Proceedings of the IASTED International Conference
ARTIFICIAL INTELLIGENCE AND APPLICATIONS
February 16-18, 2004, Innsbruck, Austria

STEPWISE LINEAR REGRESSION FOR DIMENSIONALITY REDUCTION IN NEURAL NETWORK MODELLING

J F Dale Addison, K. J McGarry, Stefan Wermter, J MacIntyre
University of Sunderland, Centre for Adaptive Systems, School of Computing and Technology, The Sir Tom Cowie Campus at St Peters, David Goldman Informatics Centre, St Peters Way, Sunderland, SR6 0DD
England

Abstract

This work considers the applicability of applying the derivatives of stepwise linear regression modelling (specifically the *p*-values which indicate the importance of a variable to the modelling process) as a feature extraction technique. We utilise it in conjunction with several data sets of varying levels of complexity, and compare our results to other dimensionality reduction techniques such as genetic algorithms, sensitivity analysis and linear principal components analysis prior to data modelling using several different neural network models. Our results indicate that stepwise linear regression is highly effective in this role with results comparable to and sometimes superior then more established techniques

Key Words

Linear regression, Feature extraction/selection, neural networks

1. Introduction

Data modelling techniques such as neural networks are highly efficient function approximators capable of modelling any non-linear function to a high degree of accuracy. However high dimensional input spaces can interfere with the accuracy of the classification or prediction accuracy of the model being developed. To address this problem, dimensionality reduction methods are now regarded as an essential activity prior to modelling the feature space as a means of ensuring that highly correlated attributes are eliminated from the input space. Alternatively the most representative features from a particular attribute can be removed and condensed into a smaller attribute set.

Many of these techniques such as genetic algorithms [1] and self organising feature maps [2] are highly heuristic in nature, whilst principal components analysis makes assumptions as to the linearity of relationships between the data items in the input set [3]. Techniques such as sensitivity analysis [4] though highly effective are computationally intensive in their constant replacement of attributes in the modelling process in an effort to find the optimal input configuration. This may be a serious constraint if speed is of crucial importance. In this paper we extend previous work on the use of stepwise linear regression for classification problems [5] by considering the by-products of stepwise linear regression, namely the standard deviation and variance of the X input values in relation to the Y target output value, the degree of correlation between data in the input set and the target regression value, but most importantly the stepwise feature allows the user to observe which of the input features should be retained. We utilise several publicly obtainable data sets which vary in both, the size of the input feature space and the degree of overlap in the class structure. The paper is organised as follows. Section 2 introduces stepwise linear regression as a modelling technique, with emphasis on the residuals produced to determine parameter retention. Section 3 considers other feature extraction/selection techniques such as genetic algorithms, linear principal components analysis and sensitivity analysis. Section 4 considers neural network architectures used for regression modelling, specifically generalised regression networks, Radial basis function networks and multi-layer perceptrons. Section 5 discusses in more detail the data sets used. Section 6 outlines the methodology used and presents the results of each feature extraction/selection techniques. The paper concludes by considering the results and their implications for data modelling.

2. Stepwise Linear Regression

Stepwise regression techniques are designed to add or remove variables which are inputs to the regression model. The objective of which is to identify a useful subset of the predictors which can be regressed onto a single output variable as stated in (1)

$$Y' = a + b_1 X_1 + b_2 X_2 + \ldots + b_k X_k \quad (1)$$

where Y' represents the predicted value of *Y, X* are the input variables, and *a* and *b* determine the degree of correlation. This formula assumes that all values of *Y* are *X* values the variance of *Y* is fixed, whilst for any fixed combination of X values, the value of Y is normally distributed. The maximum value of the correlation coefficient between observed *Y* values

the predicted values for Y' are the obtained values of $a, b_1, b_2 .... b_k$ that can be used to minimise the sum of squared errors (SS) of prediction of the residual sum of squares.

$$SS_{res} = (Y - Y')^2 \quad (2)$$

To simplify the explanation we will confine this discussion to two variables for *X*. The required value of *a* is given by the equation.

$$a = \overline{Y} - b_1\overline{X}_1 - b_2\overline{X}_2 \quad (3)$$

where $\overline{X}$ and $\overline{Y}$ represent the mean values for the input parameters *X* and the regressed *Y* value. Substitution of this value into (1) gives:

$$Y' = \overline{Y} + b_1(X_1 - \overline{X}_1) + b_2(X_2 - \overline{X}_2) \quad (4)$$

or:

$$Y' = \overline{Y} + b_1x_1 + b_2x_2 \quad (5)$$

Where x_1 and x_2 represent the two input variables. Minimisation of the residual sum of squares requires that the values for b_1 and b_2 (the slope of each independent variable) satisfy (6) and (7)

$$b_1x_1^2 + b_2x_1x_2 = x_1y \quad (6)$$

and:

$$b_1x_1x_2 + b_2x_2^2 = x_2y \quad (7)$$

The solution of both equations for unknown values of b_1 and b_2 can be found by multiplying (6) by x_1^2 and (7) by x_1x_2 gives according to (6)

$$b_1 = \frac{(x_1y)(x_2^2) - (x_2y)(x_1x_2)}{(x_1^2)(x_2^2) - (x_1x_2)^2} \quad (8)$$

and for b_2,

independent of each other and that *Y* is a linear function of *X*. For a fixed combination of

$$b_2 = \frac{(x_2y)(x_1^2) - (x_1y)(x_1x_2)}{(x_1^2)(x_2^2) - (x_1x_2)^2} \quad (9)$$

By retaining the standard deviation and variance of the *X* values in relation to *Y*, together with the degree of correlation between variable allows the user to retain or remove data which is superfluous to the model. The technique was previously used by the authors on a data set of gas turbine readings consisting of 18 exhaust emission sensors, and three other readings relating to fuel intake, the mixture of fuel air in the turbines combustion chamber, and the level of steam generated by the turbine. In classification mode our findings using stepwise linear regression in this way achieved comparable accuracy to other dimensionality reduction techniques whilst utilising around 30 percent of the features. [5] In this work we utilise the same data set but this time in regression mode, and compare its performance to three other data sets comprising of measurements of abalone sea creatures, prediction of Ph levels in drinking water sample, and the prediction of vibration levels from a high speed machine drilling tool.

3. Other Feature Selection/Extraction techniques.

We compare our results for stepwise linear regression with three well known feature extraction/selection techniques namely Linear principal components analysis [3] Genetic algorithms [1] and sensitivity analysis [4]. In previous experimental and practical work the authors have found each of these to be extremely effective for dimensionality reduction [5, 13]. We briefly outline each method below.

3.1. Linear Principal Components Analysis (PCA)

For simplicity we illustrate Principal Components Analysis by projecting data from two dimensions to one. A linear projection requires the optimum choice of projection to be a minimisation of the sum-of-squares error [3, 7]. This is obtained first by subtracting the mean of the x values of the data set. The covariance matrix is then calculated and its eigenvectors and eigenvalues are found. The eigenvectors corresponding to the *M largest* eigenvalues are retained, and the input vectors x^n are subsequently projected onto the eigenvectors to give components of the transformed vectors z^n in the *M*-dimensional space. By retaining a subset $M < d$ of the basis vectors μ_i so that only M coefficients z_i are used allows for replacement of the remaining coefficients by

constants b_i. This allows each x vector to be approximated by an expression of the form.

$$\tilde{\chi} = \sum_{i=1}^{M} z_i u_i + \sum_{l=M+1}^{d} b_i u_i \qquad (10)$$

where μ_i represents a linear combination of d orthonormal vectors.

3.2 Genetic Algorithms

Genetic algorithms represent an optimisation technique based upon the Holland algorithm, [8] which uses "elitism" (defined as the best string from each generation remaining unaltered) to breed increasingly superior data strings according to a pre-defined fitness function. The breeding process also ensures that desirable qualities are passed to the next generation. The fitness function is based upon the verification error plus the unit penalty normalised linearly before the selection process, which ensures the best-worst fitness ratio, is held at a constant 2:1 ratio. Constant selection pressure is maintained for the duration of the algorithm which assists locating "epistasis" (high levels of correlation) between attributes in the data set

3.3 Sensitivity Analysis

In this study we use sensitivity analysis by treating each attribute in turn as if it were "unavailable" [4]. Each model has a defined missing value substitution procedure, which makes predictions in the absence of values for one or more attributes. The sensitivity of a particular variable, v, is defined by running the network on a set of test cases, and accumulating the network error. The network is then run again using the same cases, but this time replacing the observed values of v with the value estimated by the missing value procedure and again accumulating the network error. The measure of sensitivity is the ratio of the error with missing value substitution to the original error. The more sensitive the network is to a particular input the greater the deterioration and therefore the greater the ratio.

4. Neural Network Architectures for Regression Modelling

In this section we discuss neural network architectures specifically from a regression perspective. Neural networks are known to be highly effective function approximators to an arbitrary level of accuracy, owing to their "input-middle-layer-output structure". In this study we have consider three architectures which highlight two different approaches to the problem of regression modelling in a high dimensional feature space. Namely a) the use of models whose units compute a non-linear function of the scalar product of the input and a weight vector, and b) activation of a hidden unit which is determined by the distance between the input and prototype vector. The former belongs to the class of multi layer feed forward neural networks such as Multi-layer perceptrons [9] whilst the latter encompasses the radial basis function [10] and Generalised regression [11] architectures. We begin with a description of the Multi-layer perceptron architecture.

4.1 Multi-Layer Perceptrons

In these networks we are concerned with representing d inputs, to M hidden units and c output units. The hidden units are transformed using a weighted linear combination of the d input values and adding a bias value, whilst the activation of hidden unit j is obtained by linear summation. The activation functions can be of several types with the most commonly used being the sigmoid. The network outputs are obtained by transforming the activations of the hidden units using a second layer of processing elements. Each output unit k is pooled into a linear combination of the outputs from the hidden units. This can be expressed by (11) as:

$$y_k = \tilde{g}\left(\sum_{j=0}^{M} w_{kj}^{(2)} g\left(\sum_{i=0}^{d} w_{ji}^{(1)} x_i\right)\right) \qquad (11)$$

where y_k is the activation of the kth output unit and $\tilde{g}$ is a function which absorbs the bias of the network into its weights in the linear combination of the outputs of the hidden units, and g is a weighted linear combination of the d input values. Figure 1 shows the architecture of a typical MLP network.

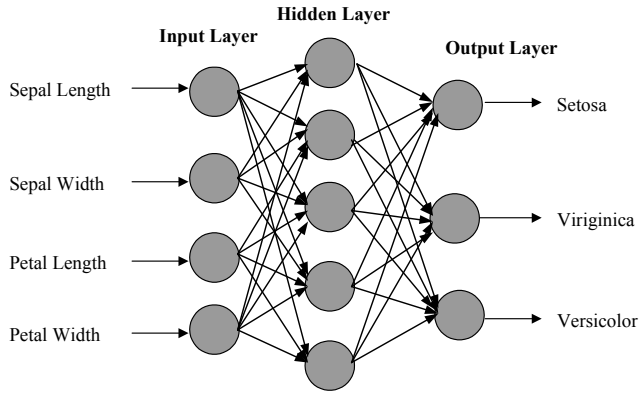

Fig 1. Multi-layer perceptron architecture for the famous Fisher iris data set problem with four inputs for petal and sepal length and width, and three outputs for the three types of flower.

4.2 Radial Basis Function Networks.

Radial basis function networks are designed to provide an interpolation function which should pass through every data point. The objective being to generate a function which is smooth enough to provide the best generalisation and average over noise within the data. In such networks the number of basis functions *M* is typically less than the number of data points *N*. In addition the centres of the basis functions are determined not by the data, but during the training process, as are the width of each basis function. Finally a bias parameter is included into the linear sum which acts as a compensator for the difference between the average value over the data set of the basis function activations and the related average value of the targets. This can be expressed by (12)

$$y_k(x) = \sum_{j=1}^{M} w_{kj}\phi_j(x) + w_{k0} \qquad (12)$$

where ϕ_j is the individual basis function width parameter, and w_{k0} is the bias parameter. Figure 2 shows the typical architecture of an RBF network.

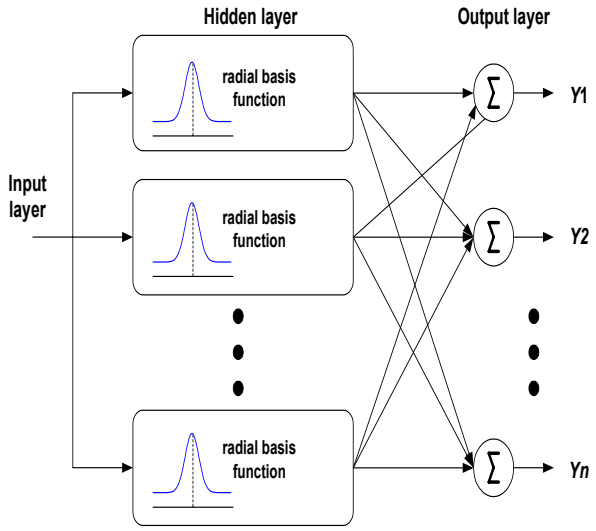

Figure 2 : Configuration of a typical radial basis function network

4.3. Generalised Regression Neural Networks

Generalized regression neural networks (GRNNs) work in a similar fashion to Probabilistic neural networks (PNN's), but are dedicated to perform regression tasks. [11] As with the PNN, Gaussian Kernel functions are located at each training case. Each case can be regarded,as evidence that the response surface is a given height at that point in input space, with progressively decaying evidence in the immediate vicinity. The GRNN copies the training cases into the network to be used to estimate the response on new points. The output is estimated using a weighted average of the outputs of the training cases, where the weighting is related to the distance of the point from the point being estimated (so that points nearby contribute most heavily to the estimate).

The first hidden layer in the GRNN contains the radial units. A second hidden layer contains units which help to estimate the weighted average. Each output has a special unit assigned in this layer which forms the weighted sum for the corresponding output. To obtain the weighted average from the weighted sum, the weighted sum must be divided through by the sum of the weighting factors. A single special unit in the second layer calculates the latter value. The output layer then performs the actual divisions (using special division units). Hence, the second hidden layer always has exactly one more unit than the output layer. In regression problems, typically only a single output is estimated, and so the second hidden layer usually has two units.

The GRNN can be modified by assigning radial units which represent clusters rather than each individual training case: this reduces the size of the network and increases execution speed. Centers can be assigned using any appropriate algorithm such as sub-sampling or K-means.

5. Experimental Data Sets

We have used four different data sets for our experiments Two of them are available from the UCL database repository [12] specifically the Abalone and Water treatment databases. The third is the original gas turbine data set but this time used in regression mode. The fourth is taken from a project within the University of Sunderland's MINICON project for predicting time to failure of a high speed machine tool with a large number of inputs. Table 1 gives further details of the data sets.

Table 1: Details of the data sets used

Data set	Inputs	Attributes	Comments
Abalone	8	4177	Highly unstructured domain
Turbine	21	240	Ill structured domain, no missing values
Water	37	527	Ill structured domain, missing values
Machine Tool	401	141	Well structured domain, missing values

6. Methods and Results

We began this experiment by inputting the data set to each architecture without the benefit of any feature extraction/selection. This provided us with a benchmark of performance. Then we applied each of the dimensionality reduction techniques to the input sets, and trained each network on the reduced data sets. Tables 2-5 detail the results of our experiments. The data sets were partitioned into training, verification and test set in the ratio 70-20-10 respectively. The networks were trained on the training set, and the accuracy of their predictions verified using the verification and test sets. We include the results of stepwise linear regression as part of the comparison. The result is expressed as the variance between training and verification sets, i.e. 31% refers to a models which explains 31% of the model accuracy in the sample set. This result is given in the *Result* field.

Table 2: Results obtained from the Abalone data set

		RBF	MLP	GRNN
Technique	***Inputs***	***Result***	***Result***	***Result***
Sen Anal	6	34%	33%	31%
Gen Alg	6	34%	33%	31%
PCA	2	19%	18%	19%
S.L.R	6	34%	33%	31%
Normal	8	31%	33%	33%

Table 3: Results obtained from Gas turbine data set.

		RBF	MLP	GRNN
Technique	***Inputs***	***Result***	***Result***	***Result***
Sen Anal	1	99%	99%	83%
Gen Alg	20	37%	25%	18%
PCA	7	14%	38%	38%
S.L.R	13	78%	80%	76%
Normal	21	88%	37%	34%

Table 4: Results obtained from Water database.

		RBF	MLP	GRNN
Technique	***Inputs***	***Result***	***Result***	***Result***
Sen Anal	7	64%	72%	62%
Gen Alg	3	81%	80%	80%
PCA	15	66%	70%	46%
S.L.R	13	73%	83%	53%
Normal	37	75%	83%	51%

Table 5: Results obtained from machine tool database.

		RBF	MLP	GRNN
Technique	***Inputs***	***Result***	***Result***	***Result***
Sen Anal	89	16%	57%	24%
Gen Alg	338	30%	39%	32%
PCA	7	32%	38%	45%
S.L.R	100	22%	46%	37%
Normal	400	29%	41%	29%

Table 6: Equivalent results using stepwise linear regression.

Data Set	Inputs	Result
Abalone	6	53%
Gas	13	78%
Water	13	94%
Machine Tool	100	100%

7. Observations and Conclusions

This work has evaluated the effectiveness of stepwise linear regression primarily as a dimensionality reduction technique, but also from the perspective of regression modelling. Considering the feature selection/extraction aspect, we have selected a variety of databases which are both high dimensionality but also contain anomalies such as highly unstructured domains which as can be seen from the results in table 2 make accurate regression modelling extremely difficult. The use of stepwise linear regression appears most effective when combined with multi-layer perceptron architectures, using either conjugate gradients, or a combination of back-propagation of error combined with longer training epochs of the conjugate gradient algorithm.

In most cases it is comparable if not superior to most of the other methods considered however; most notably in the gas turbine data set sensitivity analysis is clearly superior. The accuracy obtained in that analysis must be tempered by the caveat that sensitivity analysis is a time consuming process because of the necessity to substitute attributes in and out of the model according to the relative accuracy of the model. We proffer a tentative conclusion that the measure of inter-correlation used by stepwise linear regression combined with its measures of variance are more effective here than using an objective function as in genetic algorithms, or relying upon a limited number of principal components analysis to model the feature space. However this conclusion requires further work on a larger variety and number of data sets before it can be stated categorically.

As a data modelling method stepwise linear regression performs exceptionally well on all but the abalone data

sets. This suggests that there is perhaps more linearity in these data sets which make them more amenable to modelling using such linear techniques.

REFERENCES

[1] E Goldberg, *Genetic Algorithms in Search Optimisation a*nd Machine Learning, Addison-Wesley, 1989.

[2] T. Kohonen, Self-organised Formation of Topologically correct feature maps. *Biological Cybernetics 43*, 56-69, 1982

[3] C. Bishop,. Neural Networks for Pattern Recognition. Oxford: University Press. 1995

[4] A Hunter,. L, Kennedy., J., Henry, and R.I , Ferguson,.: Application of Neural Networks and Sensitivity Analysis to improved prediction of Trauma Survival Computer Methods and Algorithms in *Biomedicine 62,* , 11-19, 2000.

[5] J F D., Addison, S., Wermter, and J, MacIntyre, Effectiveness of feature extraction in neural network architectures for novelty detection, *ICANN-99, Ninth International Conference on Artificial Neural Networks, E*dinburgh, UK, , pp976-981, 1999

[6] M.R Hestenes, and E Steiffel Methods of Conjugate Gradients for Solving Linear Systems, *Journal of Research of the National Bureau of Standards 49*(6): 409-436, 1952

[7] W.H.; Press, S. A. Teukolsky.; W.T Vetterling and B.P Flannery.*: Numerical recipes in C: The art of scientific computing* (second edition), Cambridge University press 1992.

[8] J. Holland, *Adaptation in Natural and Artificial systems* . MIT Press, 1975.

[9] Haykin, S.: *Neural Networks - A Comprehensive Foundation.:* Maxwell Macmillan International Publishing Company, (1995) 138

[10] Lowe, D.: 1995 *Radial Basis Function Networks.: The Handbook of Brain Theory and Neural Networks,* Cambridge, MA: MIT Press

[11] D.F. Specht, Probabilistic Neural Networks, *Neural Networks 3*(1), pp 109-118, 1990.

[12] C.L Blake. & C.J, Merz, UCI Repository of machine learning databases [http://www.ics.uci.edu/~mlearn/MLRepository .html]. Irvine, CA: University of California, Department of Information and Computer Science 1998

[13] Odin Taylor and J F Dale Addison Novelty Detection for Condition Monitoring *COMADEM 2000, 13th International Congress and Exhibition on Condition Monitoring and Diagnostic Management, Houston*, USA, pp 731-743, December 2000