Estimating Probabilities of Diabetes Mellitus Using Neural Networks [1]

Murali Shanker [2]
College of Business
Kent State University
Kent, OH 44242-0001
USA
Phone: +1 (330) 672-2750 Ext. 347
Fax: +1 (330) 672-2448
E-mail: *mshanker@kent.edu*

M.Y. Hu
Kent State University and Chinese University of Hong Kong

M. S. Hung
Optimal Solutions Technologies, Inc.

November 13, 1999

[1]Estimating Diabetes Probability using ANN
[2]Author for correspondance

Abstract

Classification problems are often encountered in medical diagnosis. This paper presents an introduction to classification theory and shows how artificial neural networks can be used for classification. We also map out a bootstrapped procedure for interval estimation of posterior probabilities. The entire procedure is illustrated using the diabetes mellitus data in Pima Indians.

Keywords: Neural networks, classification, posterior probability, medical diagnosis

INTRODUCTION

Statistical classification involves the assignment of an object to an appropriate group based on a number of variables (also called attributes) describing that object. Recent developments [1][2] [3], to name just a few, show that with neural networks, not only can we assign an object to a group, but we can also estimate the probability that the object belongs to that group. This probability, called the *posterior probability*, is much more useful to the decision maker than the classification itself. Clearly, medical diagnosis is a natural application of statistical classification and the primary purpose of this paper is to demonstrate the methodology for the estimation of posterior probability for medical diagnosis.

The organization of this paper is as follows. In the next section, basic concepts of statistical classification and least squares estimation are reviewed, so as to justify the use of neural networks for the estimation of the posterior probability. Then artificial neural networks are introduced. As there are many aspects of neural networks existing in the literature, only those directly related to our work are presented. Special attention is given to how to build the appropriate neural network model. Issues such as feature selection and network architecture selection are discussed. To illustrate the methodology of building neural networks for medical diagnosis, an actual data set obtained from a diabetes study of Pima Indians is used [4]. The process of building neural networks for the diagnosis of diabetes is described next. It turns out that the "optimal" networks are fairly simple. The results are then analyzed in the following section.

LEAST SQUARES ESTIMATION OF THE POSTERIOR PROBABILITY

In this section, a very brief review of classification theory is presented in order to define the necessary terminology and notation. Then the least squares method for the estimation of the posterior probability is developed.

Let each object be associated with a d-vector x of attributes. Assume that the sample space X consists of m groups of the objects. Following Duda and Hart [1973], let ω_j denote the fact that an object is a member of group j. Define

$P(\omega_j)$ = *prior* probability of group j, the probability that a randomly selected object belongs to group j;

$f(x \mid \omega_j)$ = conditional probability density function for x being a member of group j.

The posterior probability $P(\omega_j \mid x)$, which is the probability that object x belongs to group j, is obtained using the *Bayes* rule:

$$P(\omega_j \mid x) = \frac{f(x,\omega_j)}{f(x)} = \frac{f(x \mid \omega_j)P(\omega_j)}{\sum_{j=1}^{m} f(x,\omega_j)} \tag{1}$$

For medical diagnosis, one group may be positive (e.g., diabetic) whereas the other group may be negative (e.g., not diabetic). Then the posterior probability of the first group will be the probability that the case is positive.

Suppose a particular x is observed and is to be assigned to a group. Let $c_{ij}(x)$ be the cost of assigning x to group i when it actually belongs to group j. The expected cost of assigning x to group i is

$$C_i(x) = \sum_{j=1}^{m} c_{ij}(x) P(\omega_j \mid x). \tag{2}$$

To minimize the expected cost, the optimal assignment of x should follow the *Bayesian decision rule* shown below.

$$\text{Decide } \omega_k \text{ for } x \text{ if } C_k(x) = \min_{\mathbf{i}} C_i(x). \tag{3}$$

A particular case for the rule is when the cost is binary: $c_{ij}(x) = 0$ if $i = j$, and 1 otherwise. The cost function $C_i(x)$ can be simplified to

$$C_i(x) = \sum_{i \neq j} P(\omega_j \mid x) = 1 - P(\omega_i \mid x), \tag{4}$$

and the Bayesian decision rule is reduced to

$$\text{Decide } \omega_k \text{ for } x \text{ if } P(\omega_k \mid x) = \max_{\mathbf{i}} P(\omega_i \mid x). \tag{5}$$

It is clear that "optimal" classification should be made based on the posterior probability. Unfortunately, it is usually a nonlinear function of the attributes x and the functional form can be very complex. So far as we know, neural networks are the only practical method for approximating the posterior probabilities directly.

The theoretical basis for the neural network estimation is the method of least squares. Continuing the notation from the previous development, but recasting the problem in a more general form, let $x \in \Re^d$ and $y \in \Re^m$ be vectors of random variables and $f_{x,y}(x, y)$ be their joint distribution. The objective of the least squares method is to determine a mapping $F : \Re^d \rightarrow \Re^m$ so as to

$$Minimize\ Q = E\left[\sum_{i=1}^{m} (y_i - F_i(x))^2\right], \tag{6}$$

where the expectation E is defined over the joint density function, and y_i and $F_i(x)$ are the i^{th} component of y and $F(x)$, respectively. By definition,

$$Q = \int_{x \in X} \int_y \left[\sum_{i=1}^{m} (y_i - F_i(x))^2\right] f_{x,y}(x, y) dy dx$$

For classification, x is an object and y is the *target membership value* defined as

$$y_i = 1 \text{ if } x \text{ belongs to group } i, \text{ 0 otherwise.}$$

For the following development, we will assume there are only two groups so that the results can be used directly for our problem of determining whether a patient is diabetic or not. For a more general derivation, please see [1]. For a two-group classification problem, we can reduce the membership variable to only one and let $y = 1$ if object x belongs to group 1 and $y = 0$ otherwise. As a result, there is only one component of the estimated function and it will be denoted by $F(x)$. Then

$$Q = \int_{x \in X} \left[(1 - F(x))^2 f(x, \omega_1) + (0 - F(x))^2 f(x, \omega_2) \right] dx$$

where $f(x, \omega_1)$ and $f(x, \omega_2)$ are the joint density functions for groups 1 and 2, respectively. Since $f(x, \omega_j) = P(\omega_j \mid x) f(x)$ and $P(\omega_2 \mid x) = 1 - P(\omega_1 \mid x)$, the expected sum of squares can be written as

$$\begin{aligned} Q &= \int_{x \in X} \left[(1 - F(x))^2 P(\omega_1 \mid x) f(x) + F(x)^2 \left(1 - P(\omega_1 \mid x)\right) f(x) \right] dx \\ &= \int_{x \in X} \left[F(x)^2 - 2F(x) P(\omega_1 \mid x) + P(\omega_1 \mid x) \right] f(x) dx \\ &= \int_{x \in X} \left[(F(x) - P(\omega_1 \mid x))^2 + \left(1 - P(\omega_1 \mid x)\right) P(\omega_1 \mid x) \right] f(x) dx. \end{aligned}$$

Let

$$\sigma_A^2 = \int_{x \in X} P(\omega_1 \mid x)(1 - P(\omega_1 \mid x)) f(x) dx \tag{7}$$

and

$$\sigma_\varepsilon^2 = \int_{x \in X} (F(x) - P(\omega_1 \mid x))^2 f(x) dx \tag{8}$$

and therefore,

$$Q = \sigma_A^2 + \sigma_\varepsilon^2 \tag{9}$$

The quantity σ_A^2 is termed the *approximation error* and σ_ε^2 the *estimation error* by Barron [1989] .

It is clear that Q is minimized when σ_ε^2 is zero and that is achieved when

$$F(x) = P(\omega_1 \mid x). \tag{10}$$

Hence, the least squares solution is the posterior probability.

NEURAL NETWORKS

Artificial neural networks have been used to solve a wide variety of problems including classification. A neural network is composed of nodes and arcs connecting a node to another. Nodes are artificial representations of *neurons* and arcs are those of *synapses* in a biological brain. The *feedforward* neural networks, the kind considered here, are networks without closed feedback loops. The node set

is typically partitioned into three subsets: input nodes, output nodes, and hidden nodes. A *multi-layer perceptron* is a feedforward network where hidden nodes are arranged in layers and only nodes between nodes of neighboring layers are connected.

Each arc in a neural network is associated with a weight. Let (i, j) denote the arc going from node i to node j and w_{ij} be its weight. A hidden or output node may be assigned a scalar called *bias*, which is similar to the intercept in a linear regression function. For node j, let w_{0j} denote the bias and S_j be the set of nodes (including node 0 if the node has a bias) connected into it. The *input* received at node j is defined as

$$x_j = \sum_{i \in S_j} w_{ij} a_i,$$

where a_i is the output of node i and is defined as

$$a_i = F_i(x_i).$$

The function F_i is called the *activation function* and is usually one of two forms: logistic or linear. The logistic function is defined as $F_i(x) = (1 + e^{-x})^{-1}$ and the linear function is $F_i(x) = x$. (Some authors add an intercept and a slope to the linear function.) In almost all applications, including this one, the input nodes use linear activation functions. The activation value of node 0 is fixed at $a_0 = 1$.

One can view a neural network as a mapping function $F : \Re^d \to \Re^m$ when a d-dimensional input x is submitted to the network and an m-dimensional output a is obtained after using the above definitions to transform inputs to outputs. The attractiveness of neural networks is its simplicity. All one needs to do for a feedforward network is to define the *network architecture* – here it means how the nodes are connected, what nodes have biases, what activation function is used in each node.

The weights of a neural network are determined by *training* the network. In neural network literature, training is based on a *learning law* which prescribes necessary algorithms to carry out the computation. For feedforward networks, training is a mathematical minimization problem, usually with the following objective function:

$$\text{Minimize} \sum_{l=1}^{L} \sum_{i \in N_O} (y_i^l - a_i^l)^2$$

where L is the number of *patterns* (sample size) and y_i^l is the *target value* for pattern l and output node i. For classification problems, the target values can be defined as

$$y_i^l = \begin{cases} 1 & \text{if pattern } l \text{ belongs to group } i \\ 0 & \text{otherwise} \end{cases}$$

So neural networks are one of the least squares methods. The remaining issue is whether neural networks can provide an accurate estimate. Cybenko [1989] and Funahashi [1989] , among others, have shown that a neural network, with sufficient number of hidden nodes, can approximate any function

arbitrarily closely. In practice, the number of hidden nodes required for a good approximation can be quite small.

For two-group classification problems, one output node is sufficient. So the objective function is simplified to

$$\text{Minimize} \sum_{l=1}^{L} (y^l - a^l)^2 \tag{11}$$

where $y^l = 1$ if pattern l belongs to group 1 and $y^l = 0$ otherwise, and a^l denotes the network output for pattern l. The fact that we are minimizing the sum here instead of the mean does not affect the solution. Therefore, we expect the network output to be an unbiased estimator of the posterior probability.

Finally, since the objective function is not necessarily convex with respect to the weights w, none of the nonlinear optimization algorithms can guarantee to yield the global minimizing solution. The most popular algorithm, the back-propagation method [9], suffers from lack of clear stopping criteria and long running time. The algorithm used here was developed by Ahn [10] and is called *forward additive algorithm*. It uses a strategy where the starting solution is determined by the data set. When there are no hidden nodes and the activation functions at the output node are linear (i.e., $a(x) = x$), the network becomes a multiple linear regression model and the global solution can be found easily, similar to solving a system of linear equations defined by the data set. The next step is to change the activation functions at the output nodes to logistic function (or any nonlinear function), using the linear model solution as the starting solution for this logistic regression model. Minimization is achieved by a convergent algorithm called *limited memory quasi-Newton* developed by Nocedal [11] and adapted for neural network training by Denton [12]. The next step is to add one hidden node at a time, along with the arcs from the input nodes to the new hidden node and the arcs from the hidden node to the output nodes. The solution of the previous network is retained as the starting value for the same parameters in the new network and new parameters are assigned starting values based on the distribution of the input features. (There are four different methods in Ahn's algorithm for assigning starting values of the new parameters. We used only the most basic method. See Ahn [10] for the details of all the methods.)

Building Neural Network Models

There are many questions to be answered when building a neural network model: the number of input and output nodes, the number of hidden layers and the nodes in each layer, the arcs (which nodes are to be connected), the bias terms (which nodes are to have them), and the activation function at each node. There are few concrete guidelines for any of the decisions. For this study, we concentrate on networks of one hidden layer and one output node. There are arcs from every input node to every hidden node and from every input node to the output node. The activation function for the input

nodes is linear, whereas it is logistic for every other node. Only the output node has bias. So the remaining key questions are the input nodes and the number of hidden nodes.

The issue in input nodes is not merely about its number and about what they are. We would like to use the fewest features in a model, but without sacrificing the model's power for prediction or explanation (known as Occam's Razor). Feature selection is a difficult issue, and the difficulty is compounded by the nonlinear nature of neural networks. Nonlinearity renders the typical hypothesis testing on the significance of feature variables – such as in linear regression – impossible. In several studies [13][14][15], we have used a backward elimination method and shown it to be successful in determining the right mix of feature variables. The method is to train a neural network with the full set of features. The network is evaluated on a holdout set, called the *validation set.* Let the SSE of the validation set be denoted as SSE_F. Next, one feature variable is deleted and a new neural network is trained. Let its SSE on the same validation set be denoted as SSE_R. Then the quantity $SSE_R - SSE_F$ represents the contribution of the feature variable. The larger the difference $SSE_R - SSE_F$, the greater the contribution. (One would expect the difference to be positive on the training set, if the training algorithm yields global or near global minimum, but it can be either positive or negative on the holdout set.) After this quantity is calculated for each feature in turn, the variable with the smallest contribution is the logical candidate for removal and it is removed if some threshold value is crossed. After a feature is removed, the remaining set can now be considered the full set. The above process is then repeated.

In a previous study of the same diabetes data set [15], the measure of contribution is based on the well-known F ratio of regression theory; namely,

$$F = \left(\frac{\mathrm{SSE}_R - \mathrm{SSE}_F}{df_R - df_F}\right) \Big/ \left(\frac{\mathrm{SSE}_F}{df_F}\right)$$

where *df* stands for degrees of freedom and is equal to the sample size (number of cases in the validation set) minus the number of parameters in the neural network (number of arcs plus the biases). In that study, the threshold for removal is 3.0; therefore a feature whose F ratio is less than 3.0 is removed.

The selection of the hidden nodes is difficult because it involves a trade-off. Typically, a network with larger number of hidden nodes yields greater precision for function approximation (smaller SSE). But the larger network may overfit the data and thus not generalize as well. Generalizability refers to the ability of a model to predict unseen objects. Our approach also relies on the use of the validation set. After the features have been selected, they are used to train neural networks of increasing hidden nodes. The architecture with the smallest SSE on the validation set is then selected.

The reliability of a model can be improved with more samples. For example, instead of relying on one training set and one validation set for the selection of feature variables and network architecture, the decisions would be more robust if more samples could be used. One approach to create more samples from a data set is called *bootstrap* [16]. Given a sample of size n, bootstrap generates a new sample of the same size by sampling with replacement. In such a way, as many replicated samples as desired can be created. Several advantages of boostrapping will be discussed later.

DATA

Pima Indian Diabetes

To illustrate the application of neural networks to the estimation of posterior probability in medical diagnosis, the diabetes data set of Pima Indian females is used [4][17]. This population, near Phoenix, Arizona, has been under continuous study since 1965 by the National Institute of Diabetes and Digestive and Kidney Diseases because of its high incidence rate of diabetes [18][19]. Each community resident over 5 years of age was asked to undergo a standardized examination every two years, which included an oral glucose tolerance test. Diabetes was diagnosed according to the World Health Organization criteria [20]; that is, if the 2 hour post-load plasma glucose was at least 200 mg/dl (11.1 mmol/l) at any survey examination, or if the Indian Health Service Hospital serving the community found a glucose concentration of at least 200 mg/dl during the course of routine medical care [18]. This database provides a well validated data resource for exploring the prediction of the date of onset of diabetes [4][17].

The data set consists of 768 subjects who are females over 21 years old. Of these, 268 were diagnosed with diabetes.

Feature Variables

Eight variables were considered to be significant risk factors for diabetes among Pima Indians and other populations [4]. These are:

- Number of times pregnant (PREGNANT)
- Plasma glucose concentration at 2 hours in an oral glucose tolerance test (GTT)
- Diastolic blood pressure (mm Hg) (BP)
- Triceps skin fold thickness (mm) (SKIN)
- 2-hour serum insulin (μU/ml) (INSULIN)
- Body mass index (weight in Kg/(height in m)^2) (BMI)
- Diabetes pedigree function (DPF)
- AGE (years)

The Diabetes Pedigree Function (DPF) was developed by Smith, *et al.* [4] to provide a synthesis of the diabetes mellitus history in relatives and the genetic relationship of those relatives to the subject. The DPF uses information from parents, grandparents, siblings, aunts and uncles, and first cousins. It provides a measure of the expected genetic influence of affected and unaffected relatives on the subject's eventual diabetes risk.

In a previous study, Shanker (1996) used the feature selection method discussed earlier to show that three variables – GTT, BMI and AGE – could provide as much classification rate as the eight variables. Hence these same three variables are used here.

NEURAL NETWORK MODELS FOR DIABETES DIAGNOSIS

The target value of each subject is coded 1 if diabetes is diagnosed, and 0 otherwise. Then the data set is divided into three subsets: training, validation and test sets. The training set is used to train neural networks and the validation set is used to evaluate the quality of each trained network. The test set will be used later to evaluate the predictive power of the chosen models. The division of the data set is shown in Table I.

(Insert Table I here)

As mentioned before, all the neural networks have three input nodes, one output node, and one layer of hidden nodes. The activation function for the hidden and the output node is logistic. Only the output node has a bias. To determine the appropriate number of hidden nodes, 60 bootstrap replicates of the training set and 10 bootstrap replicates of the validation set were generated. So for each network architecture, we have 60 trained networks, each of which is evaluated by 10 validation sets. The average of these $60 \times 10 = 600$ SSE's for each network architecture is plotted in Figure 1. The average SSE is 23.17 for networks of 0 hidden nodes, and it goes up to 23.69 for those with 1 hidden node, and it goes up higher with more hidden nodes. The results clearly suggest the appropriate architecture is with 0 hidden nodes. So that is the one selected for further study.

(Insert Figure 1 here)

ANALYSIS OF RESULTS

So far, one appropriate model for the diabetes data set is a neural network with three inputs and zero hidden nodes. The next steps involve the evaluation of the models on the thus far unseen test set. First, a large training sample was created by combining the training and validation sets. A network was trained with this set and used to estimate the posterior probability of each case in the test set. Based on an arbitrary classification scheme: a case is declared diabetic if the estimated posterior probability is greater than or equal to .7, it is declared non-diabetic if the probability is less than or equal to .3, and it is declared indeterminate otherwise, the results are summarized in Table II.

(Insert Table II here)

The table shows that 65 of the nondiabetics were correctly classified, whereas 5 were incorrectly classified as diabetic. Out of the total of 148 cases, 83 (56.081%) were correctly classified, and 22 (14.865%) were incorrectly classified. with the remaining 43 cases indeterminate. The previous study [15] reported a 80.21% correct classification rate (based on a different test set) with both neural networks and logistic regression. The classification rate is higher than what we have here. But since it did not allow for indeterminate diagnosis, the result there implies a 19.79% incorrect classification, which is higher than our figure here. Since indeterminate cases can be referred back for further tests or examinations, we feel the more important statistic should be the incorrect classification rate. Based on this, the new model shows improvement in the neural network's ability to make diagnosis.

The next step is to ascertain the reliability of the posterior probability estimates. We combined the original training and validation sets, and thirty bootstrap replicates were generated from this combined set. Each bootstrapped set resulted in a trained neural network, which provided an estimated probability for each case in the test set. Figure 2 shows the mean posterior probability from the bootstrapped networks and the probability estimated from the original combined training set.

The graph shows a close match between the two estimates. Not surprisingly, the classification table based on the bootstrap mean posterior probability is also very close (Table III). The only difference is more correct classification and one less indeterminate for the nondiabetic cases.

(Insert Figure 2 here)
(Insert Table III here)

For this example, it seems that bootstrap did not offer much more useful results than the traditional non-bootstrapped methods. There are a few important advantages of the bootstrap method, however [21]. First, it provides useful information regarding the robustness of the estimates . Figure 3 shows the mean and the standard deviation of test set SSE from each bootstrap replicated network. The small range in both statistics assure us that the network estimates are reliable; in other words, there is good stability.

(Insert Figure 3 here)

The second advantage of the bootstrap method is that it offers a direct way to estimate the variance of the estimation error σ_ε^2. Recall its definition,

$$\sigma_\varepsilon^2 = \int_{x \in X} (F(x) - P(\omega_1 \mid x))^2 f(x) dx$$

A sample estimate can be defined as

$$s_\varepsilon^2 = \frac{\sum_{i=1}^{n} \sum_{j=1}^{r} \left(\theta_{ij} - \overline{\theta}_i\right)^2}{(n-1)(r-1)} \tag{12}$$

where θ_{ij} is the posterior probability for case i estimated by bootstrap sample j, $\overline{\theta}_i$ is the mean estimate for case i, and n and r are test set sample size and number of bootstrap replications, respectively. Of course, we can estimate the approximation error σ_A^2 also, with the following formula:

$$s_A^2 = \frac{\sum_{i=1}^{n} \left(1 - \overline{\theta}_i\right) \overline{\theta}_i}{(n-1)} \tag{13}$$

Without bootstraps, there wouldn't be the mean probability $\overline{\theta}_i$, neither would there be estimates of variances. For the test set, we obtained $s_\varepsilon^2 = 0.0013$ and $s_A^2 = 0.1603$.

An important application of s_ε^2 is to offer interval estimations. As there is no reason to assume the posterior probabilities to have any known distributions, no interval estimate can be assigned an exact confidence level. However, based on the bootstrap samples, approximate confidence levels are possible. For each case in our test set, we counted the number of estimated posterior probabilities to fall within a multiple of s_ε around the mean. The following are the "coverages" for this test set.

	Percentage of points within		
	$\pm 1 s_\varepsilon$	$\pm 2 s_\varepsilon$	$\pm 3 s_\varepsilon$
Actual coverage	72.65%	94.71%	99.39%
Normal distribution	68.26%	95.44%	99.72%

The table shows that 72.65% of all bootstrap estimates are within one standard deviation of the mean, as compared to 68.26% in the normal distribution. The coverages are lower in the actual data than the normal function for larger multiples; suggesting that the distribution of points is more clustered towards the middle. But the results are still quite useful. Since $s_\varepsilon = .036$, we can state that 99.39% of the estimates are within 0.11 of the mean. For example, if the mean is .8, we are over 99% certain that the individual estimates will be between .91 and .69.

CONCLUSIONS

Classification problems are routinely applied to medical diagnosis. This paper provides an introduction to the theory of classification and highlights the importance of posterior probabilities in classification. Posterior probabilities correspond to the likelihood of an object belonging to a group. Thus, these probabilities contain more useful information for a medical practitioner than the classification itself.

Artificial neural networks have been successfully applied to classification problems. This paper shows that not only can neural networks be used to estimate posterior probabilities, as with other traditional statistical procedures such as discriminant analysis and logistic regression, but that neural networks estimates are direct and distribution-free. Yet, the primary drawback in using these neural estimates has been the absence of a reliability measure. The bootstrapped approach is proposed and used here to construct interval estimates for the predicted probabilities.

Data on diabetes mellitus in Pima Indian is used to illustrate the application of neural classifiers. Variable selection was performed in a previous study. Thus, for model selection, this study concentrates on the identification and selection of model architecture. Bootstrap replicates of the training and validation samples lead to the selection of the simplest architecture with no hidden nodes. Interesting results are captured in the test sample. Overall, classification results are comparable to those

previously reported. Here, we set up an indeterminate category based on posterior probabilities. Patients in this gray area are recommended to go through further medical testing. Standard deviation of the estimate is used to construct interval estimates of the posterior probabilities.

References

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

[2] M. D. Richard and R. Lippmann (1991). Neural network classifiers estimate bayesian a posterior probabilities. *Neural Computation*, 3, 461–483.

[3] D. W. Ruck, S. K. Rogers, M. Kabrisky, M. E. Oxley, and B. W. Suter (1990). The multi-layer perceptron as an approximation to a bayes optimal discriminant function. *IEEE Transactions on Neural Networks*, 1, 296–298.

[4] J. W. Smith, J. E. Everhart, W. C. Dickson, W. C. Knowler, and R. S. Johannes (1988). Using the ADAP learning algorithm to forecast the onset of diabetes mellitus. In *Symposium on Computer Applications in Medical Care*, 261–265.

[5] R. O. Duda and P. Hart (1973). *Pattern Classification and Scene Analysis*. Wiley & Sons, New York.

[6] A. R. Barron (1989). Statistical properties of artificial neural networks. In *28th IEEE Conference on Decision and Control*, 280–285.

[7] G. Cybenko (1989). Approximation by superpositions of a sigmoidal function. *Mathematics of Control, Signals, and Systems*, 2, 303–314.

[8] K.-I. Funahashi (1989). On the approximate realization of continuous mappings by neural networks. *Neural Networks*, 2, 183–192.

[9] D. E. Rumelhart, G. E. Hinton, and R. J. Williams (1986). *Parallel Distributed Explorations in the Microstructure of Cognition*, Chapter in Learning Internal Representation by Error Propagation. MIT Press, Cambridge, MA.

[10] B-H Ahn (1996). *Forward Additive Neural Network Models*. PhD thesis, Kent State University, College of Business, Kent, OH, USA.

[11] J Nocedal (1980). Updating quasi-newton matrices with limited storage. *Mathematics of Computation*, 35, 752–773.

[12] J. Denton (1991). *A Robust and Efficient Training Algorithm for Feedforward Neural Networks*. PhD thesis, Kent State University, Kent, OH, USA.

[13] M. Hu, M. S. Hung, M. Shanker, and H. Chen (1996). Using neural networks to predict the performance of sino-foreign joint ventures. *International Journal of Computational Intelligence and Organizations*, 1(3), 134–143.

[14] M. Hu, E. Patuwo, M. S. Hung, and M. Shanker (1999). Neural network analysis of performance of Sino-Hong Kong joint ventures. *Annals of Operations Research*, 87, 213–232.

[15] M. S. Shanker (1996). Using neural networks to predict the onset of diabetes mellitus. *Journal of Chemical Information and Computer Sciences*, 36(1), 35–41.

[16] B. Efron and R. J. Tibshirani (1993). *An Introduction to Bootstrap*. Chapman & Hill, N.Y.

[17] P. M. Murphy and D. W. Aha (1994). UCI repository of machine learning databases (machine-readable data depository). Department of Information and Computer Science, University of California, Irvine, CA.

[18] W. C. Knowler, P. H. Bennett, R. F. Hamman, and M. Miller (1978). Diabetes incidence and prevalence in Pima Indians: A 19-fold greater incidence than in Rochester, Minnesota. *American Journal of Epidemiology*, 108(6), 497–505.

[19] W. C. Knowler, D. J. Pettitt, P. J. Savage, and P. H. Bennett (1981). Incidence in Pima Indians: Contributions to obesity and prenatal diabetes. *American Journal of Epidemiology*, 113(2), 144–156.

[20] WHO technical report series (1985). Technical Report 727, WHO Study Group.

[21] Leo Breiman (1996). Bagging predictors. *Machine Learning*, 24, 123–140.

TABLES and FIGURES

	Non Diabetic	Diabetic	Total
Training	301	160	461
Validation	105	54	159
Test	94	54	148
Total	500	268	768

Table I: Data Distribution

	True State		
Classified as	Non Diabetic	Diabetic	Total
Negative	65	17	82
Indeterminate	24	19	43
Positive	5	18	23
Total	94	54	148

Table II: Test Set Classification using Combined Training + Validation Sets

	True State		
Classified as	NonDiabetic	Diabetic	Total
Negative	66	17	83
Indeterminate	23	19	42
Positive	5	18	23
Total	94	54	148

Table III: Test Set Classification using Bootstrapping on the Training+Validation set

Figure 1: Mean Validation Set SSE versus Hidden Nodes

Figure 2: Test Set Posterior Probability: Bootstrapped Mean (Y) versus Non-Bootstrapped Mean (X) Estimates

Figure 3: Test Set SSE Across Bootstrap Replicates

Figure Titles

Figure 1 Mean Validation Set SSE versus Hidden Nodes

Figure 2 Test Set Posterior Probability: Bootstrapped Mean (Y) versus Non-Bootstrapped Mean (X) Estimates

Figure 3 Test Set SSE Across Bootstrap Replicates