A PRINCIPLED APPROACH TO BUILDING AND EVALUATING NEURAL NETWORK MODELS FOR E-BUSINESS APPLICATIONS

Victor L. Berardi, B. Eddy Patuwo
Dept. of Management & Information Systems, Kent State University, Kent, Ohio 44242-0001, USA, Telephone: 1-330-672-1163, Fax: 1-330-672-2448, epatuwo@bsa3.kent.edu

Michael Y. Hu
Dept. of Marketing, Kent State University, Kent, Ohio 44242-0001, USA

ABSTRACT

Every commercial transaction generates large amounts of data on consumers for use by organizations. Data from e-business is typified by its complexity, quantity, and noisiness. Neural networks are ideally suited for these problem characteristics. Furthermore, the fact that neural networks can estimate the posterior probabilities associated with the group membership of objects of interest, makes them a powerful tool of great potential for e-business applications.

As with all classification approaches, though, the neural network's utility is based upon its generalization performance on new data. In this paper, we propose a principled approach to building and evaluation neural network models for e-business applications. First, the usefulness of neural networks for e-commerce applications and Bayesian classification is discussed. Next, the theory concerning model accuracy and generalization is presented. Then the principled approach is described including illustrative examples.

INTRODUCTION

Every commercial transaction generates large amounts of data on consumers 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. 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.

Decision support systems and data mining approaches have been dominant approaches for improving data utilization. Decision support systems have enjoyed years of success in manufacturing and organizational management. Their track record, ease of use, and numerous vendor offerings make them a natural choice in this process. More recently, data mining approaches have gained in popularity as commercial offerings have entered mainstream use. Utilizing sophisticated statistical correlation procedures, organizations can find and exploit relationships behind customer behaviors and characteristics that are seemingly unrelated.

Another approach, though much less commonly utilized for e-business applications, lies in (artificial) neural networks. Neural networks possess many characteristics that make them appealing for e-business applications as discussed next.

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-business applications can generate dozens of variables on individual customers with each transaction, and that each system may contain thousands of customers with transactions from multiple 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 [1][2][3][4][5][6] that neural networks can indeed approximate any function to desired accuracy. Richard and Lippmann [7] and Hung, Hu, Patuwo and Shanker [8], meanwhile, show that neural networks are capable of estimating posterior probability distributions. Given the large amounts of data with complex relationships generated in e-business settings, the usefulness of neural network applications in this area is promising.

Another strength of neural networks for e-business 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 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 to appropriately entice the customer into becoming a more regular purchaser.

The above points show that neural networks are useful in e-business applications and they should make neural networks a valuable, though currently under utilized, 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 to generalize well to unseen future data as well.

The purpose of this paper is to present an approach to building and evaluating neural network models for e-business decision-making. To achieve this goal, issues related to Bayesian classifiers, model estimation error, model bias and model variance are reviewed first. Then a principled approach to building and evaluating neural network models is discussed. Examples will be presented to illustrate the approach.

BAYESIAN CLASSIFICATION AND POSTERIOR PROBABILITIES

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

Posterior probability forms the basis for the well-known Bayesian classification theory [9]. According to Bayes rule, if we obtain an observation **x**, the prior probability of belonging to group *j*, $P(\omega_j)$, will be modified into the posterior probability, $P(\omega_j \mid \mathbf{x})$, that object **x** belongs to group *j* by the following equation:

$$P(\omega_j \mid \mathbf{x}) = \frac{f(\mathbf{x}, \omega_j)}{f(\mathbf{x})}$$

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

Bayes rule shows how observing the value of **x** changes the prior probability $P(\omega_j)$ to the posterior probability $P(\omega_j \mid \mathbf{x})$ upon which the classification decision is based. For example, consider an e-mailed based mass marketing campaign that might generate a two percent response rate (i.e., prior probability $P(\omega_j)$ of response). Upon learning demographic and psycho graphic information on individuals (i.e., object attributes **x**) the probability of response can be modified up or down from $P(\omega_j)$ to $P(\omega_j \mid \mathbf{x})$ to reflect this new information.

Furthermore, suppose that a particular **x** is observed and is to be assigned to a group. Let $\lambda_{ij}(\mathbf{x})$ be the cost of misclassifying **x** to group *i* when it actually belongs to group *j*. Since $P(\omega_j \mid \mathbf{x})$ is the probability that the object belongs to group *j* given **x**, the expected loss associated with assigning **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 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 [10] shows that the mapping function F: $\mathbf{x} \rightarrow \mathbf{y}$, which minimizes the expected squared error is the conditional expectation $E[\mathbf{y} \mid \mathbf{x}]$. Since in a classification problem the output **y** is a vector of binary values, it can be easily shown (see, for example, [8]) that $E[\mathbf{y} \mid \mathbf{x}] = P(\mathrm{w} \mid \mathbf{x})$. Since neural networks can approximate any function F arbitrarily closely (universal approximators), then neural network outputs are indeed good estimators of the posterior probabilities $P(\mathrm{w} \mid \mathbf{x})$.

Many recent papers have provided linkage between neural networks and posterior probabilities [7][8][11][12][13] [14][15] for squared error functions and on the cross-entropy [16] error function . It should be noted most of these articles assume infinite sample sizes. It is Hung et al. [8] and Richard and Lippmann [7] who show that neural networks minimizing squared-error and cross-entropy cost functions are capable 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.

We desire to use neural networks to approximate accurately the Bayesian posterior probabilities in order to make good classification decisions. We see from (1) that computing the posterior probabilities requires specification or estimation of prior probabilities $P(\omega_j)$ and conditional likelihood functions $f(\mathbf{x} \mid \omega_j)$. While $P(\omega_j)$ can be directly estimated from observed data, the strength of a neural network approach is that neural network directly estimates the posterior probability $P(\omega_j \mid \mathbf{x})$ based upon the data presented.

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 model estimation error relative to the posterior probability distribution as 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 ω_j and their attributes $\mathbf{x}$ to forecast the behavior of a new customer, which is based upon $P(\omega_j \mid \mathbf{x})$. 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 benefits as well.

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 training set composition can have significant impact on neural network performance for unseen objects. 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 [17]

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

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 $E[\mathbf{y} \mid \mathbf{x}] = P(\mathrm{w} \mid \mathbf{x})$ 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. Interested readers are directed to Duda and Hart [1973] for detailed coverage of Bayesian classification theory.

Therefore, (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 (2) into components to measure the average performance of a model and performance variation resulting from different data sets gives

$$E_D\left[(f(\mathbf{x};D) - E[\mathbf{y} \mid \mathbf{x}])^2\right] = \qquad (3)$$
$$(E_D[f(\mathbf{x};D)] - E[\mathbf{y} \mid \mathbf{x}])^2 + E_D[(f(\mathbf{x};D) - E_D[f(\mathbf{x};D)])^2]$$
"Model Bias" + "Model Variance".

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 $E[\mathbf{y} \mid \mathbf{x}]$. In other words, model bias directly considers the neural network's ability to learn the underlying generating process. 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. 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. [17] 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. 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-business environments can be utilized in a principled approach to building neural networks for specific application and in more fully predicting the performance level expected in practice.

A PRINCIPLED APPROACH

Determining the model bias and the model variance in (3) requires estimating $E[\mathbf{y} \mid \mathbf{x}] = P(\mathrm{w} \mid \mathbf{x})$, the posterior probabilities. The existence of substantive amount of data typically available in e-business application presents an opportunity to construct the group likelihood function $f(\mathbf{x} \mid \omega_j)$ and to compute the posterior probabilities directly from (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 ω_j, such as customer segments, that are contained in an e-commerce database. Commercially available distribution fitting procedures, such as ExpertFit [18] 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, we first break up $\mathbf{x}$ into its independent components $\mathbf{x} = [\mathbf{x}^1, \mathbf{x}^2, \ldots, \mathbf{x}^k]$. After determining the joint densities of the individual components, we can compute the likelihood function from $f(\mathbf{x}|\omega_j) = f(\mathbf{x}^1|\omega_j) f(\mathbf{x}^2|\omega_j) \ldots f(\mathbf{x}^k|\omega_j)$. 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. This also allows us to generate multiple data sets of the same size as the original set. Otherwise, by splitting up the original data, one is only able to provide conservative estimates of expected model errors.

The ability to determine the object level posterior probabilities represents an outstanding potential use for neural networks in e-business applications. In this context, decision makers can estimate not only how well a neural network model will perform with given data but also in the context of long-term, continuous use. Model performance can be thoroughly investigated while minimizing the common problem of overestimating the model's true utility. To realize this advantage, though, multiple data sets are necessary to estimate total model error, model bias, and model variance and will be used within 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. [1992] and is described now. Recall from equation (3) that

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

and that the variance is

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

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 (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

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

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

$$\text{Model Bias}\ (\mathbf{x}) \approx \left(\bar{f}(\mathbf{x}) - E[\mathbf{y}|\mathbf{x}]\right)^2 \qquad (7)$$

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

Total Estimated Model Error ($\mathbf{x}$) =

Model Bias ($\mathbf{x}$) + Model Variance ($\mathbf{x}$) (9)

An approach equivalent to the hold-out method of train and test 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 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 (6) to (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 object 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 percent. If the problem has an expected Bayesian classification rate of 66 percent, it is seen that the Bayesian classification efficiency is nearly 91 percent—a much better model classification performance than first appears.

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

1. Compute the posterior probabilities directly from (1) as described earlier.
2. Divide the data set into S training sets and one large test set. Or generate a data set of the desired size from the fitted distributions, and divide it into S training sets. Then use the original data set as the test set.
3. Train the neural networks on the S training sets to get neural network estimators $f(\mathbf{x}; D^1)$, $f(\mathbf{x}; D^2), \ldots, f(\mathbf{x}; D^S)$.
4. Using the test set, compute model bias, model variance and total estimated model error as in (6) to (9).
5. Decide on the best neural network structure based on the error measures in step 4.
6. Evaluate the performance of the neural network model using the Bayesian Classification Efficiency.

ILLUSTRATIVE EXAMPLES

For illustration purposes, in this paper we use simulated data sets generated with eight input variables which 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 $\{0, 1, 2, \ldots, 10\}$.

These random variables are chosen to represent those likely to be encountered in e-business 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.

The illustrative problems contain two- and three-group settings. The two-group example represents a special case problem from a neural network and classification standpoint and may be appropriate for an e-business situation 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.

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-business 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.

A model order selection procedure commonly used in neural network applications is used. 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.

RESULTS

The performance of the neural network models in estimating posterior probabilities and classification performance will be analyzed in the following manner. First, 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 investigated. 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 from using 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.

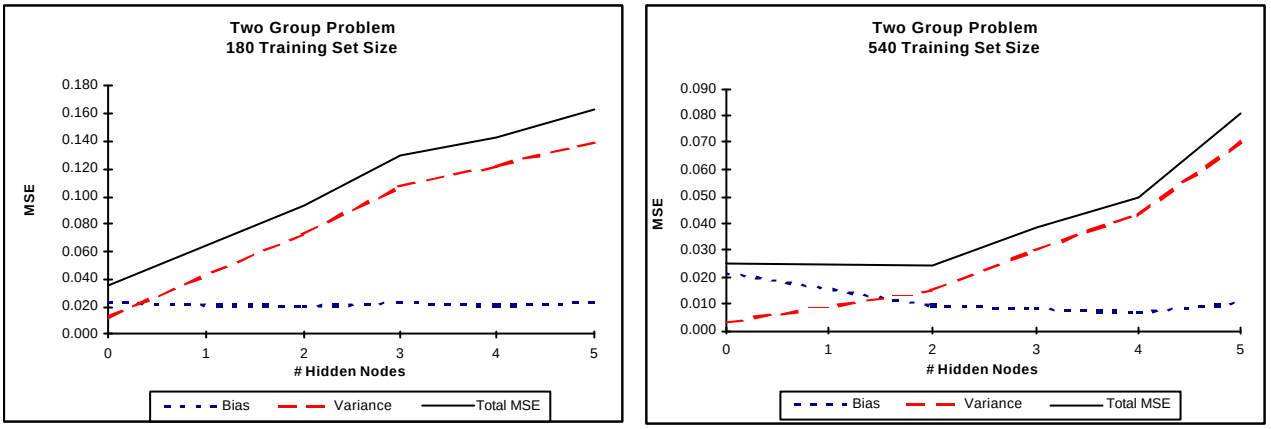
Model Estimation Error

Figure 1 contains the total estimation error, the model bias, and model variance performance of the neural network models across the hidden nodes. The top panel contains results of the two-group problem, while the bottom panel presents those of the three-group problem. The scale has been set to facilitate visualizing the error components.

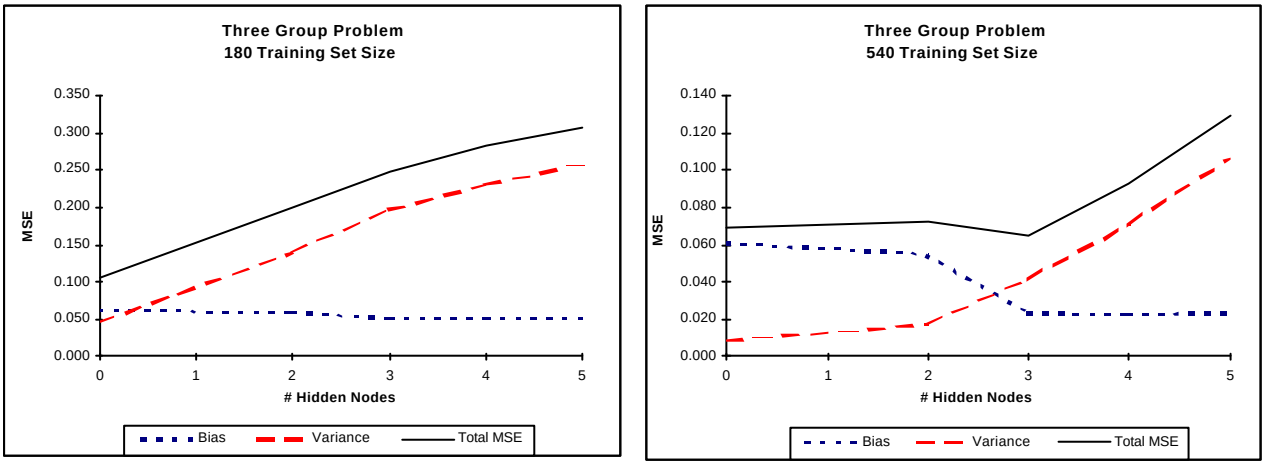
In Panel A for the two-group problem with a training sample size of 180 objects, total estimation error rises steadily as the hidden nodes increase. It is seen that this increase in estimation error results almost entirely from the variance component as the bias remains essentially unchanged. For a smaller sample size, it is seen that small 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 merely leads to large increases in model variance.

FIGURE 1

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



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.

As the training set size is increased to 540 objects, 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 as the number of hidden nodes increases from zero to two, then they exhibit a rise in error as hidden nodes increase from three through five. From zero to two hidden nodes, it is seen that decreases in model bias are cancelled by increases in the variance component. Beyond two hidden nodes, the variance component dominates any changes in bias, resulting in overall estimation error increases. It should also be noted that the impact of changing the model order in the 540 training sample size case is approximately one-half of what is seen in the 180 training sample size case. Furthermore, the minimum total error in the 180 training set size case at zero hidden nodes is nearly 45 percent greater than that achieved with 540 objects in training at two hidden nodes.

Given this available information, model builders could trade off costs associated with obtaining more data for training against the posterior probability estimation performance gains 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 percent) 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. It is interesting to note, as more information on the generating process is presented via the larger training sample size, higher-order models are 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.

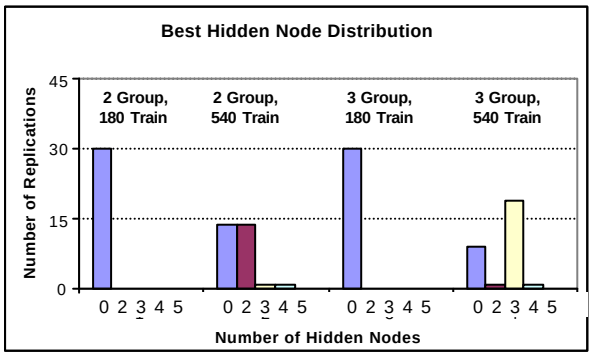
Hidden Node Distribution

The impact of the error dynamics discussed above becomes even more important for practitioners when one considers it in the context of the commonly applied model order selection procedure. 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 model would then be used in practice.

Figure 2 contains a count of the number of data sets that yielded each hidden node as "best" for each problem type. 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.

FIGURE 2

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



If, for the two-group problem with 540 training sample size, we happen to train the neural network using the one data set which yields four hidden nodes instead of two as the best, an increase of about 100 percent can be expected in the total model estimation error (see Figure 1). The implications for practical applications are significant when one considers the complexity of the underlying generating process the neural networks are approximating in e-business situations and the large number of training data likely to be employed. It would not be surprising to find large variations in the number of hidden nodes being chosen merely because of data set variations and that this could have significant practical performance consequences in terms of cost and efficacy. Without utilizing multiple data sets this would not be apparent and related performance anomalies, therefore, would go unexplained.

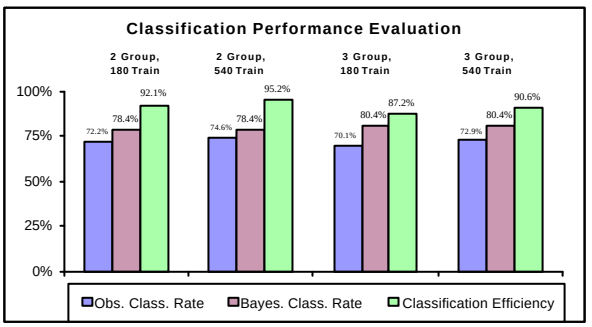
Classification Performance Evaluation

In evaluating the classification performance of neural network classifiers, 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, as object 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. Figure 3 presents the classification performance results for the example problem.

The observed classification rate for the two- and three-group problems is in the 70 percent to 75 percent 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. While 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 —which is Bayesian classification rate and is 78.375 percent in the two-group problem and 80.417 percent in the three-group case—the classification efficiency is seen to be much higher. Efficiency runs from 92.1 percent in the small training sample two-group problem to over 95 percent when the training size is increased. In the three-group case, the classification efficiency is 87.2 percent in the small training sample example to 90.6 percent for the larger training sample size. The neural network model performance in relation to problem classification difficulty can be used to decide if more observations are cost effective, if additional input attributes should be collected to effect problem difficulty, or some combination of the two is indicated.

FIGURE 3

Classification performance evaluation.



CONCLUSION

The neural network is a promising modeling tool for e-business applications. Data from e-business is typified by its complexity, quantity, and noisiness. Neural networks are ideally suited for these problem characteristics. It has been pointed out in this paper that neural network modeling is not a trivial task, though. 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 to building and evaluating neural network models for e-

business classification settings. The aim is to facilitate practical construction of models with better generalization capability. The approach is then illustrated with simulated data sets for a two-group problem and a three-group problem. The total model estimation error is used in the model order selection to determine the number of hidden nodes.

Results from this study show that a larger training sample size will inevitably lead to more complex neural networks and in turn yield a reduction in the total model estimation error. We have also proposed the use of Bayesian classification efficiency for the evaluations of neural network classification models. Based on the output measures, our proposed procedure for building neural network models seems to be conceptually sound and is an integrated approach with definite promising benefits.

REFERENCES

[1] Cybenko, G. Approximation by superpositions of a sigmoidal function. *Mathematical Control Signals Systems*, 1989, 2, pp. 303-314.

[2] Hornik, K. Approximation capabilities of multilayer feedforward networks. *Neural Networks*, 1991, 4, pp. 251-257.

[3] Hornik, K., Stinchcombe, M., and H. White.. Multilayer feedforward networks are universal approximators. *Neural Networks*, 1989, 2, pp. 359-366.

[4] Rumelhart, D.E., G.E. Hinton, and R.J. Williams. Learning internal representations by error propagation. *In Parallel Distributed Processing: Exploration in the Microstructure of Cognition. Vol. 1: Foundations*, D.E. Rumelhart, J.L. McClelland, and the PDP group, eds., pp. 318-362. MIT Press, Cambridge, MA, 1986.

[5] Rumelhart, D.E., G.E. Hinton, and R.J. Williams. Learning representation by backpropagating errors. *Nature (London)*, 1986, 323, pp. 533-536.

[6] White, H. Connectionists nonparametric regression: multilayer feedforward networks can learn arbitrary mappings. *Neural Networks*, 1990, 3, pp. 535-549.

[7] Richard, M.D. and R.P. Lippmann. Neural network classifiers estimate Bayesian a-posteriori probabilities. *Neural Computation*, 1991, 3 (4), pp. 461-483.

[8] Hung, M.S., M.Y. Hu, B.E. Patuwo, and M. Shanker. Estimating posterior probabilities in classification problems with neural networks. *International Journal of Computational Intelligence and Organizations*, 1996, 1, pp. 49-60.

[9] Duda, R.O., and P. Hart. *Pattern Classification and Scene Analysis*. New York: Wiley, 1973.

[10] Papoulis, A. *Probability, Random Variables, and Stochastic Processes*, 3rd Edition, pp. 175. McGraw-Hill, 1991.

[11] Bourlard, H., and C.J. Wellekens. Links between Markov models and multilayer perceptrons. In D.S. Toretzky (Ed.) *Advances in Neural Information Processing Systems*, Volume 1, pp. 502-510. San Mateo, CA: Morgan Kaufmann, 1989.

[12] Ruck, D.W., S.K. Rogers, M. Kabisky, M.E. Oxley,, and B.W. Suter. The multilayer perceptron as an approximation to a Bayes optimal discriminant function. *IEEE Transactions on Neural Networks*, 1990, 2 (1), pp. 296-298.

[13] Shoemaker, P.A. A note on least-squares learning procedures and classification by neural networks. *IEEE Transactions on Neural Networks*, 1990, 2 (1), pp. 158-160.

[14] Wan, E.A. Neural network classification: A Bayesian interpretation. *IEEE Transactions on Neural Networks*, 1990, 1 (4), pp. 303-375.

[15] White, H. Learning in artificial neural networks: A statistical perspective. *Neural Computation*, 1, pp. 425-464.

[16] Baum, E.B. and F. Wilczek. Supervised learning of probability distributions by neural networks. In D. Anderson (Ed.) *Neural Information Processing Systems* pp. 52-61. American Institute of Physics, New York, 1988.

[17] Geman, S., E. Bienenstock, and R. Doursat. Neural Networks and the Bias/Variance Dilemma. *Neural Computation*, 1992, 4, pp. 1-58.

[18] Law, A.M. *ExpertFit*, Averill M. Law & Associates, *www.averill-law.com*, 2001.