Spam Detection using Generalized Additive Neural Networks

Jan V. du Toit[1,†], David A. de Waal[2]
School of Computer, Statistical and Mathematical Sciences
North-West University[1], Potchefstroom Campus
Private Bag X6001, Potchefstroom, 2520
South Africa, Tel: +27 18 2992548
Fax: +27 18 2992570
and Centre for Business, Mathematics and Informatics
North-West University[2], Potchefstroom Campus
email: {Tiny.DuToit, Andre.DeWaal}@nwu.ac.za[1,2]
†Primary recipient of correspondence

ABSTRACT

During the last decade the number of spam messages sent has increased significantly. These undesired emails place a heavy burden on end users and email service providers. As a result, a tenacious struggle to outsmart each other exists between people who send spam and the spam filter providers. Constant innovation is therefore of vital importance to curb the rapid increase of spam. In this article a Generalized Additive Neural Network (GANN) is harnessed to detect spam. This relative new type of neural network has a number of strengths that makes it a suitable classifier of spam. An automated GANN construction algorithm is applied to a spam data set. This method can perform in-sample model selection or cross-validation and produces results that compare favourable to other classifiers found in the literature. Even though neural networks in general are regarded as black box techniques, results obtained by GANNs can be interpreted by graphical methods. Partial residual plots assist the spam researcher to gain insight into the models constructed. Finally, by performing variable selection the temporal evolution of spam can be tracked. This feature ensures that the models can adapt to the ever-changing tactics of people who send spam with greater ease.

Index terms - Generalized Additive Model, GAM, Generalized Additive Neural Network, GANN, Neural Network, Spam, Spam detection.

1 INTRODUCTION

Since the late 1990s the quantity of email sent has grown exponentially. Moreover, the amount of spam (the Concise Oxford English Dictionary defines spam as "irrelevant or inappropriate messages sent on the Internet to a large number of newsgroups or users.") has increased even more. In 1998 approximately 10% of the overall mail volume was comprised of spam. By 2007 this number has increased to as much as 80% (Cranor and LaMacchia, 1998); (Goodman, Cormack and Heckerman, 2007). More than a billion spam messages are sent daily to large email services such as Microsoft's Hotmail. This deluge of unsolicited messages creates a heavy burden on both tens of millions of end users worldwide and tens of thousands of email service providers (ESPs). Spam takes away resources from users and service providers without providing any remuneration or obtaining authorization (Kiran and Atmosukarto, n.d.). Spam emails are normally sent using bulk mailers and address lists that are acquired from web pages and newsgroup archives. Their content range from deals to real estate to pornographic material.

Fortunately, considerable progress has been made in stopping spam. Of that 80% only a small fraction actually reaches end users. Currently, nearly all ESPs and most email programs include filters for spam. The problem is stabilizing from the point of view of the end user. As a result, spam is an annoyance for most users today rather than a threat to their use of email. At the same time, a rapid increase of the technology used by both spammers (people who send spam) and spam filter providers is taking place behind the scenes. Whenever spam-filtering software is improved, spammers devise more advanced techniques to defeat the filters. Constant innovation from spam researchers is therefore crucial to restrain spam from overwhelming users' inboxes.

In this article the proliferation of spam is attacked by a relatively new type of neural network. Generalized additive neural networks have a number of favourable properties which provide grounds for an investigation into the domain of spam detection. An automated construction algorithm has been developed which utilizes a greedy best-first search procedure that identifies good models in short time periods. These models proved to have high predictive accuracy and are comparable to other models found in the literature that distinguish between spam and good emails. With the automated algorithm, in-sample model selection, cross-validation, and feature selection can be performed.

The rest of the article is organized as follows. In Section 2 a brief overview of spam detection procedures is provided, ranging from early techniques to current advanced methods. The generalized additive neural network architecture and construction methodologies are discussed in Section 3. This special type of neural network is then applied to a spam data set in Section 4 and the results are considered in Section 5. Finally, some concluding remarks are presented in the last section.

2 BACKGROUND

Nearly all spam filtering systems utilize at least one machine learning component (Goodman et al., 2007) where computer

programs are presented examples of both spam and good email. The characteristics of the spam email versus the good email is then determined by a learning algorithm. Accordingly, future incoming messages can be automatically classified as probably spam, probably good, or somewhere in between.

Learning approaches were initially fairly simple and used techniques like the Naive Bayes algorithm to count how frequently each feature or word appeared in spam messages or good messages. Naive Bayes and other similar techniques require training data - known spam and known good mail - to train the system. When spam was becoming a major problem around 1998, it was relatively static. A trained filter did not need to be updated for a number of months. Certain words like "free" or "money" were sufficient indicators of spam and functioned for a lengthy period. Unfortunately, spammers adapted to the more widely deployed spam filters. They quickly learned the most obvious words to avoid and the most innocent words to add to lead the filter astray. To keep up with the spammers, it became necessary to collect increasing amounts of email as spammers made use of a wider variety of terms. Filters also had to be updated frequently. Currently, Hotmail uses more than 100,000 volunteers who are daily asked to label an email that was sent to them as either "spam" or "good" email. This feedback loop system provides Hotmail with new messages to train their filters, allowing them to respond quickly to new spammer attacks and schemes.

Apart from getting more training data, faster, much more advanced learning algorithms are currently being used. For example, algorithms based on logistic regression and support vector machines can bring down the amount of spam that bypass filtering by half, compared to Naive Bayes (Goodman et al., 2007). With these algorithms the messages are broken down into individual words and weights are "learned" for each word. The weights are carefully adjusted to obtain the most accurate results from the training data. The learning process can be a potentially time-consuming operation as tens of thousands or even hundreds of thousands of weights may require repeated adjusting. Fortunately, such computation has been made possible by advances in machine learning over the past few years. Complex algorithms like Sequential Conditional Generalized Iterative Scaling allows Hotmail to learn a new filter from scratch in about one hour with training data of more than a million emails.

In the following section a recently developed neural network architecture is employed to detect spam. This design does not suffer from the black box perception ascribed to artificial neural networks in general as visual diagnostics provide insight into the models created. Furthermore, no user input is required while an automated algorithm searches for the best model.

3 Generalized Additive Neural Networks

Spam detection can be regarded as an instance of the generic supervised prediction problem which consists of a data set having a number of cases (messages) (Potts, 1999). Each case is associated with a vector of input variables (features) x_1, x_2, ..., x_k and a target variable y. The latter represents a class label that indicates whether an email is spam or non-spam. A predictive model maps the inputs to the expected value of the target and is built on a training set where the target is known. The objective is to apply the model to new data where the target is unknown.

Generalized linear models (McCullagh and Nelder, 1989),

$$g_0^{-1}(E(y)) = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \ldots + \beta_k x_k,$$

are often used for predictive modeling. The range of predicted values are restricted by the link function, g_0^{-1}. For spam detection, the logit link

$$g_0^{-1}(E(y)) = ln(\frac{E(y)}{1-E(y)})$$

is appropriate as the expected target (probabilities) is bounded between zero and one. The parameters are usually estimated by maximum likelihood.

Multilayer perceptrons (Bishop, 1995); (Ripley, 1996); (Zhang, Patuwo and Hu, 1998) are the most widely used type of neural network for supervised prediction. A multilayer perceptron (MLP) with a single hidden layer with h hidden neurons has the form

$$g_0^{-1}(E(y)) = w_0 + w_1 tanh(w_{01} + \sum_{j=1}^{k} w_{j1} x_j) + \ldots$$

$$+ w_h tanh(w_{0h} + \sum_{j=1}^{k} w_{jh} x_j),$$

where the link function is the inverse of the output activation function. Although other sigmoidal functions could be used, the activation function in this case is the hyperbolic tangent. The unknown parameters are estimated by numerically optimizing some appropriate measure of fit to the training data such as the negative log likelihood.

A generalized additive model (GAM) is defined as

$$g_0^{-1}(E(y)) = \beta_0 + f_1(x_1) + f_2(x_2) + \ldots + f_k(x_k),$$

where the expected target on the link scale is expressed as the sum of unspecified univariate functions (Hastie and Tibshirani, 1986); (Hastie and Tibshirani, 1990); (Wood, 2006). Each univariate function can be regarded as the effect of the corresponding input while holding the other inputs constant. When a GAM is implemented as a neural network it is called a *generalized additive neural network* (GANN).

The main architecture of a GANN is comprised of a separate MLP with a single hidden layer of h units for each input variable:

$$f_j(x_j) = w_{1j} tanh(w_{01j} + w_{11j} x_j) + \ldots$$

$$+ w_{hj} tanh(w_{0hj} + w_{1hj} x_j).$$

The individual bias terms of the outputs are incorporated into the overall bias β_0. Each individual univariate function contains $3h$ parameters, where h could be different across inputs. This architecture can be extended to include an additional parameter for a direct connection (skip layer):

$$f_j(x_j) = w_{0j} x_j + w_{1j} tanh(w_{01j} + w_{11j} x_j) + \ldots$$

$$+ w_{hj} tanh(w_{0hj} + w_{1hj} x_j).$$

A backfitting algorithm is used by (Hastie and Tibshirani, 1986); (Hastie and Tibshirani, 1990) to estimate the individual univariate functions f_j. Backfitting is not required for GANNs. Any method that is suitable for fitting more general

MLPs can be utilized to simultaneously estimate the parameters of GANN models. The usual optimization and model complexity issues also apply to GANN models.

Neural network construction and interpretation in general is not a trivial task. The most common way to determine the number of hidden nodes is via trial-and-error or experiments (Zhang et al., 1998). A number of rules of thumb have also been suggested. Two examples of these are that each weight should have at least ten cases and the number of hidden nodes depends on the number of cases. To help curb the overfitting problem, some researchers have produced empirical rules to limit the number of hidden nodes. Unfortunately, none of these heuristic choices performs well for all problems. Deciding on the number of inputs is also not evident. Ideally, a small number of essential nodes are desired which can expose the unique features embedded in the data. Too little or too many input nodes can have an adverse affect on the learning or prediction capability of the network. To compound these matters, neural networks in a broad sense are regarded as black box methods. There is no explicit form to analyze and explain the relationship between inputs and the target. This causes difficulty in interpreting results from the networks. Fortunately, these concerns are addressed by the automated construction algorithm for GANNs.

Presently two algorithms exist to estimate GANN models. Potts (1999) suggested an interactive construction algorithm that makes use of visual diagnostics to determine the complexity of each univariate function. Plots of the fitted univariate functions, $\hat{f}_j(x_j)$ overlaid on the partial residuals

$$pr_j = g_0^{-1}(y) - \hat{\beta}_0 - \sum_{l \neq j} \hat{f}_l(x_j)$$

$$= (g_0^{-1}(y) - g_0^{-1}(\hat{y})) + \hat{f}_j(x_j),$$

versus the corresponding jth input are utilized for model selection (Berk and Booth, 1995); (Ezekiel, 1924); (Larsen and McCleary, 1972). When GANNs are constructed interactively, human judgment is required to interpret the partial residual plots. For a large number of inputs this can become a daunting and time consuming task. Also, human judgment is subjective which may result in the creation of models that are suboptimal. Consequently, Du Toit (2006) developed an automated method based on the search for models using objective model selection criteria or cross-validation. With this approach, partial residual plots are not used primarily for model building, but as a tool to provide insight into the models constructed. When given adequate time to evaluate candidate models, this best-first search technique is complete and optimal. Du Toit showed the algorithm is powerful, effective and produces results comparable to other non-linear model selection techniques found in the literature.

In the next section the implementation of the automated construction algorithm, called *AutoGANN*, is used to classify incoming email into spam or good messages.

4 EXAMPLE

The Spambase data set (Asuncion and Newman, 2007) has 4,601 instances where each instance denotes a single message and 39.4% are classified as spam. There are 57 continuous non-missing inputs and a binary target indicating spam (1) or non-spam (0). Most of the inputs (54) indicate how frequently a particular word or character occurred in each email and was encoded as a percentage in [0, 100]. Examples of words and characters are “business”, “credit”, “edu”, “free”, “internet”, “!”, “#” and “$”. Finally, there are three run-length inputs that measure the length of sequences of consecutive capital letters.

Kiran and Atmosukarto (n.d.) performed a number of experiments on the Spambase data set to analyze various implementation and design aspects of spam filtering. They considered eight classification algorithms, namely decision trees, support vector machines, Naive Bayes, neural networks, ensemble decision trees, boosting, bagging and stacking. For all the experiments the data was partioned into a training set and a testing set. The latter was unseen by the classifiers and performance was measured by evaluating the accuracy (Table 1). Although no indication was given how the data was partitioned, additional experiments were conducted to determine if the accuracies could be improved. A random 50% - 50% split into spam and good emails for the training set, a leave-one-out cross-validation and k-fold cross-validation with $k = 10$ were carried out without significant improvements from the above results. It was decided to split the data into 70% (training) and 30% (testing) subsets. The AutoGANN system constructed a model with an accuracy of 94.28%, thereby establishing a third place on the list of classifiers in Table 1. Of the 42 inputs selected, 31 were identified as having linear relationships with the target and 11 inputs had non-linear relationships with the target. Examples of inputs (words) removed from the model are *addresses*, *direct*, *mail*, *people*, *table* and *your*. It would seem as if these words do not differentiate between spam and good emails.

Classifier	Accuracy (%)
Ensemble decision tree	96.40
Adaboost	95.00
Stacking	93.80
Support vector machine	93.40
Bagging	92.80
Decision tree	92.58
Neural network	90.80
Naive Bayes	89.57

Table 1: Classifier accuracy results.

Although the Ensemble decision tree and the Adaboost methods produce more accurate results than the AutoGANN system, their results are very difficult to interpret. The Ensemble decision tree method creates a magnitude of small models and combines the results into one complicated final model. The separate models, from which the final model is constructed, are usually not available and the results are therefore very difficult to interpret. Adaboost calls a classifier (such as a decision tree method) repeatedly, where incorrect classified cases are given more weight in subsequent iterations/models. The final model is also difficult to interpret as the individual models constructed during each iteration of the algorithm are not available and the resulting final model may by very complex. The AutoGANN system loses some predictive accuracy over that of the Ensemble decision tree and Adaboost methods, but it is a price worth paying for the increased interpretability that is further elaborated on in the next section.

5 DISCUSSION

The automated construction algorithm solves the problem of architecture selection by organising the GANN models into a search tree and performing a greedy best-first search (De Waal and Du Toit, 2007). Out-of-sample performance or an in-sample model selection criterion can be optimized. In addition, two heuristics are applied to speed up the search. First, a stepwise regression identifies significant inputs and their relationships with the target. This information is combined to create a clever starting point (GANN model) from which search can commence. Basically, this heuristic performs an intelligent guess of the best architecture. The better the guess, the less search must be performed to obtain the best model. In the example, this particular GANN model had an accuracy of 91.03%, a result which outperformed the Naive Bayes and neural network classifiers of Table 1. One-half (21) of the 42 inputs selected in the best GANN model, were already identified by the intelligent start. A second heuristic allows multiple changes to successive architectures examined. This rule of thumb enables the algorithm to make systematic leaps in the search tree.

AutoGANN can perform input selection which is especially useful to track the temporal evolution of spam. This phenomenon is a recently observed trend where spammer techniques have evolved in response to the appearance of more and better filters. Kiran and Atmosukarto (n.d.) noticed that most of the evaluations of existing spam detection algorithms found in the literature completely ignored this "evolution-in-time" issue by randomly selecting training and testing sets from an email corpus (a collection of written texts). By learning this temporal evolution, spam filters can adapt to the ever-changing tactics of spammers with greater ease. One way to uncover temporal evolution is to analyse input-level changes in spam over time. Unfortunately, the Spambase data set does not specify any time periods thus making such an analysis infeasible.

Neural networks are usually regarded as black boxes with respect to interpretation. The influence of a particular input on the target can depend in complicated ways on the values of the other inputs. In some applications, such as voice recognition, pure prediction is the goal; understanding how the inputs affect the prediction is not important. In many scientific applications, the opposite is true. To understand is the goal, and predictive power only validates the interpretive power of the model. Some domains such as spam detection often have both goals. Evaluating new cases is the main purpose of predictive modeling. However, some understanding, even informal, of the factors influencing the prediction can be helpful in making progress towards developing better spam filters.

Figures 1, 2, 3 and 4 present partial residual plots for the inputs *edu*, *free*, *hp* and *internet* respectively. These diagrams allow the modeller to gain insight into the constructed model. All four inputs were identified as having non-linear relationships with the target (non-straight lines in the partial residual plots). Figure 1 shows a reverse trend between the frequency of the word *edu* and the probability that the email is spam. On the other hand, Figures 2 and 4 indicate that an increase in the frequencies of the words *free* and *internet* raises the probability of the email being classified as spam. Figure 3 shows a sharp reduction in probability for small frequencies of the word *hp*, followed by a constant probability for increasing frequencies. Although a non-linear relationship between *internet* and the target was identified, Figure 4 emphasises that a linear relationship would also suffice. Such a modification would result in a more parsimonious model.

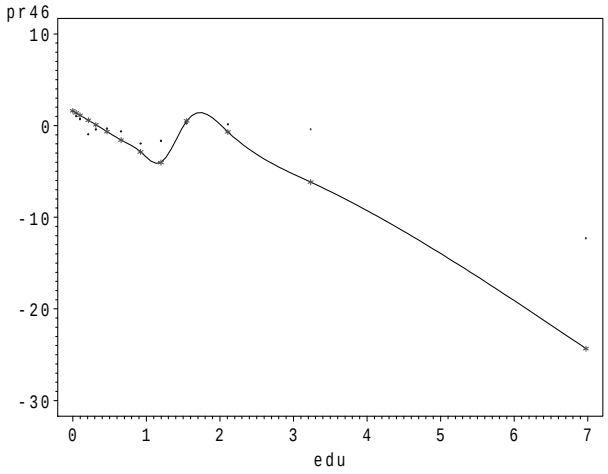

Figure 1: Partial residual plot for *edu*.

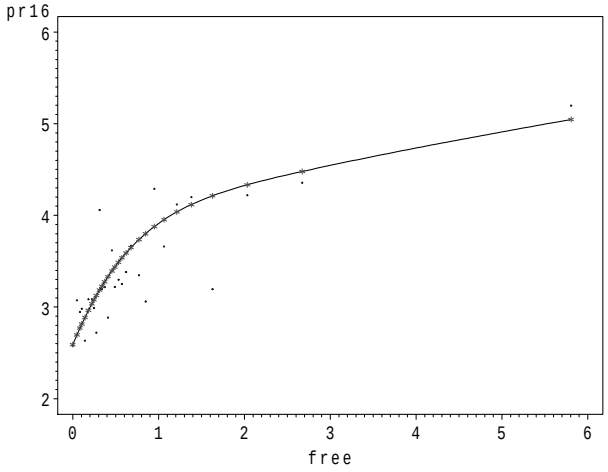

Figure 2: Partial residual plot for *free*.

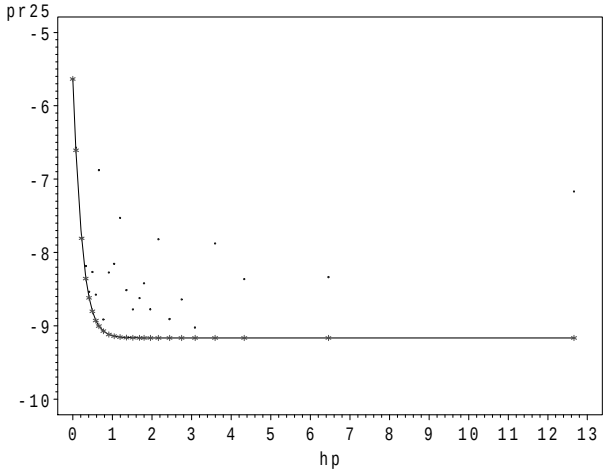

Figure 3: Partial residual plot for *hp*.

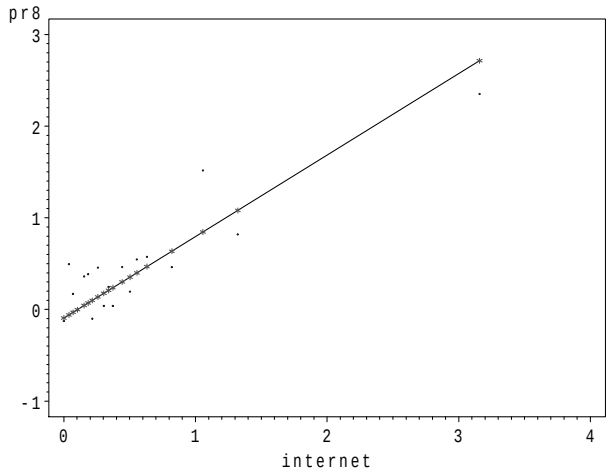

Figure 4: Partial residual plot for *internet*.

In the next section, some final conclusions are made.

6 Conclusions

Scientific evaluation is a crucial component of research as researchers must be able to compare methods using standard data and measurements. This type of evaluation is particularly difficult for spam filtering. Building a standard benchmark for use by researchers is difficult in view of the sensitivity of email. Few organisations and individuals would allow their own messages to be publicly shared and those that would are hardly representative. This predicament is exemplified by the Spambase data set. Although this particular data set serves as an adequate testbed for evaluating a new spam filter, the inputs were selected on the basis of email arriving at one specific individual at one specific corporate organization. As a consequence the attributes are not representative of a general spam sample. Extracting these attributes from other email corpora could result in rather sparse data.

Fortunately, a special spam track within the context of the larger Text REtrieval Conference (TREC), a U.S.-government-supported program that facilitates analysis, evaluates participants' filters on real email streams (Goodman et al., 2007). In addition, standard measures and corpora for tests in the future are defined. The spam track depends on two types of email corpora. The first is synthetic, made up of a rare public corpus of non-spam messages and combined with a carefully modified set of recent spam. Researchers run their filters on it and it may be freely shared. With the second private corpora, researchers submit their code to testers who run it on the corpora and return summary results only, thereby guaranteeing privacy.

Acknowledgments

The authors wish to thank SAS® Institute for providing them with Base SAS® and SAS® Enterprise Miner™ software used in computing all the results presented in this paper. This work forms part of the research done at the North-West University within the TELKOM CoE research program, funded by TELKOM, GRINTEK TELECOM and THRIP.

References

Asuncion, A. and Newman, D. J. (2007), UCI machine learning repository. University of California, Irvine, School of Information and Computer Sciences, http://www.ics.uci.edu/∼mlearn/MLRepository.html, Date of access: 17 February 2010.

Berk, K. N. and Booth, D. E. (1995), 'Seeing a curve in multiple regression', *Technometrics* **37**(4), 385–398.

Bishop, C. M. (1995), *Neural Networks for Pattern Recognition*, Oxford University Press, Oxford.

Cranor, L. F. and LaMacchia, B. A. (1998), 'Spam!', *Communications of the ACM* **41**(8), 74–83.

De Waal, D. A. and Du Toit, J. V. (2007), Generalized additive models from a neural network perspective, *in* 'Workshops Proceedings of the 7th IEEE International Conference on Data Mining (ICDM 2007), October 28-31, 2007, Omaha, Nebraska, USA', IEEE Computer Society.

Du Toit, J. V. (2006), Automated Construction of Generalized Additive Neural Networks for Predictive Data Mining, PhD thesis, School for Computer, Statistical and Mathematical Sciences, North-West University, South Africa.

Ezekiel, M. (1924), 'A method for handling curvilinear correlation for any number of variables', *Journal of the American Statistical Association* **19**(148), 431–453.

Goodman, J., Cormack, G. V. and Heckerman, D. (2007), 'Spam and the ongoing battle for the inbox', *Communications of the ACM* **50**(2), 25–33.

Hastie, T. J. and Tibshirani, R. J. (1986), 'Generalized additive models', *Statistical Science* **1**(3), 297–318.

Hastie, T. J. and Tibshirani, R. J. (1990), *Generalized Additive Models*, Vol. 43 of *Monographs on Statistics and Applied Probability*, Chapman and Hall, London.

Kiran, R. and Atmosukarto, I. (n.d.), Spam nor not spam - that is the question.

Larsen, W. A. and McCleary, S. J. (1972), 'The use of partial residual plots in regression analysis', *Technometrics* **14**(3), 781–790.

McCullagh, P. and Nelder, J. A. (1989), *Generalized Linear Models*, Vol. 37 of *Monographs on Statistics and Applied Probability*, 2nd edn, Chapman and Hall, London.

Potts, W. J. E. (1999), Generalized additive neural networks, *in* 'KDD '99: Proceedings of the fifth ACM SIGKDD international conference on Knowledge discovery and data mining', ACM, New York, NY, USA, pp. 194–200.

Ripley, B. D. (1996), *Pattern Recoqnition and Neural Networks*, Cambridge University Press, Cambridge, United Kingdom.

Wood, S. N. (2006), *Generalized Additive Models: An introduction with R*, Texts in Statistical Science, Chapman & Hall/CRC, London.

Zhang, G., Patuwo, B. E. and Hu, M. Y. (1998), 'Forecasting with artificial neural networks: The state of the art', *International Journal of Forecasting* **14**, 35–62.

BIOGRAPHY

Dr. Tiny du Toit obtained his Ph.D. in Computer Science at the North-West University in 2006. Currently, he is a senior lecturer in Computer Science and part of the Telkom CoE program at the university's Potchefstroom campus where he lectures and performs research in the field of Artificial Intelligence.