Available online at www.sciencedirect.com
SCIENCE DIRECT®
Decision Support Systems 38 (2004) 233–246
Decision Support Systems
www.elsevier.com/locate/dsw

# A principled approach for building and evaluating neural network classification models

Victor L. Berardi[a,*], B. Eddy Patuwo[a], Michael Y. Hu[b]

[a] Department of Management and Information Systems, Graduate School of Management, Kent State University, 6000 Frank Road, Canton, OH 44270, USA
[b] Department of Marketing, Kent State University, Canton, OH 44270, USA

Received 3 September 2002; received in revised form 12 June 2003; accepted 12 June 2003
Available online 23 July 2003

**Abstract**

In this paper, we propose a principled approach to building and evaluating neural network classification models for decision support system (DSS) implementations. First, the usefulness of neural networks for use with e-commerce data and for Bayesian classification is discussed. Next, the theory concerning model accuracy and generalization is presented. Then, the principled approach, which is developed with consideration of these issues, is described. Through an illustrative problem, it is seen that when problem complexity is considered, the classification performance of the neural networks can be much better than what is observed. Furthermore, it is seen that model order selection processes based upon a single dataset can lead to an incorrect conclusion concerning the best model, which impacts model error and utility.


*Keywords:* E-commerce; Decision processes; Data utilization; Classification; Artificial neural networks; Model error; Model variance; Model bias

## 1. Introduction

Commercial transactions generate data on a wide array of customer attributes for use by organizations in creating and improving organizational processes and customized marketing programs. Web-related retailers can link customer credit card purchases to their site's search and browsing records, and even to external databases, to develop more sophisticated and accurate customer profiles. This information can then be used to develop individualized advertising schemes, email distributions, and suggested add-on purchase items, in addition to any number of sophisticated programs and processes (e.g., Refs. [4,12,16,22]). These repositories of data must be properly managed to ensure the highest return on an organization's investment. Data architectures and algorithms are evolving to more efficiently store and retrieve data and to effectively transform it into the information and knowledge necessary to provide superior economic returns and increased customer satisfaction. As Raymond and Blili [17] note, ''organizational learning must be adapted to network enterprises, virtual enterprises, and other new organizational forms enabled by information technology'' to reap full benefits and value.

* Corresponding author. Tel.: +1-330-244-3439; fax: +1-330-494-6121.
*E-mail address:* vberardi@kent.edu (V.L. Berardi).

0167-9236/03/$ - see front matter © 2003 Elsevier B.V. All rights reserved.
doi:10.1016/S0167-9236(03)00093-9

Data mining has been a successful approach for improving data utilization and for supporting organizational learning. With data mining, which utilizes sophisticated statistical correlation procedures, organizations can find and exploit relationships behind customer behaviors and characteristics that are seemingly unrelated. Appropriate organizational and individual decisions can then be made. Data mining has gained in popularity with data availability and as customizable commercial offerings have entered mainstream use. They are particularly applicable when large data repositories from well-established organizational processes are aggregated in data warehouses and data marts. Unfortunately, these characteristics are often not available in newly emerging markets or for rapidly changing products and e-commerce processes with increasingly short lifecycles.

More generally, decision support systems (DSSs) have also enjoyed years of successful application to organizational management. Their track record, ease of use, and numerous vendor offerings make them a natural choice in this process. DSS models have become increasingly sophisticated to address issues related to data complexity as well as to organizational process, product, and marketplace changes. Not only can DSSs incorporate the aforementioned data mining approaches, they might also include intelligent agents, and artificial intelligence techniques, among others, used in isolation or combination.

Intelligent agents have been successfully incorporated into many decision support processes. According to a purchasing decision-making process framework, electronic commerce agents can be classified into need identification, product brokering, merchant brokering, negotiation, purchase and delivery, and product/service evaluation. In addition, electronic commerce agents are being employed in fraud protection, auction support, learning agents, and behavioral agents among others [24]. Indeed, the Spring 2000 issue of the *International Journal of Electronic Commerce* is dedicated to electronic commerce agents. Subsequently, Wu [29] applies artificial agents to pricing, investment, and operating strategies. The author notes that "artificial agents can sometimes only be tested by pushing the limits of currently available nonlinear optimization software" thereby acknowledging both the difficulty and importance in achieving useful, reliable solutions to these important problems.

Verhoef and Donkers [25] propose a conceptual model underlying DSSs that combines electronic commerce purchase information with socio-demographic information to predict not only the purchase of products or services but also the potential value of customers. In their study, the authors show how univariate and multivariate probit models can improve prediction of customer potential value. It is worth noting that, as discussed below, neural networks are an excellent theoretical fit for inclusion in their framework and for improving the significance of predictive performance. Kim et al. [13] discuss the application of decision-tree induction techniques to personalized advertisements. The authors propose rule-extraction, machine learning, and decision-tree techniques to personalization while noting that artificial intelligence (neural networks) is an alternative approach.

The previous discussion, while not intended as a comprehensive survey of the many ways in which e-commerce data is utilized to improve decision making, does illustrate the important role that artificial intelligence techniques, such as (artificial) neural networks, can play in organizational knowledge management and learning. Neural networks possess many characteristics that make them appealing for e-commerce. Neural networks simulate the tabula rasa, or clean slate, learning processes of biological systems including the human brain. Unlike traditional statistical methods such as discriminant analysis or regression methods, tabula rasa learning is appealing because it makes no prior assumptions on the form of a solution. Neural networks allow the data itself to determine the appropriate model form. Considering that e-commerce applications can generate dozens of variables on individual customers with each transaction, and that each system may contain customers with transactions from multiple product and market sources, the limitations of traditional fixed-form models is readily apparent.

A second desirable property of neural networks is the fact that they are consistent estimators. A consistent estimator is one that converges to the object of estimation (e.g., a multivariate function) asymptotically for large sample sizes. It has been shown by several authors [5,9,10,20,21,28] that neural networks can indeed approximate any function to desired accuracy. Richard and Lippmann [18] and Hung et al. [11], meanwhile, show that neural networks are capable of estimating posterior probability distributions. Given

the large amounts of data with complex relationships generated in e-commerce settings, the usefulness of neural networks as classification models is promising.

Another strength of neural networks for e-commerce settings is the fact that they can handle "noisy" data that might cause errors in traditional computer programming approaches. In addition, they can provide output for decision making when clear choices of right and wrong may not exist. Given the variability inherent in human choice and actions, noisy data is the rule rather than the exception. Furthermore, the output from a neural network for a classification problem represents a posterior probability of group membership that can be used to determine appropriate company actions. For example, based upon input factors, a customer might be identified as having a low probability of making additional purchases in the near term. This information could be used to quickly generate a short-term buying incentive tailored toward highly profitable related items. Furthermore, sensitivity analysis can be performed to determine what effect changes on the customer's input characteristics might have on purchase behavior. Opportunities then could be crafted, for example, to appropriately entice the customer into becoming a more regular purchaser.

The above points show that neural networks are useful for e-commerce applications which should make them a valuable, though currently underutilized, addition to the decision maker's tool chest. In many cases, neural networks might not be used simply because they are perceived as a "black box" that is not fully understood. In other cases, one might point to instances in which neural networks performed well during model building and even testing but then failed to live up to expectations during actual use. Indeed, it is the very advantages of neural networks—in particular, their reliance on the data itself to determine appropriate model form and their universal approximation capabilities—that can sometimes limit their usefulness in practice. Hence, one must be concerned not only with the ability of neural networks to learn presented data accurately but also to generalize well to unseen future data. The task of building models to achieve consistent classification performance is especially difficult—and of great strategic and economic value to organizations—early in the product/market lifecycle or in highly dynamic systems where large repositories of data simply may not exist.

The purpose of this paper is to present an approach for building and evaluating neural network models for DSS implementations and e-commerce applications. Understanding how to utilize e-commerce data within a neural network framework to yield more accurate and reliable classification decisions is a primary concern in this endeavor. The improved models could be utilized more quickly after a new product launch or a process change as part of intelligent agents or other DSS components. To achieve this goal, issues related to Bayesian classifiers, model estimation error, model bias and model variance are reviewed first. Then methodology for building and evaluating neural network models classification is discussed. Examples will be presented to illustrate the approach.

## 2. A principled approach for building and evaluating neural network models

### 2.1. Bayesian classification and posterior probabilities

In a classification problem setting, the underlying population generating process characterizes the relationship between the input attributes $\mathbf{x}$ and the output classes $\omega$. This relationship defines the *posterior probability* distribution. At this point, simply note that the posterior probability $P(\omega_j|\mathbf{x})$ is the probability that an object belongs to a specified group $\omega_j$ after we have observed information $\mathbf{x}$ related to the object.

Posterior probability forms the basis for the well-known Bayesian classification theory [6]. According to Bayes rule, if we obtain an observation $\mathbf{x}$, the prior probability of belonging to group $j$, $P(\omega_j)$, will be modified into the posterior probability, $P(\omega_j|\mathbf{x})$, that object $\mathbf{x}$ belongs to group $j$ by the following equation:

$$P(\omega_j \mid \mathbf{x}) = \frac{f(\mathbf{x}, \omega_j)}{f(\mathbf{x})} = \frac{f(\mathbf{x} \mid \omega_j)P(\omega_j)}{\sum_{j=1}^{M} f(\mathbf{x} \mid \omega_j)P(\omega_j)}, \quad j = 1, 2, \ldots, M. \tag{1}$$

Bayes rule shows how observing the value of $\mathbf{x}$ changes the prior probability $P(\omega_j)$ to the posterior

probability $P(\omega_j|\mathbf{x})$ upon which the classification decision is based. For example, consider an e-mail based mass marketing campaign that might generate a 2% response rate (i.e., prior probability $P(\omega_j)$ of response). Upon learning demographic and psychographic information on individuals (i.e., object attributes $\mathbf{x}$) the probability of response can be modified up or down from $P(\omega_j)$ to $P(\omega_j|\mathbf{x})$ to reflect this new information.

Furthermore, suppose that a particular $\mathbf{x}$ is observed and is to be assigned to a group. Let $\lambda_{ij}(\mathbf{x})$ be the cost of misclassifying $\mathbf{x}$ to group $i$ when it actually belongs to group $j$. Since $P(\omega_j|\mathbf{x})$ is the probability that the object belongs to group $j$ given $\mathbf{x}$, the expected loss associated with assigning $\mathbf{x}$ to group $i$ can be minimized by following the *Bayesian decision rule* for classification,

$$\text{Decide } \omega_k \text{ for } \mathbf{x} \text{ if } L_k(\mathbf{x}) = \min_{i=1,2,\ldots,M} L_i(\mathbf{x}) = \min_{i=1,2,\ldots,M} \sum_{j=1}^{M} \lambda_{ij}(\mathbf{x}) P(\omega_j \mid \mathbf{x}).$$

Assuming equal misclassification costs, then the Bayesian decision rule is to assign an object to the group associated with the maximum posterior probability:

$$\text{Decide } \omega_k \text{ for } \mathbf{x} \text{ if } P(\omega_k \mid \mathbf{x}) = \max_{j=1,2,\ldots,M} P(\omega_j \mid \mathbf{x}).$$

This decision rule yields a minimum expected misclassification rate or, in other words, the maximum overall number of correct classifications in the long run. The Bayesian decision rule can be generalized to include unequal misclassification costs as described in Ref. [6]. For example, when considering the potential value of customers, as in Verhoef and Donkers [25], the costs of misidentifying a truly valuable customer as less important might not have the same consequence to the organization as the reverse error. Berardi and Zhang [2] demonstrate the efficacy of neural network classifiers when unequal misclassification costs are relevant.

The above discussion clearly shows the important role of posterior probabilities in the Bayesian classification decision. The theoretical relationship linking estimation of Bayesian posterior probabilities to minimizing squared error cost functions has long been known. Papoulis [15] shows that the mapping function F: $\mathbf{x} \rightarrow \mathbf{y}$, which minimizes the expected squared error is the conditional expectation $E[\mathbf{y}|\mathbf{x}]$. Since in a classification problem the output $\mathbf{y}$ is a vector of binary values, it can be easily shown (see, for example, Ref. [11]) that $E[\mathbf{y}|\mathbf{x}] = P(\boldsymbol{\omega}|\mathbf{x})$. Since neural networks can approximate any function $F$ with arbitrary accuracy (universal approximators), then neural network outputs are indeed good estimators of the posterior probabilities $P(\boldsymbol{\omega}|\mathbf{x})$.

Many recent papers have provided linkage between neural networks and posterior probabilities [3,11,18,19,23,26,27] for squared error functions and for the cross-entropy [1] error function. It should be noted that most of these articles assume infinite sample sizes. It is Hung et al. [11] and Richard and Lippmann [18] who show that neural networks minimizing squared-error and cross-entropy cost functions are capable of estimating posterior probabilities for finite sample sizes. The fact that neural networks can estimate posterior probabilities makes them powerful classification tools. It helps to explain their many reported successes and is a major reason for the high level of research activity.

For e-commerce applications, we desire to use neural networks to approximate accurately the Bayesian posterior probabilities in order to make consistently accurate classification decisions. We see from Eq. (1) that computing the posterior probabilities requires specification or estimation of prior probabilities $P(\omega_j)$ and conditional likelihood functions $f(\mathbf{x}|\omega_j)$. While $P(\omega_j)$ can be directly estimated from observed data, the strength of a neural network approach is that the neural network directly estimates the posterior probability $P(\omega_j|\mathbf{x})$ based upon the data presented. While all decision making models require the sample data to reflect the true population, as noted earlier, neural networks are especially data dependent making the evaluation approach presented in this research particularly relevant.

An object's posterior probability of belonging to a specific group is of great interest, since it allows us to make optimal decisions regarding the class membership of new data. For neural network classification problems, accurately modeling the underlying generating process is analogous to closely fitting the posterior probability distribution, which in turn, is

key to maximizing expected classifications. Therefore, from a perspective of statistical classification generalization, we are concerned with the model estimation error relative to the posterior probability distribution because this directly impacts the ability to generalize results. In e-commerce applications, this means for example, we might be utilizing a transactional database of customer segments $\omega_j$ and their attributes $\mathbf{x}$ to forecast the behavior of a new customer, which is based upon $P(\omega_j|\mathbf{x})$. Maximum expected classification rates are achieved when the neural network model accurately estimates the posterior probabilities of group membership, which results in not only minimized marketing costs but also in maximizing the value to the organization as well.

### 2.2. Model estimation error: model bias and model variance

As noted above, neural networks are intimately dependent upon the data used for model building. Variations in the training data set composition can have significant impact on neural network performance for unseen objects. This is especially salient for new products and emerging processes where the number of objects in a database is still relatively few given problem complexity. In addition, e-commerce applications are often characterized by changing websites or DSS infrastructures that may continually impact the data generating process. Therefore, for each problem domain, we must be concerned with the fact that a large number of data sets are possible and that our current database represents but one particular realization. To acknowledge this fact, the network model's estimation error in the context of multiple data sets can be written as [8]

$$E_D[(f(\mathbf{x};D) - E[\mathbf{y} \mid \mathbf{x}])^2]. \tag{2}$$

$E_D$ represents the expectation with respect to all training sets, $D$. In other words, it is the average over all possible training sets with fixed sample size $N$. The term $f(\mathbf{x}, D)$ represents the neural network estimate of the true function $\mathbf{y}$ given inputs $\mathbf{x}$. The term $E[\mathbf{y}|\mathbf{x}] = P(\boldsymbol{\omega}|\mathbf{x})$ represents the best possible estimate, which for a neural network classification problem setting, is the posterior probabilities upon which the classification decision will be made.

Therefore, Eq. (2) represents the expected model estimation error of the neural network classifiers over all possible data sets that could be used in network training. Decomposing the model estimation error from Eq. (2) into components to measure the average performance of a model and performance variation resulting from different data sets gives

$$\begin{aligned} &E_D[(f(\mathbf{x};D) - E[\mathbf{y} \mid \mathbf{x}])^2] \\ &\quad = \underbrace{(E_D[f(\mathbf{x};D)] - E[\mathbf{y} \mid \mathbf{x}])^2}_{\text{``Model Bias''}} + \underbrace{E_D[(f(\mathbf{x};D) - E_D[f(\mathbf{x};D)])^2]}_{\text{``Model Variance''}}. \end{aligned} \tag{3}$$

As can be seen, the total mean-squared estimation error for a classifier can be thought of as consisting of two components, *model bias* (*squared*) and *model variance*. The model bias measures the extent to which the average of the network function $E_D[f(\mathbf{x}; D)]$ differs from the best possible function $E[\mathbf{y}|\mathbf{x}]$. In other words, model bias directly considers the neural network's ability to learn the underlying generating process. Recalling that neural networks are consistent estimators, it is apparent that the bias component can be affected by altering the model complexity. This is achieved by changing the model order via the number of hidden nodes. Model variance, meanwhile, measures how sensitive the network estimates $f(\mathbf{x}; D)$ are to specific data sets. High model variance is indicated when model performance changes greatly based upon data set changes. Hence, to evaluate the variance component requires multiple data sets for training. Generalization performance of a classifier can suffer if one or both of these components are large.

Often a tradeoff exists between the bias and variance contributions to the model estimation error which Geman et al. [8] called the *bias/variance dilemma*. Many methods have been proposed in dealing with the issue of balancing the bias and variance components. Often, these efforts attempt to "smooth" network outputs thereby reducing the variance component. Such approaches are indicated only when model variance dominates the total error as the price to pay is typically an increase in bias. Therefore, the effect on total model estimation error can be ambiguous. It bears repeating that this issue is especially important for neural network modeling, which

relies heavily on the data set in determining the appropriate model form.

With specific reference to neural network classifiers, these efforts seek to improve generalization capabilities for unseen objects. The fact that both model bias and model variance contribute to model estimation error—and hence the generalization performance of neural networks—suggests that an approach utilizing multiple training data sets is necessary to fully evaluate the performance of proposed models. Next we will discuss how data available in e-commerce environments can be utilized in the principled approach.

### 2.3. Data utilization

Determining the model bias and the model variance in Eq. (3) requires estimating $E[\mathbf{y}|\mathbf{x}]=P(\boldsymbol{\omega}|\mathbf{x})$, the posterior probabilities. The data available from e-commerce applications present an opportunity to construct the group likelihood function $f(\mathbf{x}|\omega_j)$ and to compute the posterior probabilities directly from Eq. (1). This is indeed a potentially valuable fact that should be fully exploited and to which neural networks are particularly well suited.

Let $\mathbf{x}=[x_1, x_2, \ldots, x_p]$ be customer attributes associated with an identified class $\omega_j$, such as customer segments, that are contained in an e-commerce database. Knowledge Discovery and Data Mining procedures for variable and observation selection may be employed as desired here. Next, commercially available distribution fitting procedures, such as ExpertFit [14] can be used to determine the marginal distributions of $x_1, x_2, \ldots, x_p$. Assuming independence, the likelihood function can be computed as follows: $f(\mathbf{x}|\omega_j)=f(x_1|\omega_j)f(x_2|\omega_j)\ldots f(x_p|\omega_j)$. If the marginal random variables are believed to be correlated, first break up $\mathbf{x}$ into its independent components $\mathbf{x}=[\mathbf{x}^1, \mathbf{x}^2, \ldots, \mathbf{x}^{\mathbf{k}}]$. After determining the joint densities of the individual components, compute the likelihood function from $f(\mathbf{x}|\omega_j)=f(\mathbf{x}^1|\omega_j)f(\mathbf{x}^2|\omega_j)\ldots f(\mathbf{x}^{\mathbf{k}}|\omega_j)$. This information, combined with the prior probabilities $p(\omega_j)$ of each group in the database, is all that is needed to calculate the object level posterior probabilities of group membership $P(\omega_j|\mathbf{x})$.

Knowing the group likelihood function $f(\mathbf{x}|\omega_j)$ gives us the ability to generate data from the fitted distributions, which allows us to generate multiple data sets of the same size as the original set. In addition to ExpertFit [14], methods such as bootstrap resampling or the jack-knife procedure can be used to generate multiple data sets for neural network training. Interested readers should see Efron and Tibshirani [7] for a review of implementing these procedures. The multiple data sets are used to decompose the model error into its bias and variance components. This, in turn, is critical to understanding the dynamics of error changes related to, for example, hidden node changes. This information can then be used to improve the model building process.

### 2.4. Model performance evaluation

The ability to determine the object level posterior probabilities represents an outstanding potential use for neural networks in e-commerce applications. Decision makers can estimate not only how well a neural network model will perform with a single data set of given size but also in the generalization of the model to future observations. Model performance, therefore, can be thoroughly investigated while minimizing the common problem of overestimating the model's true utility. To realize this advantage, though, it is necessary to estimate total model error, model bias, and model variance using multiple data sets in the proposed approach as described next.

The Monte Carlo procedure used to evaluate the bias, variance, and total estimation error is adopted from Geman et al. [8] and is described now. Recall from Eq. (3) that

$$\text{Model Bias} = \left(E_D[f(\mathbf{x};D)] - E[\mathbf{y} \mid \mathbf{x}]\right)^2, \tag{4}$$

and that the variance is

$$\text{Model Variance} = E_D\left[\left(f(\mathbf{x};D) - E_D[f(\mathbf{x};D)]\right)^2\right], \tag{5}$$

where $f(\mathbf{x};D)$ is the neural network estimator for any given training set $D$ and $E[\mathbf{y}|\mathbf{x}]$ is the true function—the posterior probabilities, which are computed from Eq. (1).

The model bias and variance components are estimated by generating $S$ independent random training sets $D^1, D^2, \ldots, D^S$ used for training $S$ neural network estimators $f(\mathbf{x};D^1), f(\mathbf{x};D^2), \ldots, f(\mathbf{x};D^S)$. The expected response of the neural networks over all data

sets, $E_D[f(\mathbf{x};D)]$ is denoted by $\bar{f}(\mathbf{x})$, the average response at $\mathbf{x}$, and is calculated as

$$\bar{f}(\mathbf{x}) = \frac{1}{S}\sum_{s=1}^{S} f(\mathbf{x}, D^S). \quad (6)$$

The model bias and model variance components of an object $\mathbf{x}$ are estimated using:

$$\text{Model Bias } (\mathbf{x}) \approx (\bar{f}(\mathbf{x}) - E[\mathbf{y} \mid \mathbf{x}])^2 \quad (7)$$

$$\text{Model Variance } (\mathbf{x}) \approx \frac{1}{S}\sum_{s=1}^{S}[f(\mathbf{x}, D^S) - \bar{f}(\mathbf{x})]^2 \quad (8)$$

$$\text{Total Estimated Model Error } (\mathbf{x}) = \text{Model Bias } (\mathbf{x}) + \text{Model Variance } (\mathbf{x}) \quad (9)$$

An approach equivalent to the hold-out method of training and testing typically employed for real data sets is utilized in this framework. The hold-out method typically is used to deal with the bias–variance tradeoff. In most practical problems, the neural network is trained on the majority of the data while a small hold-out sample is used to test model performance. Sometimes a second hold-out sample, called a validation set, is also used. This additional procedure stems from concerns that the small test set may not adequately cover the generating process input–output space and hence may itself contribute to bias and variance problems.

Note that bias and variance in the test set can be mitigated by having a large test set to provide adequate coverage of the input–output space. Therefore, training on $S$ independent data sets generates the neural network models. These trained models are then presented with each of the test set object attributes $\mathbf{x}$. The neural network estimated outputs $f(\mathbf{x};D^S)$ are compared to $E[\mathbf{y}|\mathbf{x}]$ and the model bias and model variance components of total estimated model error are computed using Eqs. (6)–(9).

While the error measures just described provide insights into the estimation performance of neural network models, results based upon classification rates provide a link to a more traditional evaluation basis. The observed classification rate is probably the most commonly reported measure. Dividing the correct classifications by the total number of observations yields the *observed classification rate*. Usually, only the observed classification rate is reported because a single set of real data is available but it may be misleading as a performance metric as discussed below. However, for a data set with known posterior probabilities, it is possible to determine the expected classification rate using the Bayesian decision rule. This *Bayesian classification rate* represents the optimal long-run classification rate one can expect to achieve using the theoretical posterior probabilities for classification. Taking the ratio of these classification rate measures—observed to Bayesian—yields the *Bayesian classification efficiency (BCE)*.

The classification efficiency provides a clearer picture of the actual classification performance of the neural network model than does the observed classification rate because it puts the performance in context of problem difficulty. For example, consider a two-group problem that results in an observed classification rate of only 60%. If the problem has an expected Bayesian classification rate of 66%, it is seen that the Bayesian classification efficiency is nearly 91%—a much better model classification performance than first appears.

Below we summarize a principled approach for building and evaluating neural network models.

1. Obtain data related to decision process of interest.
2. If data set is sufficiently large, divide it into $S$ training sets and one large test set. Or, if limited data are available, use a distribution fitting program such as ExpertFit [14] to generate a data set of a desired size from the fitted distributions, and divide it into S training sets. Then use the original data set as the test set.
3. Compute the posterior probabilities for all data points directly from Eq. (1) as described earlier.
4. Train the neural networks on the S training sets to get neural network estimators $f(\mathbf{x};D^2)$, …, $f(\mathbf{x};D^S)$ Repeat this step for a range of hidden nodes to sufficiently capture the dynamics of the total model error, the bias component, and the variance component. Specific guidelines concerning how many hidden nodes to investigate cannot be given as this is dependent upon the sample size and the complexity of the underlying generating process.

5. Using the test set, compute model bias, model variance and total estimated model error as in Eqs. (6)–(9).
6. Decide on the best model structure based on the error measures in step 5. For neural networks, this means choose the hidden nodes (model order) that minimize the total model error.
7. Evaluate the performance of the neural network model using the observed classification rate, the Bayesian classification rate, and the Bayesian classification efficiency.

## 3. Simulation examples

### 3.1. Experimental design

A two-way factorial experiment is specified in this research with the number of hidden nodes and training data sample size as major factors. The number of hidden nodes is a prime consideration in neural network models, given that they are consistent estimators. It is expected that hidden node changes will directly impact total model estimation error, model bias, and model variance. Five levels of hidden nodes are investigated as this proved sufficient for the problems investigated. Training data set sample size is another important consideration in the performance and specification of neural network models. The impact of training sample size on estimation performance can determine if a model has practical usefulness to decision makers. This is evident when too little data yields even the best models as not accurate enough for making the desired decisions. Furthermore, given that neural networks are consistent estimators, some interaction between the model order (hidden nodes) and training data sample size might be expected. Two levels of training sample sizes (180 and 540 objects) are used. Therefore, the $2 \times 5$ experimental design carried out in this project results in 10 experimental cells. Thirty replications within each cell are achieved via 30 training data set replications with model performance evaluated on a test set of 2400 objects. The test set is an independent sample obtained from the same data generating processes as the training data, where each object in the training and test sets could represent a customer to be classified.

The illustrative problems contain independent two- and three-group settings that are analyzed separately. The two-group example represents a special case problem from a neural network and classification standpoint. For an e-commerce situation it may be appropriate where one might be trying to decide the probability of a subsequent customer purchase (yes or no) so that appropriate enticements can be offered. A three-group problem sufficiently represents the general case classification problem for neural network modeling and might be appropriate in more complex customer segmentations. The three-group problem is a more complex generating process and the error component patterns observed might be expected to differ as neural network model order is changed.

The performance of the neural network models in estimating posterior probabilities and classification performance will be analyzed in the following manner. First, the statistical significance of changes in total estimation error related to the experimental factors, sample size and neural network hidden nodes will be investigated within the ANOVA framework. Then, the total estimation error and the model bias and variance components will be discussed. In particular, the impact of altering the model complexity via hidden node changes will be explored graphically. Next, the effect of the commonly employed model order selection procedure on expected performance will be covered. It is seen that the process of selecting the ''best'' number of hidden nodes can lead to variable network performance in actual use, a fact not obvious when utilizing a single data set only. Finally, the network classification performance from observed (neural networks) and Bayesian classification rate perspectives will be presented and it will be seen that the models do a more efficient job at correctly classifying objects than the observed classification rate initially indicates. It is important to note that all results are reported for a ''hold-out'' test sample that is independent of the training data sets and has not been presented to the neural networks until after the models have been fixed.

### 3.2. Data set information

For illustration purposes, in this paper we use simulated data sets generated with eight input varia-

bles that are grouped into five independent components $\mathbf{x}=[\mathbf{x}^1, \mathbf{x}^2, \ldots, \mathbf{x}^5]$. Here,

| | |
|---|---|
| $\mathbf{x}^1=[x_1, x_2, x_3]$ | Correlated trivariate normal random variables |
| $\mathbf{x}^2=[x_4, x_5]$ | Correlated bivariate Bernoulli random variables |
| $\mathbf{x}^3=[x_6]$ | Weibull random variable with concave density function |
| $\mathbf{x}^4=[x_7]$ | Weibull random variable with convex density function |
| $\mathbf{x}^5=[x_8]$ | Binomial random variable with state space {0, 1, 2, . . ., 10} |

These random variables are chosen to represent those likely to be encountered in e-commerce applications. The normal distribution is widely applicable to many problems while correlation between the three normally distributed input variables yields additional modeling realism and flexibility. Bernoulli variables model yes/no and true/false features of a transaction, while the binomial component represents a countable characteristic. The Weibull components are highly flexible as parametric choices change the distribution from convex to concave. Choices leading to convex shapes model features distributed in exponential fashion such as seen in many service situations. A concave Weibull distribution is appropriate in features where rates of occurrence increase over time. Posterior probabilities for each object are calculated as in Eq. (1) above.

In the example problems, training data set sizes of 180 and 540 objects were used to train the neural networks $f(\mathbf{x};D^S)$, where $S=30$ training set replications are used. The hold-out test set contains 2400 objects and is the basis for all reported results. Sample sizes are chosen merely to facilitate investigation of the bias and variance components of estimation error in this illustrative problem setting and can be any size for specified problems of interest. For e-commerce applications, though, smaller customer databases would be expected early in the new product development cycle and where reliable corporate intelligence could have the most economic and strategic value. The smaller sample size is a subset of the larger one as often occurs in practice when one must consider tradeoffs between the cost and value of obtaining more data for the decision making process.

A model order selection procedure commonly used in neural network applications is employed here. This is achieved by varying the model architectures from zero to five hidden nodes. In typical fashion, the number of hidden nodes resulting in the lowest total estimated model error is retained and is used in the reporting of classification results. Networks with one hidden node are excluded from the presented analysis because the performance of these models proved to be inferior and would readily be rejected by decision makers. Note that alternative model order selection approaches, such as cascade correlation, can readily be used within the proposed framework.

## 4. Results

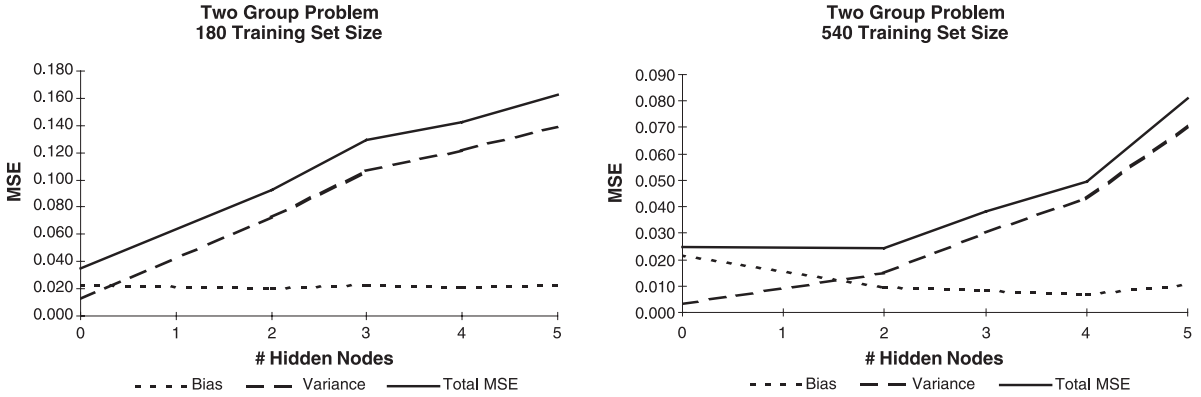
### 4.1. Model estimation error

The two-way ANOVA results for the dependent measure, total model error, of the two-group and three-group problems indicate strong significance for the main effects (sample size and hidden nodes) and the interaction term as all $p$-values $< 0.001$. The strong interaction between hidden nodes and sample size is reassuring given that neural networks theoretically are consistent estimators. This means that as sample size increases, the appropriate model complexity can increase concurrently. In neural network classification models, model complexity is increased by adding hidden nodes and the resulting connections.

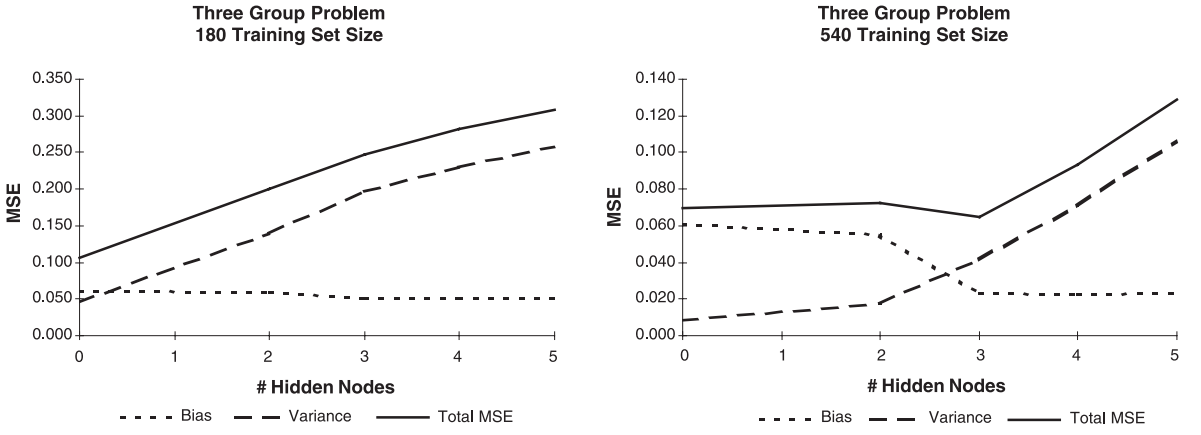
As a practical matter, though, the presence of interaction necessitates utilizing one-way ANOVA of hidden nodes conditioned on the training sample size. The one-way ANOVA results on hidden nodes for all four cases—the two-group and the three-group problems at training sample sizes of 180 and 540 objects each—show strong statistical significance of hidden nodes as all $p$-values $< 0.0001$. In other words, for both the two- and three-group problems altering model order via hidden node changes results in significant changes in model estimation performance. As expected, the model complexity is an important factor in posterior probability estimation performance. Practitioners, naturally, will be interested in finding the best number of hidden nodes for their particular situation.

Multiple comparison tests were run to identify homogeneous subsets of hidden nodes, however, a graphical analysis will be presented instead as it effectively reflects these model error results and it includes bias and variance information in a concise presentation, too. Fig. 1 contains the total estimation error, the model bias, and model variance performance of the neural network models across the hidden nodes. Panel A, contains results of the two-group problem, while Panel B presents those of the three-group problem. The scale has been set to facilitate visualizing the dynamics of the error components.

In Panel A for the two-group problem with a training sample size of 180 objects, total estimation error rises steadily as the hidden nodes increase. Total model estimation error is minimum (0.0351) at zero hidden nodes and increases fivefold to 0.1622 at five hidden nodes. At zero hidden nodes, it is seen that model bias is the largest error component. As the hidden nodes are changed, the increase in model estimation error results almost entirely from the variance component as the bias remains essentially unchanged. For a smaller sample size, it is seen that simple models adequately approximate the generating process as it is represented via the training data. Insufficient information exists in the data to reduce model bias through more complex models and trying to do so merely leads to large increases in model variance.




**Panel A. Two group problem with 180 and 540 objects in training sample sets.**




**Panel B. Three group problem with 180 and 540 objects in training sample sets.**

Fig. 1. Total model estimation error and bias and variance components across hidden nodes.

As the training set size is increased to 540 objects for the two-group problem in Panel A, more information concerning the generating process is presented to the neural networks and more complex models can be utilized. The total model estimation error remains flat for this two-group problem as the number of hidden nodes increases from 0 to 2, then they exhibit a rise in error as hidden nodes increase from three through five. The two hidden node model yields minimum model error at 0.0242. The error increases more than threefold to 0.0809 at five hidden nodes. At zero hidden nodes, the bias component dominates the model error. From zero to two hidden nodes, it is seen that a decrease in model bias is cancelled by an increase in the variance component. Beyond two hidden nodes, the variance component dominates any changes in bias, resulting in overall estimation error increases.

Note that the minimum total error (0.0351) when training on 180 objects, which occurs at zero hidden nodes, is nearly 45% greater than that achieved with 540 objects in training at two hidden nodes (0.0242). Furthermore, the impact on error of changing the model order in the 540 training sample size case is about one-third less than the fivefold increase seen in the 180 training sample size case. Given this information, model builders could trade off costs associated with obtaining more data for training against the posterior probability estimation improvements expected. In addition, with 540 objects in training, modelers could make informed decisions concerning the impact of selecting the more parsimonious zero hidden node model as opposed to the two hidden node model. Decisions could be made as to whether the slight (approximately 2%) improvement in overall estimation performance, and the concurrent decrease in model bias at two hidden nodes, is more important than the increase in prediction variability expected.

The three-group case in Panel B exhibits similar patterns, particularly for the smaller training sample size. At 180 objects, the minimum model error is at 0.1066 and increases steadily to 0.3081 at five hidden nodes. Once again, while the bias component is larger than variance at zero hidden nodes, it remains flat as hidden nodes are changed. The variance component, meanwhile, comprises nearly all of the model error changes throughout the range. When 540 objects are used for training, minimum model error (0.0648) is achieved at three hidden nodes.

It is interesting to note that, as more information on the generating process is presented via the larger training sample size, a higher-order model is preferred. From two to three hidden nodes a large impact on the bias and variance components is observed even though from zero to two hidden nodes both components are only moderately impacted. The three-group problem is more complex for the network to approximate than the two-group structure is. However, while models built on a single data set might not observe large changes in overall estimation performance, the dynamics of the individual components are apparent and brought to the forefront via the proposed approach. As before, model builders could decide if a higher-biased, less-variable model is preferred, in which case the more parsimonious zero hidden node model would be chosen. If decision makers emphasize low bias and can accept some prediction variability, a model with three hidden nodes would be selected. The value of utilizing multiple data sets, which is possible in this proposed framework, becomes even more evident when considering it in context of the commonly applied model order selection procedure.

### 4.2. Hidden node distribution

Recall that neural network modelers often train networks of various hidden nodes and use a hold-out sample to select the ‘‘best’’ number of hidden nodes. This is the model that would then be used in practice. Fig. 2 contains a frequency distribution of the number

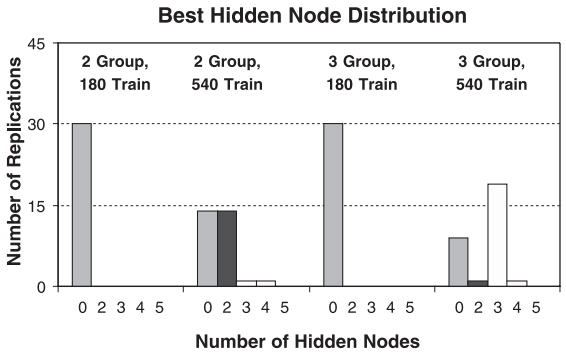



Fig. 2. Number of replications ($S=30$) at each number of hidden nodes (0, 2, 3, 4, 5) resulting in the lowest estimation error.

of data sets that yielded each hidden node as "best". For example, in the two-group case when trained on 180 objects, all 30 data set replications yielded zero hidden nodes as the "best" choice in the model order selection process. The same is true in the three-group case when the training set size is 180 objects. However, when the training sample size is increased to 540 objects, the selection process varies from zero to four hidden nodes, which exhibits even greater variability in the more complex three-group case.

Consider the two-group problem with a training sample size of 540 objects. If we happen to train the neural network using a data set which yields four hidden nodes as the best instead of two, an increase of more than 100% can be expected in the total model estimation error (see Fig. 1, Panel A). The implications for practical applications are significant when one considers the complexity of the data generating process the neural networks are approximating in e-commerce situations and the large number of training data likely to be employed. It would not be surprising to find even larger variations in the number of hidden nodes being chosen as "best" merely because of random data set variations. This could have significant practical performance consequences in terms of cost and model efficacy. Indeed, this may well be a reason why neural networks have sometimes not performed as well as anticipated based upon a single data set even when that data set is a "hold-out" test sample. Therefore, without utilizing multiple data sets, this fact is not apparent and related performance anomalies would go unexplained.

In the next section, performance evaluation is presented from an object classification perspective rather than from the model error. It should be noted that when equal costs of misclassification are assumed, classification takes place by assigning an object to the group that corresponds to the maximum neural network output. For a two-group problem, the decision threshold is at the 0.5 cutoff and posterior probability estimation performance is most critical near this cutoff. However, when unequal costs of misclassification are indicated the cutoff points may vary. Indeed, accurate estimation of object posterior probabilities throughout the [0, 1] interval is critical for generalized decision models that can be utilized for a wide range of organizational objectives.

### 4.3. Classification performance evaluation

In evaluating the classification performance of neural networks, the observed classification rate is the most commonly reported measure. Within the proposed framework, classification performance is expanded to include consideration of the best-expected classification performance, the Bayesian classification rate, because the posterior probabilities are known. Considering the observed rate in relation to the Bayes rate gives a much more accurate picture of the true classification performance of the models employed and is called Bayesian classification efficiency. Fig. 3 presents the classification performance results for each problem at the hidden node/sample size combination yielding the lowest error. Recall from Fig. 1 that when the training sample size was 180 objects, zero hidden nodes was best for both the two-group and three-group problems. With 540 objects in the training set, meanwhile, the two-group problem resulted in two hidden nodes as best while the three-group problem yielded three hidden nodes.

Fig. 3 shows the observed classification rate for the two- and three-group problems is in the 70% to 75% range. Increasing training sample size yields an intuitively expected increase in correct classifications, while the more complex three-group problem achieves a slightly lower classification performance than the two-group problem. Whereas individual decision makers would need to decide if these rates are sufficient for their specific application, it can be seen that when the long-run classification rates to be expected are factored in—the Bayesian classification rate which is 78.375% in the two-group problem and 80.417% in the three-group case—the classification efficiency is much higher. Classification efficiency runs from 92.1% in the small training sample two-group problem to over 95% when the training size is increased. In the three-group case, the classification efficiency is 87.2% in the small training sample example to 90.6% for the larger training sample size. The neural network model performance in relation to problem classification difficulty can be used by model builders to decide which of several courses of action are indicated. This might include deciding if more observations are cost effective, if additional input attributes should be collected to effect problem diffi-

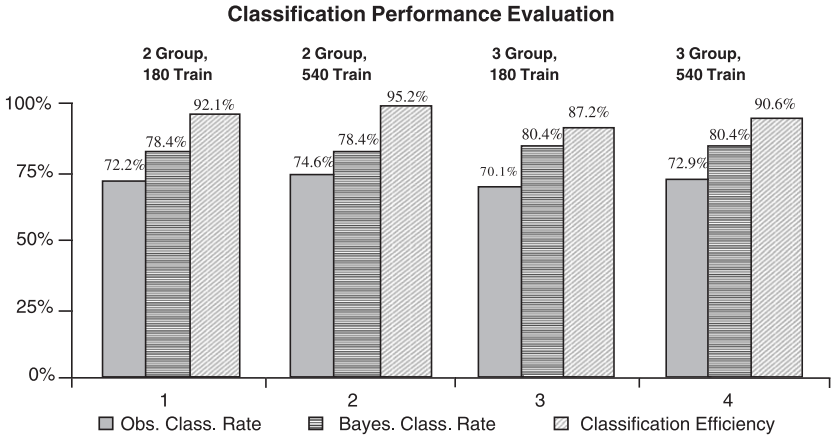



Fig. 3. Classification performance evaluation at the best hidden node.

culty, or if some combination of the two is to be specified.

## 5. Conclusion

The neural network is a promising modeling tool for e-commerce applications. Electronic commerce data is typified by its complexity and high level of noise. Neural networks are ideally suited for these problem characteristics and can be incorporated into intelligent agents and other decision support system components. But as has been detailed in this paper, neural network model building is not an insignificant task.

While neural networks can approximate any function with arbitrary accuracy, obtaining the best model order (hidden nodes) for given data is not a trivial matter and can lead to incorrect decisions when based upon a single realized data set. Multiple data sets are required to make appropriate decisions in the model building process based upon a full understanding of the model error dynamics. The total model estimation error (model bias plus model variance) approach provides a sound conceptual framework for using neural networks for the estimation of posterior probabilities in classification.

In this paper, we have presented a principled approach for building and evaluating neural network models for e-commerce classification settings. The aim is to facilitate practical construction of models with better generalization capability. The approach, which utilizes multiple data sets for building and evaluating neural networks, is then illustrated on both a two-group problem and a three-group problem. Specific guidelines for practitioners are described throughout the paper including a step-by-step summary. Generally, the total model error is used in the model order selection procedure to determine the best number of hidden nodes and the resulting classification performance evaluated.

In this approach, model error is decomposed into model bias and model variance components to more fully understand the underlying error dynamics. This information can then be used by model builders to make necessary tradeoff decisions. Results from this study show that a larger training sample size leads to more complex neural networks and, in turn, yields a reduction in the total model estimation error. This result is consistent with theoretical prescription. More importantly, though, it is seen that decisions based upon a single data set can result in selecting models that yield significant increases in model error, which is not apparent unless multiple data sets are used. For the example problem, decisions based upon a single data set lead to selecting model orders that could more than double the total model error.

We have also proposed the use of Bayesian classification efficiency for the evaluation of neural network classification models. In most studies, only the observed classification rate is reported as a performance metric because the theoretical maximum classification rate, the Bayesian classification rate—which

requires posterior probabilities—is not known. By considering the observed classification rate relative to the Bayesian rate, a new measure called the Bayesian classification efficiency, can be developed. It is seen with this measure that classification performance can be significantly higher than initially thought. This could result in a model being judged as sufficient for use or it could motivate model builders to affect model complexity via additional input attributes or it could be used to justify costs associated with collecting additional data.

## References

[1] E.B. Baum, F. Wilczek, Supervised learning of probability distributions by neural networks, in: D. Anderson (Ed.), Neural Information Processing Systems, American Institute of Physics, New York, 1988, pp. 52–61.
[2] V.L. Berardi, G.P. Zhang, The effect of misclassification costs on neural network classifiers, Decision Sciences 30 (13) (1999) 659–682.
[3] H. Bourlard, C.J. Wellekens, Links between Markov models and multilayer perceptrons, in: D.S. Toretzky (Ed.), Advances in Neural Information Processing Systems, Morgan Kaufmann, San Mateo, CA, 1989, pp. 502–510.
[4] J.R. Bult, T. Wansbeek, Optimal selection for direct mail, Marketing Science 14 (4) (1995) 378–394.
[5] G. Cybenko, Approximation by superpositions of a sigmoidal function, Mathematical Control Signals Systems 2 (1989) 303–314.
[6] R.O. Duda, P. Hart, Pattern Classification and Scene Analysis, Wiley, New York, 1973.
[7] B. Efron, R.J. Tibshirani, An Introduction to the Bootstrap, Chapman & Hall, New York, 1993.
[8] S. Geman, E. Bienenstock, R. Doursat, Neural networks and the bias/variance dilemma, Neural Computation 4 (1992) 1–58.
[9] K. Hornik, Approximation capabilities of multilayer feedforward networks, Neural Networks 4 (1991) 251–257.
[10] K. Hornik, M. Stinchcombe, H. White, Multilayer feedforward networks are universal approximators, Neural Networks 2 (1989) 359–366.
[11] M.S. Hung, M.Y. Hu, B.E. Patuwo, M. Shanker, Estimating posterior probabilities in classification problems with neural networks, International Journal of Computational Intelligence and Organizations 1 (1996) 49–60.
[12] G.G. Karauga, A.M. Khraban, S.K. Nair, D.O. Rice, AdPalette: an algorithm for customizing online advertisements on the fly, Decision Support Systems 32 (2001) 85–106.
[13] J.W. Kim, B.H. Lee, M.J. Shaw, H.-L. Chang, M. Nelson, Application of decision-tree induction techniques to personalized advertisements on Internet storefronts, International Journal of Electronic Commerce 5 (3) (2001) 45–62.
[14] A.M. Law, ExpertFit, Averill M. Law and Associates, http://www.averill-law.com (2002).
[15] A. Papoulis, Probability, Random Variables, and Stochastic Processes, 3rd ed., McGraw-Hill, New York, 1991.
[16] T.S. Raghu, P.K. Kannan, H.R. Rao, A.G. Whinston, Dynamic profiling of consumers for customized offerings over the Internet: a model and analysis, Decision Support Systems 32 (2001) 117–134.
[17] L. Raymond, S. Blili, Organizational learning as a foundation of electronic commerce in the network organization, International Journal of Electronic Commerce 5 (2) (2001) 29–45.
[18] M.D. Richard, R.P. Lippmann, Neural network classifiers estimate Bayesian a-posteriori probabilities, Neural Computation 3 (4) (1991) 461–483.
[19] D.W. Ruck, S.K. Rogers, M. Kabisky, M.E. Oxley, B.W. Suter, The multilayer perceptron as an approximation to a Bayes optimal discriminant function, IEEE Transactions on Neural Networks 2 (1) (1990) 296–298.
[20] D.E. Rumelhart, G.E. Hinton, R.J. Williams, Learning internal representations by error propagation, in: D.E. Rumelhart, J.L. McClelland, and the PDP group (Eds.), Parallel Distributed Processing: Exploration in the Microstructure of Cognition, Foundations, vol. 1, MIT Press, Cambridge, MA, 1986, pp. 318–362.
[21] D.E. Rumelhart, G.E. Hinton, R.J. Williams, Learning representation by backpropagating errors, Nature (London) 323 (1986) 533–536.
[22] D.C. Schmittlein, R.A. Peterson, Customer base analysis: an industrial purchase process application, Marketing Science 13 (1) (1994) 41–67.
[23] P.A. Shoemaker, A note on least-squares learning procedures and classification by neural networks, IEEE Transactions on Neural Networks 2 (1) (1990) 158–160.
[24] E. Turban, Electronic Commerce 2002, Prentice-Hall, Upper Saddle River, NJ, 2002, pp. 154–163.
[25] P.C. Verhoef, B. Donkers, Predicting customer potential value and application in the insurance industry, Decision Support Systems 32 (2001) 189–199.
[26] E.A. Wan, Neural network classification: a Bayesian interpretation, IEEE Transactions on Neural Networks 1 (4) (1990) 303–375.
[27] H. White, Learning in artificial neural networks: a statistical perspective, Neural Computation 1 (1989) 425–464.
[28] H. White, Connectionists nonparametric regression: multilayer feedforward networks can learn arbitrary mappings, Neural Networks 3 (1990) 535–549.
[29] D.J. Wu, Artificial agents for discovering business strategies for network industries, International Journal of Electronic Commerce 5 (1) (2000) 9–36.