VERİ BİLİMİ DERGİSİ
www.dergipark.gov.tr/veri

An Application on Researching the Usability and Efficiency of Artificial Neural Network Models in Statistical Applications

Halil ERYILMAZ[1]*, Ali Fuat YÜZER[2]

[1]Eskişehir Technical University, Faculty of Science, Department of Statistics, Eskişehir
[2]Anadolu University, Faculty of Science, Department of Statistics, Eskişehir

Makale Bilgisi

Başvuru:
06/07/2023
Kabul:
29/08/2023

Abstract

Artificial neural networks can produce solutions for samples that have not been addressed before, by generalizing the relationship between inputs and outputs related to a problem from existing samples. In the studies conducted in this area, it has been pointed out that the artificial neural network models are similar to some statistical methods, while the working principles of some of them are almost the same. Because of the similarities between these two disciplines, they need to be compared with each other to show how important one is in the development of the other. In this study, it is aimed to investigate the usability and effectiveness of artificial neural network models in binary classification problems. For this purpose, firstly, artificial neural network models are briefly mentioned. Then, the similarities between them and some statistical methods were taken into consideration. In the application phase, logistic regression analysis, which is frequently used in binary classification problems, and artificial neural network models were applied on a data set obtained from patients who consulted the Internal Medicine polyclinic of Osmangazi University Health, Application and Research Hospital, and the results were compared. According to the results obtained, it was observed that artificial neural network models gave better results than logistic regression analysis in binary classification problems.

Keywords: *Artificial Neural Networks, Backpropagation, Logistic Regression Model, Binary Classification*

Yapay Sinir Ağı Modellerinin İstatistiksel Uygulamalardaki Kullanılabilirliği ve Etkinliğinin Araştırılması Üzerine Bir Uygulama

Özet

Yapay sinir ağları, var olan numunelerden bir problem ile ilgili girdi ve çıktılar arasındaki ilişkiyi genelleştirerek önceden ele alınmamış numuneler için çözümler üretebilmektedirler. Bu alanda yapılan çalışmalarda yapay sinir ağları modellerinin bazı istatistiksel yöntemlere benzerlik gösterdiğine, bazılarının ise çalışma prensiplerinin hemen hemen aynı olduğuna dikkat çekilmiştir. Bu iki disiplinin birbirleriyle benzerlikleri nedeniyle, birinin diğerinin gelişiminde ne kadar önemli olduğunu göstermek için birbirleriyle karşılaştırılmaları gereklidir. Bu çalışmada yapay sinir ağı modellerinin ikili sınıflandırma problemlerinde kullanılabilirliği ve etkinliğinin araştırılması amaçlanmıştır. Bu amaçla, ilk olarak kısaca yapay sinir ağı modellerinden bahsedilmiş ve bazı istatistiksel yöntemler ile aralarındaki

* İletişim e-posta: haeryilmaz@eskisehir.edu.tr

benzerlikler göz önüne alınmıştır. Uygulama aşamasında ise Osmangazi Üniversitesi Eğitim ve Uygulama Hastanesi İç Hastalıkları polikliniğine başvuran hastalardan elde edilmiş bir veri seti üzerinde iki sınıflı sınıflandırma problemlerinde sıkça kullanılan lojistik regresyon analizi ve yapay sinir ağ modelleri uygulanmış, elde edilen sonuçlar karşılaştırılmıştır. Elde edilen sonuçlara göre, yapay sinir ağı modelleri ikili sınıflandırma problemlerinde lojistik regresyon analizine göre daha iyi sonuçlar verdiği gözlenmiştir.

Anahtar Kelimeler: *Yapay Sinir Ağları, Geriye Yayılım, Lojistik Regresyon Modeli, İkili Sınıflandırma*

1 Introduction

Artificial neural networks, one of the branches of artificial intelligence, are used directly not only in artificial intelligence studies but also in many branches of science. Today, software developed using artificial neural networks can show how it can change the future [1,2].

In 1943, Warren McCulloch and Walter Pitts created the first artificial neuron network. The 1980s were a breakthrough period for studies on artificial neural networks. Artificial neural networks are effectively employed in place of traditional techniques in business, finance, industry, and education [1].

In some studies published in the early 1990s, it was noted that some artificial neural network models and some statistical techniques are similar or even the same. Further studies have shown that this is not a coincidence, but that these two areas are highly correlated. Comparison of artificial neural network models and statistical techniques revealed that one is important in the development of the other. Some scientists suggest that some neural network models such as single-layer and multi-layer may be useful for statistical applications, and that some statistical techniques such as estimation criteria, confidence intervals, and diagnostic methods can be applied to artificial neural network applications. Improving communication between artificial neural networks and statistical methodology provides great benefits for both fields [1, 3, 4].

The aim of this study is to examine how well artificial neural network models work in some statistical applications. In order to achieve this, artificial neural network models are described, and similarities between some artificial neural network models and some statistical techniques are considered. In the application phase, logistic regression analysis, which are frequently used in binary classification problems, and artificial neural network models, were applied on a data set obtained from the health field and the obtained results were compared.

2 Artificial neural networks

2.1 An artificial neural network

An artificial neural network is an information processing system with characteristics similar to those of biological neurons. Artificial neural network models have been created based on the following assumptions as a result of generalizing the mathematical models of human cognitive and biological neuron structure:

- Information processing takes place in many simple elements called neurons.
- Signals occur over connections that provide the relationship between neurons.
- Each link has a weight value. As with biological neurons, these weight values are multiplied by the signals sent.
- Every neuron uses an activation function, which is often a nonlinear function, to produce an output signal.

Thus, an artificial neural network can be defined as a combination of neurons and the connection between them (neuron structure), methods for calculating weights on connections (learning algorithms) and activation function [5, 6, 7].

2.2 Components of an artificial neural network

Artificial neural network models are composed of dense interconnections of simple computational elements, which ensures a high performance, taking into account the known structure of the biological nervous system. An artificial neural network model, as shown in Figure 1, typically comprises of layers, computing components that, in each layer, execute tasks comparable to those in human nerve cells and can vary in number, and extensive connections between these computational elements across the layers. Node, unit, processing element, and artificial neuron (or simply neuron) are the names given to the computational components utilized in various artificial neural network models.

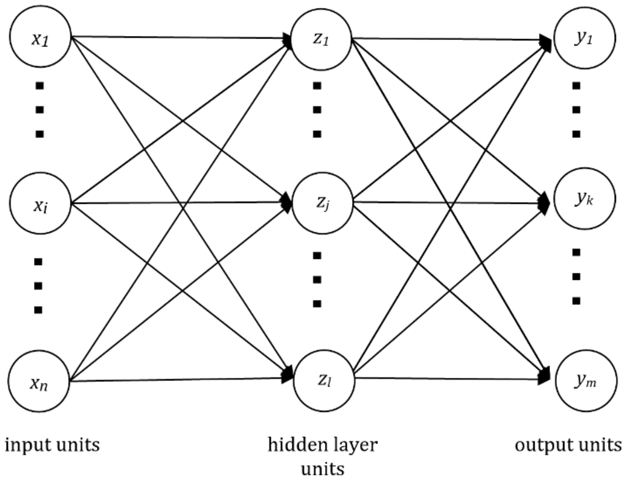


Figure 1. A general neural network model [6].

The neuron is the basic processing element of a network. All neurons in the network receive one or more inputs and give a single output. This output can be outputs to the outside of the artificial neural network, or it can be used as input to other neurons [8].

One of the most important elements of an artificial neural network is connections. Each link also has a weight value. All connections that provide the transmission of inputs between neurons in the artificial neural network have different weight values. In short, the weights represent the information required for the neural network to solve a problem [9, 10].

Layers are formed when neurons come together in the same direction. The first layer in the artificial neural network is the input layer and it provides the outside data to be imported into the artificial neural network. The other layer is the output layer, where the information is transmitted to the outside. If there are layers between the input and output layers, they are called hidden layers. An artificial neural network does not need to have a hidden layer, but can have more than one hidden layer [9].

Every neuron has an internal state. This is called the activation level and is a function of the input values received. Usually, one neuron sends its activation as a signal to other neurons. At each stage, a neuron only sends a single signal, but this signal can be sent to more than one neuron at the same time [8].

The *y* neuron, shown as an example in Figure 2, receives input signals from x_1, x_2 and x_3 neurons. The activations (output signals) of these neurons are x_1, x_2 and x_3, respectively. The weights connecting the *y* neuron with the x_1, x_2 and x_3 neurons are w_1, w_2 and w_3, respectively. *y_in* is the sum of the weighted signals from neurons x_1, x_2 and x_3 to the *y* neuron, although it is the network input:

$$y_{in} = w_1x_1 + w_2x_2 + w_3x_3 \qquad (1)$$

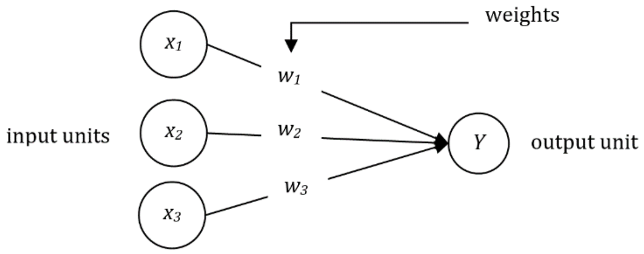


Figure 2. A simple artificial neuron [11].

The activation of the *y* neuron is defined as a function of the input values of the network and different activation functions can be used for this function [1]:

$$y = f(y_{in}) \qquad (2)$$

2.3 Activation functions

The most basic operations to be done in an artificial neural network are to collect the weighted input values and apply an activation function. Sigmoid functions are powerful activation functions. The two sigmoid functions that are widely used are logistic and hyperbolic tangent functions, because the computational load during learning is reduced by the relation between the value of the function at a given point and the value of its derivative at that point [11].

The logistic sigmoid function is a sigmoid function whose values range from 0 to 1 and is frequently used as an activation function for artificial neural networks. To emphasize the interval value of the function, this function is called binary sigmoid, but also called logistic sigmoid [12].

The sigmoid logistic function is calculated by the formula given in Equation (3) or the formula given in Equation (4), which is its derivative.

$$f(x) = \frac{1}{1 + e^{-\sigma x}} \qquad (3)$$

$$f'(x) = \sigma f(x)[1 - f(x)] \qquad (4)$$

Figure 3 shows binary sigmoid curves for different values of σ, which is the step parameter of the logistic sigmoid function. The logistic sigmoid function can be scaled to the appropriate value range and thus becomes a function suitable for any problem.

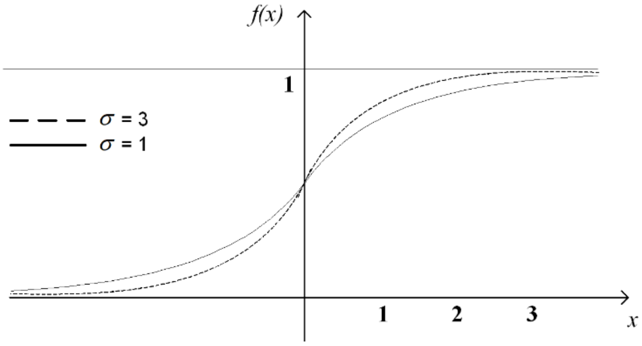


Figure 3. The sigmoid logistic function (for σ =1 and σ =3) [11].

2.4 Training the neural network (adjusting the weights)

An important factor that distinguishes different neural network characteristics from each other is the methods used to adjust the connection weights values during their training. For this, there are generally two training methods: "*supervised learning*" and "*unsupervised learning*" [13].

In supervised learning, a correct output is given as an example of artificial neural networks. According to the difference (error) between the desired and actual output, the weights of the connections between neurons can be adjusted later to obtain the most appropriate output. For this reason, the supervised learning algorithm needs a “supervisor (teacher)”. The delta rule developed by Widrow-Hoff, the generalized delta rule developed by Rumelhart and McClelland, and the backpropagation learning algorithm can be given as examples of supervised learning algorithms [13].

In unsupervised learning, the network develops the classification itself according to the output information obtained from the sample given as input. In these learning algorithms, the desired output value does not need to be known. Only input information is given during the learning process. The network then adjusts the link weights to create patterns that show the same characteristics. Kohonen's self-organizing maps and adaptive resonance theory can be given as examples of unsupervised learning [11, 14].

Single-layer networks fail to solve inseparable problems. In this, scientists have examined multi-layer artificial neural network models. Using the generalized delta rule and nonlinear activation function for multi-layer networks, the backpropagation method has the ability to perform any nonlinear function approximation suitable for specific inputs and targets. This method provides great opportunities for better forecasting, classification and forecasting problems [15].

2.4.1 The backpropagation algorithm with momentum coefficient

In backpropagation with momentum coefficient, the direction of weight change is the combination of the current gradient and the previous gradient. This is a modified version of the gradient reduction method and is an advantage if some training data differ from the majority of the training data. If some unconventional data is to be used, it is good to use this change with a small learning rate. However, although the training data are similar, the speed of the approach can be increased using this change. To use momentum coefficient, the weights of one or previous training samples must be stored [11].

For example, in a simple form of backpropagation with momentum coefficient, the new weights for the training step *t+1* are based on the weights at the training steps *t* and *t–1*. Weight update formulas for backpropagation with momentum coefficient are given in in Equation (5) and Equation (6). The value of the momentum coefficient μ is limited in the range of 0–1.

$$\Delta w_{jk}(t+1) = \alpha\delta_k z_j + \mu\Delta w_{jk}(t) \qquad (5)$$

$$\Delta v_{ij}(t+1) = \alpha\delta_j x_i + \mu\Delta v_{ij}(t) \qquad (6)$$

The momentum coefficient allows the network to vary in weight within an acceptable range and, when used in conjunction with a small learning rate, prevents responses that would lead to large errors for any given sample. When momentum coefficient is used, the network does not trade in the direction of the gradient. It trades in the direction of the combination of the aspects of the current and previous weight correction [11, 13].

2.4.2 Conjugate gradient backpropagation algorithm

In conjugate gradient backpropagation algorithms, a search is usually performed along the conjugate directions, which converge faster than the stepwise reduction directions.

Many training algorithms use a learning rate to determine the weight update. In most conjugate gradient algorithms, the weight update is done at each iteration. A search is made along the conjugate gradient directions for this process, which minimizes the network performance function. There are many search functions. Some search functions are well suited for certain training functions.

All conjugate gradient algorithms start searching in the opposite direction of the gradient on the first iteration:

$$\mathbf{p}_0 = -\mathbf{g}_0 \qquad (7)$$

A straight search is performed to determine the best length to move along the current search direction:

$$\mathbf{x}_{k+1} = \mathbf{x}_k + \alpha_k \mathbf{p}_k \qquad (8)$$

Then the next search direction, which is conjugate to the previous search directions, is determined. The usual method for determining the new search direction is to combine the previous search direction with the new stepwise decrease direction:

$$\mathbf{p}_k = -\mathbf{g}_k + \beta_k \mathbf{p}_{k-1} \qquad (9)$$

The conjugate gradient has several variants, depending on behaves of the calculated constant β_k. The Fletcher-Reeves algorithm is one of them. In Fletcher-Reeves algorithm, the formula for updating weights is given in Equation (10). This equation is the ratio of the square norm of the current gradient to the square norm of the previous gradient.

$$\beta_k = \frac{\mathbf{g}_k^T \mathbf{g}_k}{\mathbf{g}_{k-1}^T \mathbf{g}_{k-1}} \qquad (10)$$

Another variant of the conjugate gradient algorithm was proposed by Polak and Ribiére. As in the Fletcher-Reeves algorithm, the direction search in each iteration is determined by formula is given in Equation (9). The constant β_k in the Polak-Ribière update process, is calculated with the Equation (12).

$$\beta_k = \frac{\Delta \mathbf{g}_k^T \mathbf{g}_k}{\mathbf{g}_{k-1}^T \mathbf{g}_{k-1}} \qquad (11)$$

The search direction is periodicity with respect to the gradient's negative direction in all conjugate gradient algorithms. When the total number of repetitions equals the total number of network parameters, the standard repetition point is reached. However, there are various approaches to repetition that can boost training efficiency. One of these methods is the one proposed by Powell, which builds on Beale's previous methods. In this method, processes are restarted if there is a small verticality between the current gradient and the previous gradient. This is tested by the inequality in Equation (10).

$$\left|\mathbf{g}_{k-1}^T \mathbf{g}_{k-1}\right| \geq 0.2\left\|\mathbf{g}_k\right\|^2 \qquad (12)$$

If this condition is provided, the search direction is repeated according to the opposite direction of the gradient [10, 12].

2.5 Neural network architectures

The location of neurons in layers and the way they connect with neurons in other layers is called network architecture. Each neuron has a input layer, and the activation function of neurons in this layer is equal to the input signal for each neuron. In other words, they accept inputs directly without applying any function.

It is often thought that neurons are arranged in layers. In general, neurons in the same layer behave the same. The main factors that determine the behavior of neurons are the activation functions and the weights on the connections to which the signals are sent. Neurons within each layer usually have the same activation function and are connected to other neurons by the same connection pattern. All neurons in a layer may be fully connected to all other neurons, or they may be unconnected. If every neuron in one layer (for example, the hidden layer) is connected to a neuron in another layer, the hidden unit will be connected to every output neuron.

Neural networks are often classified as single-layer or multi-layer artificial neural networks. In addition to this classification, artificial neural networks with a competitive layer can be added.

When determining the number of layers, the input units is not counted as a layer because there is no computational operation on them. The number of layers in a network is equal to the number of layers of weighted connections which connect neurons [11].

2.5.1 Single-layer neural network models

A single-layer neural network has one weighted connection layer. Often, the units are divided into two as the input unit that receives the signals and the output units from which the network's response is to be received. A typical single-layer neural network model is given in Figure 4.

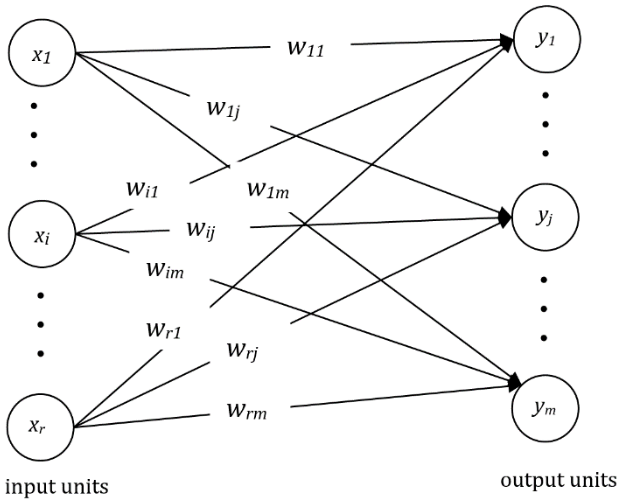


Figure 4. A single-layer neural network model [11].

In single-layer neural network models, the weights that affect one output unit do not affect another output unit. This architecture can be used for sample classification problems. It makes available to take the response for the sample is taken from the output units through the input signals produced in relation to the sample.

Many real-world problems require more complex architectures and complex training rules, and in general, single-layer neural networks are not sufficient to solve such problems. However, if the conditions are suitable for using these networks, it is possible to get accurate results [9, 11].

2.5.2 Multi-layer neural network models

In multi-layer neural networks, there is one or more layers between the input units and the hidden units. Typically, such network architectures with a weighted connection layer between the hidden and output units can solve more complex problems than single-layer network architectures. Multi-layer networks, which can be much more difficult to train, can be extremely successful when some single-layer networks cannot be trained to solve the problem [9, 11]. A typical multi-layer neural network model is shown in Figure 5:

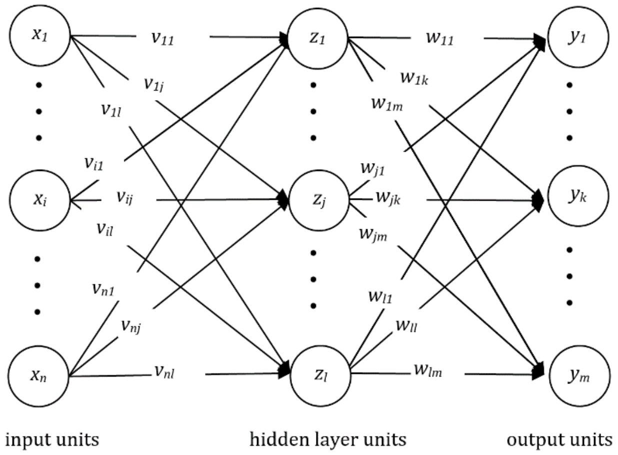


Figure 5. A two-layer neural network model [11].

2.5.3 Neural networks models with a competitive layer

A competitive layer is a form of part of a large number of neural networks. Usually, the connections between neurons in the competitive layer are not shown in the architectural diagram of the network. Competitive links have $-\in$ weights. An artificial neural network model with a competitive layer is shown in Figure 6:

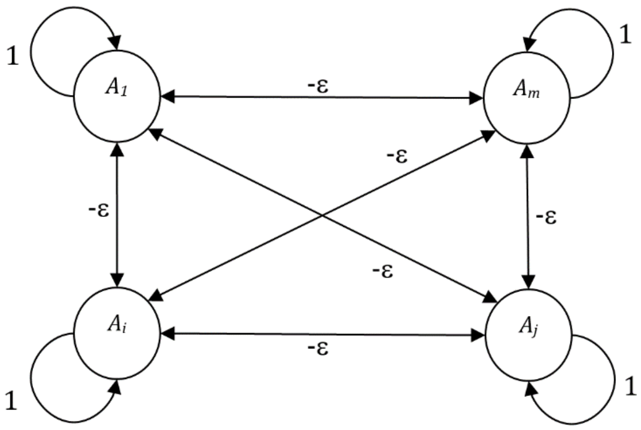


Figure 6. A neural network model with a competitive layer [11].

3 Neural networks and statistical models

In this section, the similarities between neural network models and some statistical techniques will be considered. As it is known, in regression analysis, it is aimed to determine a functional relationship between dependent variables and independent variables with the help of obtained observations, in other words, to find regression parameters and to make predictions with using these parameters. This process is a problem of approximating a (multiple multivariate) function given mathematically specific "input" and "output" values. Kolmogorov proved with a theorem that a multivariable continuous function can be expressed with the help of a finite number of one-variable continuous functions. This theorem is also called “Kolmogorov's Mapping Neural Network Existence Theorem”. This theorem states that any continuous function can be precisely described by a feedforward neural network with three layers (two hidden and output layers). In this respect, multi-layer neural networks seem to be a good approximator, and this leads to the use of neural network models in regression analysis [16]. Therefore, it is necessary to compare appropriate techniques and models for the use of statistical techniques and artificial neural network models in sample classification problems. In this regard, it is of great importance to determine artificial neural network models suitable for statistical techniques such as discriminant analysis, logistic regression etc, and to explain their advantages and disadvantages [1,16]. Also there is increasing interest in the potential role that data science and machine learning can play in healthcare. Machine learning encompasses a range of approaches ranging from applied statistical methods of supervised learning such as logistic regression models to more computationally complex models such as various types of neural networks [17].

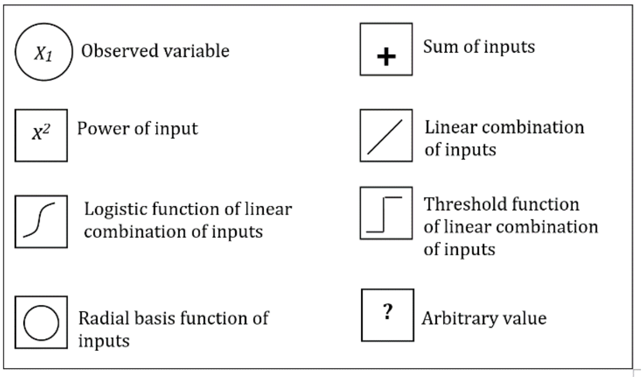
3.1 Comparison of appropriate neural network and statistics terms and symbols

Some neural network models are similar or identical to some statistical techniques [18]. For this purpose, it is necessary to compare the appropriate neural network and statistical terms with different names. Appropriate terms for both fields are given in Table 1.

Table 1. Appropriate neural network and statistical terms [18].

Statistical terms	Neural network terms
variables	features
independent variables	inputs
predicted values	outputs
dependent variables	targets
residuals	errors
estimation	learning (training)
estimation criterion	error function
observations	sample (training pair)
Parameter estimates	weights
transformations	functional links
regression	supervised learning
data reduction	unsupervised learning

Some symbols used in the diagram representation of artificial neural network models are given in Figure 7:



X1
Observed variable
+
Sum of inputs
x2
Power of input
Linear combination of inputs
Logistic function of linear combination of inputs
Threshold function of linear combination of inputs
Radial basis function of inputs
?
Arbitrary value

Figure 7. Some symbols used in the diagram representation of neural network models [18].

Various neural networks can be represented in the form of network diagrams. For example, the diagram, which is given in Figure 8, illustrates neural network and statistical terminology for a simple linear regression model. In Figure 8, neural network terms are written above the symbols in the diagram, and appropriate statistical terms are written below. The circles represent the observation variables whose name is written inside it, and the boxes represent a single or multivariate (activation) function. The sign in the box indicates the type of function.

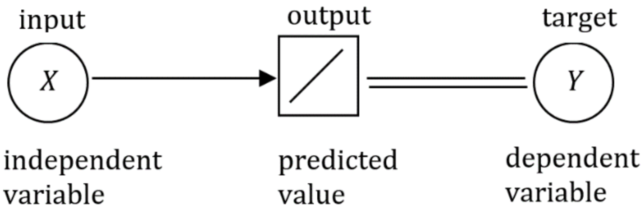


Figure 8. Simple linear regression model represented in the form of a neural network diagram [18].

3.2 Logistic regression model

In the logistic regression model, the expected value of the dependent variable *y* is expressed as *E(y)*, and calculated with the formulas given in Equation (13) or Equation (14).

$$E(y) = \frac{1}{1 + e^{-[b_0 + \sum_{i=0}^{r} b_i x_i]}} \tag{13}$$

$$E(y) = \frac{1}{1 + e^{-\mathbf{x}'\boldsymbol{b}}} \tag{14}$$

Since *y* is a random variable that takes values (0,1), its expected value *E(y)* is equal to the probability

$\pi = p(y=1)$. Accordingly, Equation (14) can be written as Equation (15). Here are $\mathbf{x}' = (1, x_1, \dots, x_r)$ ve $\mathbf{b} = (b_0, b_1, \dots, b_r)$.

$$\pi = p(y=1) = \frac{1}{1+e^{-\mathbf{x}'\boldsymbol{b}}} \qquad (15)$$

The logistic model can be easily linearized. Let Equation (16) be defined to linearize it.

$$\eta = \mathbf{x}'\mathbf{b} \qquad (16)$$

η defined by Equation (16) is a linear estimate and is obtained from Equation (15) as Equation (17).

$$\eta = ln\frac{\pi}{1-\pi} \qquad (17)$$

The transformation given in Equation (17) is called the logit transformation for the probability of π, and the ratio of $\frac{\pi}{1-\pi}$ in this transformation is called the "odds ratio (OR)".

The general form of the logistic regression model is as given in Equation (18).

$$y_j = E(y_j) + \varepsilon_j \ , \qquad j = 1, 2, \dots, n \qquad (18)$$

The expected values of the independent y_j variables in Equation (18) are Bernoulli random variables given in Equation (19).

$$E(y_j) = \pi_j = \frac{1}{1+e^{-\mathbf{x}_\mathbf{j}'\mathbf{b}}} \qquad (19)$$

In the linear expression $\eta = \mathbf{x}'\mathbf{b}$, the maximum likelihood method is applied to estimate the parameters. Since each y_j variable is Bernoulli distributed, the probability distribution of each of them will be as given in Equation (20) and each y_j variable will take the value "0" or "1".

$$f_j(y_j) = \pi_j^{y_j}(1-\pi_i)^{1-y_j} \ , \qquad j = 1, 2, \dots, n \qquad (20)$$

Since the variables are independent, the likelihood function can be expressed as given in Equation (21).

$$L(y_1, y_2, \dots, y_n, b) = \prod_{j=1}^{n} \pi_j^{y_j}(1-\pi_i)^{1-y_j} \qquad (21)$$

Since it is more convenient to work with log-likelihood, Equation (21) is obtained if the logarithm of both parts of Equation (20) is taken.

$$\ln L(y_1, y_2, \dots, y_n, b) = \textstyle\sum_{j=1}^{n} \left[y_j \ln\frac{\pi_j}{1-\pi_j} \right] + \sum_{j=1}^{n} \ln(1-\pi_j) \qquad (22)$$

On the other hand, since $1 - \pi_j = \left[1 + e^{\mathbf{x}_\mathbf{j}'\mathbf{b}}\right]^{-1}$ and $\eta_j = \ln\left[\pi_j/(1-\pi_j)\right] = \mathbf{x}_\mathbf{j}'\mathbf{b}$, log-likelihood can be expressed as Equation (23).

$$\ln L(y, b) = \textstyle\sum_{j=1}^{n} y_j x_i' b - \sum_{j=1}^{n} \ln\left[1 + e^{x'b}\right] \qquad (23)$$

In the logistic regression model, observations or trials are usually repeated for the same level of the variable *x*. if it indicates that y_j is the *j.* observation is “1” and n_j is the number of trials in which such events are provided, log-likelihood takes the form given in Equation (24)

$$\ln L(y, b) = \textstyle\sum_{j=1}^{n} y_j \pi_j + \sum_{j=1}^{n} n_j \ln(1-\pi_j) - \sum_{j=1}^{n} y_j \ln(1-\pi_j) \qquad (24)$$

To find the $\hat{\mathbf{b}}$ maximum likelihood estimators, the numerical search method is applied using Equation (24).

The vector $\hat{\mathbf{b}}$ is assumed to be the final prediction vector. If valid assumptions are provided for the model, the linear estimator is $\hat{\boldsymbol{\eta}} = \mathbf{x}_\mathbf{j}'\hat{\mathbf{b}}$ when $E(\hat{\mathbf{b}}) = \mathbf{b}$ and $Var(\hat{\mathbf{b}}) = (\mathbf{X}'\mathbf{V}^{-1}\mathbf{X})^{-1}$ and the fitted value of the logistic regression model is usually calculated as given in Equation (25) [19, 20].

$$\hat{y_j} = \hat{\pi}_j = \frac{1}{1+e^{\hat{\boldsymbol{\eta}}}} = \frac{1}{1+e^{-\mathbf{x}_\mathbf{j}'\mathbf{b}}} \qquad (25)$$

3.3 Neural network models vs. logistic regression model

A simple single-layer neural network model with *r* number of input units and an output unit with logistic function as activation function is shown in Figure 9. In this case, the weighted signal to the output unit is as given in Equation (26).

$$y\_in = b_0 + \sum_{i=1}^{r} b_i x_i \qquad (26)$$

The logistic function $y = \frac{1}{1+e^{-y\_in}}$ is applied to this signal. The output signal is calculated by the formula given in Equation (27).

$$y = \frac{1}{1+e^{-[b_0 + \sum_{i=1}^{r} b_i x_i]}} \qquad (27)$$

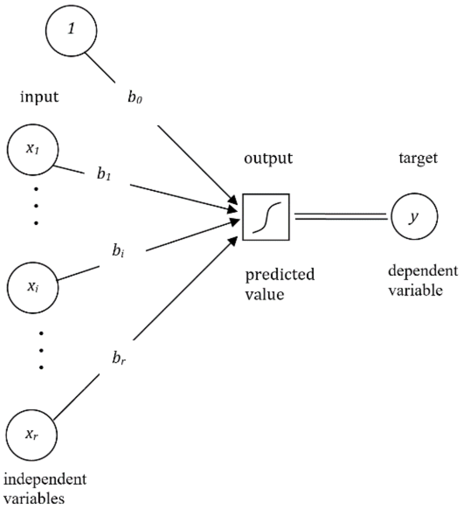


Figure 9. A single-layer neural network model with logistic function as activation function vs. logistic regression model [1].

If the number of input samples is *n*, the output signal y_j correspond to the *j.* input sample $\left[x^j = \left(x_{j1}, x_{j2}, \ldots, x_{jr}\right); j = 1, 2, \ldots, n\right]$ is calculated as given in Equation (28). Here, x_{ji} $(j = 1, 2, \ldots, n;\ \ i = 1, 2, \ldots, r)$ input variables are independent, and y_j $(j = 1, 2, \ldots, n)$ output variables are dependent variables.

$$y_j = \frac{1}{1 + e^{-\left[b_0 + \sum_{i=1}^{r} b_i x_{ji}\right]}} \quad , j = 1, 2, \ldots, n \qquad (28)$$

The model expressed by Equation (28) is similar to the logistic regression model by adding the error term. According to the samples given in the neural network, training is done with the generalized delta rule and appropriate weights are found [11].

A two-layer neural network model with *r* input units, a hidden layer, and an output unit with a logistic function as activation function is shown in Figure 10. In this model, there are *r* neurons in the input layer, *n* neurons in the hidden layer and one neuron in the output layer. The weight connecting *i.* input unit to *j.* hidden layer unit denoted by b_{ij} $(i = 1, 2, \ldots, r)$; $j = 1, 2, \ldots, n)$ and the weight connecting *j.* hidden layer unit to output layer denoted by c_j $(j = 1, 2, \ldots, n)$. b_{0j} $(j = 1, 2, \ldots, n)$ is the deviation weights correspond to the hidden layer, and c_0 is the deviation weight correspond to the output layer.

Let the activation functions of the output and hidden layer be denoted by $f_0(.)$ and $f_h(.)$, respectively. These functions are logistic functions. In this case, for the neural network model shown in Figure 10, the network output vector correspond to p. $\left[x^p = \left(x_{p1}, x_{p2}, \ldots, x_{pr}\right); \ p = 1, 2, \ldots, P\right]$ input sample is calculated with the formula given in Equation (29) [11].

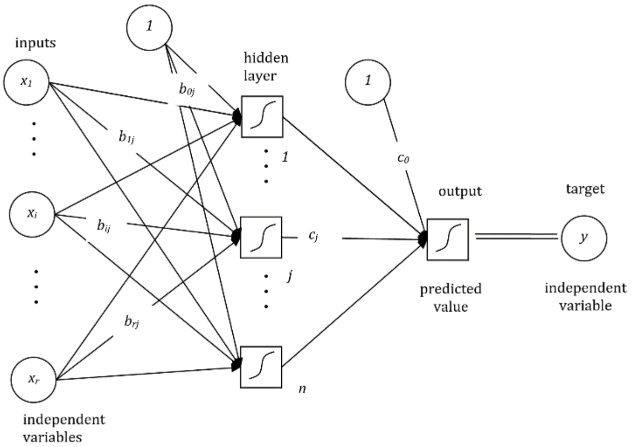


Figure 10. A multi-layer neural network model with a hidden layer and logistic function as activation function vs. logistic regression model [1]

4 Application

In the application phase, the data set of 225 patients who applied to the Internal Medicine polyclinic of Osmangazi University (OGU) Health, Application and Research Hospital was used. Risk factors affecting hypertension were investigated through the obtained data set. For this purpose, some variables thought to be effective on hypertension were included in the study, and it was tried to determine whether these variables were effective on hypertension and their effects on human health as a risk factor. The variables used in the data set obtained from 225 patients are shown in Table 2.

Table 2. The variables used in the data set.

Variable	mean±st.error	min-max
age (*x₁*)	61.81±12.035	(30-87)
height (*x₂*)	166.29±9.634	(140-190)
weight (*x₃*)	72.58±13,696	(45-110)
SBP(systolic blood pressure) (*x₄*)	147.12±31.683	(80-290)
DBP(diastolic blood pressure) (*x₅*)	86.90±15.071	(50-140)
HDL(high density lipoprotein) (*x₆*)	45.14±13.897	(15-88)
total cholesterol (*x₇*)	187.04±54.351	(89-414)
triglyceride (*x₈*)	108.18±70.845	(18-371)
albumin(*x₉*)	3.81±0,584	(2.1-5.0)
glucose (*x₁₀*)	166.55±80.66	(70-440)
gender (*x₁₁*)	(1: male, 2: female)	
smoking (*x₁₂*)	(0: no, 1: yes)	
alcohol use (*x₁₃*)	(0: no, 1: yes)	
exercise (*x₁₄*)	(0: no, 1: yes)	
Heart disease (*x₁₅*)	(0: no, 1: yes)	
hypertension (*y*)	(0: no, 1: yes)	

Among the variables indicated in Table 2, the hypertension variable was considered as the dependent variable and the other variables as the independent variables. Since the hypertension variable has two categories (yes or no), firstly binary logistic regression analysis, which is frequently used in binary classification problems, was applied and then neural network models trained with backpropagation algorithm with momentum coefficient and conjugate gradient backpropagation algorithm were used in the classification of the data. Finally, the results of the applied logistic regression analysis and the neural network models were compared.

SPSS 11.5 statistical package program was used for logistic regression analysis and NeuroSoulitions 4.21 package program was used for neural network model calculations.

4.1 Classification by logistic regression analysis

The y dependent variable of patients with hypertension was taken as 0 for patients without hypertension and 1 for patients with hypertension. Wald test was used to determine the independent variables affecting the hypertension variable, which was taken as the dependent variable in the logistic regression analysis. $\hat{b}$ parameters, standard errors, Wald statistics, degrees of freedom (df), significance levels (p) and Exp($\hat{b}$) for these parameters are given in Table 3.

Table 3. Values obtained as a result of logistic regression analysis.

variable	$\hat{b}$	**st.error**	**Wald**	**df**	p	**Exp($\hat{b}$)**
constant	-10.577	5.119	4.269	1	0.039	0.000
x_1	0.045	0.017	7.419	1	0.006	1.046
x_2	0.029	0.030	0.958	1	0.328	1.030
x_3	0.032	0.017	3.864	1	0.049	1.033
x_4	0.000	0.009	0.003	1	0.959	1.000
x_5	0.035	0.019	3.253	1	0.071	1.035
x_6	0.000	0.015	0.001	1	0.982	1.000
x_7	0.003	0.004	0.572	1	0.449	1.003
x_8	0.000	0.003	0.000	1	0.984	1.000
x_9	-0.508	0.348	2.140	1	0.144	0.601
x_{10}	0.002	0.002	0.676	1	0.411	1.002
x_{11}	-1.744	0.538	10.509	1	0.001	0.175
x_{12}	-0.409	0.428	0.916	1	0.338	0.664
x_{13}	0.485	0.660	0.540	1	0.462	1.624
x_{14}	0.362	0.631	0.329	1	0.566	1.436
x_{15}	-0.473	0.380	1.545	1	0.214	0.623

Exp($\hat{b}$) values given in Table 3 are OR (Odds Ratio) values. This value indicates how many times or what percentage of the probability the dependent variable will be observed with the effect of the independent variables. The significance of the $\hat{b}$ coefficients is also evaluated as the significance of OR values. Variables with an OR value close to 1 are not a significant contributor to the change of the dependent variable. If the coefficients of these variables are not significant, they are interpreted as "the variable is not an important risk factor". If there are OR values greater than 1 (provided that the coefficient is significant), it is interpreted that "the variable is an important risk factor". Values close to zero indicate that the variable is an important risk factor, provided that the coefficient is significant, but that it is a negatively effective factor that causes the dependent variable to have low values. According to the OR values obtained, women have a 0.175 times higher risk of having hypertension than men. In addition, 1 unit increase in age increases the risk of hypertension 1.046 times. Similarly, 1 unit increase in weight increases the risk of hypertension 1.033 times, and 1 unit increase in diastolic blood pressure increases the risk of hypertension 1.035 times.

Hosmer and Lemeshow suggested using 0.15 or 0.25 as significance level for the Wald test in logistic regression [21, 22]. p=0.15 was taken as the significance level in this study. Accordingly, the variables affecting the dependent variable with significance levels less than 0.15 are x_1 (age) [p=0.006], x_3 (weight) [p=0.049], x_5 (DBP) [p=0.071], x_9 (albumin) [p=0.144] ve x_{11} (gender) [p=0.001].

As a result of the logistic regression analysis, the estimated model is as given in Equation 30.

$$\hat{y} = \frac{1}{1+e^{-\begin{bmatrix} -10.577+0.045x_1+0.029x_2+0.032x_3+0.035x_5 \\ +0.003x_7-0.508x_9+0.002x_{10}-1.744x_{11} \\ -0.409x_{12}+0.485x_{13}+0.362x_{14}-0.473x_{15} \end{bmatrix}}} \quad (30)$$

The validity of the estimated model was tested with the Hosmer-Lemeshow test. The Hosmer-Lemeshow chi-square value for the estimation model was calculated as 5.599 (p=0.692). Since p=0.692>α=0.05, the model was accepted as best fit. Accordingly, the coefficient of at least one of the independent variables was different from zero, so the estimation equation is significance.

Classification is made by Equation (30). For this process, the dependent variable estimation values were calculated, and values less than 0.5 were assigned a value of "0" and those greater than 0.5 were assigned a value of "1". These assignment values were compared with the observation values and the correct classification rates were calculated. The results obtained are given in Table 4.

Table 4. Classification table for logistic regression analysis.

			estimated values			
			hypertension			
			no	yes	total	**accuracy (%)**
Observation values	hypertension	no	43	39	82	52.4
		yes	21	122	143	85.3
total			64	161	225	**73.3**

As given in Table 4, 82 of 225 patients were not actually hypertensive. The remaining 143 patients are hypertensive. 43 of 82 patients without hypertension were classified with logistic regression analysis as correct and 39 of them as incorrect, with an accuracy percentage of 52.4%. 122 of 143 patients with hypertension were classified as correct and 21 of them as incorrect, with an accuracy rate of 85.3%.

In the classification process made with logistic regression analysis, the overall accuracy rate was calculated as 73.3% by correctly classifying 165 of 225 patients.

4.2 *Classification by neural network models*

For the classification of the data set, a two-layer neural network model with 15 units in the input layer and one unit in the output layer and one hidden layer was established. As a result of the trials, it was observed that the best accuracy value is observed when the number of units in the hidden layer is 7, and the accuracy percentage decreases when the number of units in the hidden layer is greater than 7. Therefore, the number of units in the hidden layer is taken as 7. Since the hypertension variable has two categories (yes, no), the sigmoid logistic function was used as the activation function of the hidden and the output layers. The established model is shown in Figure 11:

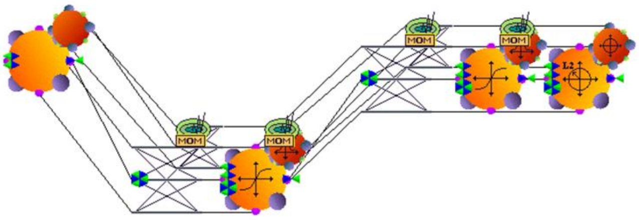


Figure 11. The model set up in the NeuroSolutions 4.21 package program.

In the neural network model shown in Figure 11, there are 15 neurons in the input layer, 7 neurons in the hidden layer, and one neuron in the output layer. The sigmoid logistic function is used as the activation functions of the hidden and output layers. It has been trained with the backpropagation algorithms, which are methods of the supervised learning. During the training, two different methods were used, namely backpropagation algorithm with momentum coefficient and conjugate gradient backpropagation algorithm.

4.2.1 *Classification by the backpropagation algorithm with momentum coefficient.*

The weights calculated after training the network with the backpropagation algorithm with momentum coefficient are given in Table 5.

By using the weights calculated after the network is trained, the network output is calculated with the Equation (31) (As noted here before, the functions f_0 ve f_h are sigmoid logistic functions).

$$\hat{y} = f_0\left[c_{0k} + \sum_{j=1}^{n} c_{jk} f_h\left(b_{0j} + \sum_{i=1}^{r} b_{ij} x_i\right)\right] \quad (31)$$

The $\hat{y}$ values are assigned by giving a value of "0" to those less than 0.5 and a value of "1" to those greater than 0.5 for the classification process. These assignment values were compared with the observation values and the correct classification rates were calculated. The accuracy percentages calculated in the classification process for the data set used after the network was trained are given in Table 6.

Table 5. The weights calculated after training the network with the backpropagation algorithm with momentum coefficient.

b_{ij}	**weights between input layer and hidden layer**						
j / i	**1**	**2**	**3**	**4**	**5**	**6**	**7**
1	0.038183	-0.377079	-0.092539	-0.844849	1.088823	0.000232	0.340129
2	-0.047652	-0.142448	-0.076438	0.206520	-1.179121	-0.472014	-0.325958
3	-0.018952	-0.426096	-0.585312	-0.207449	0.397902	0.145364	-0.154750
4	0.014908	-0.077396	-0.932752	0.427744	0.030066	-0.362468	0.054019
5	0.033981	-0.489262	0.625766	-0.324133	0.689294	0.617243	-0.005147
6	0.002287	0.120196	0.225298	0.262354	-0.404114	0.168880	0.007833
7	-0.028441	0.263126	-0.171264	0.148204	-0.341717	0.024915	-0.033381
8	-0.017017	0.508269	0.384997	-0.008662	-0.759987	-0.511327	-0.215143
9	-0.003526	-0.018455	-0.599053	-0.484052	-0.683164	-0.286301	-0.071966
10	0.004560	0.071364	-0.690455	-0.308165	1.074355	-0.147219	-0.066622
11	0.016332	-0.303747	-0.752405	0.016994	0.582696	0.530000	0.105504
12	0.021496	0.336081	0.283152	0.030311	0.555092	0.522999	0.293574
13	0.023933	0.421452	0.730063	-0.042374	-0.011385	-0.303856	0.449919
14	0.005816	0.514841	-0.918895	0.267335	-0.460008	0.719241	-0.336804
15	-0.053653	-0.375008	-0.190207	0.348217	-1.206198	-0.020581	-0.566735
c_{jk}	**weights between hidden layer and output layer**						
j / k	**1**	**2**	**3**	**4**	**5**	**6**	**7**
1	0.127995	-0.836024	-1.120090	-1.601550	-1.57311	0.837367	1.375225

Table 6. Classification table for neural network model trained with backpropagation algorithm with momentum coefficient.

			estimated values			
			hypertension			
			no	yes	total	**accuracy (%)**
Observation values	hypertension	no	63	19	82	76.8
		yes	12	131	143	91.6
total			75	150	225	**86.2**

As given in Table 6, 82 of 225 patients were not actually hypertensive. The remaining 143 patients are hypertensive. 63 of 82 patients without hypertension were classified as correct and 19 of them as incorrect in the classification process made with the neural network model trained with backpropagation algorithm with momentum coefficient, with an accuracy percentage of 76.8%. 131 of 143 patients with hypertension classified as correct and 12 as incorrect, with an accuracy rate of 91.6%.

In the classification process made with the neural network model trained with backpropagation algorithm with momentum coefficient, the overall accuracy value was calculated as 86.2% by correctly classifying 194 of 225 patients.

4.2.2 *Classification by the conjugate gradient backpropagation algorithm*

The weights calculated after training the network with the conjugate gradient backpropagation algorithm are given in Table 7.

By using the weights calculated after the network is trained, the network output is calculated with the Equation (31). As in the neural network model trained with backpropagation algorithm with momentum coefficient, the $\hat{y}$ values are assigned by giving a value “0” to those less than 0.5 and a value of “1” to those greater than 0.5 for the classification process. These assignment values were compared with the observation values and the correct classification rates were calculated. The accuracy percentages calculated in the classification process for the data set used after the network was trained are given in Table 8.

Table 7. The weights calculated after training the network with the conjugate gradient backpropagation algorithm.

b_{ij}	**weights between input layer and hidden layer**						
j / i	**1**	**2**	**3**	**4**	**5**	**6**	**7**
1	0.50044	-1.730519	-0.432385	-2.212496	-0.310118	-0.536704	-0.797588
2	-0.089624	1.59597	-1.718343	0.194532	-3.885706	-1.279213	-1.481379
3	0.073586	0.176147	-2.741652	0.638694	-1.827383	-0.964285	-2.103983
4	-0.650837	-0.115309	-3.471644	0.639703	0.244935	-0.653154	-1.481175
5	-0.192319	-0.979086	1.14439	1.29368	-0.984991	0.506407	-1.323641
6	0.119219	1.328376	-0.461388	0.72299	-1.144382	0.045564	-0.932604
7	-0.597068	0.609235	-2.262588	-0.§68023	0.341118	0.071685	-0.917533
8	-0.831742	-0.473473	0.386028	0.55305	-0.576316	-1.043114	-1.102681
9	0.520319	-1.427755	-2.942906	-0.360444	-1.911586	-0.921742	-1.922553
10	-0.264529	0.101774	-2.476718	-1.508757	0.731404	-1.119844	-0.99558
11	0.088826	0.331771	-4.073246	-0.216949	0.621034	-0.121464	-2.126909
12	-0.897644	0.109378	-0.875451	-0.716155	1.556789	-0.189914	-1.367429
13	-0.307702	-1.136922	3.832603	-1.293699	0.138086	-1.617161	-0.455787
14	-1.509156	0.012926	-2.93491	0.230133	-2.5357	-0.735346	-2.920688
15	0.454645	1.265555	0.10662	0.764483	-1.667776	-0.019351	-1.419592
c_{jk}	**weights between hidden layer and output layer**						
j / k	**1**	**2**	**3**	**4**	**5**	**6**	**7**
1	1.180347	-2.461950	-5.509069	-2.862042	-4.694672	0.832300	-1.448453

Table 8. Classification table for neural network model trained with conjugate gradient backpropagation algorithm.

			estimated values			
			hypertension			
			no	yes	total	**accuracy (%)**
Observation values	hypertension	no	60	22	82	73.1
		yes	4	139	143	97.2
total			64	161	225	**88.4**

As given in Table 8, 82 of 225 patients were not actually hypertensive. The remaining 143 patients are hypertensive. 60 of 82 patients without hypertension were classified as correct and 22 of them as incorrect in the classification process made with the neural network model trained with conjugate gradient backpropagation algorithm, with an accuracy percentage of 73.1%. 139 of 143 patients with hypertension classified as correct and 4 as incorrect, with an accuracy rate of 97.2%.

In the classification process made with the neural network model trained with conjugate gradient backpropagation algorithm, the overall accuracy value was calculated as 88.4% by correctly classifying 199 of 225 patients.

5 Conclusions

The data set of 225 patients who consulted the Internal Medicine polyclinic of OGU Health, Application and Research Hospital was first classified by logistic regression analysis, and then by artificial neural network models trained with backpropagation algorithms. The accuracy percentages calculated as a result of the application are given in Table 9.

Table 9. Accuracy percentages calculated as a result of the application

	hypertension		
	no	yes	
	accuracy (%)	accuracy (%)	accuracy (%)
Logistis regression analysis	52.4	85.3	73.3
Neural network model trained with the backpropagation algorithm with momentum coefficient	76.8	91.6	86.2
Neural network model trained with the conjugate gradient backpropagation algorithm	73.1	97.2	88.4

When Table 9 is examined, it is seen that the overall accuracy percentages in the correct classification of the data are calculated as 73.3% for the logistic regression analysis, 86.2% for the neural network model trained with the backpropagation algorithm with momentum coefficient, and 88.4% for the artificial neural network model trained with the conjugate gradient backpropagation algorithm.

According to these results, it has been observed that classification with appropriate neural network models in binary classification problems tends to give better results than classification with logistic analysis, which is frequently used in the field of statistics.

Since the activation function applied to the units of the hidden layer in multi-layer neural network models is a nonlinear function, these models are real nonlinear models. Accordingly, classifications made with multi-layer neural network models give better results. Based on this result, the multinomial logistic regression model and the appropriate multi-layer neural network model can be compared in further studies to investigate the usability and effectiveness of neural network models in classification problems where the dependent variable has more than two categories.

In addition, statistical models also affect the development of artificial neural network models. For this reason, it is more beneficial to use statistical models and artificial neural network models together. In this respect, the mutual application of both fields is one of the important issues and requires further research.

Acknowledgment

This study was produced from the master's thesis named “Sinir Ağları ile İstatistiksel Modelleme ve Bir Uygulama Denemesi”, completed by Dr. Halil Eryılmaz, at Anadolu University, Institute of Science and Technology, Department of Statistics, in 2004.

References

[1] Eryılmaz H. Sinir Ağları ile İstatistiksel Modelleme ve Bir Uygulama Denemesi. Yüksek Lisans Tezi, Anadolu Üniversitesi, Eskişehir, Türkiye, 2004.

[2] Zizi Y, Jamali-Alaoui A, Goumi BE, Oudgou M & Moudden AE. "An Optimal Model of Financial Distress Prediction: A Comparative Study between Neural Networks and Logistic Regression" *Risks, MDPI,* 9(11), 2021.

[3] Stern HS. “Neural Networks in Applied Statistics”. *Technometrics*, 38(3), 205–214, 1996.

[4] Ripley BD. *Statistical Aspects of Neural Networks.* Editors: Barndorff-Nielsen OE, Jensen JL, Kendall WS. Networks and Chaos–Statistical and Probabilistic Aspects, 105–126, Great Britain, Chapman & Hall, 1993.

[5] Halaç A, Erbil T, Falay T. “İnsan Zekasına Sayısal Rakip Yapay Zeka”. *PC Net*, 59, 2002.

[6] Akpınar H. Yapay Sinir Ağları ve Kredi Taleplerinin Değerlendirilmesinde Bir Uygulama Önerisi. Yüksek Lisans Tezi, İstanbul Üniversitesi, İstanbul, 1993.

[7] Özden K & Tahsin A. Line Balancing Based on Error Rate Estimation with Artificial Neural Networks in Assembly Line Operations. *İstanbul Gelişim Üniversitesi Sosyal Bilimler Dergisi*, 10(1), 16-32, 2023.

[8] Freemann JA, Skapura DM. *Neural Networks: Algorithms, Applications and Programming Techniques*. New York, USA, Addison-Wesley Publishing Company, 1991.

[9] Elmas Ç. *Yapay Sinir Ağları*, Ankara, Türkiye, Seçkin Yayıncılık, 2003.

[10] Öztemel E. *Yapay Sinir Ağları*. İstanbul, Türkiye, Papatya Yayıncılık, 2003.

[11] Fausett LV. *Fundamentals of Neural Networks Architectures, Algorithms, and Applications*. New Jersey, USA, Printice-Hall, 1994.

[12] Montesinos López OA, Montesinos López A, Crossa J. “*Multivariate Statistical Machine Learning Methods for Genomic Prediction*”, Cham: Springer; 2022.

[13] Hertz J, Krogh A, Palmer RG. *Introduction to the Theory of Neural Computation*. Redwood City, CA: Addison-Wesley, 1991.

[14] Bishop CM. *Neural Networks for Pattern Recognition*. Oxford, UK, Clarendon Press, 1995.

[15] Wang S. “An Adaptive Approach to Market Development Forecasting”. *Neural Computing & Applications*, 8(1), 3–8, 1999.

[16] Kolmogorov AN, "On the Representation of Continuous Functions of Many Variables by Superposition of Continuous Functions of One Variable and Addition", *Doklady Akademii. Nauk SSSR*, 114(5), 953-956, 1957.

[17] Issitt RW, Cortina-Borja M, Bryant W, et al. “Classification Performance of Neural Networks Versus Logistic Regression Models: Evidence From Healthcare Practice. *Cureus* 14(2): e22443, 2022.

[18] Warren SS. “Neural Networks and Statistical Models”. *Proceedings of the Nineteenth Annual SAS Users Group International Conference*, Dallas, Texas, 10-13 April 1994.

[19] Rawlings JO. *Applied Regression Analysis: A research Tool*, California, USA, Wadsworth & Brooks, 1988.

[20] Montgomery DC. *Introduction to Linear Regression Analysis*. 3rd ed. New York, USA, J.Wiley-Interscience, 2001.

[21] Smita M. “Logistic Regression Model For Predicting Performance of S&P BSE30 Company Using IBM SPSS” *International Journal of Mathematics Trends and Technology*, 67(7), 118-134, 2021.

[22] Şahin S, Pehlevan Özel H, Yuksek Y & Tütüncü T. The Predictive Factors of Malignancy in Follicular Lesion of Undeterminated Significance. *Muğla Sıtkı Koçman Üniversitesi Tıp Dergisi*, 10(1), 15-18, 2023.