Bootstrapping Neural Discriminant Models

Masaaki Tsujitani and Takashi Koshimizu

Abstract In this article the role of the bootstrap is highlighted for nonlinear discriminant analysis using a feed-forward neural network model. Statistical techniques are formulated in terms of the principle of the likelihood of a neural network model when the data consist of ungrouped binary responses and a set of predictor variables. We illustrate that the information criterion based on the bootstrap method is shown to be favorable when selecting the optimum number of hidden units for a neural network model. In order to summarize the measure of goodness-of-fit, the deviance on fitting a neural network model to binary response data can be bootstrapped. We also provide the bootstrap estimates of the biases of excess error in a prediction rule constructed by fitting to the training sample in the neural network model. We additionally propose bootstrap methods for the analysis of residuals in order to identify outliers and examine distributional assumptions in neural network model fitting. These methods are illustrated through the analyses of medical diagnostic data.

Index Terms Bootstrapping methods, binary data, goodness-of-fit test, cross-entropy, bias correction, residual analysis.
\section*{\textbf{\Large INTRODUCTION}}

In recent years, neural computing techniques have received increasing attention from statisticians, and some limiting properties of the back-propagation technique have been noted\cite{1-4}. Statistical techniques are formulated in terms of the principle of the likelihood of neural network models regarding the connection weights of the network as unknown parameters in classification problems, as well as the probabilistic interpretation of a neural network model\cite{5,6}. In this article the role of the bootstrap is highlighted for nonlinear discriminant analysis using a feed-forward neural network model when the data consist of a binary response and a set of explanatory variables.

When unknown parameters of a model are estimated by the maximum likelihood method, the maximum value of the log likelihood has a positive bias as an estimator of the expected log likelihood\cite{7,8}. In the application of multi-layer neural network models, an issue has been identified concerning the non-uniqueness of connection weights. We illustrate that the information criterion based on the bootstrap method\cite{9} is shown to be favorable when selecting the optimum number of hidden units for a neural network model. It may provide a better bias estimate in certain cases, since analytic approximations are not used in the bootstrap method.

Introducing the maximum likelihood principle into the neural network model, the deviance allows us to test the goodness-of-fit of the neural network model. In the case of ungrouped binary responses, however, the deviance cannot be used as a summary measure of the goodness-of-fit test of a neural network model\cite{10-12}. We can provide bootstrapping on the null distribution of the deviance with a specified significance level.

The primary aim in discriminant analysis is to assign an individual to one of two or more distinct groups on the basis of measurements of several characteristics. As the same observations are used for forming and assessing the prediction rule, this proportion of errors underestimates the error rate. To correct this bias and estimate the error rates for the prediction rule, many studies have been made such as cross-validation, the jackknife or the bootstrap for estimating excess errors \cite{13-16}. We provide bootstrap estimates of the biases of excess error in prediction rule constructed by fitting to the training sample and the bias-corrected estimates of the actual error rates in a neural network model.

We additionally propose residual analysis, which has been developed in classical regression models in order to detect outliers and quantify their effects. The wealth produced in classical linear regression provides guides and suggestions that may be carefully applied to neural network models. The non-existence of standardized residuals with asymptotically normal or chi-squared\cite{17} for binary data results in the bootstrapping of residuals to assess the discrepancies between a target value and the corresponding fitted value in neural network model fitting.
A feed-forward neural network with a single hidden layer for two-class classification problems is shown as in Fig.1. Each case has input variables \( x_1, \ldots, x_I \) representing predictor variables, with the input weights to hidden units being the \( \alpha \) terms and the weights to the output unit being the \( \beta \) terms. The connection weight vector between the \( i \)-th unit in the input layer \( (i=1, \ldots, I) \) and the \( j \)-th unit in the hidden layer \( (j=1, \ldots, H) \), is \( \mathbf{w}_j = (\alpha_{0j}, \alpha_{1j}, \ldots, \alpha_{Ij})^T \), where \( \alpha_{0j} \) is a bias. Similarly, the weight vector connected to the output unit is \( \mathbf{w}_o = (\beta_0, \beta_1, \ldots, \beta_H)^T \), where \( \beta_0 \) is a bias and \( T \) denotes the transposition of the matrix.

![Feed-forward neural network with a single hidden layer for two-class classification problems](image)

Fig.1 Feed-forward neural network with a single hidden layer for two-class classification problems

The input to the \( j \)-th hidden unit is a linear projection of the input vector \( \mathbf{x} = (x_1, \ldots, x_I) \), i.e.,

\[
    u_j = \sum_{i=0}^{I} \alpha_{ij} x_i, \quad x_0 = 1.
\]

The output of the hidden unit is
where $f(\cdot)$ is a nonlinear activation function. The most commonly used activation function is the logistic sigmoid function

$$f(u_j) = \frac{1}{1 + \exp(-u_j)}.$$  

The input to output unit is

$$v = \sum_{j=0}^{H} \beta_j y_j$$

and the output is

$$o = g(v) = g\left(\sum_{j=0}^{H} \beta_j y_j\right)$$

where $g(\cdot)$ is an alternative nonlinear activation function. When the logistic sigmoid function is used as $f(\cdot)$ and $g(\cdot)$, the functional representation of the output from the neural network for a given input vector $x$ with $\square = (0, 0) = (\alpha_j, \beta_j)$ is

$$o(x; \square) = g\left(\sum_{j=0}^{H} \beta_j f\left(\sum_{i=0}^{I} \alpha_i x_i\right)\right) = \frac{1}{1 + \exp\left(-\sum_{j=0}^{H} \beta_j \left(\frac{1}{1 + \exp\left(-\sum_{i=0}^{I} \alpha_i x_i\right)}\right)\right)}. \quad (1)$$

The number of parameters included in the function (1) is $p = H(I+1)+H+1$.

The sum-of-squares error function has been frequently used. With this error function, the unknown parameters are estimated by minimizing

$$E = \frac{1}{2} \sum_{d=1}^{D} \left| o^{(d)} - t^{(d)} \right|^2.$$  

where $o^{(d)}$ is the output(1) and $t^{(d)}$ is a target value in the neural network context for the $d$-th case. In the case of the sum-of-squares error function, we assume that the distribution of the target value can be defined by a smoothing deterministic function, added to normally distributed noise,

$$t = z(x) + \varepsilon, \varepsilon \sim N(0, \sigma^2) \quad (3)$$

and seek to model $z(x)$, with outputs $E(t^{(d)}|x) = o^{(d)}(x; \square)$, which maximize the likelihood function. The principle of this maximum likelihood method is based on the assumption of normally distributed target data. For classification problems, however, the target values are binary variables, and the normally distributed noise model does not provide a good description of their distribution. A more appropriate choice of error function is needed.

When an input vector, $x^{(d)} = (x_1^{(d)}, x_2^{(d)}, \ldots, x_I^{(d)})$, $d=1, \ldots, D$ is given in two-class classification problems, an output of a classifier, $o^{(d)} = o^{(d)}(x; \square)$, is viewed as
an estimator of the conditional probability that $i^{cd} = 1$, i.e.,
$$\Pr\{i^{cd} = 1 \mid x^{cd}\} = o^{cd}.$$  

The conditional probability of $i^{cd} = 0$ is given by
$$\Pr\{i^{cd} = 0 \mid x^{cd}\} = 1 - o^{cd}.$$  

With this interpretation of network outputs, we can present a particular case of the binomial distribution, called the Bernoulli distribution
$$f(i^{cd}) = \begin{cases} o^{cd} & \text{if } i^{cd} = 1 \\ 1 - o^{cd} & \text{if } i^{cd} = 0 \end{cases}.$$  

The variance of this distribution can be given by
$$\Var[f(i^{cd})] = o^{cd}(1 - o^{cd}).$$  

Assuming that data are drawn independently from the distribution, the likelihood of observing the training sample $t = (t^{c1}, t^{c2}, \ldots, t^{cD})$ and $x = (x^{c1}, x^{c2}, \ldots, x^{cD})$ is
$$L(x, t; \Theta) = \prod_{d=1}^{D} \left\{ o^{cd} \cdot [1 - o^{cd}] \right\}^{t^{cd} - o^{cd}}.$$  

The logarithm of the likelihood is written in the form
$$\ln L(x, t; \Theta) = \sum_{d=1}^{D} \left\{ t^{cd} \ln o^{cd} + (1 - t^{cd}) \ln(1 - o^{cd}) \right\}.$$  

The estimated outputs $\hat{o}^{cd}$ are hence the maximum likelihood estimators of these parameters. The back-propagation algorithm [18] enables us to give an explicit updating rule for estimating $\hat{o}^{cd}$ (i.e., the connection weights, $\hat{\alpha}_{ij}, \hat{\beta}_{ij}$).

The negative logarithm of the likelihood is a cross-entropy error function [19-22]:
$$F(\Theta) = -\sum_{d=1}^{D} \left\{ t^{cd} \ln o^{cd} + (1 - t^{cd}) \ln(1 - o^{cd}) \right\}.$$  

The Kullback-Leibler measure
$$I(\Theta) = \sum_{d=1}^{D} \left\{ t^{cd} \ln \left( \frac{t^{cd}}{o^{cd}} \right) + (1 - t^{cd}) \ln \left( \frac{1 - t^{cd}}{1 - o^{cd}} \right) \right\}$$  

may be expanded explicitly as
$$\sum_{d=1}^{D} \left\{ t^{cd} \ln \left( \frac{t^{cd}}{o^{cd}} \right) + (1 - t^{cd}) \ln \left( \frac{1 - t^{cd}}{1 - o^{cd}} \right) \right\} = -\sum_{d=1}^{D} \left\{ t^{cd} \ln \left( \frac{1}{t^{cd}} \right) + (1 - t^{cd}) \ln \left( \frac{1}{1 - t^{cd}} \right) \right\} + \sum_{d=1}^{D} \left\{ t^{cd} \ln o^{cd} + (1 - t^{cd}) \ln(1 - o^{cd}) \right\}.$$  

Since the first term of the right side in (10), which depends on only $t^{cd}$, is zero, the smaller second term, i.e. the cross-entropy, yields the smaller Kullback-Leibler measure. Maximizing the logarithm of the likelihood function thus corresponds to minimizing the cross-entropy error function and minimizing the Kullback-Leibler measure [23]. The outputs are good estimators of the conditional expectation of target values.
The Kullback-Leibler measure also has a relationship with the sum-of-squares error function as in Theorem 1.

**Theorem 1.** The Kullback-Leibler measure is approximately equivalent to the sum-of-squares error function with the weighting factor of the reciprocal of the variance of Bernoulli distribution according to

\[
\sum_{d=1}^{D} \left( t^{d} \cdot \ln \left( \frac{t^{d}}{o^{d}} \right) + (1 - t^{d}) \cdot \ln \left( \frac{1 - t^{d}}{1 - o^{d}} \right) \right) \approx \frac{1}{2} \sum_{d=1}^{D} \left( o^{d} \cdot \frac{1}{1 - o^{d}} \left( t^{d} - o^{d} \right)^{2} \right). \tag{11}
\]

**[Proof]** Using the formula,

\[
\ln \left( \frac{a}{b} \right) \approx (a - b) + \frac{1}{2} \left( a - b \right)^{2} + \ldots,
\]

it follows that

\[
\sum_{d=1}^{D} \left( t^{d} \cdot \ln \left( \frac{t^{d}}{o^{d}} \right) + (1 - t^{d}) \cdot \ln \left( \frac{1 - t^{d}}{1 - o^{d}} \right) \right) = \sum_{d=1}^{D} \left( t^{d} - o^{d} \right) + \frac{1}{2} \left( t^{d} - o^{d} \right)^{2} + \left( t^{d} - o^{d} - 1 + o^{d} \right) \left( \frac{1}{2} - o^{d} \right) \right) \approx \frac{1}{2} \sum_{d=1}^{D} \left( o^{d} \cdot \frac{1}{1 - o^{d}} \left( t^{d} - o^{d} \right)^{2} \right) \tag{Q.E.D.}
\]

In the case of the sum-of-squares error function, the variance of added normal distribution noise \( \varepsilon \) in (3) is constant \( \sigma^{2} \). However the variance of the Bernoulli distribution is not constant and depends on \( \hat{o}^{d} \) as in (5). Theorem 1 can thus be derived by giving weights of the reciprocal of the variance (5) to the sum-of-squares error function.

### ☛. BOOTSTRAPPING

**A. Determination of the Optimum Number of Hidden Units**

In conventional statistics, various criteria have been developed for assessing the generalization performance. The AIC (Akaike Information Criterion) provides us with a decision as to which of several competing network architectures is best for a given problem. From the point of entropy maximization principle, the goodness of statistical models is evaluated by the expected log likelihood \( E_{Z} \[ \ln L(Z; \hat{\theta}) \] \). However we cannot derive the expected log likelihood since the true distribution of \( Z \) is unknown. A natural estimate of the expected log likelihood is provided by \( \ln L(X; \hat{\theta}) \). Its bias is defined by

\[
C = E_{Z} \left[ \ln L(X; \hat{\theta}(X)) - E_{Z} \ln L(Z; \hat{\theta}(X)) \right]. \tag{12}
\]

If an estimator \( \hat{C} \) of \( C \) is available, an unbiased estimator of the expected log likelihood of the distribution is obtained by

\[
\ln L(X; \hat{\theta}(X)) - \hat{C}. \tag{13}
\]

The bias \( C \) can be asymptotically approximated by the dimension of the
parameter vector and defined as
\[ AIC = -2 \ln L(X; \hat{\theta}(X)) + 2p, \] (14)
where \( p \) is the number of parameters in the neural network model. The \( AIC \) approach selects the model with minimum value of (14).

Let the (original) training sample \( X = \{X^{<d>, X^{<e>}, \ldots, X^{<D>}\} \) with \( X^{<d>} = (x^{<d>}; t^{<d>}) = (x_1^{<d>, x_2^{<d>}, \ldots, x_I^{<d>}; t^{<d>}) \) be independently distributed in an unknown distribution \( F \), where \( x_1^{<d>, x_2^{<d>}, \ldots, x_I^{<d>}) \) are \( I \)-dimensional explanatory variables and \( t^{<d>} \) is a target value on the corresponding individual indicating the presence/absence. Let \( \hat{F} \) be the empirical distribution with probability distribution of \( 1/D \) at each point \( X^{<d>, X^{<e>},\ldots, X^{<D>}. \) The bootstrap pairs sampling algorithm is given as follows:

**Step 1.** Generate \( B \) bootstrap samples \( X^* = \{X^{<d>*}, X^{<e>*}, \ldots, X^{<D>*}\} \) from \( \hat{F} \). We denote the \( b \)-th bootstrap sample by \( X^*(b) \).

**Step 2.** For each bootstrap sample, \( b = 1, \ldots, B \), fit a model and estimate parameters \( \hat{\theta}(X^*(b)) \).

**Step 3.** The bootstrap estimator of bias \( C \) in (12) is given by
\[ C^* = E_X \left[ \ln L(X; \hat{\theta}(X)) - \ln L(X; \hat{\theta}(X^*(b))) \right] \]
\[ = \frac{1}{B} \sum_{b=1}^{B} \left[ \ln L(X^{<d>}; \hat{\theta}(X^*(b))) - \ln L(X; \hat{\theta}(X^*(b))) \right] \] (15)
where \( \ln L(X^{<d>}; \hat{\theta}(X^*(b))) \) is the log likelihood on the bootstrap sample and \( \ln L(X; \hat{\theta}(X^*(b))) \) is the log likelihood on the training sample given \( \hat{\theta}(X^*(b)) \).

The bootstrap bias correction of the log likelihood is then
\[ \ln L(X; \hat{\theta}(X)) - C^*, \]
and the \( EIC \) (Extended Information Criterion) based information criterion [9] is defined by
\[ EIC = -2\ln L(X; \hat{\theta}(X)) + 2C^*. \] (16)

Note that the bootstrap algorithm requires refitting the model \( B \) times. Typically \( B \) is in range of \( 50 \leq B \leq 400 \) for estimating some biases or standard error. A bootstrap number \( B \) of 400 is perhaps reasonable when fitting complicated models like neural network models [14,24].

**B. Bootstrapping the Log Likelihood Test Statistic**

Introducing the maximum likelihood principle into the neural network model, the deviance allows us to test the goodness-of-fit of the neural network model:
\[ Dev = 2 \left[ \ln L(\text{max}) - \ln L(X; \hat{\theta}) \right] = -2 \sum_{d=1}^{D} \left\{ t^{<d>} \ln \hat{\delta}^{<d>} + \left( 1 - t^{<d>} \right) \ln (1 - \hat{\delta}^{<d>}) \right\} \] (17)
where \( \ln L(X; \hat{\theta}) \) denotes the maximized log likelihood under some current neural network model, and the log likelihood for the saturated model \( \ln L(\text{max}) \) is zero. The degrees of freedom, \( D-p \), equal the difference between the number of
independent observations and the number of unknown parameters in the neural network model. The greater the deviance, the poorer the model fits. It should be noted that the deviance (17) is equivalent to twice the Kullback-Leibler measure (9), i.e.,

$$\text{Dev} = 2 \times I(t; o).$$

The deviance (17) is, however, not even approximately distributed $\chi^2$ in the case where ungrouped binary responses are available [10,11,17]. The number of degrees of freedom, which is required for the test for significance using the assumed $\chi^2$ distribution for the deviance, is a contentious issue. No adequate distribution theory for the deviance exists[12]. Consequently, the deviance on fitting a model to binary response data cannot be used as a summary measure of the goodness-of-fit test of a model.

From the above points of view, we employ bootstrapping the deviance (17) for the goodness-of-fit test:

**Step 1.** Generate $B$ bootstrap samples $X^* = \{X^*_{q(b)}, X^*_{p(b)}, ..., X^*_{b(b)}\}$ from $F$. Let $X^*_{b(b)}$ denote the $b$-th bootstrap sample.

**Step 2.** For the bootstrap sample $X^*_{b(b)}$, the deviance (17) is computed by

$$\text{Dev}(b) = 2 \left[ \ln L(\text{max}) - \ln L\left(X^*_{q(b)}; \hat{\theta}^*(b)\right) \right].$$

This process is repeated independently $B$ times, and these computed values are arranged in ascending order.

**Step 3.** The value of the $j$-th order statistic $\text{Dev}^*$ of the $B$ replications can be taken as an estimate of the quantile of order $j/(B+1)$.

**Step 4.** The estimate of the $100(1-\alpha)-th$ percentile of $\text{Dev}^*$ is used to test the goodness-of-fit of a model with a specified significance level $\alpha = 1 - j/(B+1)$. The greater value of the deviance (17) than the estimate of the percentile indicates that the model fits poorly [25].

**C. Excess Error Estimation**

The primary aim in discriminant analysis is to assign an individual to one of two or more distinct groups on the basis of measurements of several characteristics. As the same observations are used for forming and assessing the prediction rule, this proportion of errors underestimates the error rate. To correct this bias and estimate the error rates for the prediction rule, we provide the bootstrap estimates of the biases of excess error in the prediction rule constructed by fitting to the training sample and the bias-corrected estimates of the actual error rates in neural network model.

We can apply a prediction rule on the training sample. Let $e(F; X)$ be the error rate, the probability of incorrectly predicting the outcome of a new observation drawn from $F$, given a prediction rule on a training sample of $X$. It is defined as the actual error rate, which is of interest for performance assessment of prediction rules. Let $e(\hat{F}; X)$ be the error rate, the probability of incorrectly predicting the outcome on the sample drawn from the empirical distribution of
the training sample, \( \hat{F} \), which is called the *apparent error rate*. The usage of the training sample for both forming and assessing the prediction rule results in the underestimation of the error rate. The difference

\[
b(F) = E[\hat{e}(\hat{F}; X) - e(F; X)]
\]

is bias, called the *excess error* of a given prediction rule. This excess error must be estimated when the training sample is small relative to the number of unknown parameters.

The bootstrap method can be used to estimate the excess error and actual error rate when fitting neural network models to data of classification problems. The bootstrap sampling algorithm is given as follows:

**Step 1.** Generate \( B \) bootstrap samples \( X^* = \{X^{<\cdot>_1}, X^{<\cdot>_2}, ..., X^{<\cdot>_B}\} \) from \( \hat{F} \). Let \( \hat{F}^* \) be the empirical distribution of \( X^{<\cdot>_1}, X^{<\cdot>_2}, ..., X^{<\cdot>_B} \).

**Step 2.** For each bootstrap sample, \( b = 1, ..., B \), fit a model and estimate parameters \( \hat{\theta}(X^*(b)) \).

**Step 3.** The bootstrap estimate of the excess error in (18) is given by

\[
\hat{b}(\hat{F}) = E[\hat{e}(\hat{F}^*; X^*) - \hat{e}(\hat{F}; X^*)] \approx \frac{1}{B} \sum_{b=1}^{B} [\hat{e}(\hat{F}^*(b); X^*(b)) - \hat{e}(\hat{F}; X^*(b))];
\]

where \( \hat{e}(\hat{F}^*; X^*) \) is the error rate on the bootstrap sample and \( \hat{e}(\hat{F}; X^*) \) is the error rate on the training sample given a prediction rule constructed with the bootstrap sample.

**Step 4.** Using bias correction (19), the bootstrap estimator of the actual error rate can be obtained as

\[
\hat{e}(\hat{F}; X) - \hat{b}(\hat{F}).
\]

**D. Standardized Residuals**

To assess the discrepancies between a target value and the corresponding fitted value, the wealth produced in classical linear regression provides guides and suggestions that may be carefully applied to neural network models. The target value \( t^{\cdot\cdot} \) has a Bernoulli distribution (4). This setting suggests that the existence of “standardized residuals” with asymptotically normal or chi-squared is not valid [17]. The estimated \( d \)-th raw residual \( \hat{e}^{d\cdot\cdot} = t^{d\cdot\cdot} - \hat{\theta}^{d\cdot\cdot} \) for the training sample can be standardized by

\[
\hat{e}^{d\cdot\cdot} = \frac{t^{d\cdot\cdot} - \hat{\theta}^{d\cdot\cdot}}{\sqrt{\text{Var}(t^{d\cdot\cdot} - \hat{\theta}^{d\cdot\cdot})}}.
\]

The asymptotic variance \( \text{Var}(t^{d\cdot\cdot} - \hat{\theta}^{d\cdot\cdot}) \) is estimated by applying the bootstrap method as follows:

**Step 1.** Generate \( B \) bootstrap samples, each one of size \( D \) drawn with replacement from the training sample \( X = \{X^{\cdot\cdot}_1, X^{\cdot\cdot}_2, ..., X^{\cdot\cdot}_D\} \) with \( X^{\cdot\cdot} = (x^{\cdot\cdot}_1, x^{\cdot\cdot}_2, ..., x^{\cdot\cdot}_D) = (t^{\cdot\cdot}_1, t^{\cdot\cdot}_2, ..., t^{\cdot\cdot}_D) \). Denote the \( b \)-th sample by \( X^*(b) \).
Step 2. For each bootstrap sample, $b = 1, \ldots, B$, maximize the log likelihood function (7) giving $\hat{\mathbf{\beta}}_{\text{boot}} = \left(\hat{\beta}_{\text{boot}}^1, \hat{\beta}_{\text{boot}}^2, \ldots, \hat{\beta}_{\text{boot}}^B\right)$.

Step 3. Estimate the variance of the $d$-th raw residual $\hat{\epsilon}_{\text{raw}}^{<d>} = \hat{\epsilon}(x_{\text{raw}}^{<d>}; \hat{\bm{\mu}})$ giving $\left\{\hat{\epsilon}_{\text{raw}}^{<d>}, \hat{\epsilon}_{\text{raw}*}^{<d>}\right\}$ for the training sample by

$V\text{ar}(\hat{\epsilon}_{\text{raw}}^{<d>}) = \frac{1}{B-1} \sum_{b=1}^{B} \left\{ \hat{\epsilon}(x_{\text{raw}}^{<d>}; \hat{\bm{\mu}}) - \hat{\epsilon}(x_{\text{raw}}^{<d>}; \bullet) \right\}^2$,

where

$\hat{\epsilon}(x_{\text{raw}}^{<d>}; \hat{\bm{\mu}}) = \hat{\epsilon}_{\text{raw}}^{<d>}(x_{\text{raw}}^{<d>}; \hat{\bm{\mu}}), \quad \hat{\epsilon}(x_{\text{raw}}^{<d>}; \bullet) = \frac{1}{B} \sum_{b=1}^{B} \hat{\epsilon}(x_{\text{raw}}^{<d>}; \hat{\bm{\mu}})$.

Step 4. The standardized residual of the $d$-th estimated raw residual $\hat{\epsilon}_{\text{est}}^{<d>}$ is defined as

$\hat{\epsilon}_{\text{std}}^{<d>} = \frac{\hat{\epsilon}_{\text{est}}^{<d>}}{\sqrt{V\text{ar}(\hat{\epsilon}_{\text{est}}^{<d>})}}$. \hspace{1cm} (21)$

The Pearson residual is alternatively estimated by

$\hat{\epsilon}_{\text{p}}^{<d>} = \frac{\hat{\epsilon}_{\text{est}}^{<d>}}{\sqrt{\hat{\rho}_{\text{est}}^{<d>}(1-\hat{\rho}_{\text{est}}^{<d>})}}$. \hspace{1cm} (22)$

It should be noted that the estimation of standard error of the output $\hat{\rho}_{\text{est}}^{<d>}$ in the sum-of-squares error function (2) is also discussed by use of bootstrapping[24].

\section{APPLICATION}

We used the sample of developing Kyphosis with three predictor variables, “Age of child”, “The number of vertebrae” and “The beginning of the range” defined as follows [26]:

Kyphosis: A binary variable indicating the presence/absence of a postoperative spinal deformity called Kyphosis
Age of child: The age of the child in months
The number of vertebrae: The number of vertebrae involved in the spinal operation
The beginning of the range: The beginning of the range of the vertebrae involved in the operation

The feed-forward neural network with a single hidden layer as shown in Fig. 2 is applied to the sample of 81 observations through learning with the cross-entropy error function.
Table I summarizes the results after fitting a neural network model with one hidden unit to seven hidden units. The number of misclassifications with one hidden unit, two hidden units, and three hidden units are 13, 9 and 5, respectively. No misclassification is observed when fitting the neural network models with six or more hidden units. The bias estimates of the log likelihood, $C^*$, and the values of $EIC$ computed from the estimates of biases are also given through refitting to each of $B=400$ bootstrap samples.

Table I

<table>
<thead>
<tr>
<th>Number of Hidden Units</th>
<th>EIC($C^*$)</th>
<th>Number of Apparent Errors</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>93.88(5.98)</td>
<td>13</td>
</tr>
<tr>
<td>2</td>
<td>91.76(9.76)</td>
<td>9</td>
</tr>
<tr>
<td>3</td>
<td>95.36(14.79)</td>
<td>5</td>
</tr>
<tr>
<td>4</td>
<td>99.33(20.30)</td>
<td>2</td>
</tr>
<tr>
<td>5</td>
<td>99.90(22.10)</td>
<td>1</td>
</tr>
<tr>
<td>6</td>
<td>102.29(23.60)</td>
<td>0</td>
</tr>
<tr>
<td>7</td>
<td>106.67(26.99)</td>
<td>0</td>
</tr>
</tbody>
</table>

Fig.1 shows the change in $EIC$ value when the number of hidden units varies from one to seven. Inspecting the results in Fig.1, the minimum value is obtained for the model with two hidden units.
Fig. 3: The values of EIC calculated by 400 bootstrap samples.

Table II provides the bootstrap estimates of the $100(1-\alpha)-th$ percentile of $\text{Dev}^*$ for Kyphosis data. By comparing $\text{Dev}^*(\alpha = 0.05) = 24.22$ for two hidden units in Table II with the value 19.59 of the deviance (17) for Kyphosis data, it suggests that the applied model with two hidden units fits the data fairly well.

<table>
<thead>
<tr>
<th>Number of Hidden Units</th>
<th>$\alpha = 0.05$</th>
<th>$\alpha = 0.10$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>33.20</td>
<td>31.36</td>
</tr>
<tr>
<td>2</td>
<td>24.22</td>
<td>21.66</td>
</tr>
<tr>
<td>3</td>
<td>19.72</td>
<td>16.62</td>
</tr>
<tr>
<td>4</td>
<td>16.23</td>
<td>12.97</td>
</tr>
<tr>
<td>5</td>
<td>11.93</td>
<td>8.80</td>
</tr>
<tr>
<td>6</td>
<td>11.69</td>
<td>5.91</td>
</tr>
</tbody>
</table>

The bootstrap estimates of excess error rate (19) and the bias-corrected error rate (20), i.e., the bootstrap estimates of actual error rates are -0.084 and 0.195, respectively.

Residuals are often displayed on half-normal plots in order to detect outliers and quantify their effects in neural network model fitting. The standardized residual (21) based on the bootstrap method after fitting a neural network model with two hidden units to the data of developing Kyphosis are shown in Fig. 4. The residual plots display no outliers and give evidence supporting the network model with two hidden units.
Fig.4 Half-normal plot of the standardized residual (21) based on bootstrap method after fitting neural network model with two hidden units

### CONCLUSION

In this article, we introduced the probabilistic interpretation of a network output and constructed the maximum likelihood principle of the network model based on the probabilistic approach in classification problems. We also discussed the learning algorithm by maximizing the log likelihood function, that is, minimizing the cross-entropy error function and minimizing the Kullback-Leibler measure. We then derived the theorem of the relationship of two types of error function, i.e., the sum-of-squares error function and the Kullback-Leibler measure.

We considered a feed-forward neural network model with a single hidden layer for two-class classification problems, and suggested the method for estimating some biases on the expected log likelihood and the excess error by the use of bootstrap resampling. We suggested that the information criterion be based on the bootstrap method when selecting the optimal number of hidden units in application of neural network models in classification problems.

Introducing the maximum likelihood principle into the neural network model, the deviance allows us to test the goodness-of-fit of the neural network model. In the case of ungrouped binary responses, however, the deviance cannot be used as a summary measure of the goodness-of-fit test of a neural network model. We can bootstrap the null distribution of the deviance with a specified significance level.

We also presented bootstrap estimates of the biases of excess error in a
prediction rule constructed from fitting to the training sample, and the bias-corrected estimates of the actual error rates in neural network models.

We finally proposed residual analysis which has been developed in classical regression models in order to detect outliers and quantify their effects in neural network model fitting. To assess the discrepancies between a target value and the corresponding fitted value, we derived the standardized residuals from comparison with Pearson residuals.

REFERENCES


