Neural Network Credit Scoring Models

Article *in* Computers & Operations Research · September 2000

DOI: 10.1016/S0305-0548(99)00149-5 · Source: DBLP

David West
East Carolina University
32 PUBLICATIONS **2,042** CITATIONS

PERGAMON

Computers & Operations Research 27 (2000) 1131–1152

computers & operations research

www.elsevier.com/locate/dsw

Neural network credit scoring models

David West*

Department of Decision Sciences, College of Business Administration, East Carolina University, Greenville, NC 27836, USA

Abstract

This paper investigates the credit scoring accuracy of five neural network models: multilayer perceptron, mixture-of-experts, radial basis function, learning vector quantization, and fuzzy adaptive resonance. The neural network credit scoring models are tested using 10-fold crossvalidation with two real world data sets. Results are benchmarked against more traditional methods under consideration for commercial applications including linear discriminant analysis, logistic regression, *k* nearest neighbor, kernel density estimation, and decision trees. Results demonstrate that the multilayer perceptron may not be the most accurate neural network model, and that both the mixture-of-experts and radial basis function neural network models should be considered for credit scoring applications. Logistic regression is found to be the most accurate of the traditional methods.

Scope and purpose

In the last few decades quantitative methods known as credit scoring models have been developed for the credit granting decision. The objective of quantitative credit scoring models is to assign credit applicants to one of two groups: a "good credit" group that is likely to repay the financial obligation, or a "bad credit" group that should be denied credit because of a high likelihood of defaulting on the financial obligation. The first model employed for credit scoring, and a commonly used method today, is linear discriminant analysis, a simple parametric statistical method. With the growth of the credit industry and the large loan portfolios under management today, the industry is actively developing more accurate credit scoring models. Even a fraction of a percent increase in credit scoring accuracy is a significant accomplishment. This effort is leading to the investigation of nonparametric statistical methods, classification trees, and neural network technology for credit scoring applications. The purpose of this research is to investigate the accuracy of five neural network architectures for the credit scoring applications and to benchmark their performance against the models currently under investigation today. © 2000 Elsevier Science Ltd. All rights reserved.

Keywords: Credit scoring; Neural networks; Multilayer perceptron; Radial basis function; Mixture-of-experts

* Corresponding author. Tel.: + 1-252-328-6370; fax: + 1-252-328-4092.
E-mail address: westd@mail.ecu.edu (D. West)

1. Introduction

The credit industry has experienced two decades of rapid growth with significant increases in installment credit, single-family mortgages, auto-financing, and credit card debt. Credit scoring models have been widely used by the financial industry during this time to improve cash flow and credit collections. The advantages of credit scoring include reducing the cost of credit analysis, enabling faster credit decisions, closer monitoring of existing accounts, and prioritizing collections [1]. Today credit scoring is used by 97% of banks that approve credit card applications and by 82% of banks that determine whom to solicit for credit cards. Both the Federal Home Loan Mortgage Corporation and Federal National Mortgage Corporation are actively encouraging the use of credit scoring for mortgage origination, and GE Capital Mortgage uses credit scoring for mortgage insurance applications [2]. With the growth in financial services there have been mounting losses from delinquent loans. For example, in 1991 $1 billion of Chemical Bank's $6.7 billion in real estate loans were delinquent and the bank held $544 million in foreclosed property [3]. Manufacturers Hanover's $3.5 billion commercial property portfolio was burdened with $385 million in non-performing loans [3]. In response, many organizations in the credit industry are developing new models to support the credit decision. The objective of these new credit scoring models is increased accuracy, which means more creditworthy applicants are granted credit thereby increasing profits; non-creditworthy applicants are denied credit thus decreasing losses.

Linear discriminant analysis (LDA), a simple parametric statistical model, was one of the first credit scoring models. The appropriateness of LDA for credit scoring has been questioned because of the categorical nature of the credit data and the fact that the covariance matrices of the good and bad credit classes are not likely to be equal. The credit data is usually not normally distributed, although Reichert reports this may not be a critical limitation [4]. More sophisticated models are being investigated today to overcome some of the deficiencies of the LDA model. Henley [5] explores a logistic regression model for credit scoring applications. Other researchers are currently investigating nonparametric statistical models like k nearest neighbor [6,7] classification trees [7–10] and neural network models for credit scoring applications [1,7,8,11–17].

A central concern of these applications is the need to increase the scoring accuracy of the credit decision. An improvement in accuracy of even a fraction of a percent translates into significant future savings. To pursue even small improvements in credit scoring accuracy, the practitioner must explore other neural network architectures beyond the conventional MLP, as well as nonparametric statistical models and classification trees. The purpose of this research is to investigate the suitability of five neural network architectures for credit scoring and to benchmark their performance against the following models: two parametric models (linear discriminant analysis and logistic regression), two nonparametric methods (k nearest neighbor and kernel density), and classification trees.

Section 2 of this paper discusses recent research in the use of neural network models to predict financial distress. This is followed by a discussion of the conceptual differences between the five neural network models investigated, the experimental design used to estimate credit scoring accuracy for each of the quantitative models, the results, and conclusions to guide the practitioner developing quantitative credit scoring models.

2. Predicting financial distress: neural network models

American Express is using a neural network-based system to detect incidences of credit card fraud, and Lloyds Bowmaker Motor Finance has deployed a neural network credit scoring system for automobile financing decisions. Lloyds claims that the neural-based credit scoring system is 10% more accurate than the system it replaced [18]. Security Pacific Bank (SPB) is also using a neural network intelligent system for credit scoring of small business loans [18]. The specific neural network credit scoring model developed by SPB is a multi-layer perceptron (MLP) trained by the back-propagation learning algorithm. SPB believes the advantage of the neural network scoring system is the improved function-fitting capability due to the intrinsic non-linear pattern recognition capability of the neural network. They state that even a small improvement in predictive accuracy of the neural network credit scoring model is critical; a 1% improvement in accuracy will reduce losses in a large loan portfolio and save millions of dollars.

Due to the proprietary nature of credit scoring, there is a paucity of research reporting the performance of commercial credit scoring applications. The research that exists today focuses on two areas, the prediction of firm insolvency and the prediction of individual credit risk. Research at the firm level is reported first, followed by research at the level of the individual consumer. Altman et al. employs both linear discriminant analysis and a multilayer perceptron neural network to diagnose corporate financial distress for 1000 Italian firms [8]. The authors conclude that neural networks are not a clearly dominant mathematical technique compared to traditional statistical techniques such as discriminant analysis, and that linear discriminant analysis compares rather well to the neural network model in decision accuracy [8]. Coats and Fant report a different experience, contrasting a multilayer perceptron neural network with linear discriminant analysis for a set of firms labeled viable or distressed obtained from COMPUSTAT for the period 1970–1989 [11]. They find the neural network to be more accurate than linear discriminant analysis, particularly for predicting firms in financial distress. Lacher et al. utilizes the Coats and Fant data to predict financial distress with Altman's Z score [15]. They state that the neural network developed by cascade correlation more accurately predicts the financial health of a firm [15]. Salchenberger et al. investigate the use of a multilayer perceptron neural network to predict the financial health of savings and loans [17]. The authors compare a multilayer perceptron neural network with a logistic regression model for a data set of 3429 S&L's from January 1986 to December 1987. They find that the neural network model performs as well as or better than the logistic regression model for each data set examined. Tam and Kiang studied the application of a multilayer perceptron neural network model to Texas bank failure prediction for the period 1985–1987 [7]. The neural network prediction accuracy is compared to linear discriminant analysis, logistic regression, k nearest neighbor, and a decision tree model. Their results suggest that the multilayer perceptron is most accurate, followed by linear discriminant analysis, logistic regression, decision trees, and k nearest neighbor.

Desai et al. [13] investigate a multilayer perceptron neural network, a mixture of experts neural network, linear discriminant analysis, and logistic regression for scoring credit applicants in the credit union industry. Their methodology consists of two-fold cross validation of field data obtained from three credit unions and the assumption of equal costs for good and bad credit risks. They conclude that the neural network models outperform linear discriminant analysis, but are only marginally better than logistic regression models. The authors also report that the neural network models classify bad loans more accurately than either linear discriminant analysis or logistic regression [13]. Piramuthu [16] compares a multilayer perceptron neural network and a neural-fuzzy model in three credit scoring applications: Quinlan's credit card data [19], a small loan default data set [20], and the Texas bank failure data [7]. The neural network model achieved an overall accuracy of 83.56% on Quinlan's credit card data based on 10 repetitions and a single data partition. The neural fuzzy model was almost 6% less accurate than the multilayer perceptron across all three data sets [16]. The use of decision trees and a multilayer perceptron neural network for credit card application scoring are studied by Davis et al. [9]. Their results are based on a single data partition and a single neural network trial. The authors conclude that the multilayer perceptron neural network and the decision tree model both have a comparable level of decision accuracy [9]. Jensen [14] develops a multilayer perceptron neural network for credit scoring with three outcomes: obligation charged off (11.2%), obligation delinquent (9.6%), and obligation paid-off. Jensen reports a correct classification result of 76–80% with a false positive rate (bad credit risk classified as good credit) of 16% and a false negative rate (good credit risk classified as bad credit) of 4%. Jensen concludes that the neural network has potential for credit scoring applications based on results from a single data partition tested on only 50 examples [14].

The research available on predicting financial distress, whether conducted at the firm or individual level suggests that neural network models show potential yet lack an overwhelming advantage over classical statistical techniques. In the quest for small fractional improvement in predictive accuracy, it is necessary to investigate several neural network architectures and to use a rigorous experimental methodology to establish performance differences between models. Many of the previous studies are exploratory in nature, using only a single data partition to establish training and test samples. This may lead to bias in the estimation of classification accuracy on holdout samples. In many cases, only a single trial of the neural network model is employed. The stochastic nature of the neural network training process requires a number of repetitions to estimate an expected accuracy level. The most common test used to establish statistically significant differences between credit scoring models is a difference of two proportions or a paired difference t test. Dietterich has shown these tests should never be used for comparing supervised learning models because of the high probability of Type 1 error associated with these tests [21].

This research investigates the potential for small improvements in credit scoring accuracy by exploring five neural network architectures, with two real world data sets partitioned into training and test sets with 10-fold cross validation. Ten repetitions of each neural network trial are used and then the models are tested for significant differences with McNemar's Chi Square test which has been shown to be the most powerful test of model differences for supervised learning algorithms [21].

3. Neural network credit scoring models

While MLP is the most frequently used neural network architecture in commercial applications including credit scoring, several other neural network architectures can be considered. Five neural network architectures are investigated in this research: the traditional MLP network, mixture of experts (MOE), radial basis function (RBF), learning vector quantization (LVQ), and fuzzy adaptive resonance (FAR). It is natural to expect different levels of credit scoring accuracy from the different neural network architectures. They use different algorithms to estimate the unknown credit scoring function and employ different training methods to acquire information from the credit scoring examples available. Depending on the neural network model selected, the estimation of the credit scoring function may result from a global response, a local response, a partitioning of the problem domain, a nearest-neighbor prototype, or a dynamic resonance of patterns and prototypes. Credit scoring accuracy, therefore, is expected to vary with the neural network model choice. The conceptual differences between these five neural network models are highlighted next. References are provided for the reader interested in more details concerning specific network architecture.

The MLP architecture for a single hidden layer, two-input two-output network is shown in Fig. 1 [22, 23]. Rectangles represent the neurons; arrows represent the weight values of the feed-forward connections between neurons. The four connections that have no source are all from a bias unit that has not been shown for simplicity. When a data value is applied to the input neurons, calculated values at each neuron are propagated layer by layer through the network, resulting in the activation of the output neurons. In each layer, the signal propagation is accomplished as follows. First, a weighted sum is calculated at each neuron, that is the output value of each neuron in the proceeding network layer times the respective weight of the connection with that neuron. A transfer function $g(x)$ is then applied to this weighted sum to determine the neurons output value. The transfer function used in this research is the hyperbolic tangent. The output value, Y, for output neuron, k, can be expressed as a function of the input values and network weights, w, as follows:

$$Y_k = \sum_{h=1}^{2} w_{kj}\left(g\left(\sum_{i=1}^{2} w_{ji} X_i\right) + w_{jb}\right) + w_{ib}, \quad k = 1, 2, \tag{1}$$

where i indexes the input neurons, j indexes the hidden layer neurons, and b indexes the respective bias values.

During the backpropagation training process the MLP weights are initialized to small random variables. Weight updates are accomplished by calculating derivatives of the network MSE error with respect to the network weights. Using the chain rule, gradients can be estimated at each network layer beginning at the output layer. Propagating error components backward through the network and using the gradient information to adjust the weights reduces the network error function. A characteristic of the MLP architecture is that the weight matrix is a global function, that is an error adjustment results in updates to all network weights.

The MOE [24] differs from the MLP neural network in that it decomposes the credit scoring task and uses local experts to learn specific parts of the problem. In this research, a local expert is dedicated to each of the credit scoring outcomes (grant credit or deny credit). The MOE gating network determines which of the two local networks should represent any given input. An advantage of the MOE architecture is that during the learning process weight changes are localized to the respective expert network, minimizing the possibility of learning interference between the experts. A second advantage is that the local expert must learn a smaller local region of input space. By contrast, when a single multi-layer network is trained with back-propagation, there can be strong interference effects leading to slow learning and poor generalization. A typical architecture for the MOE classification network is shown in Fig. 2. The output of the MOE network (y_1, y_2) is

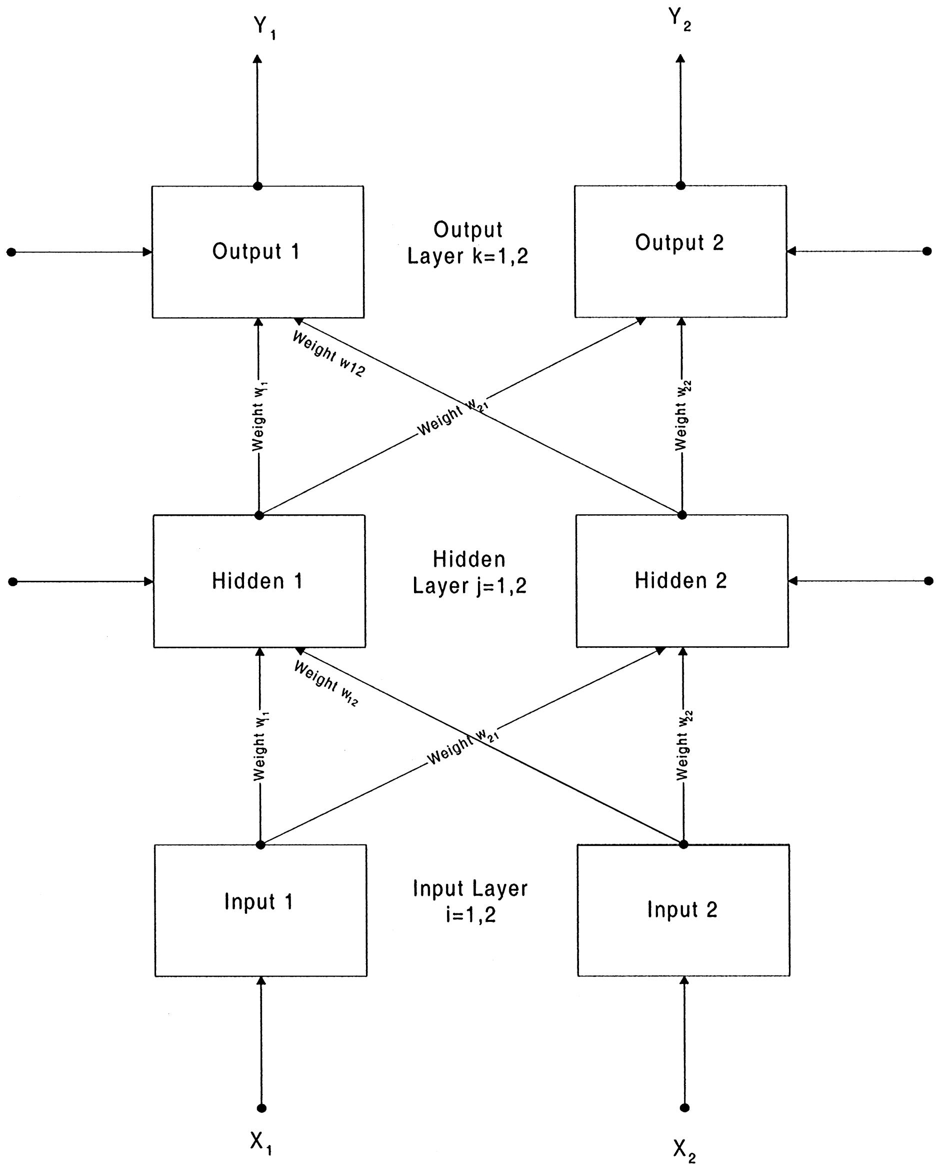

Fig. 1. Multilayer perceptron network (MLP).

a weighted average of the respective local expert outputs defined as

$$y_1 = \sum_{o=1}^{2} g_o y_{o1} \quad \text{and} \quad y_2 = \sum_{o=1}^{2} g_o y_{o2}, \tag{2}$$

where

$$g_1 = \frac{e^{s_1}}{\sum_{l=1}^{2} e^{s_l}} \quad \text{and} \quad g_2 = \frac{e^{s_2}}{\sum_{l=1}^{2} e^{s_l}}. \tag{3}$$

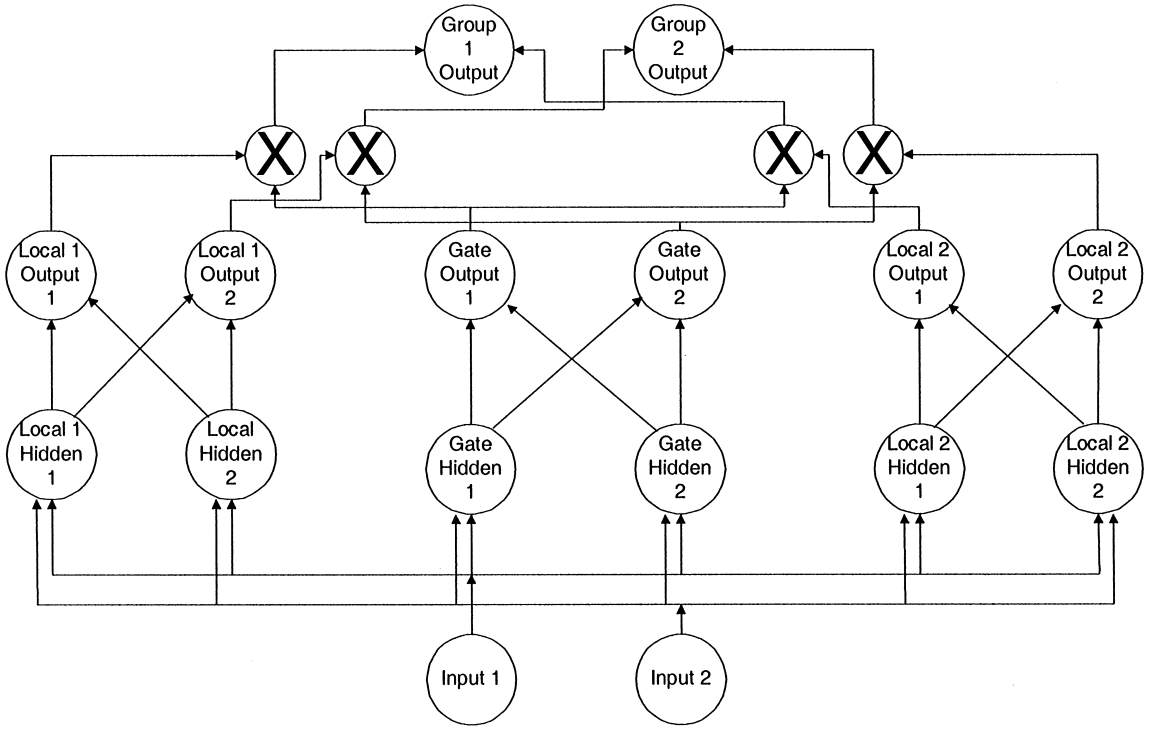

Fig. 2. Mixture-of-experts network (MOE).

In Eq. (3), s_1 and s_2 represent the activation level of the respective gating network output node.

It is important for the MOE network error function to promote localization of weights during the learning process. Jacobs et al. [24] found that the objective function, J, defined in Eq. (4), gives the best performance for the MOE architecture.

$$J = -\sum_{o=1}^{2} g_o e^{-0.5(\mathbf{d}-\mathbf{y}_o)^{\mathrm{T}}(\mathbf{d}-\mathbf{y}_o)}. \tag{4}$$

The vector **d** is the desired network output vector (0, 1 or 1, 0), and $\mathbf{y}_o$ $(o = 1, 2)$ is the activation vector for the respective local expert network. The weights of all local expert networks are modified simultaneously using the backpropagation algorithm. At each training epoch the weights of the expert networks are modified to reduce the sum of squared error difference between the system output and the desired output or target. The weights of the gating network are modified to reduce a more complicated error function. For each training pattern, the expert network that comes closest to producing the desired output is called the winner. For a given training pattern, if the MOE's performance is significantly improved, then the weights of the gating network are adjusted to make the output corresponding to the winning expert increase towards one, and the output corresponding to the losing experts decrease towards zero. Conversely, if the systems performance has not improved, the gating network's output are adjusted to move all its outputs toward some neutral value.

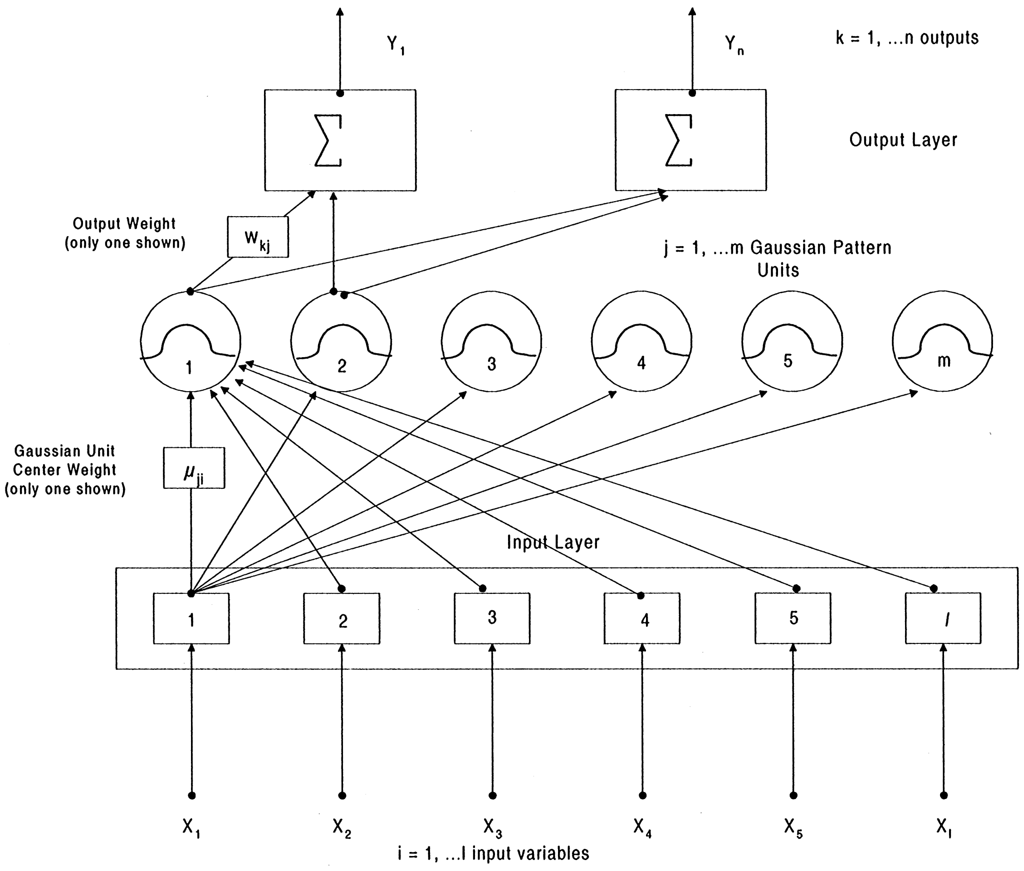
Y1
Yn
k = 1, ...n outputs
Output Layer
Output Weight (only one shown)
Wkj
j = 1, ...m Gaussian Pattern Units
1
2
3
4
5
m
Gaussian Unit Center Weight (only one shown)
μji
Input Layer
1
2
3
4
5
l
X1
X2
X3
X4
X5
Xl
i = 1, ...l input variables

Fig. 3. Radial basis function network (RBF).

The RBF network [25] is also quite different in that the hidden layer is restricted to radially symmetric functions, which produce a local response to the input data. The architecture for a radial basis function network is shown in Fig. 3 with m Gaussian basis functions in the pattern layer (note that not all weights are shown in Fig. 3). The output of the Gaussian unit j is determined as

$$G(\mathbf{X}) = \exp\left(-\frac{\|\mathbf{X} - \mu_j\|^2}{2\sigma_j^2} \right) \quad \forall j = 1, 2, \dots, m, \tag{5}$$

where $\mathbf{X}$ is the l-dimensioned input vector (l represents the number of input process variables) and μ_j is the vector weights from the l-inputs to Gaussian unit j. The parameter σ_j effects the distance over which the Gaussian function will respond and thereby influences the smoothness of the approximating function. The output of the RBFN, Y_k, is a weighted linear combination of the m Gaussian basis functions and is expressed as

$$Y_k(X) = \sum_{j=1}^{m} w_{kj} \exp\left(-\frac{\|X - \mu_j\|^2}{2\sigma_j^2} \right) + w_{k0} \quad \forall k = 1, 2, \dots, n, \tag{6}$$

where Y_k is the calculated value for each of the n output neurons, w_{k0} is the bias term associated with Y_k, and w_{kj} and μ_{ji} are elements of the weight vector $\mathbf{W}$. The RBF network is frequently referred to as a local network because the output, Y_k, is calculated from only those basis functions whose centers, μ_j, are close to the input vector $\mathbf{X}$.

Training of the RBF network involves two critical processes. First, the centers of each of the m Gaussian basis functions is fixed to represent the density function of the input space using a dynamic k means clustering algorithm. This phase of RBF training places the weights of the radial basis function units in only those regions of the input space where significant data are present. The parameter σ_j is set for each Gaussian unit to equal the average distance to the two closest neighboring Gaussian basis units. The intention is to size this parameter so there are no gaps between basis functions and allow for only minimal overlap between adjacent basis functions. After the Gaussian basis centers are fixed, the second step of the RBFN training process is to determine the weight vector $\mathbf{W}$ which best approximates the limited sample data $\mathbf{X}$. Interestingly, this is a linear optimization problem that can be solved by ordinary least squares. This avoids the problems of gradient descent methods and local minima characteristics of MLP, and uses a direct optimization procedure.

The LVQ network [26] is a simple three-layer network that produces a credit scoring decision by using the hidden layer neurons as a set of codebook vectors; a subset of these codebook vectors is assigned to each credit group. A new credit applicant is classified by identifying the group, c, of the nearest codebook vector, W_i, that is a minimum Euclidean distance from the input vector, $\mathbf{X}$.

$$c = \underset{i}{\operatorname{argmin}} \{\|\mathrm{X} - W_i\|\}. \tag{7}$$

Training of the LVQ network proceeds by presenting training examples to the network and determining the closest codebook vector. If the input vector X, and winning codebook vector, W_c, belong to the same credit group, then the weights of the codebook vector are adjusted to move it closer to the input vector as follows:

$$W_c(t+1) = W_c(t) + \alpha(t)[\mathbf{X} - W_c(t)], \tag{8}$$

where α is a learning rate that decreases over time. In similar fashion, if the pair of vectors belong to different credit groups, the codebook vector is moved away from the input vector. During this process, all codebook vectors other than the winner, c, remain stationary.

The FAR [27] is a dynamic network where the input patterns resonate in a two-layer attentional system. A credit decision will be determined based on the strength of the feedback resonance with the two output profiles, grant credit or deny credit. A gain subsystem and an orienting subsystem control the dynamics of the network and determine the granularity of the credit decision. The FAR network, depicted in Fig. 4, consists of two totally interconnected layers of neurons, identified as the complement layer and the category layer, in addition to the input and output layers. When an input vector is applied to the network, it creates a short-term activation of the neurons in the complement layer. This activity is transmitted through the weight vector μ to neurons in the category layer. Each neuron in the category layer then calculates the inner product of the respective weights and input values. These calculated values are then resonated back to the complement layer. Resonance of signals between these two layers continues to reinforce the signal until there is a match with one of the output weight vectors $\mathbf{W}_j$. The winning category neuron is determined by

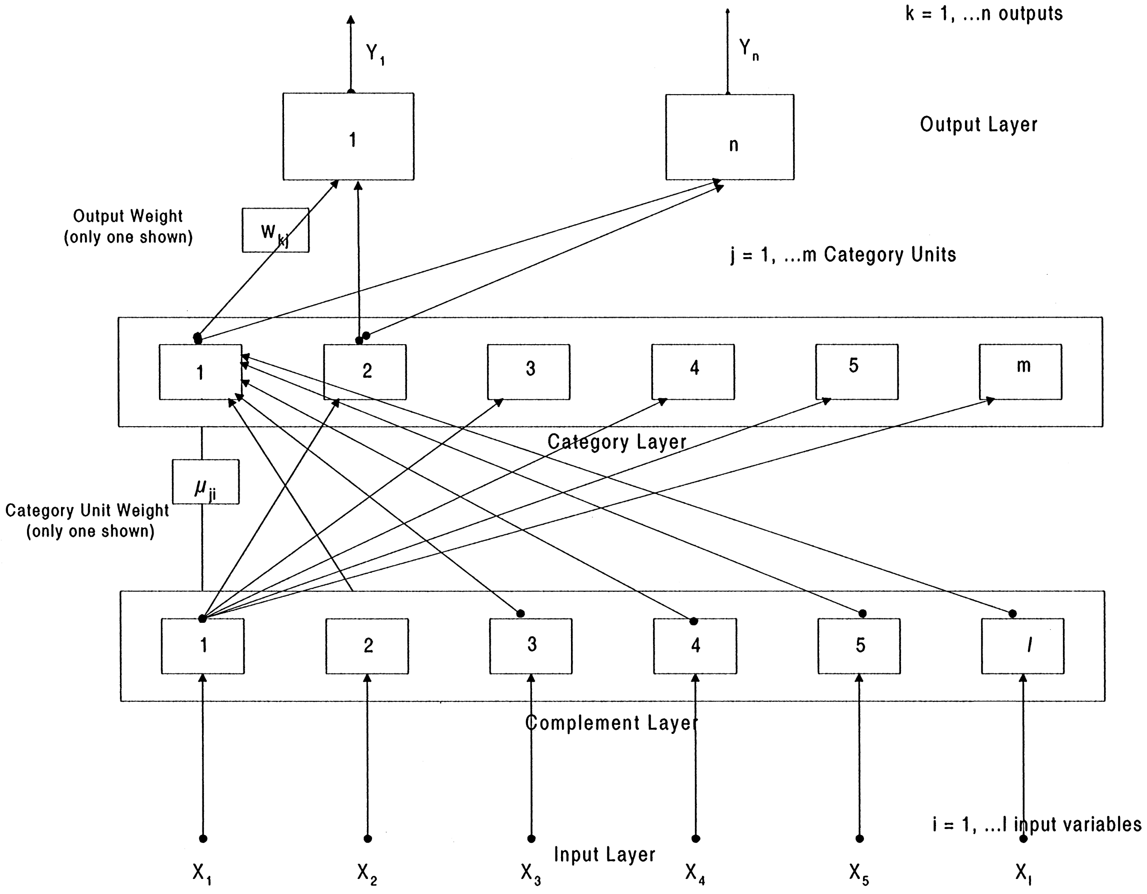
k = 1, ...n outputs
Y1
Yn
1
n
Output Layer
Output Weight (only one shown)
Wkj
j = 1, ...m Category Units
1
2
3
4
5
m
Category Layer
μji
Category Unit Weight (only one shown)
1
2
3
4
5
I
Complement Layer
i = 1, ...I input variables
Input Layer
X1
X2
X3
X4
X5
XI

Fig. 4. Fuzzy art network (FAR).

the choice function below.

$$\mathbf{T}_j(\mathbf{I}) = \frac{|\mathbf{I} \wedge \mathbf{W}_j|}{\alpha + |\mathbf{W}_j|}. \tag{9}$$

In Eq. (9), α is a constant determined by the number of neurons activated, $\wedge$ is the fuzzy set AND function, and $|\ |$ represents fuzzy set cardinality. Under some conditions, if the resonated pattern cannot closely match the weight vector of an existing output neuron, the winning category is committed to a new output neuron. This condition is determined by the following threshold test:

$$\frac{|\mathbf{I} \wedge \mathbf{W}_j|}{|\mathbf{I}|} \leqslant \rho. \tag{10}$$

When a new output node is first committed to represent a data pattern, the fast commit learning process occurs.

$$\mathbf{W}_j^{(\mathrm{new})} = \mathbf{I}. \tag{11}$$

This simply assigns the weight vector of the newly committed output node to the value of the input feature vector. After an output node has been committed, further learning is accomplished using the following relationship:

$$\mathbf{W}_j^{(\text{new})} = \beta(\mathbf{I} \wedge \mathbf{W}_j^{(\text{old})}) + (1-\beta)\mathbf{W}_j^{(\text{old})}, \quad 0 < \beta < 1. \tag{12}$$

The new weight vector is an exponentially smoothed average of the old weight vector and the result of the fuzzy AND operator applied to the input vector and the old weight vector.

4. Experimental design

This research investigates the potential for small improvements in credit scoring accuracy by testing five neural network architectures, with two real world data sets partitioned into training and independent test sets using 10-fold cross validation. Ten repetitions are used for each neural network trial and the results from the independent holdout samples are tested for significant differences between models with McNemar's chi-square test [21]. In addition to the neural network models, several quantitative models currently under consideration for credit scoring applications also are included: two parametric methods (linear discriminant analysis and logistic regression), two nonparametric methods (k nearest neighbor and kernel density), and decision trees (CART).

Two real world data sets are used to test the predictive accuracy of the credit scoring models investigated. Dr. Hans Hofmann of the University of Hamburg contributed the German credit scoring data. It consists of 700 examples of creditworthy applicants and 300 examples where credit should not be extended. For each applicant, 24 variables describe credit history, account balances, loan purpose, loan amount, employment status, personal information, age, housing, and job. The Australian credit scoring data [19] is similar but more balanced with 307 and 383 examples of each outcome. To protect the confidentiality of this data, attribute names and values have been changed to symbolic data. The data set contains a mixture of six continuous and eight categorical variables. The reader is cautioned that neither data set contains credit bureau information, which is usually used to determine credit decisions. The German credit data also contains some information like gender, marital status, and nationality that cannot legally be used in the US. This research employs a data mining strategy, using all the variables as raw data. Some of the previous research in financial distress use summary variables like the financial ratios identified by Altman [8].

To minimize the impact of data dependency and improve the reliability of the resultant estimates, 10-fold cross validation is used to create random partitions of the data sets. Each of the 10 random partitions serves as an independent holdout test set for the credit scoring model trained with the remaining nine partitions. The training set is used to establish the credit scoring model's parameters, while the independent holdout sample is used to test the generalization capability of the model. The overall scoring accuracy reported is an average across all ten test set partitions. An advantage of cross validation is that the credit scoring model is developed with a large proportion of the available data (90% in this case) and that all of the data is used to test the resulting models.

Several key design decisions involving the topology and the learning process are required to define the neural network models. Topology decisions establish the network architecture and include the number of hidden layers and number of neurons in each layer. The number of neurons in the input layer of the neural models is simply the number of variables in the data set. For the neural output layer, 1 of 2 coding is used with an output neuron dedicated to each of the credit decision outcomes. The hidden layer is more difficult to define. A relatively large hidden layer creates a more flexible credit scoring model. The scoring error for such a model will tend to have a low bias component with a large variance caused by the tendency for the model to over-fit the training data. A relatively small hidden layer results in a model with a higher error bias and a lower variance. The design of the hidden layer, therefore, involves a tradeoff between error components. For each of the two data sets, the number of neurons in the MLP hidden layer is determined using a cascade learning process. The cascade learning process is constructive, starting with an empty hidden layer and adding neurons to this layer one at a time. The addition of hidden neurons continues until there is no further improvement in network performance. The MOE is configured with two local experts (one per credit decision outcome) and a simple gating network. Each local expert is comparable to the MLP topology determined by cascade learning. The hidden layer of the LVQ model follows the heuristic to size the hidden layer with the number of neurons equal to 10% of the number of training cases. The hidden layers for the RBF and FAR models are established experimentally and vary by data set from 20 to 60 neurons. Scoring results are also dependent on the dynamics of the network learning process. The most accurate scoring results typically do not coincide with network error convergence for the training data. Initial experiments were conducted with training lengths varying from 30 000 to 300 000 iterations. Based on these experiments, the following training guidelines are used in this research. Network weights are updated after each learning epoch, defined as the size of the training data set. The network learning parameters such as learning rates, momentum, etc. are decreased every 10 learning epochs, and training is terminated after 50 learning epochs. Ten repetitions are conducted for each of the 10 partitions to reduce the stochastic variability of the network training process.

The choice of model parameters is also an important determinant of the generalization ability for the nonparametric methods. These were established by trial and error with the following results. The k nearest-neighbor model used the three nearest neighbors for the German credit and seven neighbors for Australian credit. The width of the density kernel is 1.0 for both data sets while the size of the classification tree is limited by a chi-square stopping rule with $p = 0.05$. A threshold value of 0.5 is used to distinguish between credit groups for all quantitative models tested.

5. Results and discussion

The results for each credit-scoring model are reported in Table 1 for both the German and Australian credit data. These results are averages of errors determined for each of the 10 independent holdout data set partitions used in the cross validation methodology. Since the training of any neural network model is a stochastic process, the neural network error determined for each data set partition is itself an average of 10 repetitions. The error for each neural network credit scoring model is therefore an average of 100 partition-repetition trials. In the compilation of results in Table 1, there was no attempt to edit the neural network repetitions to eliminate instances where high credit scoring errors were caused by convergence problems during training. The results of Table 1 are therefore labeled "Average Case Neural Network Models" to denote that the results may include some instances of poorly trained networks. In Table 2, the results include only the

Table 1
Credit scoring error, average case neural network models

	German credit data[b]			Australian credit data[b]		
	Good credit	Bad credit	Overall	Good credit	Bad credit	Overall
Neural models[a]						
MOE	0.1428	0.4775	0.2434	0.1457	0.1246	0.1332
RBF	0.1347	0.5299	0.2540	0.1315	0.1274	0.1286
MLP	0.1352	0.5753	0.2672	0.1540	0.1326	0.1416
LVQ	0.2493	0.4814	0.3163	0.1710	0.1713	0.1703
FAR	0.4039	0.4883	0.4277	0.2566	0.2388	0.2461
Parametric models						
Linear discriminant	0.2771	0.2667	0.2740	0.0782	0.1906	0.1404
Logistic regression	0.1186	0.5133	0.2370	0.1107	0.1409	0.1275
Non-parametric models						
K nearest neighbor	0.2257	0.5533	0.3240	0.1531	0.1332	0.1420
Kernel density	0.1557	0.6300	0.3080	0.1857	0.1514	0.1666
CART	0.2063	0.5457	0.3044	0.1922	0.1201	0.1562

[a]Neural network results are averages of 10 repetitions.
[b]Reported results are group error rates averaged across 10 independent holdout samples.

Table 2
Credit scoring error, best case neural network models

	German credit data[b]			Australian credit data[b]		
	Good credit	Bad credit	Overall	Good credit	Bad credit	Overall
Neural models[a]						
MOE	0.1301	0.4457	0.2243	0.1330	0.1146	0.1239
RBF	0.1424	0.4821	0.2437	0.1238	0.1219	0.1222
MLP	0.1291	0.5308	0.2496	0.1362	0.1127	0.1232
LVQ	0.2085	0.4480	0.2780	0.1499	0.1508	0.1500
FAR	0.2908	0.5814	0.3771	0.2251	0.1761	0.1971
Parametric models						
Linear discriminant	0.2771	0.2667	0.2740	0.0782	0.1906	0.1404
Logistic regression	0.1186	0.5133	0.2370	0.1107	0.1409	0.1275
Non-parametric models						
K nearest neighbor	0.2257	0.5533	0.3240	0.1531	0.1332	0.1420
Kernel density	0.1557	0.6300	0.3080	0.1857	0.1514	0.1666
CART	0.2063	0.5457	0.3044	0.1922	0.1201	0.1562

[a]Neural network results are an average of three best from 10 repetitions.
[b]Reported results are group error rates averaged across 10 independent holdout samples.

three best results from the 10 neural network repetitions for each data partition. This is referred to as the "Best Case Neural Network Models". Both tables report the overall error and two group errors labeled "good credit" and "bad credit". The "good credit" error is the proportion of applicants that are creditworthy but are classified as a bad credit risk. Conversely, the "bad credit" error is the proportion of applicants that are not creditworthy but are incorrectly identified as creditworthy. The overall error is simply the proportion of all applicants that are incorrectly classified.

It is evident from Table 1 that logistic regression has the lowest credit scoring error of 0.2370 for the German credit data. Closely following logistic regression is MOE with an average error of 0.2434, and RBF with 0.2540. Recall that these neural network results are averages of 10 repetitions that may include a few instances of networks that did not converge properly. Linear discriminant analysis has an error of 0.2740, which is 0.037 less accurate than logistic regression, and 0.0306 less accurate than MOE. A strength of the linear discriminant model for this data, however, is a significantly lower error rate than any other model identifying bad credit risks. This is likely due to the assumption of equal prior probabilities used to develop the linear discriminant model. The implications of this disparity for group accuracy are discussed in more detail later. It is also interesting to note that the most commonly used neural network architecture, MLP, is comparable to linear discriminant analysis with an error of 0.2672. The MLP neural model is 0.0238 less accurate than the MOE model. The LVQ neural model, the nonparametric methods, and CART are all grouped at error levels from 0.3080 to 0.3240. The least accurate method for the German credit scoring data is the FAR neural model at 0.4277.

The results for the Australian credit data exhibit some of the same patterns previously discussed for the German credit data. For instance, logistic regression has the lowest overall credit scoring error at 0.1275, followed closely by the MOE (0.1332) and RBF (0.1286) neural models. The MLP neural model (0.1416) and linear discriminant analysis (0.1404) are again comparable from an overall error consideration. The MLP and linear discriminant model have credit scoring errors that are more than 0.01 greater than logistic regression and the MOE and RBF neural models. For this data set, the k nearest-neighbor method, with an error of 0.1420, can also be grouped with MLP and linear discriminant analysis. The LVQ, CART, and kernel density models all have errors ranging from 0.1562 to 0.1703. The FAR neural model finishes a distant last with an error of 0.2461.

The neural network credit scoring errors previously reported in Table 1 are averages of 10 repetitions for each independent holdout data partition. Inevitably, some of these repetitions include instances where the network converges to locations in the error surface that do not give representative results. The practitioner developing neural models for credit scoring applications would undoubtedly ignore these inferior instances and retrain the network. For this reason, Table 2 identifies the potential errors for the three best neural network models. The results of Table 2 are then an average of the three most accurate of the 10 repetitions calculated for each data set partition. The results from Table 2 suggest that not only is MOE the most accurate neural network model, but also the most accurate of all the quantitative models investigated for these two credit scoring applications. For the German credit data, the MOE error of 0.2243 is significantly lower than RBF at 0.2437, MLP at 0.2496, and logistic regression at 0.2370. The advantage in overall error of the MOE neural model for this data set is an accuracy improvement of approximately 0.05 versus linear discriminant analysis, and 0.013 versus logistic regression. For the Australian credit data, three neural network models achieve very comparable levels of accuracy that are approximately 0.005 more accurate than the logistic regression model. The most accurate neural network models for this data set are RBF with an error of 0.1222, MLP at 0.1232, and MOE at 0.1239.

The credit scoring results measured in this research support the hypothesis that neural network models can be used in credit scoring applications to improve the overall accuracy from a fraction of

Table 3
Statistically significant differences,[a] credit model errors

	German credit	Australian credit
Superior models	MOE	MOE
	RBF	RBF
	MLP	MLP
	Logistic reg.	Logistic reg.
		LDA
		K nearest neighbor
Inferior models	LVQ	LVQ
	FAR	FAR
	LDA	Kernel density
	K nearest neighbor	CART
	Kernel density	
	CART	

[a]Statistical significance established with McNemars' test, $p = 0.05$.

a percent to several percent. It also demonstrates that the most popular neural network architecture, MLP, may not be most accurate for a credit scoring domain. In particular, the practitioner should consider the MOE and RBF neural network models. To further strengthen this conclusion, we test for statistically significant differences between credit scoring models. Dietterich has shown that two commonly used statistical tests, differences between two proportions and paired-differences t tests, should never be used to analyze results from cross validation studies that create several random train and test partitions of the data [21]. For cross validation studies of supervised learning algorithms, Dietterich recommends McNemar's test [21], which is used in this paper to establish statistically significant differences between credit scoring models. McNemar's test is a chi-square statistic calculated from a 2×2 contingency table. The diagonal elements of the contingency table are counts of the number of credit applications misclassified by both models, n_{00}, and the number correctly classified by both models, n_{11}. The off diagonal elements are counts of numbers classified incorrectly by Model A and correctly by Model B, n_{01}, and conversely the numbers classified incorrectly by Model B and correctly by Model A, n_{10}. Results of McNemar's test with $p = 0.05$ are given in Table 3. All credit scoring models are tested for significant differences with the most accurate model in the data set. A model whose overall credit scoring is not significantly different from the most accurate model is labeled a superior model; those that are significantly less accurate are labeled as inferior models. Three neural network models, MOE, RBF, and MLP, are superior models for both data sets as is logistic regression. The LDA model and K neighbor are superior for only the Australian credit data. The LVQ and FAR neural network models, and kernel density and CART, are inferior models for both data sets.

A possible explanation for the superiority of the MOE model in credit scoring applications is the ability of the MOE to partition the input space and assign local expert networks to learn the partitioned subspace. This ability is thought to reduce learning interference, which results in

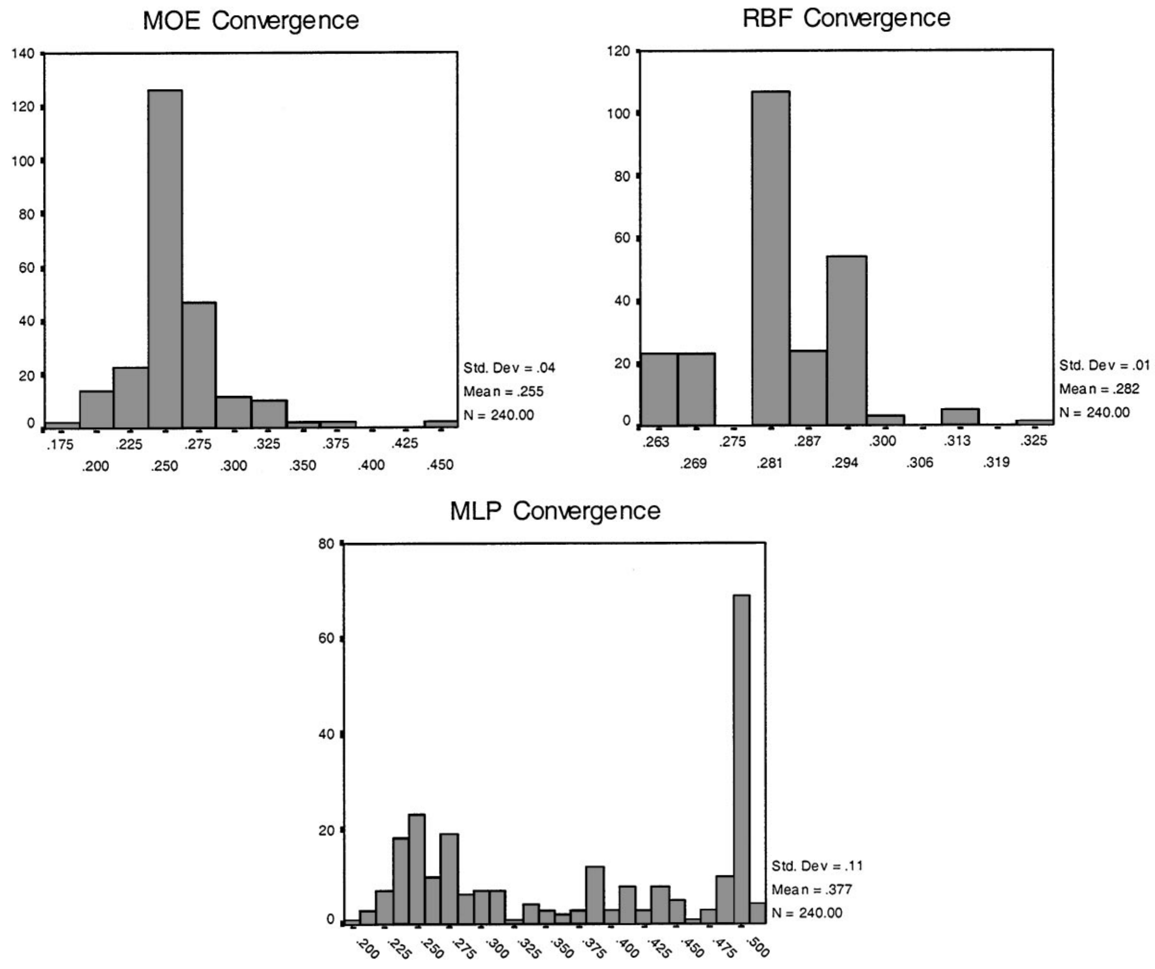

Fig. 5. Neural network convergence German credit data.

a network that converges to more accurate locations in the error surface. To test this concept, the error convergence of the MOE, RBF, and BPN networks were tracked through 240 successive training iterations. The histograms of Fig. 5 summarize these results for the German credit data. It is obvious that the MLP network error converges to a local minimum at 0.50 for a significant number of the trials (67 of 240), and that convergence is somewhat uniformly distributed between the lowest error of 0.2033 to the highest error of 0.515. The mean error convergence point for the MLP network for the 240 iterations is 0.282 with a fairly large standard deviation of 0.11. By contrast, the error convergence of the MOE network appears normally distributed about its mean of 0.255, and its distribution is much more compact with a standard deviation of 0.04. It is also interesting to note that the left-hand tail of the MOE distribution offers the potential for more accurate credit scoring models with a number of error observations of 0.20 or less. The MOE network has six iterations that converge to lower errors than the minimum error of the MLP network, and 164 iterations that converge below the minimum RBF error of 0.282.

5.1. Cost of credit scoring errors

This subsection considers the costs of credit scoring errors and their impact on model selection. Since valid estimates of costs and prior probabilities are not available, we emphasize that this is for illustrative purposes only. It is evident that the individual group accuracy of the credit scoring model can vary widely. For the German credit data, all models except LDA are much less accurate at classifying bad credit risks than good credit risks. Most pronounced is the accuracy of logistic regression with an error of 0.1186 for good credit and 0.5133 for bad credit. In credit scoring applications, it is generally believed that the costs of granting credit to a bad risk candidate, C_{12} is significantly greater than the cost of denying credit to a good risk candidate, C_{21}. In this situation it is important to rate the credit scoring model with the cost function defined in Eq. (13) rather than relying on the overall error [10]. To illustrate the cost function, relative costs of misclassification suggested by Dr. Hofmann when he compiled the German credit data are used; C_{12} is 5 and C_{21} is 1. Evaluation of the cost function also requires estimates of the prior probabilities of good credit π_1 and bad credit π_2 in the applicant pool of the credit scoring model. These prior probabilities are estimated from reported default rates. For the year 1997, 6.48% of a total credit debt of \$560 billion was charged off [28] while Jensen reports a charge off rate of 11.2% for credit applications he investigated [14]. These charge-off experiences do not reflect history with the applicant pool, but rather with applications that have been approved for credit. The error rate for the bad credit group of the German credit data (which averages about 0.45) is used to establish a low value for π_2 of 0.144 (0.0648/0.45) and a high value of 0.249 (0.112/0.45). The ratio n_2/N_2 in Eq. (1) measures the false positive rate, the proportion of bad credit risks that are granted credit, while the ratio n_1/N_1 measures the false negative rate, or good credit risks denied credit by the model.

$$\text{Cost} = C_{12}\pi_2\frac{n_2}{N_2} + C_{21}\pi_1\frac{n_1}{N_1}. \tag{13}$$

Under these assumptions, the credit scoring cost is reported for each model in Table 4. For the German credit data, the LDA (0.429) model is now slightly better than the MOE model (0.432) at the prior probability level of 14.4% bad credit. At the higher level of 24.9% bad credit, the LDA is clearly the best model from an overall cost perspective with a score of 0.54 versus 0.653 for MOE. For the German credit, the remaining neural network models have higher costs than MOE for both prior probability levels. For the Australian credit, the costs of the MOE, RBF, and MLP neural network models are nearly identical at both levels of π_2. The LDA and logistic regression costs are comparable to the neural network models at the lower value of π_2, but become relatively less efficient as the value of the prior probability increases.

The relative group classification accuracy of the neural network models is influenced by the design of the training data. The original training sets for the German credit data are created by randomly partitioning from a collection of 700 good credit examples and 300 bad credit examples. The predominance of good credit examples is a possible explanation for the confusion by the neural network models identifying bad credit risks in the test sets. To improve their accuracy with bad credit risks, two additional strategies are tested for creating neural network training sets. Strategy A is to form training data from a balanced group of 300 good credit examples and 300 bad credit

Table 4
Credit scoring model error function

	German credit $\pi_2 = 0.144$	German credit $\pi_2 = 0.249$	Australian credit $\pi_2 = 0.144$	Australian credit $\pi_2 = 0.249$
Neural models[a]				
MOE	0.432	0.653	0.196	0.243
RBF	0.469	0.707	0.194	0.245
MLP	0.483	0.758	0.198	0.243
LVQ	0.501	0.714	0.237	0.300
FAR	0.668	0.942	0.319	0.388
Parametric models				
Linear discriminant	0.429	0.540	0.204	0.296
Logistic regression	0.471	0.728	0.196	0.258
Non-parametric models				
K nearest neighbor	0.592	0.858	0.227	0.281
Kernel density	0.587	0.901	0.267	0.328
CART	0.569	0.834	0.251	0.294

[a] $C_{12} = 5, C_{21} = 1$.

examples with the remaining 400 examples of good credit used only to test the model. Strategy B involves a single replication of each example of bad credit to produce training data from a nearly balanced 600 bad examples (of which 300 are unique) and 700 good examples. Each of these strategies is tested with the MOE model using 10-fold crossvalidation. Strategy A yields the greatest improvement in the error for bad credit identification with a reduction from 0.4775 reported in Table 1 to 0.2630. The overall error rate for MOE trained with strategy A increases from 0.2434 to 0.2956. Strategy B yields a neural network model with a lower overall error rate, 0.2682, but with less improvement in the identification of bad credit, 0.3072. It is important to note that neither of these training strategies results in a more accurate classifier when overall error rate is the goal, but both are effective when widely disparate costs of misclassification need to be considered. The reader is cautioned that revising the error function defined in Eq. (4) to incorporate group costs may be a more fundamental way to deal with the cost issue.

5.2. *Explanatory ability of neural network credit scoring models*

A key deficiency of any neural network model for credit scoring applications is the difficulty in explaining the rationale for the decision to deny credit. Neural networks are frequently thought of as black-box technology devoid of any logic or rule-based explanations for the output mapping. This is a particularly sensitive issue in light of recent federal legislation regarding discrimination in lending practices. To address this problem, we develop explanatory insights for the MOE neural network trained on the German credit data. This is accomplished by clamping 23 of the 24 input values, varying the remaining input by $\pm 5\%$, and measuring the magnitude of the impact on the two output neurons. The clamping process is repeated until all network inputs have been varied. A weight can now be determined for each input that estimates its relative power in determining the resultant credit decision. Table 5 summarizes these weights for the 12 most important input

Table 5
Neural network decision analysis German credit data

Rank	Input variable	Weight	Cumulative weight
1	Account longevity	0.113	0.113
2	Credit history	0.082	0.195
3	Employment classification	0.078	0.273
4	Checking account status	0.069	0.342
5	Assets owned	0.056	0.398
6	Years in residence	0.054	0.452
7	Other existing loans	0.053	0.505
8	Housing classification	0.053	0.558
9	Amount of loan	0.051	0.609
10	Purpose of loan	0.051	0.660
11	Years employed	0.040	0.700
12	Savings account status	0.038	0.738

variables of the German credit data. These variables, listed by rank order of importance, include: account longevity, credit history, employment classification, checking account status, assets owned including home and car, years in residence, other existing loans, housing classification, amount of loan, purpose of loan, years employed, and savings account status. Other input variables with a low relative power to determine the credit decision are the loan rate as percent of disposable income, age, presence of cosigners, nationality of applicant, whether applicant has a telephone, and the marital status and gender.

With an understanding of the relative power of the input variables, a particular applicant's decision can be explained by a comparison of the mean vectors for both the good and bad credit groups. In Table 6, an applicant's score is reviewed in light of the MOE networks' recommendation to deny credit. Quantities with asterisks indicate reasons for denying credit. For this applicant, the account longevity, while greater than the mean for the bad credit group, is still significantly below the mean for credit approval. Since this is an important factor in the neural model with a weight of 0.113, it is a reason for the decision to deny credit. Other factors for the denial in the order of significance are: credit history, employment classification, assets owned, housing classification, the amount of the loan, the purpose of the loan, number of years employed, and saving account status.

6. Conclusions

This paper investigates the accuracy of quantitative models under consideration by the financial industry for credit scoring applications. The results of this research suggest that neural network credit scoring models can achieve fractional improvements in credit scoring accuracy ranging from 0.5 up to 3%. The use of neural network credit scoring models, however, will require some

Table 6
Sample credit decision explanation German credit data

Rank	Input variable	Weight	Mean of good credit	Mean of bad credit	Applicant score
1	Account longevity	0.113	16.45	2.93	6.0[a]
2	Credit history	0.082	2.8	1.6	2.0[a]
3	Employment classification	0.078	0.18	0.05	0.0[a]
4	Checking account status	0.069	3.0	1.6	4.0
5	Assets owned	0.056	1.5	1.4	1.0[a]
6	Years in residence	0.054	2.6	2.4	3.0
7	Other existing loans	0.053	0.15	0.45	0.0
8	Housing classification	0.053	1.1	1.0	1.0[a]
9	Amount of loan	0.051	28.8	39.9	46.0[a]
10	Purpose of loan	0.051	2.3	1.4	1.0[a]
11	Years employed	0.040	2.8	2.7	2.0[a]
12	Savings account status	0.038	3.5	3.4	2.0[a]

[a]Denotes reasons for denying credit to a specific application.

modeling skills to develop network topologies and devise superior training methods. While the multilayer perceptron is the most commonly used neural network model, the mixture-of experts and radial basis function neural networks should be considered for credit scoring applications. The mixture-of-experts neural network is slightly more accurate than the other credit scoring models for the two data sets investigated in this research. A possible source of advantage for the MOE is the ability to partition the input space so that network training converges closer to the global minimum in the error surface. Some explanatory features of neural network credit scoring models are demonstrated for a hypothetical credit applicant.

This research also suggests that logistic regression is a good alternative to the neural models. Logistic regression is slightly more accurate than the neural network models for the average case, which includes some inferior neural network training iterations. It is also clear that logistic regression should be the choice of the classical parametric models. Logistic regression is 0.02–0.03 more accurate than linear discriminant analysis for the two data sets investigated. The cost of misclassification is also a factor in model selection. One virtue of linear discriminant analysis in this study is the accuracy of bad credit decisions for the German credit data. The nonparametric models (k nearest neighbor and kernel density) and CART did not produce encouraging results but may demonstrate improved accuracy for very large data sets.

While we feel that credit data is somewhat homogenous or standardized in the financial industry, the reader is cautioned that these conclusions are based on two specific credit scoring data sets. We also acknowledge it is possible to improve the accuracy of any quantitative model by fine-tuning, which includes activities like data reduction, data transformation, and an exhaustive investigation of model parameters. The nonparametric methods investigated are known to suffer from the curse of dimensionality and may be more appropriate for huge data sets containing millions of examples.

References

[1] Brill J. The importance of credit scoring models in improving cash flow and collections. Business Credit 1998;1:16–7.
[2] Mester LJ. What's the point of credit scoring? Business Review-Federal Reserve Bank of Philadelphia 1997;Sept/Oct:3–16.
[3] Rosenberg E, Gleit A. Quantitative methods in credit management: a survey. Operations Research 1994;42(4):589–613.
[4] Reichert AK, Cho CC, Wagner GM. An examination of the conceptual issues involved in developing credit-scoring models. Journal of Business and Economic Statistics 1983;1:101–14.
[5] Henley WE. Statistical aspects of credit scoring. Dissertation. The Open University, Milton Keynes, UK, 1995.
[6] Henley WE, Hand DJ. A k-nearest neighbor classifier for assessing consumer credit risk. Statistician 1996;44:77–95.
[7] Tam KY, Kiang MY. Managerial applications of neural networks: the case of bank failure predictions. Management Science 1992;38(7):926–47.
[8] Altman EI. Corporate distress diagnosis: comparisons using linear discriminant analysis and neural networks (the Italian experience). Journal of Banking and Finance 1994;18:505–29.
[9] Davis RH, Edelman DB, Gammerman AJ. Machine learning algorithms for credit-card applications. IMA Journal of Mathematics Applied in Business and Industry 1992;4:43–51.
[10] Frydman HE, Altman EI, Kao D. Introducing recursive partitioning for financial classification: the case of financial distress. Journal of Finance 1985;40(1):269–91.
[11] Coats PK, Fant LF. Recognizing financial distress patterns using a neural network tool. Financial Management 1993;Autumn:142–55.
[12] Desai VS, Conway JN, Overstreet GA. Credit-scoring models in the credit-union environment using neural networks and genetic algorithms. IMA Journal of Mathematics Applied in Business & Industry 1997;8:323–46.
[13] Desai VS, Crook JN, Overstreet GA. A comparison of neural networks and linear scoring models in the credit union environment. European Journal of Operational Research 1996;95:24–37.
[14] Jensen HL. Using neural networks for credit scoring. Managerial Finance 1992;18:15–26.
[15] Lacher RC, Coats PK, Sharma S, Fant LF. A neural network for classifying the financial health of a firm. European Journal of Operational Research 1995;85:53–65.
[16] Piramuthu S. Financial credit-risk evaluation with neural and neurofuzzy systems. European Journal of Operational Research 1999;112:310–21.
[17] Salchenberger LM, Cinar EM, Lash NA. Neural networks: a new tool for predicting thrift failures. Decision Sciences 1992;23:899–916.
[18] Goonatilake S, Treleavan P. Intelligent systems for finance and business. New York: Wiley, 1995.
[19] Quinlan JR. Simplifying decision trees. International Journal of Man-Machine Studies 1987;27:221–34.
[20] Abdel-Khalik AR, El-Sheshai KM. Information choice and utilization in an experiment on default prediction. Journal of Accounting Research 1980;Autumn:325–42.
[21] Dietterich TG. Approximate statistical tests for comparing supervised classification learning algorithms. Neural Computation 1998;10:1895–923.
[22] Hand DJ. Construction and assessment of classification rules. New York: Wiley, 1997.
[23] Rumelhart DE, McClelland JL. Parallel distributed processing: explorations in the microstructure of cognition.. Cambridge, MA: MIT Press, 1986.
[24] Jacobs RA, Jordan MI, Nowlan SJ, Hinton GE. Adaptive mixtures of local experts. Neural Computation 1991;3:79–87.
[25] Moody J, Darken CJ. Fast learning in networks of locally tuned processing units. Neural Computation 1989;3:213–25.
[26] Kohonen T. Self-organizing maps. Berlin, Germany: Springer, 1997.
[27] Carpenter GA, Grossberg S, Rosen DB. Fuzzy ART: fast stable learning and categorization of analog patterns by an adaptive resonance system. Neural Networks 1991;4:759–71.
[28] Gopinathan K, O'Donnell D. Just in time risk management. Credit World 1998;2:10–2.

David West is an Associate Professor of Decision Sciences at East Carolina University in Greenville, North Carolina where he teaches operations management and management science. Dr. West received his Ph.D. in Business Administration from the University of Rhode Island. His research interests include the application of neural network technology to such areas as classification decisions, manufacturing process control, and group clustering. He has published in the *European Journal of Operational Research, Computers & Operations Research,* and *Omega — The International Journal of Management Science*.