum of redevelopment work.

INTRODUCTION

Since its foundation, Murray Computing has been developing a marketing information management system to form the core of our bureau business activities. It was initially designed to use score-card models built using linear and log-linear regression.

Since January, research into neural networks has been undertaken comparing the performance and ease of use against other techniques. The objective being to incorporate neural networks into the existing system without incurring major redevelopment overheads.

BACKGROUND

The Murray Computing system was written using object-orientated programming techniques. The benefits of this are faster development, easier maintenance, more robust software, and greater extensibility of the program code.

In order to keep run times to a minimum, the objective was to incorporate all analysis and modelling into the main system using dynamic data transfer. This would eliminate the need for importing and exporting files as would be necessary with many proprietary packages.

The regression package was implemented as a C++ class library, allowing records from the marketing databases to be passed directly rather than through external data streams. It was desirable, therefore, that a neural network package, could be used in a similar manner.

INITIAL EXPERIENCES

As an introduction to the subject, "Brainmaker" neural network simulator was purchased from California Scientific Software. This was used to build models which were then compared with regression and CHAID models built over the previous few months.

During this exercise, many of the pitfalls of neural networks were discovered, and often left me wondering whether the technique was the panacea some claim it to be.

However after several months experience, and detailed analysis of the successes and failures, I reached a stage where I can apply neural networks with confidence in most applications. All models built are exhaustively tested against validation datasets and the gains charts compared with those produced by other techniques.

Marketing data, by its nature, has a high level of noise, strong relationships between variables, and a large proportion of variables which do not contribute to the end result. Because of these factors, the optimum network architecture and parameters can be difficult to find and often several attempts may be required to obtain the best model.

Another requirement was that the software must be capable of running in a reasonable amount of time on a standard 486DX2 50Mhz PC, running Microsoft Windows 3.1, without the need for accelerator cards.

THE TESTS

The tests conducted were, unfortunately, not scientifically designed but determined by the data sources available to me.

Four predictive models were built:

1. Package holidays mailing respondents.
2. Computer software purchasers (business users).
3. Private medical insurance holders.
4. Insurance policy customers.

The package holiday model worked very well, it out-performed the comparative regression and CHAID models. The type of holiday being promoted was fairly high-brow, but still had many sub-markets e.g. young professional recently married and older 'empty nesters'. Equal numbers of respondents and non-respondents were used for this test.

The computer software purchasers model was built alongside a CHAID model, and both performed almost equally as well on the validation sample. Regression was not used because of the highly interactive nature of the business data. In addition to past purchase history, information was available about company size, turnover, number of employees, SIC code, number of personal computers, and other software packages employed. The input dataset, which had around 20% purchasers and 80% prospects, was presented to the training program in its natural state and then again in equal proportions of purchasers and prospects. The latter produced the best model, and improvement in gain of 30% was noted over the top CHAID segments.

With the private medical insurance data, the dataset was presented to the training algorithm in its natural form: i.e. 9% plan holders, 91% prospects, the results of this were disappointing. Equal proportions of plan holders and prospects were presented to the training algorithm resulting in a 250% improvement in predictive performance over the previous model and a 20% improvement over CHAID.

The insurance policy dataset, and a 35% improvement over regression was noted in the equal proportions model.

Generally, the tests showed an improvement in performance compared with other techniques, but only when the training data was presented in equal proportions.

TECHNIQUES USED

The tests were initially conducted with Brainmaker, and later tests were carried out using Murray Computing's own neural network software. This uses a simple multilayer perceptron using a back-propagation learning rule with a gain term (learning rate) and a momentum term (smoothing rate).

In dealing with marketing data, the objective is to produce the best predictive model with an ability to generalise. Forcing a model to converge during training can compromise its ability to generalise. Therefore, emphasis was placed on validating the model using unseen data.

Writing our own software gave us the flexibility to experiment with the algorithm and experimentation yielded several changes resulting in performance improvements. These were:

1. When dealing with continuous variables, data was normalised into the range +1 to -1 using the standard deviation rather than a linear compression. To allow for skews, the median rather than the mean was used as the central point. The value passed to the training function was the number of standard deviations from the median divided by 4, with limits of +4 and -4 standard deviations. The advantages of this approach are that outlying values do not extend the range too far and thereby lose the discrimination of lesser but more important values. Thus the risk of damage to the network during training caused by these values is reduced and some allowance is made for observations lying outside the scope of the data found in the training dataset.

2. Categorical data items are given the value +1 for true, -1 for false and 0 for unknown, thereby allowing for missing erroneous and incomplete variables. This is particularly useful when dealing with questionnaire derived data where questions are added, amended and removed during the life cycle of the dataset.

3. The standard sigmoid transfer function is replaced by the Hyperbolic Tangent function (tanh) producing a value in the range +1 to -1 in line with the above normalisation.

4. The gain (slope) of the transfer function was reduced during testing to provide a wider spread of values and greater discrimination at the extremes of the range.

I have been in discussion with other neural network users who have experimented with these and other adaptations with similar results.

OBSESSION WITH CONVERGENCE

In my opinion many commercial packages, particularly the cheaper ones, aim for total convergence of the training data so that the model produced gets 100% of predictions about the training dataset correct within a given tolerance.

This obsession with convergence often does not always produce the best model, and ignores the business objectives of applying the model. Some packages add hidden neurons when the convergence slows down to a predetermined rate. The resultant models lose their ability to generalise about previously unseen records and consequently are overfitted. In the worst case, the model becomes a look up table capable only of reproducing the training data.

If the aim is to find the best 10% of the file to mail, then the best model will be the one which produces the best gain in response for the top 10% of scores. In order to determine this optimum model, we built a gains chart each time the network was tested and compared the results in the area of interest only between training cycles. In our experiments the model was tested after every 10 training cycles.

COMBINATION WITH STATISTICAL TECHNIQUES

To reduce training times, and minimise the degrees of freedom in the model, experiments were conducted using statistical techniques to eliminate variables not contributing to the results. Three techniques were tested:

1. A univariate z-score of difference of proportions only using variables with a z-score greater than 1.96 (95% confidence interval). In categorical data each category would be treated as a separate binary variable and used independently of the other categories.

2. A uni-variate chi-square approach, where for categorical data, the whole variable is used if the significance is 5% or less (p-value <0.05).

3. Running a CHAID analysis of the dataset and use only those variables which feature in the CHAID model.

Of these approaches CHAID produced the best results, followed by z-score then the uni-variate chi-square. In these all cases an improvement in the model's ability to generalise was noted.

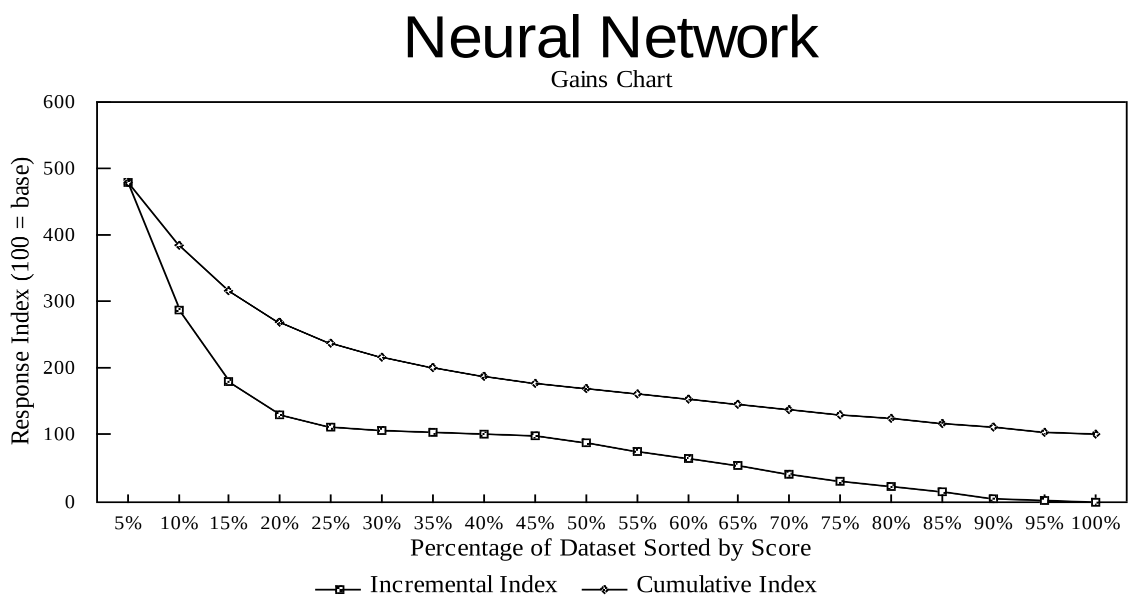
I have seen some experiments using regression to pre-set weights in a neural network prior to training rather than randomising, although I have yet to try this. Training times can be reduced considerably by this approach although the performance of the networks was generally not as good as those built using an initially randomised network. I feel that there is a danger that the underlying assumptions of regression, e.g. normal distribution, independence of variables, will prejudice the networks pattern recognition abilities.

IMPLEMENTATION

The cost effective implementation of models for prospect selection is vital if neural networks are to become commonplace in marketing applications. As a neural network is basically a scoring technique, most selection systems designed to work with regression scoring models can be adapted to use neural networks with minimal or no additional development effort. Many neural network packages produce ASCII files containing the outputs of the model in formats identical to those produced by the leading statistical packages, allowing direct import into many database packages.

The programming of one's own neural network training package should not be ruled out. The process is straight forward and well within the capabilities of the average programmer. The main advantage being the complete integration of the neural network into the core business systems. The greater understanding of the internal workings of neural networks resulting from this development will also be of significant benefit to the organisation.

The Murray Computing software was designed so that for selection, the process would be almost identical to that used for regression. A pointer to the record would be passed to the network program and a score returned to the caller. The record would then be selected or rejected according to whether that score fell within a range determined from the gains chart. Thus the only amendment necessary was to incorporate the function call to the neural network program.



An example of a neural network gains chart on a test sample of private medical insurance holders

CONCLUSIONS

Neural networks are already proving their value in marketing applications. However, one has to be prepared invest a considerable amount of time (and frustration!) in order to gain the understanding necessary to get the most from technique. But it is worth it.

Writing one's own package, which can be easily done in most high level languages, allows adaptation of the algorithms to suit specific requirements or to incorporate changes in the techniques. An object-orientated approach provides greater flexibility and a neater more adaptable implementation. There are a number of texts available describing object-orientated programming of neural network training algorithms.

Do not be afraid to experiment with the algorithms, after all this is how much of the methodology developed. A lot can be gained from sharing views and experiences through one of the many discussion forums on neural networks.

If properly planned, the implementation of a neural network model selection process can be performed with minimal additional development work.

With standard multilayer perceptron networks, better results have been obtained when presenting equal proportions of respondents/non-respondents than using 'natural' proportions.

Using standard deviation based data normalisation provides better handling of outlying values.

Treating categorical variable values as Boolean +1 (true), -1 (false) and 0 (unknown) gives greater flexibility in handling unknown or erroneous values.

Testing the network should concentrate on fulfilling the business objective rather than improving the overall performance.

Using statistical techniques in conjunction with network training algorithms can produce better models and reduce training times significantly.

REFERENCES

1. Nelson, M.M. & Illingworth, W.T. (1991) 'A Practical Guide to Neural Nets' Addison Wesley.

2. Weiss S.M. & Kulikowski C.A. (1991) 'Computer Systems That Learn' Morgan Kaufmann.

3. Beale R. & Jackson T. (1990) 'Neural Computing An Introduction' Institute of Physics Publishing.

4. Laurence J. (1992) 'Introduction to Neural Networks' California Scientific Software Press.

AUTHOR BIOGRAPHY

After graduating in Mathematics and Computer Science, John Murray has worked on the design and development of computer systems for the analysis and management of marketing databases at Reader's Digest, Lloyds Bank and NDL International. In October 1992, he founded Murray Computing, a computer bureau specialising in data analysis services.