Zanco Journal of Human Sciences (ZJHS)

TYPE: Original article
DOI https://doi.org/10.21271/zjhs.29.4.13

OPEN ACCESS

*Corresponding author

Nazeera Sedeeq Kareem Barznji
nazeera.kareem@su.edu.krd

RECEIVED :11 /09/2024
ACCEPTED :06/01/ 2025
PUBLISHED :15/08/ 2025

Prediction by Sigmoid Multilayer Perceptron Artificial Neural Network Function and Model Selection for the Risk Factors Most Affected Human Immunodeficiency

Nazeera Sedeeq Kareem Barznji / Department of Statistics and Informatics, College of Administrations and Economics, Salahaddin University-Erbil, Kurdistan Region,Iraq

Keywords:
Artificial Neural Network, Model Selection, Multilayer Perceptron, Risk Factors of the Immunodeficiency

Abstract

The Multilayer Perceptron (MLP), a widely recognized type of Artificial Neural Network (ANN), was applied in this study to forecast the risk factors associated with human immunodeficiency conditions. A sample of 500 patients with various diseases was collected from hospitals and laboratories in the Kurdistan region. Each patient's immune level was tested, and the dataset included one dependent variable, immune testing level (classified as either "good immunity" or "poor immunity"), and six independent variables representing potential risk factors (X1 to X6). Statistical analyses, including parameter estimation and variable importance ranking, revealed that X1: Genetic history had the most significant influence on immunity, followed by X5: Cancer treatments such as radiation therapy, X4: AIDS, X2: Diabetes, X3: Human Immunodeficiency Virus (HIV), and lastly, X6: Certain medications. Model selection criteria such as the Akaike Information Criterion (AIC) and Bayesian Information Criterion (BIC), along with the Likelihood Ratio Test and Chi-square test (p-value = $0.011 < 0.05$), a that these risk factors significantly affect immune deficiency outcomes. The results validate the effectiveness of the MLP model in identifying the most influential predictors of immunodeficiency.

1. Introduction

Human immunodeficiency diseases are defects of the immune system that result in poor immune defenses. They are caused by failures in various immune components, including lymphocytes, phagocytes, and the complement system. These conditions can be classified as primary, such as Bruton's disease, or secondary, such as HIV infection (Notarangelo et al., 2016, pp. 1099–1105).

Primary immunodeficiency diseases are due to intrinsic defects in immune cells, including T cells, complement components, and phagocytes. Recurrent pneumonia caused by extracellular bacteria is often a sign of antibody deficiency, while recurrent fungal infections may point to abnormalities in T lymphocyte function. Severe immunodeficiency diseases are life-threatening, with many affected children dying within the first two years of life. These conditions are more common in males, with over 50% of cases linked to mutations in genes on the X chromosome (Vaillant and Qurie, 2023, p. 6).

In theory, artificial neural network systems can approximate any mapping function given an appropriate number of nodes and connections (Hornik, Stinchcombe, and White, 1989). Therefore, they do not have inherent advantages or disadvantages over traditional machine learning systems in terms of representation or learning. A multilayer perceptron (MLP) consists of multiple layers, including an input layer, one or more hidden layers, and an output layer, where each layer contains a set of processing units called neurons. Figure 1 shows an MLP with two hidden layers (one input layer and one output layer). During training, each node exhibits certain activation patterns. The input layer contains *n* input variables, and the output layer contains *m* output variables. Both the backpropagation algorithm and induction methods do not rely on assumptions about the distribution of training data samples used in the learning process. Therefore, many of the limitations associated with statistical analysis do not apply when using backpropagation or induction learning algorithms. In addition to similarity-based learning methods, artificial neural networks are often used for similar purposes (Piramuthu, Shaw, and Gentry, 1990, p. 6).

2. Literature Review

Chai, Cheah, Goh, Chang, Shen, and Wenzhen (2021) conducted a study analyzing a multilayer perceptron (MLP) neural network model for the classification of hypertension in adolescents using simple anthropometric and sociodemographic data collected in a cross-sectional study in Sarawak, Malaysia. Out of 2,461 individuals surveyed, 741 (30.1%) were identified with hypertension, while 1,720 (69.9%) were categorized as normal. Eleven anthropometric and sociodemographic indicators were collected during the data collection process. The results illustrated that combining simple anthropometric measurements with sociodemographic data enabled effective classification of adolescent hypertension using the developed MLP model. However, to improve the model's positive predictive value—particularly at low prevalence—a model with higher sensitivity and specificity is required. The researchers concluded that this classification model may serve as a quick and simple preliminary screening tool for identifying adolescents at high risk of hypertension.

Mfitum and Ngo (2024) explored the advancement of renewable energy technologies, with a particular focus on the solar energy sector, as a solution to the energy shortage challenges in Central Africa. The study acknowledged the complexity of predicting solar irradiance due to climate change and variability in weather-related factors. It utilized a neural network to estimate solar radiation on sloped surfaces, factoring in climate variables such as temperature, wind speed, humidity, and air pressure. The model achieved a correlation coefficient of 98.883% between observed and estimated solar irradiance values, as determined by the hidden layer neurons. The authors recommend this approach for estimating solar irradiance intensity at the study site and other regions with similar climatic conditions. A logistic sigmoid activation function was employed using 50 neurons in the hidden layer.

Saravanan and Venkateswaran (2015) presented a study in which background data on medical testing and population characteristics were collected through annual antenatal checkups in the Vellore district of Tamil Nadu. The data were used to predict HIV infection status. A neural network model, specifically a multilayer perceptron with a logistic activation function, was trained using a cross-entropy error function under a probabilistic framework. Prediction performance was evaluated using sensitivity and specificity metrics. Additionally, the study employed an autoassociative neural network to handle missing data, which was imputed using global optimization methods. The effect of these imputed values on the model's predictive performance was also assessed.

3. Objective

In this study, artificial intelligence (AI), specifically the multi-layer artificial neural network method, was used to predict the risk factors causing immunodeficiency, their impact, and the relevance of the results to medical conclusions.

4. Theoretical Aspect

4.1 Artificial Neural Networks (ANN)

In machine learning, a neural network, also called an artificial neural network (ANN) or simply a neural net, is a model inspired by the structure and function of biological neural networks. It mimics how the human brain processes information by using layers of interconnected neurons to recognize patterns, make decisions, and learn from data. In statistics, a neural network is considered a machine learning algorithm that operates in a manner analogous to the human brain, replicating how biological neurons work together to identify events, evaluate options, and generate responses (Al Qadi, 2021, pp. 1–2).

The conceptual foundation for artificial neural networks can be traced back to the early 1940s, with contributions by Donald Hebb, Warren McCulloch, and Walter Pitts. These early models laid the theoretical groundwork for later developments. According to D'Agostino (2021, p. 1), neural networks consist of a large number of highly connected processing elements (neurons) that work collectively to solve specific problems in regression and classification.

Biological neurons are modeled in artificial systems through simplified components. Dendrites receive input from other neurons through synapses, which are the points of communication. These inputs are processed in the soma (cell body), and if the input signal exceeds a certain threshold, an output is generated and transmitted along the axon to subsequent neurons. The axon is a long, tube-like structure that functions as a transmission line. At its endpoint, complex structures called synapses allow the signal to pass to the next neuron. This biological analogy informs the design of artificial neural networks, where each neuron receives weighted inputs, applies a transformation, and passes the output to other neurons in the network (Al Qadi, 2021, p. 2).

There are approximately 100 billion neurons in the human brain, each with thousands of connections to other neurons, forming a highly intricate and dynamic system.

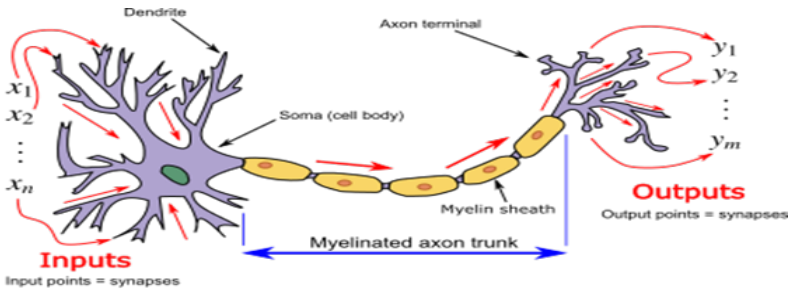

Figure 1: A biological neuron

But neural networks require minimal commaneds on model structure and assumptions, it is helpful to realize the general network constractionit is known as a feedforward planing because the relationships in the network flow forward from the input layer contains of one or more input neurons, which receive input signals from the external world or from other layers of the neural network contains the predictors to the output layer which it contains the reaction, and a hidden" layer contains unvisible nodes, or units, the value of each hidden unit, Common activation functions resolves the output of the perceptron depend on the weighted sum of the inputs and the bias term it used in perceptron include the step function. Neural networks can have more than one hidden layer. Basically, each node contains a mathematical formula, with each variable within the formula weighted differently, and each input neuron is related to a weight, which represents the strength of the connection between the input neuron and the output neuron. If the output of persisting that mathematical formula to the input exceeds a certain threshold, the node passes data to the next layer in the neural network.

4.1.1 Types of Artificial Neural Network Models:

1. Single Layer Perceptron.
2. Radial Basis Functional Neural Network, Feed Forward Neural Network.
3. Multilayer Perceptron.
4. Recurrent Neural Network.
5. Boltzmann machine
6. Hopfield Network
7. LSTM – Long Short-Term Memory.

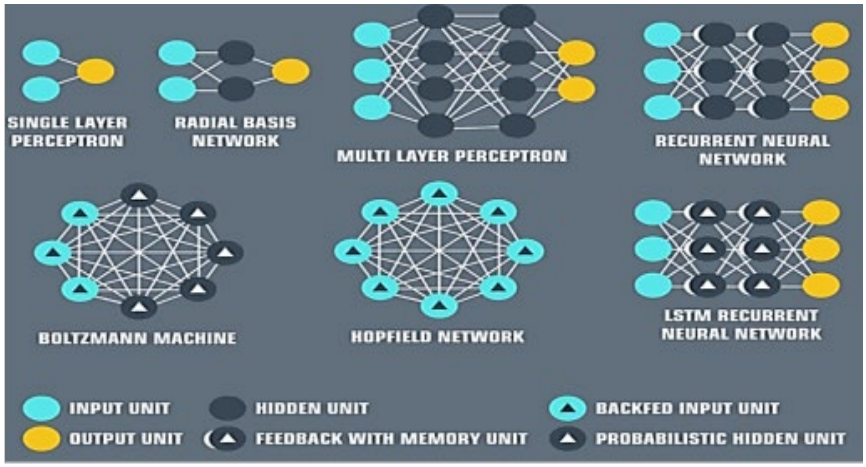

Figure 2: Types of Artificial Neural Network Models
(Hounmenou, Gneyou and Glèlè Kakaï, 2021, p. 3)

4.2 Multilayer Perceptron

4.2.1 Perceptron

It is a simple kind of neural network that can learn to classify linearly independent patterns. The fundamental building block of a neural network is the perceptron, which consists of one or more input nodes, weights associated with each input, a summation function to combine the inputs and weights, and an activation function to produce the predicted output (Rosenblatt, 1961; Sejnowski, 2018).

4.2.2 Limitations of the Perceptron in AI

1. The output of a perceptron can only be a binary number (0 or 1) due to the hard-edge transfer function.

2. It can only be used to classify the linearly separable sets of input vectors. If the input vectors are nonlinear, it is not easy to classify them correctly.

4.2.3 Types of perceptron

1. Single-Layer Perceptron is a linear function classifier and represents the simplest type of feedforward neural network (Block, 1962, pp. 123–135).

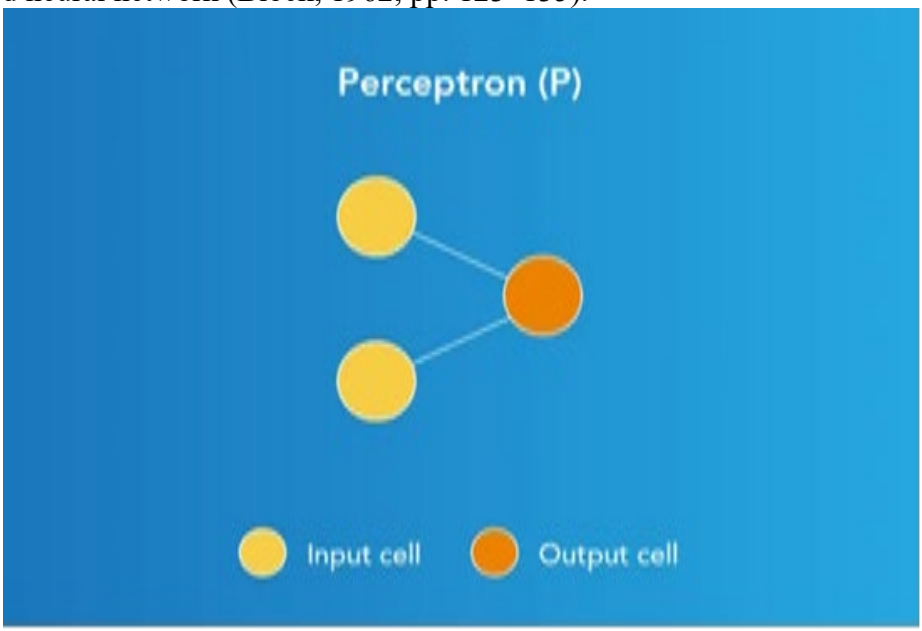

Figure 3: A Single-Layer Perceptron

2. The Multilayer Perceptron is a more complex type of neural network compared to the single-layer perceptron. It is a fully connected feedforward artificial neural network composed of at least three layers: an input layer, one or more hidden layers, and an output layer. Except for the input nodes, each node in the network functions as a neuron and utilizes a nonlinear activation function.

MLPs are typically trained using a supervised learning algorithm called backpropagation, which allows the network to learn and classify data that is not linearly separable. This means MLPs can model complex, non-linear relationships between inputs and outputs. The use of nonlinear activation functions, such as the sigmoid or tanh, enables the network to approximate any continuous function given sufficient data and layers (Rosenblatt, 1961; Rumelhart, Hinton, and Williams, 1986; Cybenko, 1989; Chan and Muhammad, 2023).

4.3 Basic Components of Artificial Neural Networks -Multi-layered Perceptron Model

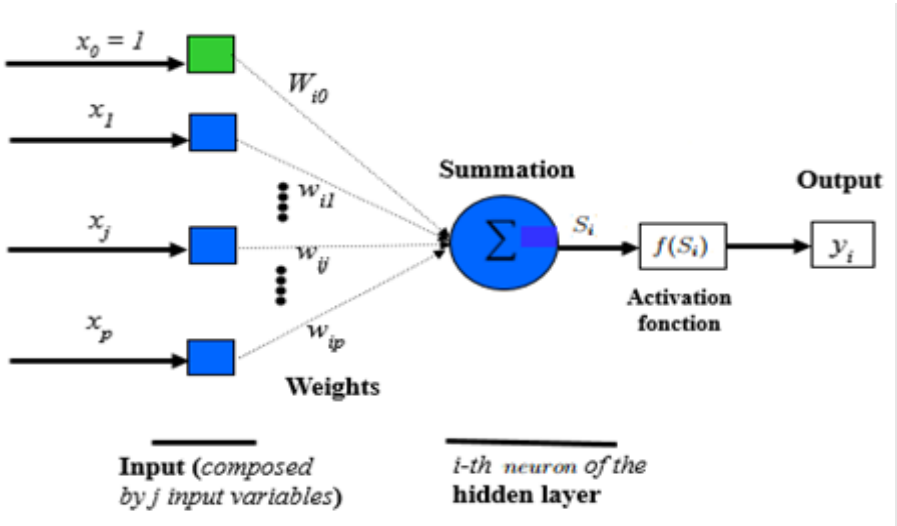

Figure 4: Diagram of Components of Artificial Neural Networks -Multi-layered Perceptron

4.3.1 Layers

The network comprises three principal components: an input layer, a hidden layer (or processing layer), and an output layer.

a. Input Layer:

The first layer is known as the input layer which accepts input for the network, The input layer are the input variables they referred to the visible layer, which includes multiple predictor variables, which accepts input for the network, with each variable corresponding to an independent neuron. At the input of hidden unit l, the input vector x is weighted by input weights W^h..

$$S_l = [X_1 W^h_{1\,l}. X_2 W^h_{2\,l}. \dots . X_n W^h_{n\,l} \dots . X_N W^h_{N\,l}] \quad \dots\dots\dots\dots (1)$$

Where: n is the index of input; l is the index of hidden units; X_n is the n^{th} input; $W^h_{n.l}$ is the input weight between the input and hidden unit; is the input weight. RBF network with N inputs, L hidden units, and M outputs (Xie et al. 2011).

b. Hidden Layer or a Processing Layer:

Comprises a specific number of nodes with RBF non-linear activation function (conventionally, it is implemented as a Gaussian function), and the last layer is responsible for generating the final output of the network. The layers of nodes situated between the input and output layers are referred to as hidden layers. There can exist one or multiple instances of these layers. The hidden layer is responsible for processing the outputs derived from the input layer, utilizing a distance metric of

$$d_l = \sqrt{\sum_{j=1}^{n} k^i_{1_l} [I_j - m^i_j]^2} \quad \dots\dots\dots\dots (2)$$

and the transfer functions in. The resultant output of the network is represented as a weighted summation of the outputs of φ(di) emanating from the hidden layer and is articulated by the hidden layer, which comprises a Radial Basis Function (RBF) centered at a specific point, contingent upon the dimensionality of the input/output predictor variables. The activations of the hidden units are described by fundamental functions that typically encompass the activation function of the Radial Basis Function. This activation function is contingent upon the Euclidean distance between the input pattern vector and the center of the hidden neuron, which possesses a multitude of neurons. The selection of neurons within this layer presents a considerable challenge. Mao and Huang have proposed a technique based on a data structure-preserving criterion for the selection of neurons in the hidden layer. The configuration of the hidden layer for RBF networks can be effectively realized through clustering methodologies. The function of clustering in the architecture of Radial Basis Function Networks (RBFN) is to establish an initial distribution of receptive fields (hidden neurons) throughout the input space of the input variables. Each neuron's output within the hidden unit l is computed by

$$\varphi_l(\overline{s_l}) = \exp\left(\frac{-\|S_l - C_l\|^2}{\sigma_l}\right) \quad \dots\dots\dots\dots (3)$$

where the activation function$\varphi_l(.)$ for the hidden unit is conventionally selected as a Gaussian function: c is the center of the hidden unit,1 and $\sigma_1$is the width of the hidden unit$\|.\|$, with denoting the Euclidean distance. This indicates$\varphi_l(s_l)$ that possesses a feasible value solely when the distance$\|s_1 - c_1\|$ is less than the width (Dhubakarya et al., 2010).

c. Output Layer:

The output layer is constituted of nodes that generate the output variables, which typically include a linear activation function that subsequently transmits the values to each neuron. The computation of the network output m is performed by:

$$o_m = \sum_{l=1}^{L} \varphi_l(s_l) W^o_{l,m} + W^o_{o,m} \quad \dots\dots\dots\dots (4)$$

Where: m represented the index of output; $W^o_{1,m}$ is the output weight between the hidden unit 1 and the output unit; $W^o_{0,m}$is the bias weight of output unit m.

One may observe from this fundamental computation the presence of four distinct types of

parameters: the input weight matrix W^h, the output weight matrixW_0, the center matrix c, and the width vectorσ. Typically, the input weights are uniformly initialized to the value of '1' [Xie et al., 2011].

4.3.2. Weights

These are the real values related to the traits. They are significant as they tell the prominence of each trait, which is threaded as an input to the artificial neural network.
Neurons in neighboring layers are fully related to each other. Each connection has a connected weight, which decides the strength of the connection. These weights are literated during the instruction operation. [Sejnowski, Terrence J. (2018). p. 47] [Andrea D'Agostino(2021) ,p1]

4.3.3. Bias Neurons

In a neural network, a bias neuron helps shift the activation function either to the left or right, thereby enabling the model to better fit the data. In addition to the input and hidden neurons, each layer (except the input layer) typically includes a bias neuron. This neuron provides a constant input to the neurons in the following layer and has its own associated weight, which is adjusted during training.

The term bias in machine learning can refer to two distinct but related concepts: one is the bias neuron itself, and the other is the statistical bias in model assumptions. Statistical bias reflects the error introduced by approximating a real-world problem using a simplified model. High bias indicates underfitting, when a model is too simplistic to capture underlying patterns, while low bias suggests that the model accurately reflects the data trends (D'Agostino, 2021, p. 1).

4.3.4. Summation Function

In a neural network, a summation function is responsible for calculating the combined input to a neuron. It does this by summing the products of each input value (also called traits or features) and their corresponding weights, then adding a bias term. This results in a single numeric value that reflects the total influence of all inputs to that neuron.

In simpler terms, the summation function aggregates the effects of the input features, each weighted by its importance, and adjusts this total using the bias. The result is then passed into an activation function, which determines the neuron's output. The summation function is a foundational component in every neuron, allowing the network to integrate and process information before applying non-linearity.

4.3.5 Activation Function

It is desired to add non-linearity to the neural network function. The activation function "links" the weighted sums of elements in a layer to the merits of elements in the passing layer. Usually, each neuron in the hidden layers and the output layer requires an activation function on its weighted sum of inputs. General activation functions consist of sigmoid, tanh, ReLU (Rectified Linear Unit), and softmax. These functions established nonlinearity into the network, permitting it to learn complex samples in the data. Activation function determines if a neuron should be activated or not by counting the weighted sum and adding bias to it. The purpose is to introduce non-linearity into the output of a neuron. If we do not consider activation function, then the output would be only a linear function (one-degree polynomial). Now, a linear function is simple to explain, but they are restricted in its difficulty, has reduced power. Without an activation function, our function model cannot learn and involve data such as images, videos, audio, speech, etc. In MLPs, some neurons use a nonlinear activation function designed to model the delay or threshold-like behavior - sometimes referred to as 'spike' - of biological neurons.

4.3.5.1 Sigmoid Activation Function

The sigmoid activation function, also known as the logistic function, is a mathematical function characterized by a distinctive "S"-shaped curve (sigmoid curve) that ranges between 0 and 1. It is commonly used in models where the output is interpreted as a probability (Cybenko, 1989, pp. 303–314).

Each element of the z vector becomes an input for the sigmoid function (Han and Morag, 1995, pp. 195–201)

$$\boldsymbol{a_m = \sigma(z_m) = \frac{1}{1+e^{zm}}} \qquad \text{..... (5)}$$

The output of $\sigma(\boldsymbol{z_m})$ is another m dimensional vector a, one entry for each unit in the hidden layer $\boldsymbol{a} = \begin{bmatrix} a_1 \\ a_2 \end{bmatrix}$

Here, **a** stands for "activation", which is a usual way to refer to the output of hidden units. This sigmoid function, "coating" the outcome of the linear function, is usually called an activation function. The plan is that a unit obtains "activated" in extra or smaller the same method that a neuron obtains activated when an adequately powerful input is accepted. The choice of a sigmoid is random. Many various non-linear functions could be chosen at this level in the network, like a Tanh or a ReLU. Unluckily, there is no moral way to select activation functions for hidden layers. It is mainly a case of trial and error. A good attribute of sigmoid functions is that they are 'mainly linear', but they saturate as they reach 1 and 0 in the alternative. Commonly, a sigmoid function is fixed and has a first derivative. Thus, the cumulative distribution functions for several general probability distributions are sigmoidal. This example is the error function, which is connected to the cumulative distribution function of a normal distribution; another is the arctan function, which is associated with the cumulative distribution function of a Cauchy distribution. In biochemistry and pharmacology, the Hill and Hill–Langmuir equations are sigmoid functions. The sigmoid function is concave (bent downward) for input values smaller than a certain point and convex (bent upward) for values larger than that point, which is typically at zero (Han and Morag, 1995, pp. 195–201).

Figure (5) in below shows Some Sigmoid Activation Function compared. In the drawing all functions are normalized in such a way that their slope at the origin is 1.

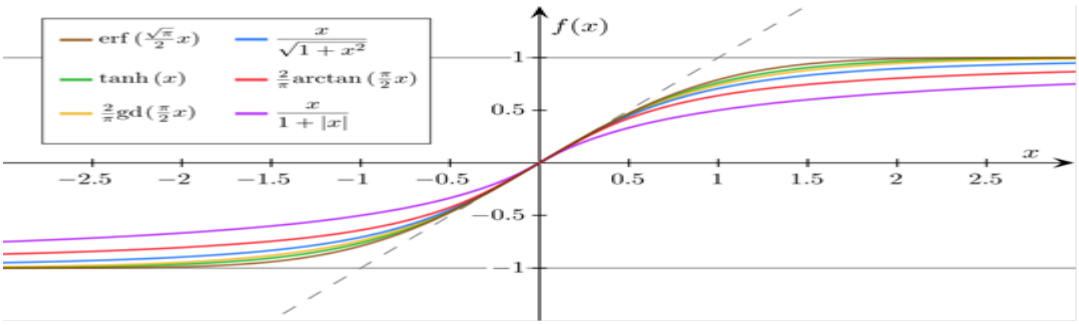

Figure (5) Some Sigmoid Activation Function Han, Jun; Morag, Claudio (1995), pp. 195–201

4.4. Linear Function

The linear gathering function is identical to that in the perceptron and ADALINE. But, with a pair of variabilities that revise the notation, now we are conducting multiple layers and dealing with units. The usual method to indicate this is with linear algebra notation. "This is not a course of linear algebra; the reason I won't pall the mathematics in descriptions. However, it establishes enough ideas and records to recognize the basic working engaged in the neural network computation" (Amin and Al, 2017, pp. 7–8).

if

$$\boldsymbol{x} = \begin{bmatrix} x_1 \\ x_2 \\ \dots \\ x_n \end{bmatrix} = (x_n) \in \mathbb{R}^n \qquad \text{.....(7)}$$

and

$$W = \begin{bmatrix} w_{11} & w_{12} & \dots & w_{1n} \\ w_{21} & w_{22} & \dots & w_{2n} \\ \dots & \dots & \dots & \dots \\ w_{m1} & w_{m2} & \dots & w_{mn} \end{bmatrix} = (w_{mn}) \in \mathbb{R}^{m \times n} \quad \text{..... (8)}$$

The input vector for our first training:

$$\boldsymbol{X} = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} \in \boldsymbol{R} \quad \text{..... (9)}$$

Since we have 3 input units relating to hidden 2 units, we have 3x2 weights. This is indicated with a matrix as:

$$W = \begin{bmatrix} w_{11} & w_{12} \\ w_{21} & w_{22} \\ w_{31} & w_{32} \end{bmatrix} \in \boldsymbol{R} \quad \text{..... (10)}$$

The result of the linear function equals the reduplication of the vector x and the matrix W. To execute the reduplication in this situation, we want to transpose the matrix W to contest the number of columns in W with the number of rows in x. Transposing means to "flip" the columns of W such that the first column befit the first row, the second column becomes the second row, and so forth.

The matrix-vector reduplication equvalents to $\boldsymbol{z = W^T * X} = \begin{bmatrix} \boldsymbol{w_{11}} & \boldsymbol{w_{12}} & \boldsymbol{w_{13}} \\ \boldsymbol{w_{21}} & \boldsymbol{w_{12}} & \boldsymbol{w_{23}} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}$

.....(11)

The preceding matrix process in collection equals:

$$\boldsymbol{z = W^T * X} = \begin{bmatrix} \boldsymbol{x_1 w_{11}} * + & \boldsymbol{x_2 w_{12}} * + & \boldsymbol{x_3 w_{13}} * \\ \boldsymbol{x_1 w_{21}} * + & \boldsymbol{x_2 w_{12}} * + & \boldsymbol{x_3 w_{23}} * \end{bmatrix} = \begin{bmatrix} \boldsymbol{z_1} \\ \boldsymbol{z_2} \end{bmatrix} \quad \text{..... (12)}$$

$$z_m = f(x_n, w_{mn}) = b + \sum_m x_n w_{mn} \quad \text{..... (13)}$$

Here, f is a function of each component of the vector x and each unit of the matrix W. The m indicator recognizes the rows in WT and the rows in z. The n indicates the columns in WT and the rows in x

Attention that we attach a b bias term, which has the role to facilitate learning a correct threshold for the function. If the "Linear gathering function" section is here.

Shortly, the linear function is a weighted quantity of the inputs plus a bias.

4.5. Designs of the artificial neural network

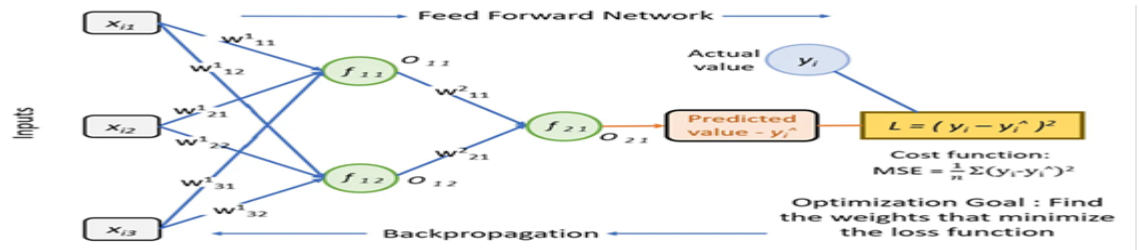

Figure (6) Branches of all Neural Networks
(Chan, Abu-Salih and Qaddoura et al., 2023, pp. 3–18)

4.5.1 Forward Propagation

The feedforward neural network (FFNN) is the earliest design of artificial neural networks. It is the simplest type of ANN, where information flows in one direction only, from input to output. There are no cycles or loops in the network architecture. Multilayer Perceptrons (MLPs) are a type of feedforward neural network (Inoussah Moungnutou Mfetoum Simon Koumi Ngoh, 2024, p. 6).

The forward propagation phase requires "chaining" all the steps we determined so far: the linear function, the sigmoid function, and the threshold function. Can be performed as

$$\hat{y} = T\left(\sigma^{(2)}\left(\lambda^{(2)}(\sigma^{(1)}(\lambda^{(1)}(x_n . w_{mn}))\right)\right) \quad \text{.....(14)}$$

All neural networks can be constituted as a structure of functions where every step is nested in the next step. For example, we can add an additional hidden layer to the network by:

$$\hat{y} = T\left(\sigma^{(3)}(\lambda^{(3)}(\sigma^{(2)}\left(\lambda^{(2)}(\sigma^{(1)}(\lambda^{(1)}(x_n . w_{mn}))\right)\right)\right) \quad \text{..... (15)}$$

4.5.2 Backpropagation

Backpropagation is an algorithm widely used in artificial intelligence and machine learning to train artificial neural networks by modifying errors. The algorithm works by minimizing the loss function, which measures the difference between the predicted output and the actual input. When Multilayer Perceptrons (MLPs) are trained using backpropagation, the algorithm computes gradients of the loss function with respect to the model parameters and iteratively updates these parameters to reduce the error (Han and Morag, 1995, pp. 195–201).

4.5.2.1 Backpropagation for Single-Unit Multilayer Perceptron

During the backward pass, the weights and biases are updated according to the model's requirements. The algorithm propagates the error from the output layer backwards to adjust parameters. A multilayer perceptron has high processing power and can handle both linear and nonlinear patterns. Designing the backpropagation computations can be complex, so simplifying assumptions can be made by considering a network with a single input unit, a single hidden unit, and a single output unit, as shown in Figure 4. We first derive backpropagation for this simplified network and then extend it to networks with multiple neurons (Rumelhart, Hinton, and Williams, 1986, pp. 533–536).

To understand how the error changes when we adjust the weights in a multilayer perceptron, consider the following points:

1. The error EE depends on the output of the sigmoid activation function aa.
2. The output aa of the sigmoid function depends on the linear combination zz of inputs and weights.
3. The linear function zz itself depends on the weights ww.

Thus, the multilayer perceptron can be seen as a composite function, where the output is a function of functions. According to the chain rule of calculus, the derivative of this composite function is found by multiplying the derivative of the outer function by the derivative of the inner function, applied recursively.

The derivative of the error with respect to the sigmoid activation function is:

$$\frac{\partial E}{\partial w^{(L)}} = a^{(L)} - y \quad \text{..... (16)}$$

Next, the derivative of the sigmoid activation function w.r.t the linear function is:

$$\frac{\partial a^{(L)}}{\partial z^{(L)}} = a^{(L)}(1 - a^{(L)}) \quad \text{..... (17)}$$

Finally, the derivative of the linear function w.r.t the weights is:

$$\frac{\partial z^{(L)}}{\partial w^{(L)}} = a^{(L-1)} \quad \text{..... (18)}$$

If we put all the pieces together and replace we obtain:

$$\frac{\partial E}{\partial w^{(L)}} = a^{(L)} - y \, X \, a^{(L)} \, (1 - \, a^{(L)}) \, X \, a^{(L-1)} \quad \text{……. (19)}$$

At this point, we have figured out how the error revision occurs as we change the weight relating the hidden layer and the output layer, w(L).Surprising advance. May still want to know how the error changes as we regulate the weight connecting the input layer and the hidden layer w(L−1). Luckily, this is fine. Simple may stratify the chain-rule again, and again until we get there

$$\frac{\partial E}{\partial w^{(L-1)}} = \frac{\partial E}{\partial a^{(L)}} X \frac{\partial a^{(L)}}{\partial z^{(L)}} X \frac{\partial z^{(L)}}{\partial a^{(L-1)}} X \frac{\partial a^{(L-1)}}{\partial z^{(L-1)}} X \frac{\partial z^{(L-1)}}{\partial w^{(L-1)}} \quad \text{…….(20)}$$

by, replacing with the real derivatives becomes:

$$\frac{\partial E}{\partial w^{(L-1)}} = a^{(L)} - y \, a^{(L)} \, (1 - a^{(L)}) \, w^{(L)} \, a^{(L-1)} \, (a1 \, -^{(L-1)}) \, x^{(L-1)} \quad \text{……. (21)}$$

The derivative of the error with regard to the bias term b. There are two methods to reach this. One method is to treat the bias as another trait (usually with value 1) and add the identical weight to the matrix W. In such a case, the derivative of the weight for the bias is computed with the weights for the other trait in the same accurate manner. The other choice is to compute the derivative individually as

$$\frac{\partial E}{\partial b^{(L)}} = \frac{\partial E}{\partial a^{(L)}} X \frac{\partial a^{(L)}}{\partial z^{(L)}} X \frac{\partial z^{(L)}}{\partial b^{(L)}} \quad \text{……..(22)}$$

Since the values for the first two derivatives are known. We just need to figure out the derivative for $\frac{\partial z^{(L)}}{\partial b^{(L)}}$. Now, think of that the slope of z does not rely at all from b, because b is just a constant value added at the end. Then, the derivative of the error w.r.t the bias minimizes to:

$$\frac{\partial E}{\partial b^{(L)}} = a^{(L)} - y \, X \, a^{(L)}(1 - a^{(L)}) \quad \text{……. (23)}$$

This is very suitable because it means we can reuse part of the computation for the derivative of the weights to calculate the derivative of the biases. The last lost part is the derivative of the error w.r.t. the bias in the (L−1) layer.

$$\frac{\partial E}{\partial b^{(L-1)}} = \frac{\partial E}{\partial a^{(L)}} X \frac{\partial a^{(L)}}{\partial z^{(L)}} X \frac{\partial z^{(L)}}{\partial a^{(L-1)}} \frac{\partial a^{(L-1)}}{\partial z^{(L-1)}} X \frac{\partial z^{(L-1)}}{\partial b^{(L-1)}} \quad \text{……. (24)}$$

Replacing with the actual derivatives for each expression:

$$\frac{\partial E}{\partial b^{(L-1)}} = a^{(L)} - y \, X \, a^{(L)}(1 - a^{(L)}) X \, w^{(L)} \, X \, a^{(L-1)}(1 - a^{(L-1)}) \quad \text{…….. (25)}$$

Same as before, we can reuse part of the calculation for the derivative of $w^{(L-1)}$ to solve this.

4.5.2.2 Backpropagation for Multiple-Unit Multilayer Perceptron

Most neural networks contain more than one neuron per layer. Up to now, we assumed a network with a single neuron in each layer. The main difference when adding more units is the need for additional indices to identify these neurons. For example, we use the letter *j* to index units in the output layer, *k* for units in the hidden layer, and *i* for units in the input layer. Correspondingly, weights require two subscripts to specify the connection between neurons in different layers.

For any network with multiple units, the number of weights exceeds the number of neurons, making this indexing necessary. This arrangement is illustrated in the weight matrix shown in Figure 2. The weights in layer LL are indexed accordingly. (Han and Morag, 1995, pp. 195–201).

With all this notation in mind, our original equation for the derivative of the error w.r.t the weights **in (L) layer suit:** $\frac{\partial E}{\partial w_{jk}^{(L)}} = \frac{\partial E_i}{\partial a_j^{(L)}} X \frac{\partial a_j^{(L)}}{\partial z_j^{(L)}} X \frac{\partial z_j^{(L)}}{\partial w_{jk}^{(L)}}$ **…….(26)**

Replacing with the derivatives:

$$\frac{\partial E}{\partial w_{jk}^{(L)}} = a_j^{(L)} - \mathrm{y}\,\mathrm{X}\,a_j^{(L)}(1 - a_j^{(L)})\mathrm{X}a_k^{(L-1)} \quad \text{..... (27)}$$

There is a second thing to consider. This time we have to take into account that every sigmoid activation from (L−1) layers affects the error via multiple pathways (assuming a network with multiple output units). To return this, we add a summation icon, and the statement for the derivative of the error w.r.t the sigmoid activation becomes

$$\frac{\partial E}{\partial a_k^{(L-1)}} = \sum_j \frac{\partial E_i}{\partial a_j^{(L)}}\mathrm{X}\frac{\partial a_j^{(L)}}{\partial z_j^{(L)}}\mathrm{X}\frac{\partial z_j^{(L)}}{\partial a_k^{(L-1)}} \quad \text{.....(28)}$$

Now, considering both the new subscripts and summation for $\frac{\partial E}{\partial a_k^{(L-1)}}$, may apply the chain-rule one more time to compute the error derivatives for w in $(L-1)$ as:

$$\frac{\partial E}{\partial w_{ki}^{(L-1)}} = \sum_j \left(\frac{\partial E_i}{\partial a_j^{(L)}}\mathrm{X}\frac{\partial a_j^{(L)}}{\partial z_j^{(L)}}\mathrm{X}\frac{\partial z_j^{(L)}}{\partial a_k^{(L-1)}}\right)\mathrm{X}\frac{\partial a_k^{(L-1)}}{\partial z_k^{(L-1)}}\mathrm{X}\frac{\partial z_k^{(L-1)}}{\partial w_k^{(L-1)}} \quad \text{.....(29)}$$

Replacing with the real derivatives for each expression we obtain:

$$\frac{\partial E}{\partial w_{ki}^{(L-1)}} = a_j^{(L)} - \mathrm{y}\,\mathrm{X}\,a_j^{(L)}\left(1 - a_j^{(L)}\right)\mathrm{X}w_{jk}^{(L)}\mathrm{X}a_k^{(L-1)}\left(1 - a_k^{(L-1)}\right)\mathrm{X}x_i^{(L-1)} \quad \text{.....(30)}$$

Considering the new indices, the derivative for the error w.r.t the bias becomes:

$$\frac{\partial E}{\partial b_j^{(L)}} = \frac{\partial E_i}{\partial a_j^{(L)}}\mathrm{X}\frac{\partial a_j^{(L)}}{\partial z_j^{(L)}}\mathrm{X}\frac{\partial z_j^{(L)}}{\partial b_j^{(L)}} \quad \text{.....(31)}$$

Replacing with the reall derivatives we get:

$$\frac{\partial E}{\partial b_j^{(L)}} = a_j^{(L)} - \mathrm{y}\,\mathrm{X}\,a_j^{(L)}\left(1 - a_j^{(L)}\right) \quad \text{.....(32)}$$

Last but not least, the description for the bias b at layer (L−1) is:

$$\frac{\partial E}{\partial b_{ki}^{(L-1)}} = \frac{\partial E}{\partial a_j^{(L)}}\mathrm{X}\frac{\partial a_j^{(L)}}{\partial z_j^{(L)}}\mathrm{X}\frac{\partial z_j^{(L)}}{\partial a_k^{(L-1)}}\mathrm{X}\frac{\partial a_k^{(L-1)}}{\partial z_k^{(L-1)}}\mathrm{X}\frac{\partial z_k^{(L-1)}}{\partial b_k^{(L-1)}} \quad \text{.....(33)}$$

Replacing with the real derivatives, we get:

$$\frac{\partial E}{\partial b_{ki}^{(L-1)}} = a_j^{(L)} - \mathrm{y}\,\mathrm{X}\,a_j^{(L)}\left(1 - a_j^{(L)}\right)\mathrm{X}\,w_{jk}^{(L)}\mathrm{X}a_k^{(L-1)}\left(1 - a_k^{(L-1)}\right) \quad \text{.....(34)}$$

4.6. Evaluation Criteria for Proposed Method: Classification, ROC Curve, and AUC

4.6.1 Classification and Confusion Matrix

During the data training process, a confusion matrix was utilized to estimate the multilayer perceptron classification model given below in Table 1.

A confusion matrix is a matrix used to predict the result of outputs (Amin and Al, 2017).

Table 1. Classification Table and Confusion Matrix

	Predictive Positive	Predictive Negative
Actual Positive	True Positive (TP)	False Negative (FN)
Actual Negative	False Positive (FP)	True Negative Rate (TN)

Important estimates were removed from the confusion matrix in order to correctly evaluate the classification model to advance a model permitting one variable to be predicted from known values of other variables. It plans the data item into an actual value variable. Several soft-computing algorithms occur for data mining, including: neuro unclear computing, genetic algorithms, neural networks, rough sets, decision trees, and crossing.

4.6.2 ROC Curve (Receiver Operating Characteristics Curve)

It is one of the maximum important estimation metrics for studying any classification model's performance. This curve plots two parameters used to decide if the model is (good or not) for a classifier's general feasible thresholds. It is created by plotting the True Positive Rate (y-axis) against the False Positive Rate (x-axis).

4.6.3 AUC (Area Under Curve)

In Machine Learning, showing estimation is a required task. So when it comes to a classification issue, we can depend on an AUC-ROC Curve. May want to study or predict the performance of the multi-class classification problem, and use the AUC.

4.7 Model information criterion or Model selection in machine learning

4.7.1 Model selection

Model selection is a machine learning process used to choose the best model for a given task from a set of candidate models. These models are evaluated using different model selection methods. The success of model selection depends heavily on a robust validation strategy and appropriate evaluation metrics. The goal of model selection is to maximize performance or accuracy (Ding, Tarokh, and Yang, 2018, pp. 16–34).

4.7.1.1 Akaike information criterion (AIC)

The Akaike Information Criterion (AIC) is a statistical measure used to evaluate and compare different models for a given dataset (Stoica and Selen, 2004, pp. 36–47). It balances the trade-off between the goodness of fit of the model on the training data and the model's complexity. AIC estimates the forecast error and provides a relative quality measure among a set of candidate models. It penalizes models with more parameters to favor simpler models that still adequately describe the data. A lower AIC value indicates a better-fitting model. Thus, AIC serves as a tool for model selection, aiming to identify the best approximating model to the unknown true data-generating process. The method of AIC is to choose the model that reduces the negative likelihood penalized by the number of parameters, as identified in the equation.

$$\mathrm{AIC} = 2k - 2\log(L) \qquad \textbf{.....(35)}$$

Where L refers to the likelihood under the fitted model and
k is the number of parameters in the model. Specifically,
k = the number of parameters in the model. Specifically,
L = the log-likelihood of the model under the fitted model, i.e., how well the model sees its training data

4.7.1.2 Bayesian Information Criterion (BIC)

Derived from Bayesian probability theory, the Bayesian Information Criterion (BIC) is a model selection criterion similar to AIC but includes a stronger penalty for model complexity. BIC is particularly suitable for models trained using maximum likelihood estimation. Models with greater complexity receive higher BIC scores, indicating poorer model quality (Stoica and Selen, 2004, pp. 36–47).
Formula:

$$\mathrm{BIC} = -2\log(\mathrm{L}) + \mathrm{k}\log(\mathrm{n}) \qquad \textbf{.....(36)}$$

K = the number of parameters
L = the log-likelihood of the model for the training data
N = the number of data points/samples

4.7.2 Model Fitting Information - Likelihood Ratio Test

The likelihood ratio test is used to assess the goodness of fit of a statistical model, particularly when the sample size is sufficiently large. In this context, the alternative hypothesis (H1) corresponds to

a 'saturated' model, where the number of parameters equals the sample size n (Glover and Dixon, 2004, pp. 791–806).

$$D = \text{likelihood ratio test} = -2 * \text{Ln}\frac{\text{Likelihood for null model}}{\text{Likelihood for alternative model}} \quad \text{.....(37)}$$

D is the test statistic.

5. Practical Aspect (Statistical Analysis of Multilayer Perceptron Neural Network)

5.1 Data Collection and Description

1. Data collection is defined as the procedure of collecting, organizing, and analyzing perfect insights for research. In our study about Multilayer Perceptron is an excellent type of Artificial Neural Networks (ANN), which was utilized to predict the risk factors of immunodeficiency problems. For this purpose, the researcher took a sample with size of 500 patients with several diseases for which they performed analyses for their Immune levels testing at the Kurdistan Hospitals and Laboratories.

2. Variables Description

Dependent Variable: Y= Immune testing level (good immunity or poor immunity) and
Independent Variables: Six risk factors, labeled as X1, X2, X3, X4, X5, and X6, which are described in detail in Table 3.

5.2 Analyses

1. Data Case Processing Summary

In this study, the multilayer perceptron (MLP) Neural Networks were used to partition Training and Testing sample data of size 500 persons as follows:

Table 2. Case Processing Summary (Proportions of data in each dataset, training and testing data)

		N	Percent
Sample data	Training Test	400	80%
	Testing Test	100	20%
Total		500	100%

The case processing summary is illustrated in Table 2. The dataset is divided into the training dataset and the testing dataset. The model summary shows that Training data is the data that will be used to predict 80% of the result of designing the model.

2. The Network Information

Since the Multilayer Perceptron (MLP) procedure produces a predictive model for one or more dependent (target) variables based on the values of the predictor variables.

Table 3. The Network Information

Input Layer (Independent) (predictor variables): total risk factors most affecting Immunodeficiency		
The risk factors most affecting Immunodeficiency.	X_1	Genetic history
	X_2	Diabetes
	X_3	Human Immunodeficiency Virus (HIV)
	X_4	AIDS
	X_5	Cancer treatments such as radiation therapy and some chemotherapy
	X_6	Certain medicines
Rescaling Method (Standardized (x−mean)/s.d)		
Hidden Layers		

Activation Function	(Sigmoid)
Number of Units in Hidden Layer	6
Output Layer (target variable) (Dependent)	
Dependent Variables	Y= Immune Levels testing
Number of Levels	1good immunity and 2 poor immunity
Activation Function	(Sigmoid) Excluding the bias unit

Table 3 illustrates the network information. The automatic architecture selection chose six units in the input layer, corresponding to the independent variables (X1, X2, X3, X4, X5, X6), which represent the risk factors most affecting immunodeficiency. These input units use the sigmoid activation function. The network includes 6 units in the hidden layer(s), and the dependent variable (Y) is represented in the output layer, which uses the softmax activation function to forecast the status of the immune system.

3. The Multilayer Perceptron Structure of a Feedforward Architecture

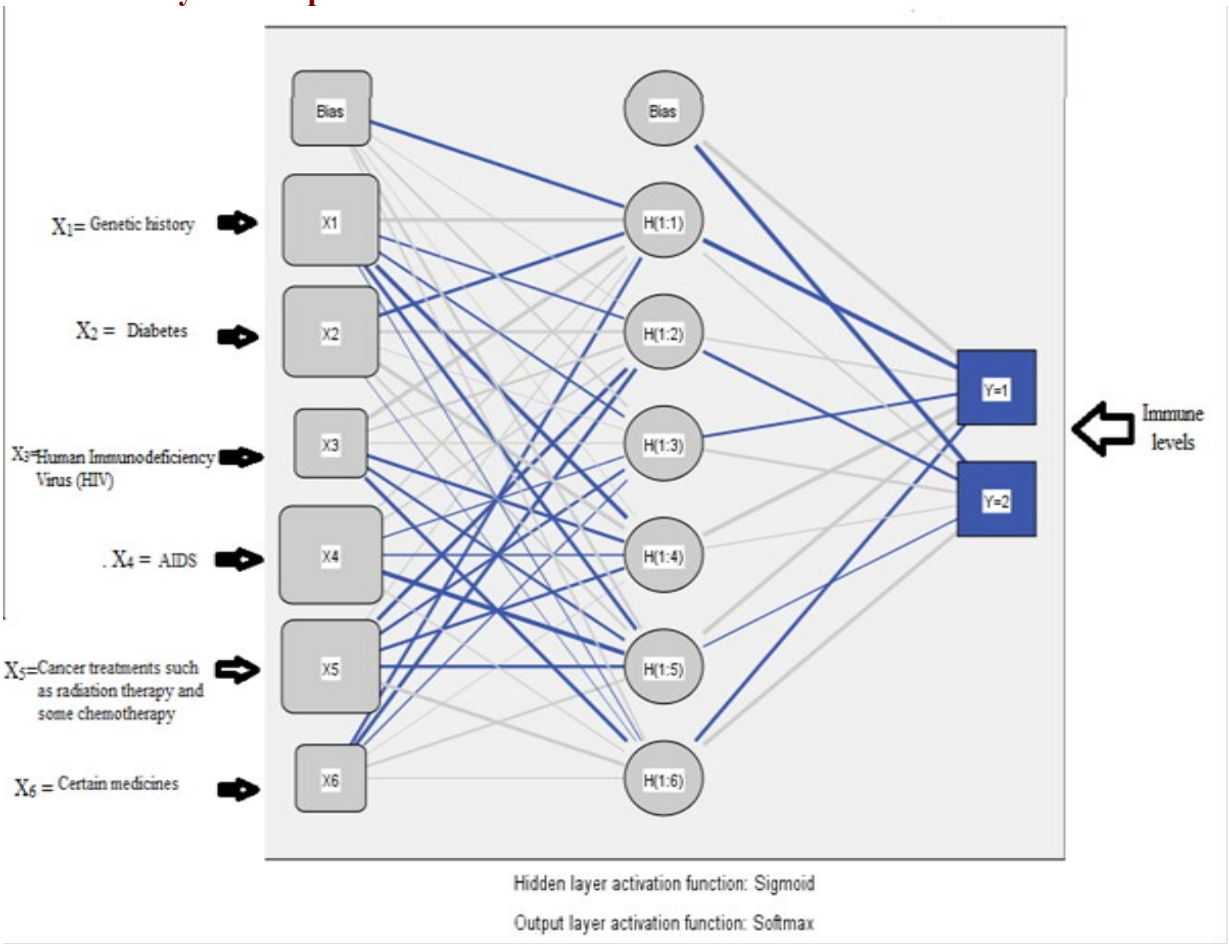

Figure 7. Multilayer perceptron structure of a feedforward architecture with one hidden layer

The multilayer perceptron (MLP) architecture is known as a feedforward network because connections flow in one direction, from the input layer to the output layer. In the figure, the input layer consists of the predictor variables (X1, X2, X3, X4, X5, X6). The hidden layer contains unobservable nodes or units, while the output layer contains the response variables. Each output unit is a function of the hidden units; the exact form of this function depends both on the network type and user-defined specifications. In the diagram, dark lines represent strong connections, while

blurred lines indicate weaker connections.

In contrast, a limitation of random forest algorithms is that, when used for regression, they cannot predict values beyond the range seen in the training data. Additionally, random forests may overfit datasets that contain significant noise.

4. Parameter Estimates

Table 4. Parameter Estimates

Parameter Estimates								
Predictors (Independent Variables)			Predicted					
			Hidden Layer 1					
			H(1:1)	H(1:2)	H(1:3)	H(1:4)	H(1:5)	H(1:6)
Input Layer	(Bias)		-0.436	0.305	-0.064	-0.085	-0.229	-0.406
	X_1	Genetic history	0.849	0.938	-0.429	0.369	-0.723	-0.181
	X_2	Diabetes	-0.055	0.704	0.213	0.049	0.091	0.161
	X_3	Human Immunodeficiency Virus (HIV)	0.010	0.245	0.012	0.125	0.637	0.204
	X_4	AIDS	0.517	0.156	0.727	0.244	-0.372	-0.007
	X_5	Cancer treatments such as radiation therapy and some chemotherapy	-0.084	0.835	-0.194	-0.486	-0.457	0.181
	X_6	Certain medicines	0.119	0.118	0.508	-0.003	-0.027	-0.292

In the multilayer perceptron structure analysis shown in Figure 8, dark links represent strong connections. The strengths of the random forest algorithm, when used for regression, include its ability to predict beyond the range of the training data. The figure illustrates the connection strengths between predictors and the output, with the following weights: X1 = 0.938, X5 = 0.835, X4 = 0.727, X2 = 0.704, X3 = 0.637, and X6 = 0.508. The training bias is -0.436, and this low bias indicates that the model effectively captures the underlying patterns in the data.

5. Classification

Table 5. Classification or Confusion Matrix

Sample	Observed	Predicted		
		weak Immunity	good Immunity	Percent Correct
Training	poor Immunity	163	33	83.2%
	good Immunity	28	125	81.7%
	Overall Percent	54.7%	45.3%	82.5%
Testing	poor Immunity	82	12	87.2%
	good Immunity	17	40	70.2%
	Overall Percent	65.6%	34.4%	80.8%

This table illustrates the classification results, highlighting that the cells along the diagonal of the cross-classification matrix represent correct predictions. For the training data, the correct predictions are 163 and 125, corresponding to an accuracy of 82.5% in predicting the risk factors most affecting human immunodeficiency. Training data is used to develop and fit the machine learning model. Similarly, for the testing data, the correct predictions are 82 and 40, which

correspond to an accuracy of 80.8%. Testing data is used to evaluate the performance and generalization ability of the trained model.

6. ROC Curve

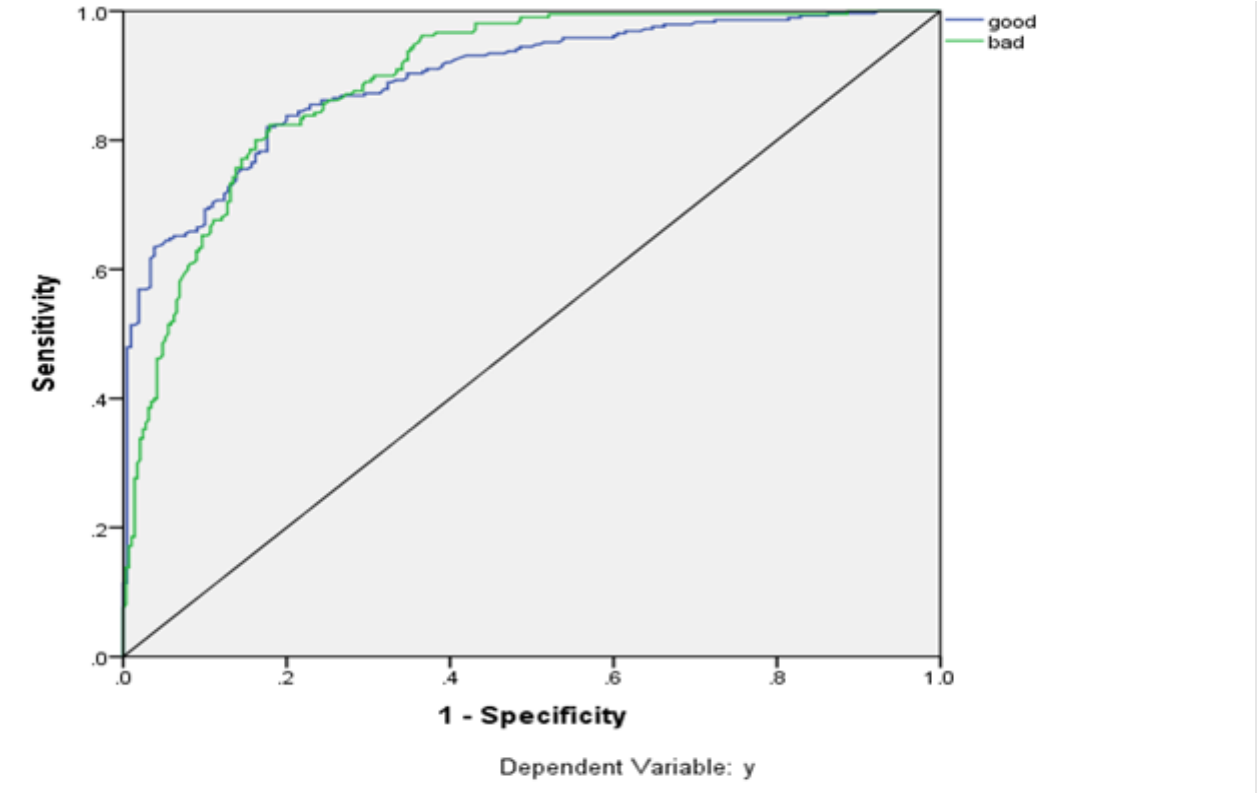

Figure 8. The ROC Curve

The ROC curve gives a visual display of the sensitivity and specificity for all possible cutoffs in a single plot, which is much cleaner and more powerful than a series of tables. The chart displays two curves, one for the category (poor) and one for the category (good). Since there are only two categories, the curves are symmetrical about a 45-degree line (not displayed) from the upper left corner of the chart to the lower right. This chart is based on the combined training and testing samples.

7. Area under ROC Curve (AUC)

Table 6. Area Under the Curve

Area Under the Curve		
		Area
Y	good Immunity	0.895
	poorImmunity	0.895

This table refers to the area under the curve is a numerical summary of the ROC curve, and the values in the table represent, for each category, the probability that the predicted pseudo-probability of being in that category is higher for a randomly chosen case in that category than for a randomly chosen case not in that category, for a randomly selected defaulter and randomly selected non-defaulter, there is a 0.895 probability that the model-predicted pseudo-probability of default will be higher for the defaulter than for the non-defaulter While the area under the curve is a useful one-

statistic summary of the accuracy of the network, you need to be able to choose a specific criterion by which customers are classified. The predicted-by-observed chart provides a visual start on this process.

8. The Importance of Independent Variable Importance

Table 7. The importance of Independent Variable

Independent Variable Importance			
Independent Variables		Importance	Normalized Importance
X_1	Genetic history	0.268	100.0%
X_2	Diabetes	0.176	65.5%
X_3	Human Immunodeficiency Virus (HIV)	0.152	56.8%
X_4	AIDS	0.183	68.2%
X_5	Cancer treatments such as radiation therapy and some chemotherapy	0.205	76.4%
X_6	Certain medicines	0.017	6.2%

The importance of an independent variable is a measure of how much the Network's model-predicted value changes for different values of the independent variable. Normalized importance is simply the importance values divided by the largest importance values and expressed as percentages.

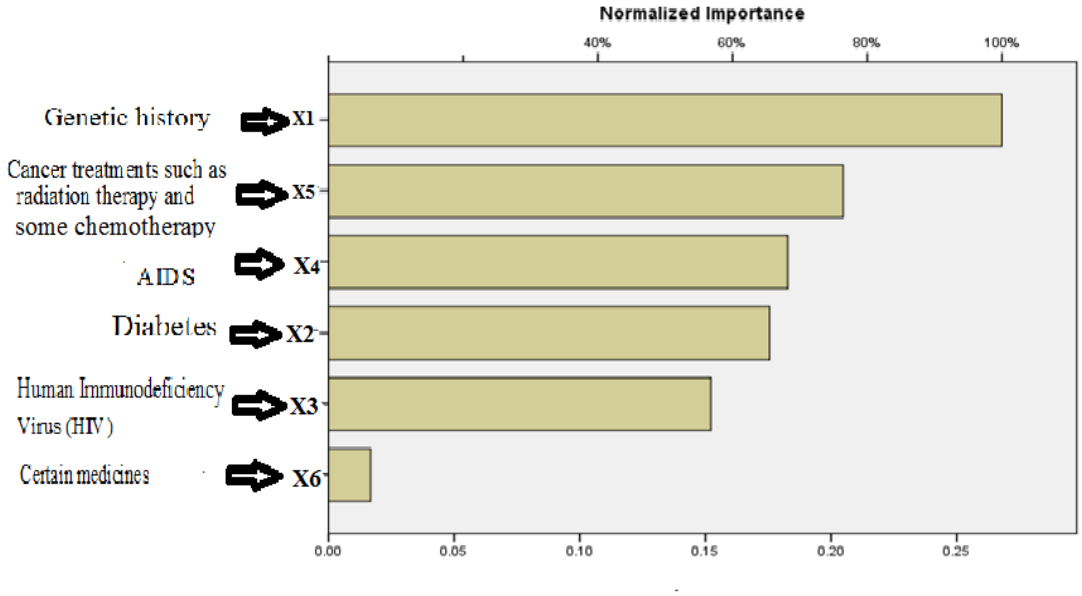

Figure 9. Table of Independent Variable Importance

It is a figure represents the Importance of the risk factors most affected on Human Immunodeficiency it indicates that the X_1= Genetic history has biggest effect (100.00%) and follow X_5= Cancer treatments such as radiation therapy and some chemotherapy 76.4%, the X_4 =AIDS percentage is 68.2%, X_2 =Diabetes percentage is 65.5%, X_3 =Human Immunodeficiency Virus (HIV) 56.8% and finally X_6 =the Certain medicines has the lowest effects by 6.2%.

9. Model Fitting Criteria

Table 8. Model Fitting Criteria

Model Fitting Information						
Model	Model Fitting Criteria(Model Selection)			Likelihood Ratio Tests		
	AIC	BIC	-2 Log Likelihood	Chi-Square	df	Sig.
Intercept Only	167.097	171.312	165.097			
Final	162.836	196.553	146.836	18.261	7	0.011

Table 8 shows that the chi-square test in (AIC) (BIC) and the likelihood ratio has [P-value =0.011< 0.05], which confirms that the risk factors have the most effect on Human Immunodeficiency

10. Model Fitting Criteria (Model Selection Criteria)

Likelihood Ratio Tests							
Effect Predictor variables		Model Fitting Criteria			Likelihood Ratio Tests		
		AIC of Reduced Model	BIC of Reduced Model	-2 Log Likelihood of Reduced Model	Chi-Square	df	Sig. p-value
Intercept		119.695	149.197	105.695	0.000	0	.
X_1	Genetic history	125.803	151.091	113.803	8.108	1	0.004
X_2	Diabetes	121.929	147.217	109.929	4.234	1	0.029
X_3	Human Immunodeficiency Virus (HIV)	122.803	146.091	113.803	4.008	1	0.044
X_4	AIDS	123.929	149.217	111.929	6.234	1	0.028
X_5	Cancer treatments such as radiation therapy and some chemotherapy	124.803	150.091	111.803	8.000	1	0.006
X_6	Certain medicines	120.295	144.583	106.295	4.600	1	0.048

Since the significance values (p-values) for all predictor variables. X1, X2, X3, X4, X5, and X6 are less than 0.05. This confirms that these risk factors significantly affect human immunodeficiency. The Akaike Information Criterion (AIC) and Bayesian Information Criterion (BIC) were used to compare different models and identify the best fit for the data. Among the factors, X1 = Genetic history has the greatest effect with a p-value of 0.004, followed by X5 = Cancer treatments such as radiation therapy and chemotherapy (p = 0.006). Next are X4 = AIDS (p = 0.028), X2 = Diabetes (p = 0.029), X3 = Human Immunodeficiency Virus (HIV) (p = 0.044), and finally X6 = Certain medicines, which has the smallest effect with a p-value of 0.048.

6. Conclusion

The Multilayer Perceptron (MLP) is a well-known type of Artificial Neural Network (ANN) that was used to predict risk factors associated with human immunodeficiency problems. Patients with various diseases underwent immune level testing at hospitals and laboratories in Kurdistan. The model utilized a feedforward architecture with parameter estimates highlighting the relative impact of different risk factors. The analysis showed that X1 = Genetic history has the greatest effect, followed by X5 = Cancer treatments (such as radiation therapy), then X4 = AIDS, X2 = Diabetes, X3 = Human Immunodeficiency Virus (HIV), and finally X6 = Certain medicines.

Model fit was assessed using Model Selection Criteria such as Akaike Information Criterion (AIC),

Bayesian Information Criterion (BIC), and Likelihood Ratio Tests, including the Chi-Square test (P-value = 0.011 < 0.05). These results indicate that the full model predicts outcomes significantly better than the null model, confirming a good model fit. This signifies that the selected risk factors significantly affect human immunodeficiency, as supported by the final model fitting criteria.

Limitation/Recommendation

The researcher used a Multilayer Perceptron (MLP) Artificial Neural Network for medical (disease) data and recommended its use across various fields due to its high accuracy. The MLP was implemented with the sigmoid activation function because the data involved binary classification. However, for multi-class classification tasks, it is recommended to use the softmax activation function.

References

- Amin, M.Z. and Al, A. (2017). 'Application of Multilayer Perceptron (MLP) for Data Mining in Healthcare Operations'. *International Conference on Biotechnology*, University of South Asia, Lahore, Pakistan, 8–9 February, pp. 7–8.
- Block, H.D. (1962) 'The Perceptron: A Model for Brain Functioning. I', *Reviews of Modern Physics*, 34(1), pp. 123–135. doi:10.1103/RevModPhys.34.123.
- Castro Gbêmêmali Hounmenou, G., Gneyou, K.E. and Glèlè Kakaï, R. (2021). 'A Formalism of the General Mathematical Expression of Multilayer Perceptron Neural Networks'. *Preprints*, doi:10.20944/preprints202105.0412.v1, pp. 3–9.
- Chan, K.Y., Abu-Salih, B., Qaddoura, R. et al. (2023). 'Deep neural networks in the cloud: Review, applications, challenges and research directions'. *Neurocomputing*, pp. 3–18.
- Chinn, I.K. and Orange, J.S. (2019). 'Immunodeficiency Disorders'. *Pediatrics in Review*, 40(5), pp. 229–242. doi:10.1542/pir.2017-0308. PMID: 31043442.
- Cybenko, G. (1989). 'Approximation by superpositions of a sigmoidal function', *Mathematics of Control, Signals, and Systems*, 2(4), pp. 303–314.
- D'Agostino, A. (2021). *Introduction to neural networks: weights, biases and activation*. Available at: https://medium.com/mlearning-ai/submission-suggestions-b51e2b130bfb (Accessed: 17 July 2025).
- Ding, J., Tarokh, V. and Yang, Y. (2018). 'Model Selection Techniques: An Overview', *IEEE Signal Processing Magazine*, 35(6), pp. 16–34. arXiv:1810.09583.
- Fernandez, J. (2023). *Overview of Immunodeficiency Disorders*. MD, PhD, Cleveland Clinic Lerner College of Medicine at Case Western Reserve University, p. 1.
- Goodfellow, I., Bengio, Y. and Courville, A. (2016). 'Softmax Units for Multinoulli Output Distributions'. In: *Deep Learning*. Cambridge, MA: MIT Press, pp. 180–184. ISBN 978-0-26203561-3.
- Han, J. and Morag, C. (1995). 'The influence of the sigmoid function parameters on the speed of backpropagation learning'. In: Mira, J. and Sandoval, F. (eds.) *From Natural to Artificial Neural Computation*. Lecture Notes in Computer Science, vol. 930. Berlin: Springer, pp. 195–201. doi:10.1007/3-540-59497-3_175.
- Hornik, K., Stinchcombe, M. and White, H. (1989). 'Multilayer feedforward networks are universal approximators', *Neural Networks*, 2(5), pp. 359–366. doi:10.1016/0893-6080(89)90020-8.
- Inoussah, M., Ngoh, S., Molu, R.J.J., Kenfack, B., Onguene, R., Dzonde Naoussi, S., Tamba, J.G., Bajaj, M. & Tuka, M., 2024. A multilayer perceptron neural network approach for optimizing solar irradiance forecasting in Central Africa with meteorological insights. Scientific Reports, 14, Article number: [if available], DOI: 10.1038/s41598-024-54181-y.
- Notarangelo, L.D., Fischer, A., Geha, R.S., Casanova, J.L. and Chapel, H. (2016). *Primary immunodeficiencies: 2009 update*. pp. 1099–1105.
- Pac, M. and Bernatowska, E. (2016). 'Comprehensive activities to increase recognition of primary immunodeficiency and access to immunoglobulin replacement therapy in Poland', *Polish Journal of Medicine*, 175(8), pp. 101–109.
- Piramuthu, S., Shaw, M.J. and Gentry, J.A. (1990). 'M Perceptrons'. *CMU-RI-TR-90-29Y*, Carnegie Mellon University, March, p. 6.
- Roberts, R.L. (2007). *Immunodeficiency*. University of California at Los Angeles, p. 1.
- Rodrigo, H.S.S. (2017). *Bayesian Artificial Neural Networks in Health and Cybersecurity*. University of South Florida. Available at: https://digitalcommons.usf.edu/etd (Accessed: 17 July 2025), pp. 120–132.
- Rosenblatt, F. (1961). *Principles of Neurodynamics: Perceptrons and the Theory of Brain Mechanisms*. Washington DC: Spartan Books, p. 4.
- Rumelhart, D.E., Hinton, G.E. and Williams, R.J. (1986). 'Learning representations by back-propagating errors', *Nature*, 323(6088), pp. 533–536.

- Sejnowski, T.J. (2018). *The Deep Learning Revolution*. Cambridge, MA: MIT Press, p. 47. ISBN 978-0-262-03803-4.
- Stoica, P. and Selen, Y. (2004). 'Model-order selection: a review of information criterion rules', *IEEE Signal Processing Magazine*, (July), pp. 36–47. doi:10.1109/MSP.2004.1311138.
- Sullivan, K. (2024). *Primary immunodeficiency*. MD, PhD, Children's Hospital of Philadelphia, p. 201.
- Vaillant, A.A.J. and Qurie, A. (2023). *Immunodeficiency*, p. 6.
- White, K. and Black, M. (2018). 'Vaccine efficacy', *Immunology Today*, 32(4), pp. 112–120.
- Yang, Z. and Yang, Z. (2014). *Comprehensive Biomedical Physics*. Stockholm: Elsevier, p. 1. ISBN 978-0-444-53633-4.
- Ziad A.A. Al Qadi (2021). *Artificial neural networks (ANN)*, pp. 1–18.

پێشبینیکردن بە شیکاری تۆڕی دەماری دەستکردی پەرسێپترۆنی فرە چینەیی سیگمۆید
ئەرک و مۆدێل هەڵبژاردن بۆ ئەو هۆکارە مەترسیدارانەی کە زۆرترین کاریگەرییان لەسەرکەمی بەرگری مرۆڤ هەیە

نەزیرە سدیق کەریم بەرزنجی
بەشی ئامار و زانیارییەکان،کۆلێژی بەڕێوەبردن و ئابووری ،زانکۆی سەلاحەددین
هەولێر، هەرێمی کوردستان،عێراق-
nazeera.kareem@su.edu.krd

پوختە
فرە چین پەرسێپترۆن (MLP)جۆرێکی زۆر بانگەشەی تۆڕەکانی دەماری دەستکرد ((ANN)ە، لەم توێژینەوەیەدا بەکارهێناوە بۆ پێشبینی هۆکارەکانی مەترسی کێشەکانی کەمی بەرگری مرۆڤ ,بۆ ئەم مەبەستە توێژەر نمونەیەکی لە قەبارەی (500) نەخۆشی چەند نەخۆشییەک وەرگرت کە ئەوانە لە نەخۆشخانە و تاقیگەکانی کوردستان ئاستی بەرگریان تاقیکردۆتەوە، ئەم نمونەی داتایە وابەستەی گۆڕاو [Y=ئاستی پشکنینی بەرگری (بەرگری باش یان بەرگری خراپ) و گۆڕاوە سەربەخۆکان(هۆکارەکانی مەترسی (X_1,X_2,X_3,X_4,X_5,X_6) لەخۆدەگرێت بۆ شیکارەکانی[،خەملاندنی پارامێتەرەکان،گرنگی گۆڕاوە سەربەخۆکان]جەختی لەوە کردەوە کە X_1=مێژووی بۆماوەیی زۆرترین کاریگەری هەیە بەدوایدا X_5=چارەسەرەکانی شێرپەنجە وەک چارەسەری تیشکی بەدوایدا دێت بە , X_4=ئایدز،پاشان X_2 = شەکرە،X_3= ڤایرۆسی کەمی بەرگری مرۆڤ ((HIV)لە کۆتاییدا X_6 =هەندێک دەرمان] و بە [هەڵبژاردنی مۆدێل [(AIC) ، (BIC)] و تاقیکردنەوەکانی ڕێژەی ئەگەری. تاقیکردنەوەی کای-چوارگۆشە [بەهای P=0.05 > 0.011 گرنگة]، ئەوە پشتڕاست دەکاتەوە کە هۆکارەکانی مەترسی زۆرترین کاریگەرییان هەیە لەسەر کەمی بەرگری هەیە

وشەی سەرەکییەکان: تۆڕی دەماری دەستکرد، هەڵبژاردنی مۆدێل، پەرسێپترۆنی فرە چین، هۆکارەکانی مەترسی کەمی بەرگری

التنبؤ باستخدام الشبكة العصبية الاصطناعية متعددة الطبقات السيني
الوظيفة واختيار النموذج لعوامل الخطر الأكثر تأثرًا بنقص المناعة البشرية

نظيرة صديق كريم برزنجي
قسم الإحصاء والمعلوماتية،كلية الإدارة و الإقتصاد،جامعة صلاح الدين-أربيل،
أقليم كوردستان،العراق
nazeera.kareem@su.edu.krd

الملخص
يعد الـ (MLP) متعدد الطبقات أحد أنواع الشبكات العصبية الاصطناعية ((ANN)المشهورة، وقد تم تطبيقه في هذا البحث للتنبؤ بعوامل الخطر لمشاكل نقص المناعة البشرية، ولهذا الغرض أخذ الباحث عينة بحجم (500) مريض مصابين بأمراض مختلفة حيث تم اختبار مستويات المناعة لديهم في مستشفيات ومختبرات كردستان، تحتوي عينة البيانات هذه على المتغير التابع [Y=مستوى الاختبار المناعي (مناعة جيدة أو مناعة ضعيفة) والمتغيرات المستقلة (عوامل الخطر (X_1,X_2,X_3,X_4,X_5,X_6) من خلال التحليلات [تقديرات المعلمات، أهمية المتغير المستقلأكد أن [X_1=التاريخ الوراثي له التأثير الأكبر يليه X_5=علاجات السرطان مثل العلاج الإشعاعي يليه X_4= الإيدز، ثم X_2= مرض السكري، X_3= فيروس نقص المناعة البشرية (HIV)، وأخيراً X_6=بعض الأدوية] ومن خلال [اختيار النموذج ((AIC، (BIC))و اختبارات نسبة الاحتمالية. اختبار مربع كاي، قيمة $P = 0.011 < 0.05$ ذات دلالة إحصائية]،]، يؤكد عوامل الخطر الأكثر تأثرًا بنقص المناعة

الكلمات المفتاحية:الشبكة العصبية الاصطناعية، اختيار النموذج، الإدراك الحسي متعدد الطبقات، عوامل خطر نقص المناعة