Advances In Multidisciplinary & Scientific Research
A Multidisciplinary & Interdisciplinary Journal
Vol. 5 No. 3, September, 2019

Artificial Neural Network Trained Architecture: An Effective Bank Loan Decision Support System

[1]Ugwu, E.A. & [2]Ezekwe, C.G.
[1]Department of Computer Science, Enugu StateUniversity, Enugu, Nigeria
[2]Electronics Development Institute, Awka, National agency for Science and Engineering Infrastructure (NASENI) - Federal ministry of Science and Technology, Nigeria
E-mails: [1]Ugwuedith2003@yahoo.com, [2]norakingchi@yahoo.com ,

ABSTRACT

Artificial Neural Network trained architecture is a neural network model that mimics the way a human brain works by selecting the best conditions that are suitable for issuing loans to customers that will not later turn to bad debts. It is made up of a number of neurons that are organized in several layers. The input layers of neurons feeds the input variables into the network through the hidden layers or processing elements which send the processed signal to the output layers. Analysis was carried out using an analytical tool called neural SPSS. We were able to determine the variables, dataset, network information, network diagram and the neural network architecture. During the process in the network, there are factors there are factors that contribute to the efficiency of the network processes; they are the network activation function (network threshold), the network activation rate (network alpha) and the network activator (the network weight). At analysis stage, neural SPSS was used to generate the right neural network architecture from the data collected. A feed-forward neural network with back propagation learning algorithm was used to build up the proposed model. This system enables banks to decide whether to grant loan to customers or not, thereby making loans available for only payable customers. The result is either 0(No loan) or 1(Loan granted). This research indicated that artificial neural networks are a successful technology that can be used in loan application evaluation in banks.

Keywords – Artificial neural network, architecture, SPSS, bank loan, decision support system

Aims Research Journal Reference Format:
Ugwu, E.A. & Ezekwe, C.G. (2019): Artificial Neural Network Trained Architecture: An Effective Bank Loan Decision Support System. Advances in Multidisciplinary Research Journal. Vol. 5. No. 3, Pp 13–32
Article DOI: dx.doi.org/10.22624/AIMS/V5N3P2

1. INTRODUCTION

Examinations of humans' central nervous systems inspired the concept of artificial neural networks. In an artificial neural network, simple artificial nodes, known as "neurons", "processing elements" or "units", are connected together to form a network which mimics a biological neural network(Hebb, 1949).There is no single formal definition of what an artificial neural network is. However, a class of statistical models may commonly be called "neural" if it possesses the following characteristics:

i. Contains sets of adaptive weights, i.e. numerical parameters that are tuned by a learning algorithm, and
ii. It is capable of approximating non-linear functions of their inputs.
iii. There must be three sections comprising of the inputs nodes, the processing elements (hidden nodes) and the output nodes.
iv. There must be a defined threshold or network activation function from the mathematical model view point.
v. There must be a defined network training parameter called alpha which is the network activation rate.

The adaptive weights can be thought of as connection strengths between neurons, which are activated during training and prediction Werbos(1975). Artificial neural networks are similar to biological neural networks in the performing by its units of functions collectively and in parallel, rather than by a clear delineation of subtasks to which individual units are assigned. The term “neural network” usually refers to models employed in statistics, cognitive psychology and artificial intelligence. Neural network models which command the central nervous system and the rest of the brain are part of theoretical neuroscience and computational neuroscience.

In modern software implementations of artificial neural networks, the approach inspired by biology has been largely abandoned for a more practical approach based on statistics and signal processing. In some of these systems, neural networks or parts of neural networks (like artificial neurons) form components in larger systems that combine both adaptive and non-adaptive elements. While the more general approach of such systems is more suitable for real-world problem solving, it has little to do with the traditional, artificial intelligence connectionist models. What they do have in common, however, is the principle of non-linear, distributed, parallel and local processing and adaptation. Historically, the use of neural network models marked a directional shift in the late eighties from high-level (symbolic) artificial intelligence, characterized by expert systems with knowledge embodied in “*if then”* rules, to low-level (sub-symbolic) machine learning, characterized by knowledge embodied in the parameters of a dynamical system.

1.1 Problem Definition

In Nigeria presently, inflation is on the high side and average Nigerians are in search of cash either for business or to sort out personal pressing needs. This led to the high demand of credit facilities from the banks. As the demand for credit facilities are high, the rate of defaults are high. This led to this research, to help banks predict good or bad credit facility. Generally, account officers rely on traditional methods to guide them in evaluating the worthiness of loan applications and also the analysis and the ability to interpret the information correctly is a huge task for these account officers.

1.2 Research Objectives

i. To develop artificial neural network architecture that will be able to predict accurately when to issue loan to customers and when not to give.
ii. To identify neural networks as an enabling tool for evaluating credit applications to support loan decisions.
iii. To develop a system that can analyze a set of data collected from customers and use it for loan prediction.

2. EVOLUTION OF ARTIFICIAL NEURAL NETWORK

Artificial Neural Network (ANN) was initiated when Warren and Walter (1943) created a computational model based on mathematics and algorithms called threshold logic. This model paved the way for neural network research to split into two distinct approaches. One approach focused on biological processes in the brain and the other focused on the application of neural networks to artificial intelligence. In the late 1940s psychologist Donald Hebb *(1949)* created a hypothesis of learning based on the mechanism of neural plasticity that is now known as Hebbian learning. Hebbian learning is considered to be a 'typical' unsupervised learning rule and its later variants were early models for long term potentiation. Researchers started applying these ideas to computational models in 1948 with Turing’s B-type machines. (Hebb, 1949).

Farley and Wesley A. Clark (1954) first used computational machines, then called “calculators,” to simulate a Hebbian network at MIT. Other neural network computational machines were created by Rochester, et al (1956). Rosenblatt(1958) created the perceptron, an algorithm for pattern recognition based on a two-layer computer learning network using simple addition and subtraction. With mathematical notation, Rosenblatt also described circuitry not in the basic perceptron, such as the exclusive-or circuit, a circuit which could not be processed by neural networks until after the back propagation algorithm was created by Werbos(1975).

Neural network research stagnated after the publication of machine learning research by Minsky and Papert(1969), who discovered two key issues with the computational machines that processed neural networks. The first was that basic perceptrons were incapable of processing the exclusive-or circuit. The second significant issue was that computers didn't have enough processing power to effectively handle the long run time required by large neural networks. Neural network research slowed until computers achieved greater processing power. A key advance that came later was the back propagation algorithm which effectively solved the exclusive-or problem, and more generally the problem of quickly training multi-layer neural networks (Werbos 1975). In the mid-1980s, parallel distributed processing became popular under the name connectionism. The textbook by Rumelhart and McClelland(1986) provided a full exposition of the use of connectionism in computers to simulate neural processes.

Artificial Neural Network

Neural networks, as used in artificial intelligence, have traditionally been viewed as simplified models of neural processing in the brain *Behnke (2003)*, even though the relation between this model (see figure 1 and figure 2) and the biological architecture of the brain (see figure 3) is debated; it's not clear to what degree artificial neural networks mirror brain function. Support vector machines and other, much simpler methods such as linear classifiers gradually overtook neural networks in machine learning popularity. But the advent of deep learning in the late 2000s sparked renewed interest in neural networks.

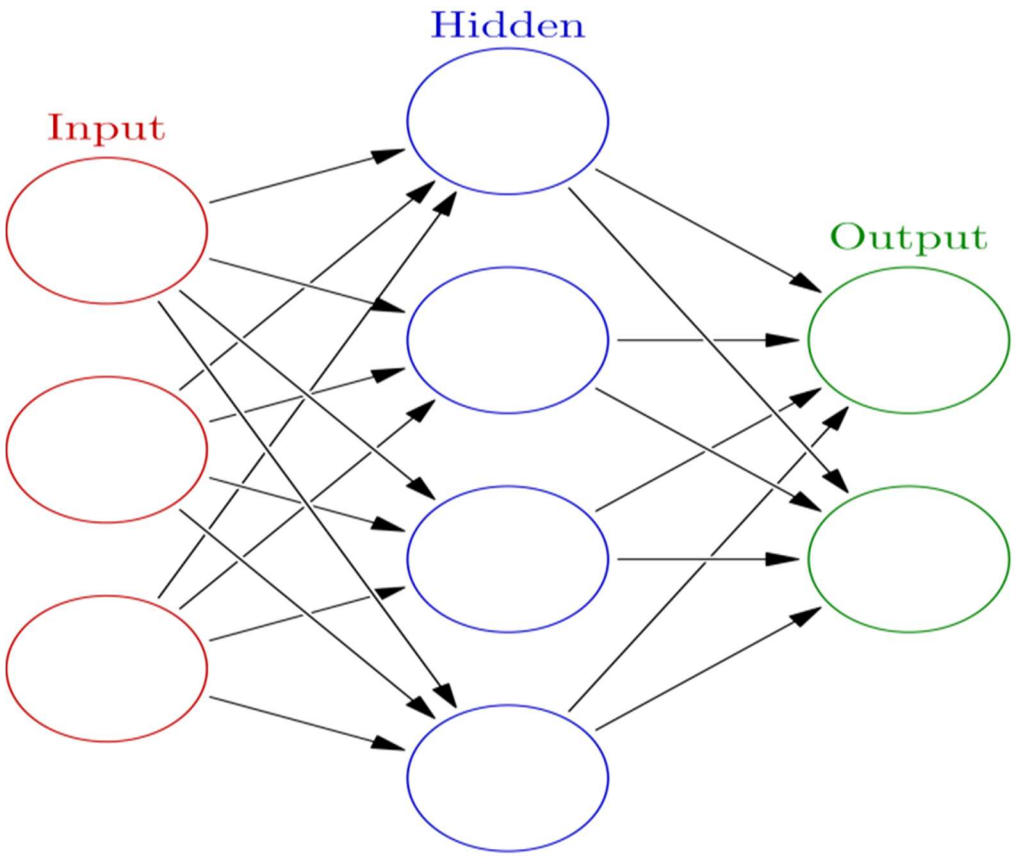


Figure 1: Artificial Neural Network model (one processing element region) *Behnke (2003).*

Artificial neural networks (**ANN**) or **connectionist systems** are computing systems that are inspired by, but not necessarily identical to, the biological neural networks that constitute animal brains *Behnke (2003)*. Such systems "learn" to perform tasks by considering examples, generally without being programmed with any task-specific rules. For example, in image recognition, they might learn to identify images that contain cats by analyzing example images that have been manually labeled as "cat" or "no cat" and using the results to identify cats in other images. They do this without any prior knowledge about cats, for example, that they have fur, tails, whiskers and cat-like faces. Instead, they automatically generate identifying characteristics from the learning material that they process.

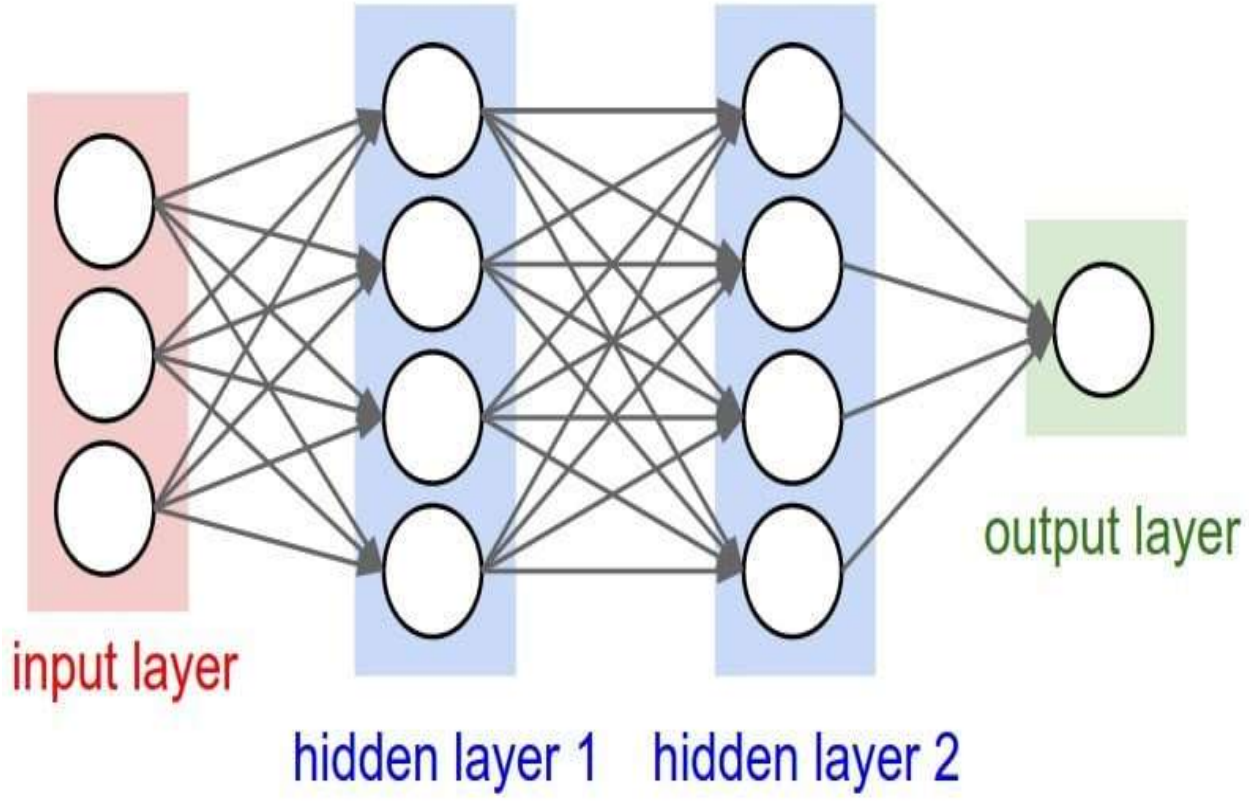


Figure 2: Artificial Neural Network model (two processing element region) *Behnke (2003).*

An ANN is based on a collection of connected units or nodes called artificial neurons (*Cireşan, 2010*), which loosely model the neurons in a biological brain. Each connection, like the synapses in a biological brain, can transmit a signal from one artificial neuron to another. An artificial neuron that receives a signal can process it and then signal additional artificial neurons connected to it. In common ANN implementations, the signals are a connection between artificial neurons which are real numbers, and the output of each artificial neuron is computed by some non-linear function of the sum of its inputs. The connections between artificial neurons are called 'edges'(*Cireşan, 2010*). Artificial neurons and edges typically have a weight that adjusts as learning proceeds. The weight increases or decreases the strength of the signal at a connection. Artificial neurons may have a threshold such that the signal is only sent if the aggregate signal crosses that threshold. Typically, artificial neurons are aggregated into layers. Different layers may perform different kinds of transformations on their inputs. Signals travel from the first layer (the input layer), to the last layer (the output layer), possibly after traversing the layers multiple times.

Neural networks are also an emerging artificial intelligence technology that imitates the human brain on the computer(*Cireşan, 2010*). These techniques are based on the parallel, distributed processing design. The parallel structure makes neural networks proficient at analyzing problems with many variables (Tafti, 1993 and Zhang, 2004). Scientists have been inspired by the capabilities of the human brain for information processing and problem solving. Therefore, neural networks designers try to put intelligence into these systems in the form of generalized ability to learn and recognize patterns to exhibit similar intelligent functionality like humans (Shachurove, 2002).

A neural network model is composed of a number of neurons that are organized in several layers: an input layer, a hidden layer(s), and an output layer (Malhotra and Malhotra, 2003). The input layer of neurons feeds the input variables into the network. The hidden layer is a bridge between the input layer and the output layer. The neurons in this layer are fundamentally hidden from view, and their number and arrangement can typically be treated as a black box to those who are carrying out the system. The function of the hidden layer is to process the input variables. This is achieved by summing up all weighted inputs, checking whether the sum meets the threshold value and applying the transformation function. The weights between the input neuron and hidden neurons determine when each unit in the hidden layer may fire or not and by modifying these weights, the hidden layer may fire or not (Zhang, 2004). In other words, the hidden layers learn the relationship between inputs and outputs in a way similar to that of the human brain by adjusting the weights during the training process (Peel and Wilson, 1996). The function of the output layer is similar to that of the hidden layer. Each input for this layer is possessed as in the hidden layer (Muller et.al, 2009). A specific neural network model is determined by its topology, learning paradigm and learning algorithm (Handzic et.al, 2003).

2.1 Biological Model of Artificial Neural Network

The original goal of the ANN approach was to solve problems in the same way that a human brain would. However, over time, attention moved to performing specific tasks, leading to deviations from biology. Artificial neural networks have been used on a variety of tasks, including computer vision, speech recognition, machine translation, social network filtering, playing board and video games and medical diagnosis. Artificial neural networks born after McCulloc and Pitts introduced a set of simplified neurons in 1943. These neurons were represented as models of biological networks into conceptual components for circuits that could perform computational tasks. The basic model of the artificial neuron is founded upon the functionality of the biological neuron. By definition, “Neurons are basic signaling units of the nervous system of a living being in which each neuron is a discrete cell whose several processes are from its cell body” (Ndinechi et al, 2011), see figure 3.

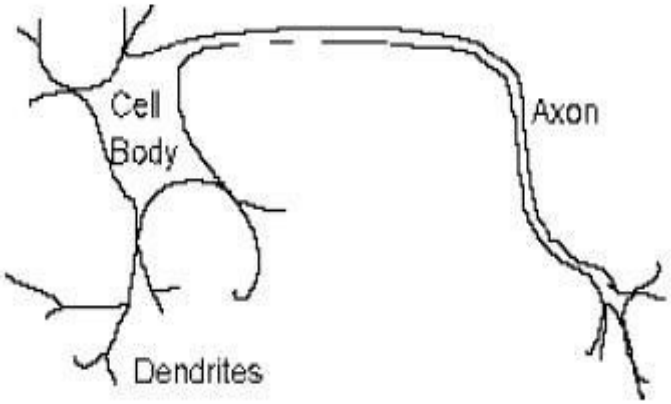


Figure 3: Artificial Neural Network biological model (Ndinechi et al, 2011).

The biological neuron has three main regions to its structure:

i. The cell body, or soma, has two offshoots from it.
ii. The dendrites and
iii. the axon end in pre-synaptic terminals.

The cell body is the heart of the cell. It contains the nucleolus and maintains protein synthesis. A neuron has many dendrites, which look like a tree structure, receives signals from other neurons. A single neuron usually has one axon, which expands off from a part of the cell body. This we called the axon hillock. The axon main purpose is to conduct electrical signals generated at the axon hillock down its length. These signals are called action potentials. The other end of the axon may split into several branches, which end in a pre-synaptic terminal. The electrical signals (action potential) that the neurons use to convey the information of the brain are all identical. The brain can determine which type of information is being received based on the path of the signal. The brain analyzes all patterns of signals sent, and from that information it interprets the type of information received. The myelin is a fatty issue that insulates the axon. The non-insulated parts of the axon area are called Nodes of Ranvier. At these nodes, the signal traveling down the axon is regenerated. This ensures that the signal travel down the axon to be fast and constant. The synapse is the area of contact between two neurons. They do not physically touch because they are separated by a cleft. The electric signals are sent through chemical interaction. The neuron sending the signal is called pre-synaptic cell and the neuron receiving the electrical signal is called postsynaptic cell. The electrical signals are generated by the membrane potential which is based on differences in concentration of sodium and potassium ions and outside the cell membrane.

Biological neurons can be classified by their function or by the quantity of processes they carry out. When they are classified by processes, they fall into three categories(Ndinechi et al, 2011):

i. Unipolar neurons,
ii. bipolar neurons and
iii. multipolar neurons.

Unipolar neurons have a single process. Their dendrites and axon are located on the same stem. These neurons are found in invertebrates.

Bipolar neurons have two processes. Their dendrites and axon have two separated processes too.

Multipolar neurons: These are commonly found in mammals. Some examples of these neurons are spinal motor neurons, pyramidal cells and purkinje cells.

When biological neurons are classified by function they fall into three categories(Ndinechi et al, 2011.

i. The first group is sensory neurons. These neurons provide all information for perception and motor coordination.
ii. The second group provides information to muscles, and glands. There are called motor neurons.
iii. The last group, the interneuronal, contains all other neurons and has two subclasses. One group called relay or protection interneurons. They are usually found in the brain and connect different parts of it. The other group called local interneurons are only used in local circuits.

An artificial neuron mimics the working of a biophysical neuron with inputs and outputs, but is not a biological neuron model. Figure 4 shows the neuron and myelinated axon, with signal flow from inputs at dendrites to outputs at axon terminals. The *network* forms by connecting the output of certain neurons to the input of other neurons forming a directed, weighted graph. The weights as well as the functions that compute the activation can be modified by a process called *learning* which is governed by a *learning rule* (Russell and Norvig, 2010).

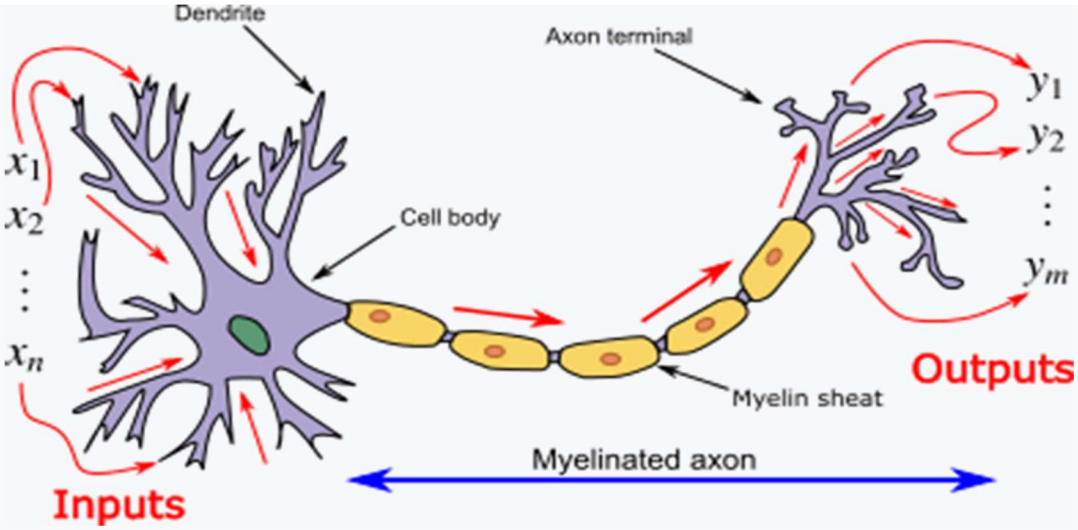


Figure 4: Neuron and myelinated axon, with signal flow from inputs at dendrites to outputs at axon terminals (en.wikipedia.org/neuron)

An ANN is based on a collection of connected units or nodes called artificial neurons, which loosely model the neurons in a biological brain. Each connection, like the synapses in a biological brain, can transmit a signal from one artificial neuron to another. An artificial neuron that receives a signal can process it and then signal additional artificial neurons connected to it. In common ANN implementations, the signal at a connection between artificial neurons are a real number, and the output of each artificial neuron is computed by some non-linear function of the sum of its inputs. The connections between artificial neurons are called 'edges' (Anyaeche and Ighravwe, 2013). Artificial neurons and edges typically have a weight that adjusts as learning proceeds. The weight increases or decreases the strength of the signal at a connection. Artificial neurons may have a threshold such that the signal is only sent if the aggregate signal crosses that threshold. Typically, artificial neurons are aggregated into layers. Different layers may perform different kinds of

2.2 The Mathematical Model

Once modeling an artificial functional model from the biological neuron, we must take into account three basic components. First off, the synapses of the biological neuron are modeled as weights. Let's remember that the synapse of the biological neuron is the one which interconnects the neural network and gives the strength of the connection. For an artificial neuron, the weight is a number, and represents the synapse. A negative weight reflects an **inhibitory connection**, while positive values **designate excitatory connections** (Ndinechi et al, 2011). The following components of the model represent the actual activity of the neuron cell. All inputs are summed altogether and modified by the weights. This activity is referred as a linear combination. Finally, an activation function controls the amplitude of the output. For example, an acceptable range of output is usually between 0 and 1, or it could be -1 and 1.

Mathematically, this process is described in the figure 5

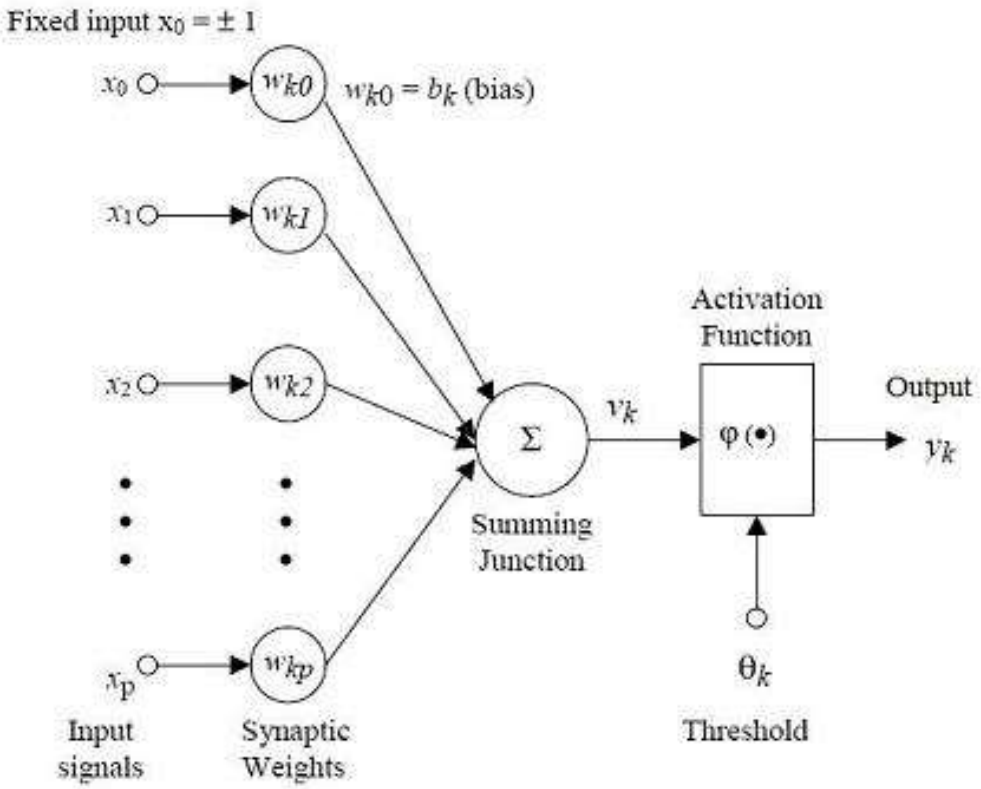


Figure 5: Mathematical Model (Ndinechi et al, 2011).

From this model the interval activity of the neuron can be shown to be:

$$v_k = \sum_{j=1}^{p} w_{kj} x_j \qquad \text{.....(1)}$$

The output of the neuron, y_k, would therefore be the outcome of some activation function on the value of v_k.

Activation functions

The activation function acts as a squashing function, such that the output of a neuron in a neural network is between certain values (usually 0 and 1, or -1 and 1). In general, there are three types of activation functions, denoted by Φ(.) . First, there is the Threshold Function which takes on a value of 0 if the summed input is less than a certain threshold value (v), and the value 1 if the summed input is greater than or equal to the threshold value.

$$\varphi(v) = \begin{cases} 1 & \text{if } v \geq 0 \\ 0 & \text{if } v < 0 \end{cases} \qquad \text{.....(2)}$$

Secondly, there is the Piecewise-Linear function. This function again can take on the values of 0 or 1, but can also take on values between that depending on the amplification factor in a certain region of linear operation.

$$\varphi(v)=\begin{cases}1 & v\geq\frac{1}{2}\\ v & -\frac{1}{2}>v>\frac{1}{2}\\ 0 & v\leq-\frac{1}{2}\end{cases} \quad \text{.....(3)}$$

Thirdly, there is the sigmoid function. This function can range between 0 and 1, but it is also sometimes useful to use the -1 to 1 range. An example of the sigmoid function is the hyperbolic tangent function.

$$\varphi(v)=\tanh\left(\frac{v}{2}\right)=\frac{1-\exp(-v)}{1+\exp(-v)} \quad \text{.....(4)}$$

2.3 The Network Activation Rate

During the training (supervised training) in the artificial neural network, the activation function also called the alpha must be present. This parameter helps to detemine the initial network weights called the initial weight (w_{i1} to w_{in}). The final weights (w_{f1} to w_{fn}) of the network is detemine by the initial weight in addition to the product of the network error vector and the chosen network input.

The final weights (w_{f1} to w_{fn}) = the initial weight (w_{i1} to w_{in}) + (the network error vector)(the chosen network input)(alpha).

The the network error vector is determinmined from the differene between the actual output and the network output.

The network error vector = the actual output - the network output.

3. PROCEDURES FOR GENERATING THE LOAN DECISION SUPPORT SYSTEM ARCHITECTURE

In financial institutions Loan applications can be categorized into good applications and bad applications. Good applications are the applications that are worthy of giving the loan. Bad applications are those ones that should be rejected due to the low probability of the applicants ever returning the loan. The institution usually employs loan officers to make credit decisions or recommendations for that institution. These officers are given some hard rules to guide them in evaluating the worthiness of loan applications. After some period of time, the officers also gain their own experiential knowledge or intuition (other than those guidelines given from their institution) in deciding whether an application is loan worthy or not. Generally, there is widespread recognition that the capability of humans to judge the worthiness of a loan is rather poor (Glorfeld and Hardgrave B 1996). Some of the reasons are:

(i) There is a large gray area where the decision is up to the officers, and there are cases which are not immediately obvious for decision making.

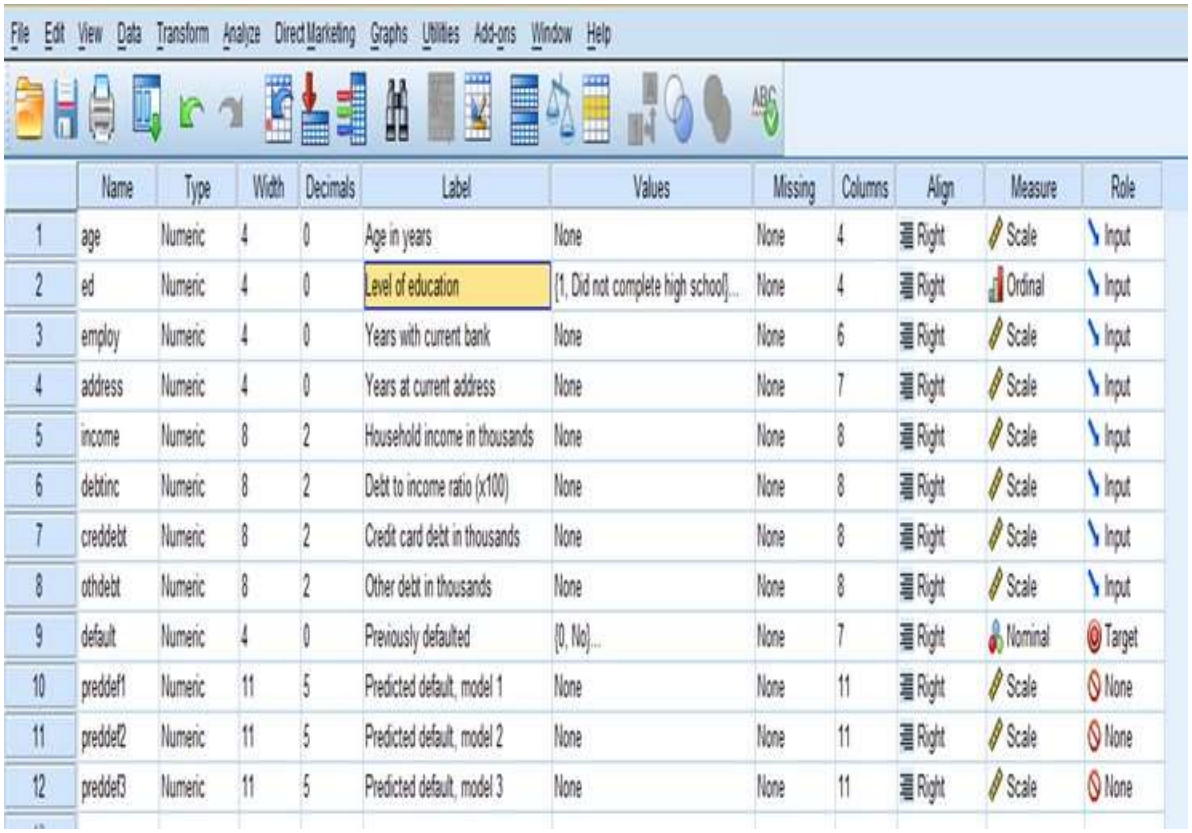
(ii) Humans are prone to bias, for instance the presence of a physical or emotional condition can affect the decision making process. Also personal acquaintances with the applicants might distort the judgmental capability.

(iii) Business data warehouses store historical data from the previous applications. It is likely that there are knowledge hidden in this data, which may be useful for assisting the decision making.

Unfortunately, the task of discovering useful relationships or patterns from data is difficult for humans. The reasons for such difficulties are the large volume of the data to be examined, and the nature of the relationships themselves that are not obvious. Given the fact that humans are not good at evaluating loan applications, a knowledge discovery tool thus is needed to assist the decision maker to make decisions regarding loan applications. Knowledge discovery provides a variety of useful tools for discovering the non-obvious relationships in historical data, while ensuring those relationships discovered will generalize to the new/future data (Bigus, 1996). This knowledge in the end can be used by the loan officers to assist them in rejecting or accepting applications. Past studies show that even the application of a simplistic linear discriminant technique in place of human judgment yields a significant, although still unsatisfactory increase in performance. Treating the nature of the loan application evaluation as a classification and forecasting problem (Thomas, L.C. 1998), have also been offered.

An analytical tool called neural SPSS was used in this research for the training and generating neural network architecture for loans. Below are the procedures that were used.

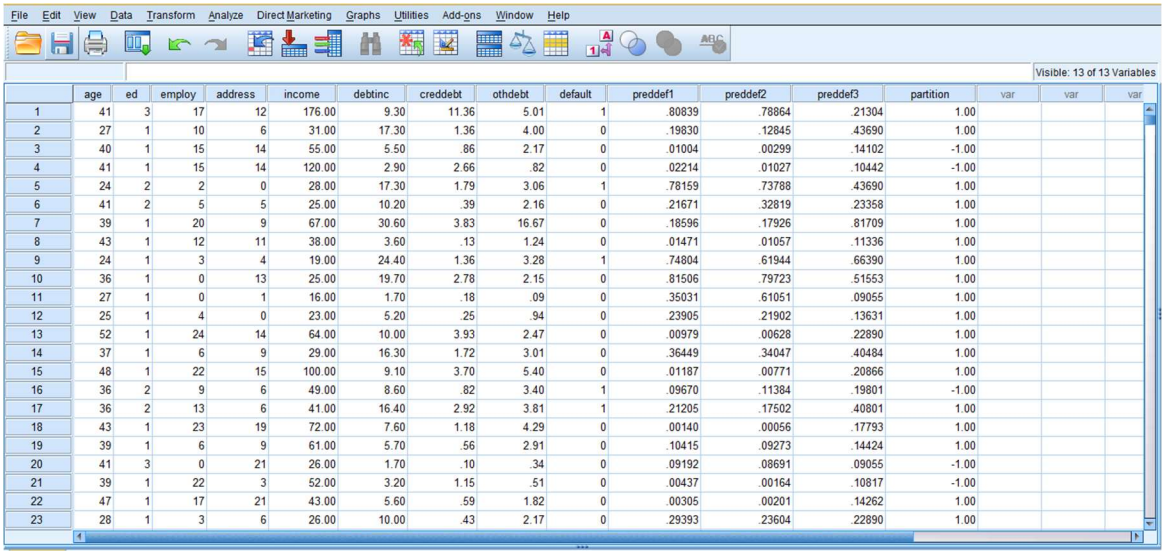
i. The input variables that were used for the analysis were generated according to central bank of Nigeria rules for commercial bank loan see figure 6.



	Name	Type	Width	Decimals	Label	Values	Missing	Columns	Align	Measure	Role
1	age	Numeric	4	0	Age in years	None	None	4	Right	Scale	Input
2	ed	Numeric	4	0	Level of education	{1, Did not complete high school}...	None	4	Right	Ordinal	Input
3	employ	Numeric	4	0	Years with current bank	None	None	6	Right	Scale	Input
4	address	Numeric	4	0	Years at current address	None	None	7	Right	Scale	Input
5	income	Numeric	8	2	Household income in thousands	None	None	8	Right	Scale	Input
6	debtinc	Numeric	8	2	Debt to income ratio (x100)	None	None	8	Right	Scale	Input
7	creddebt	Numeric	8	2	Credit card debt in thousands	None	None	8	Right	Scale	Input
8	othdebt	Numeric	8	2	Other debt in thousands	None	None	8	Right	Scale	Input
9	default	Numeric	4	0	Previously defaulted	{0, No}...	None	7	Right	Nominal	Target
10	preddef1	Numeric	11	5	Predicted default, model 1	None	None	11	Right	Scale	None
11	preddef2	Numeric	11	5	Predicted default, model 2	None	None	11	Right	Scale	None
12	preddef3	Numeric	11	5	Predicted default, model 3	None	None	11	Right	Scale	None

Figure 6:input variables for the training.

ii. The following data sets were used in order to achieve a better result.



	age	ed	employ	address	income	debtinc	creddebt	othdebt	default	preddef1	preddef2	preddef3	partition	var	var	var
1	41	3	17	12	176.00	9.30	11.36	5.01	1	.80839	.78864	.21304	1.00			
2	27	1	10	6	31.00	17.30	1.36	4.00	0	.19830	.12845	.43690	1.00			
3	40	1	15	14	55.00	5.50	.86	2.17	0	.01004	.00299	.14102	-1.00			
4	41	1	15	14	120.00	2.90	2.66	.82	0	.02214	.01027	.10442	-1.00			
5	24	2	2	0	28.00	17.30	1.79	3.06	1	.78159	.73788	.43690	1.00			
6	41	2	5	5	25.00	10.20	.39	2.16	0	.21671	.32819	.23358	1.00			
7	39	1	20	9	67.00	30.60	3.83	16.67	0	.18596	.17926	.81709	1.00			
8	43	1	12	11	38.00	3.60	.13	1.24	0	.01471	.01057	.11336	1.00			
9	24	1	3	4	19.00	24.40	1.36	3.28	1	.74804	.61944	.66390	1.00			
10	36	1	0	13	25.00	19.70	2.78	2.15	0	.81506	.79723	.51553	1.00			
11	27	1	0	1	16.00	1.70	.18	.09	0	.35031	.61051	.09055	1.00			
12	25	1	4	0	23.00	5.20	.25	.94	0	.23905	.21902	.13631	1.00			
13	52	1	24	14	64.00	10.00	3.93	2.47	0	.00979	.00628	.22890	1.00			
14	37	1	6	9	29.00	16.30	1.72	3.01	0	.36449	.34047	.40484	1.00			
15	48	1	22	15	100.00	9.10	3.70	5.40	0	.01187	.00771	.20866	1.00			
16	36	2	9	6	49.00	8.60	.82	3.40	1	.09670	.11384	.19801	-1.00			
17	36	2	13	6	41.00	16.40	2.92	3.81	1	.21205	.17502	.40801	1.00			
18	43	1	23	19	72.00	7.60	1.18	4.29	0	.00140	.00056	.17793	1.00			
19	39	1	6	9	61.00	5.70	.56	2.91	0	.10415	.09273	.14424	1.00			
20	41	3	0	21	26.00	1.70	.10	.34	0	.09192	.08691	.09055	-1.00			
21	39	1	22	3	52.00	3.20	1.15	.51	0	.00437	.00164	.10817	-1.00			
22	47	1	17	21	43.00	5.60	.59	1.82	0	.00305	.00201	.14262	1.00			
23	28	1	3	6	26.00	10.00	.43	2.17	0	.29393	.23604	.22890	1.00			

Figure 7: data set for the training

iii. Set the random seed which allows one to replicate the analysis exactly.
To set the random seed, from the menus choose: **Transform > Random Number Generators.**

iv. In the logistic regression analysis, approximately 70% of the past customers were assigned to the training sample and 30% to a holdout sample. A partition variable will be necessary to exactly recreate the samples used in those analyses.
To create the partition variable, from the menus choose: **Transform > Compute Variable**.

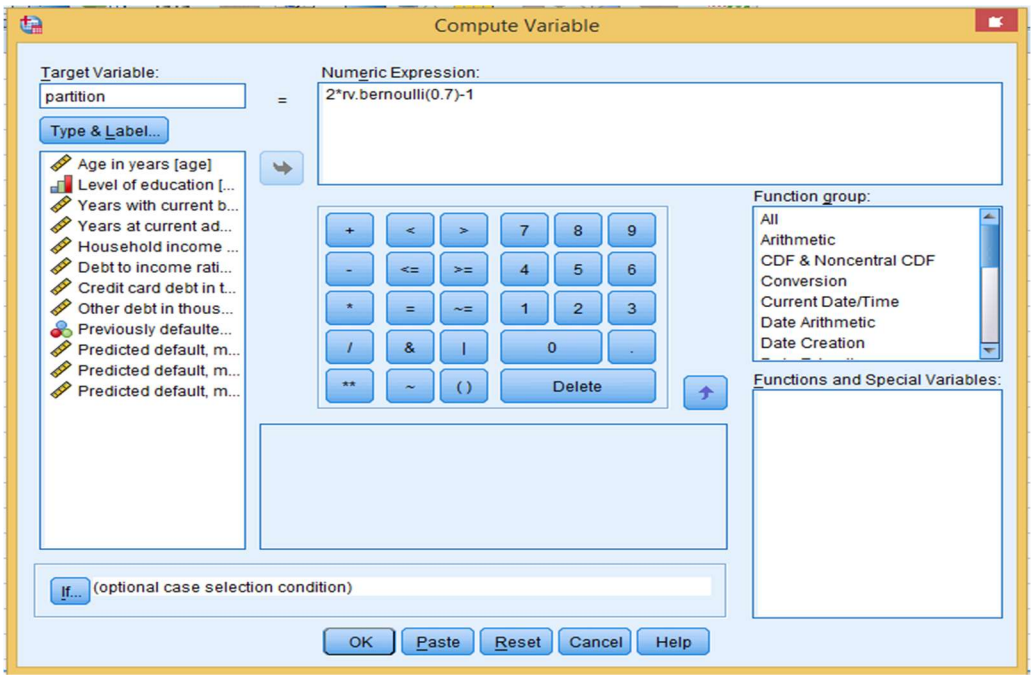


Figure 8: setting the random number

Type partition in the Target Variable text box. Type 2*rv.bernoulli(0.7)-1 in the Numeric Expression text box. This sets the values of partition to be randomly generated Bernoulli variates with a probability parameter of 0.7, modified so that it takes values 1 or −1, instead of 1 or 0. Recall that cases with positive values on the partition variable are assigned to the training sample, cases with negative values are assigned to the holdout sample, and cases with a value of 0 are assigned to the testing sample. Click OK in the Compute Variable dialog box. Approximately 70% of the customers previously given loans will have a partition value of 1. These customers will be used to create the model. The remaining customers who were previously given loans will have a partition value of −1 and will be used to validate the model results.

v. run a Multilayer Perceptron analysis as in figure 8, from the menus choose: Analyze > Neural Networks > Multilayer Perceptron. Select previously defaulted [default] as a dependent variable.

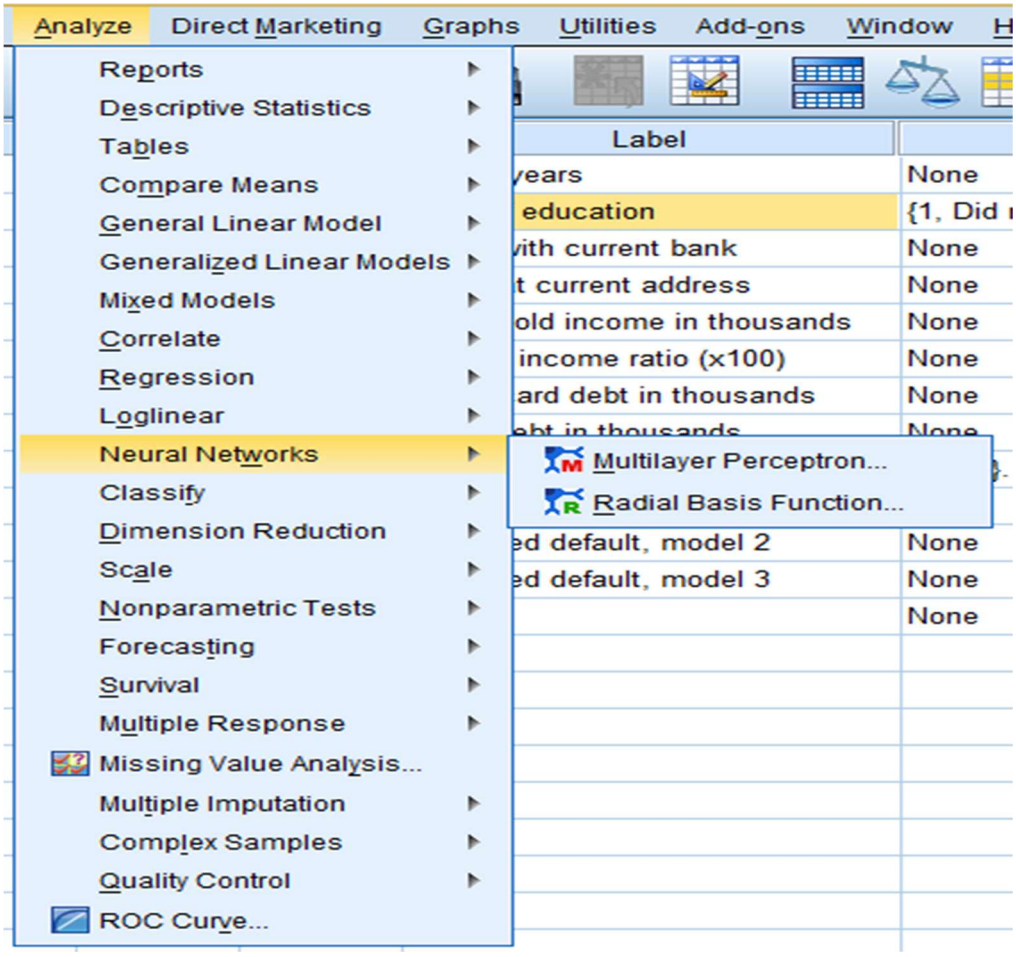


Figure 9: Running A Multilayer Perception

vi. Set network partition variable by

a. Selecting Level of education [ed] as a factor.
b. Select Age in years [age] through other debt in thousands [othdebt] as covariates.
c. Click the Partitions tab

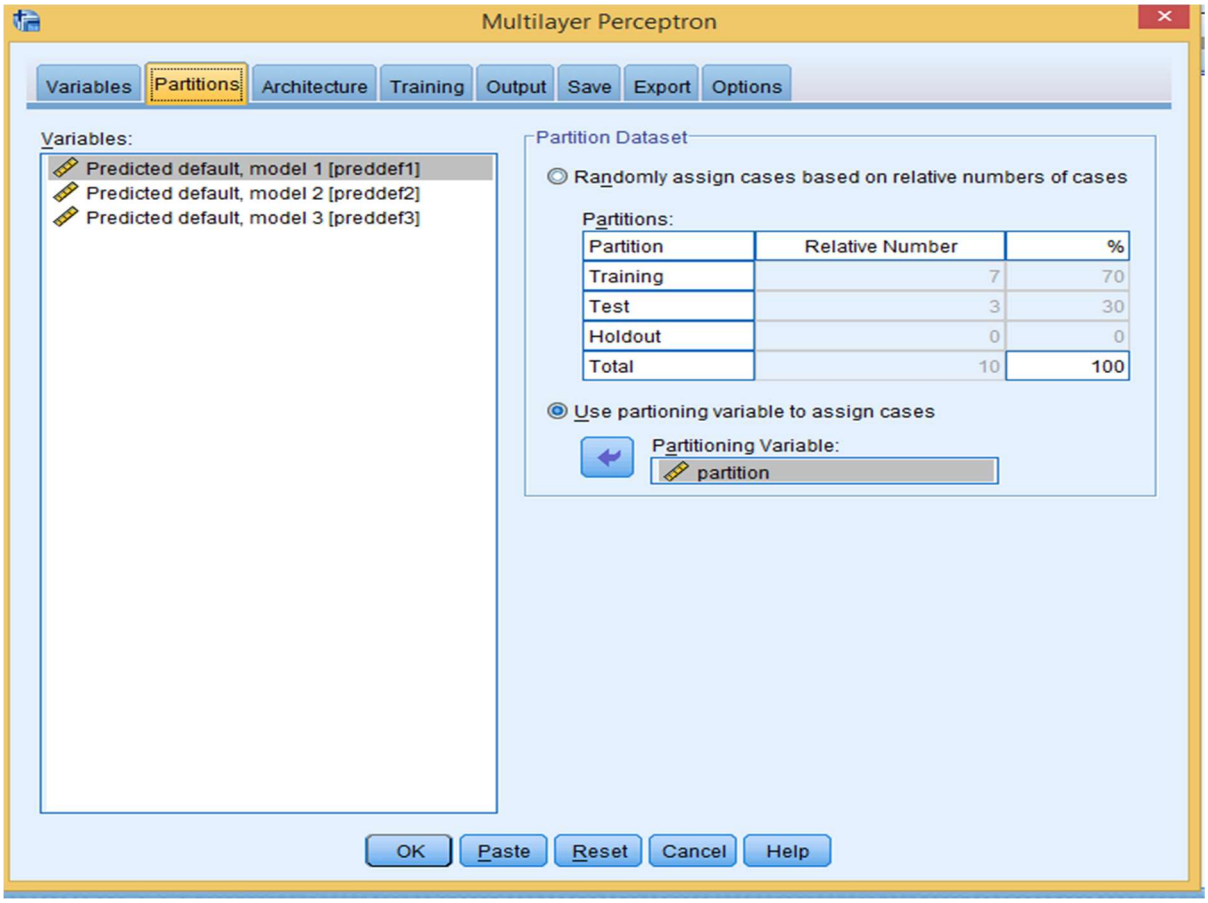


Figure 10: Setting Partition Variable

d. Select Use partitioning variable to assign cases.
e. Select partition as the partitioning variable.

vi.customize the network architecture

a. Click the Architecture tab.
b. Select >> Custom architecture
c. Go to Number of hidden layers select >> one
d. Go to Activation function select>> sigmoid
e. Go to Number of units Select>> custom and type 4 in it
f. Go to Output layer select Activation function Select>> sigmoid

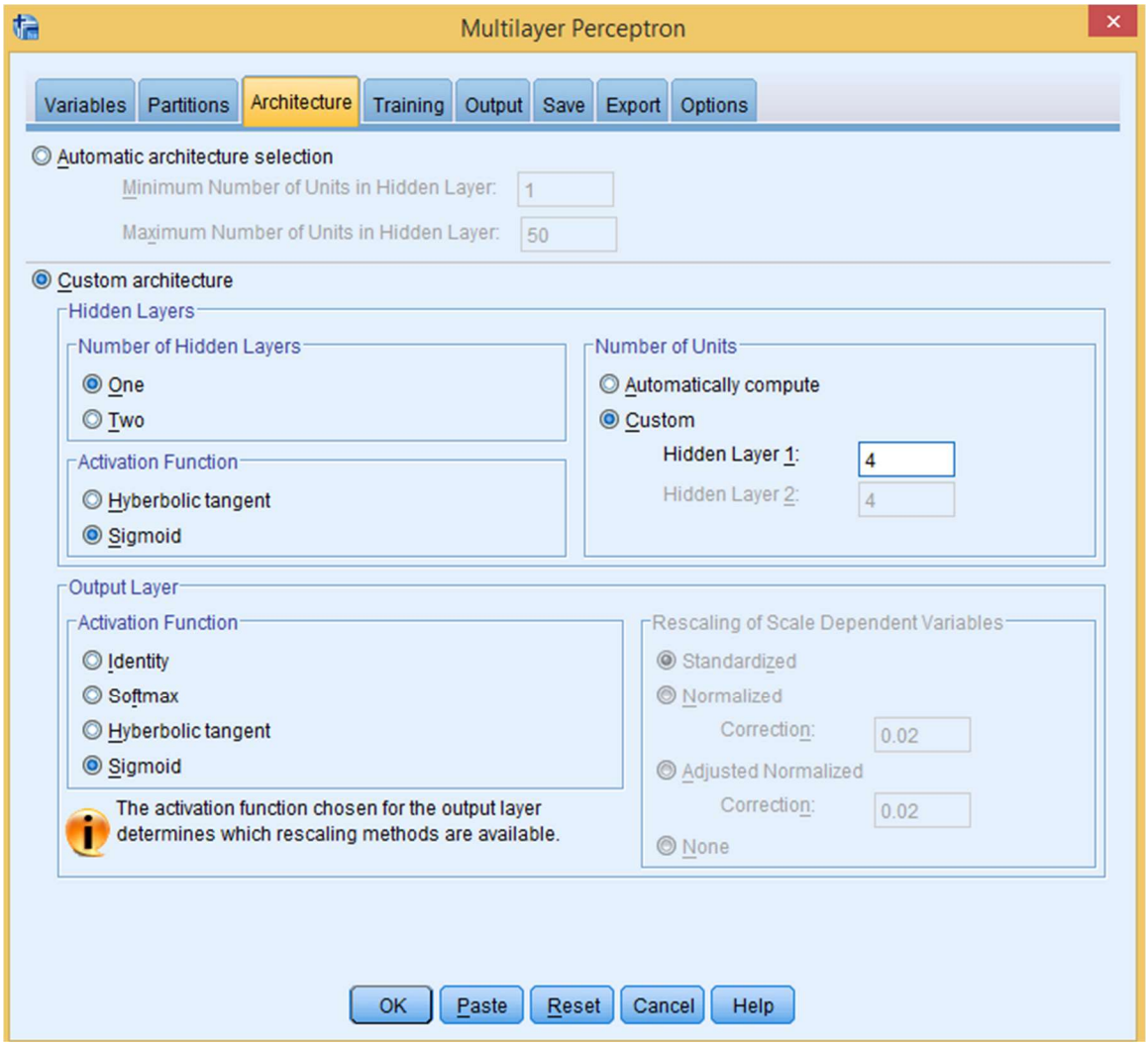


Figure 11: customize the network Architecture

vii. Training the network
   a. click on the training tab.
   b. Go to Type of training **Select>> Batch**
   c. Go to Optimization algorithm **Select>> scaled conjugate gradient**

viii. Then click the output tab.

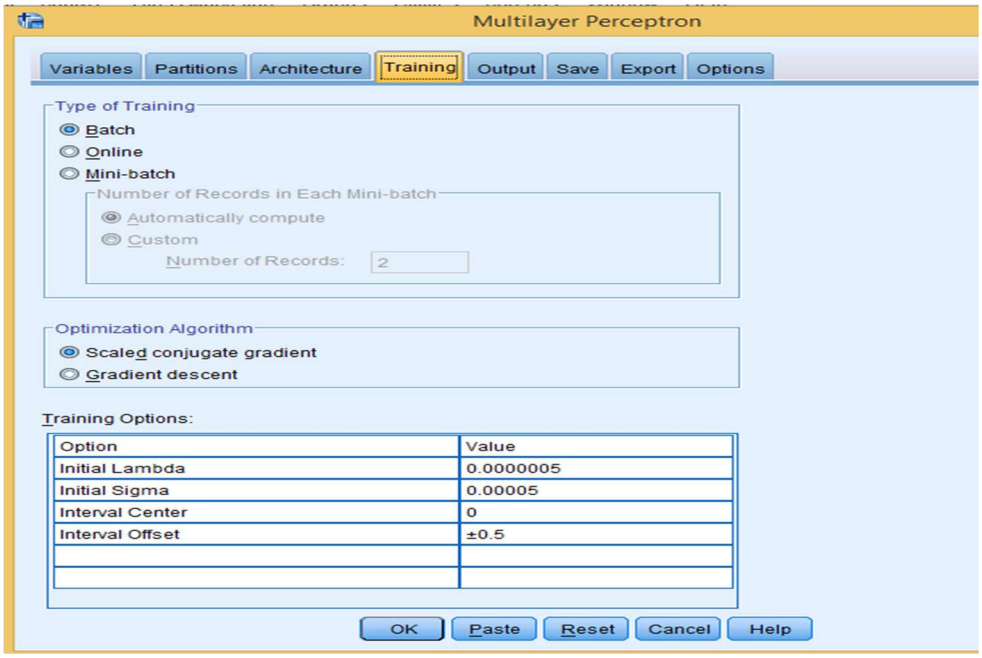


Figure 12: setting the network training parameters

4. THE RESULTS

After the training, several tables and the network architecture were generated. The case processing summary as in table 1 shows that 499 cases were assigned to the training sample and 211 to the holdout sample. The 150 cases excluded from the analysis are the prospective customers excluded from the training.

Table 1: case processing summary

N
Percent
Sample
Training
489
69.9%

Holdout
211
30.1%
Valid
700
100.0%
Excluded
150

Total
850

Table 2: Network Information

Factors	1	Level of education
Covariates	1	Credit card debt in thousands
	2	Debt to income ratio (x100)
	3	Age in years
	4	Household income in thousands
	5	Other debt in thousands
	6	Years at current address
	7	Years with current employer
Number of Units		12
Rescaling Method for Covariates		Standardized
Number of Hidden Layers		1
Number of Units in Hidden Layer 1[a]		4
Activation Function		Sigmoid
Dependent Variables	1	Previously defaulted
Number of Units		2
Activation Function		Sigmoid
Error Function		Sum of Squares

The network information table as in table 2 displays information about the neural network and is useful for ensuring that the specifications are correct. Here, note in particular that:

i. The number of units in the input layer is the number of covariates plus the total number of factor levels; a separate unit is created for each category of Level of education and none of the categories are considered "redundant" units as is typical in many modeling procedures.
ii. Likewise, a separate output unit is created for each category of previously defaulted, for a total of two units in the output layer.
iii. Custom architecture has selected four units in the hidden layer.

Table 3: Model Summary

Training	Sum of Squares Error	46.954
	Percent Incorrect Predictions	13.1%
	Stopping Rule Used	Maximum number of epochs (100) exceeded
	Training Time	0:00:00.28
Holdout	Percent Incorrect Predictions	23.2%

The model summary as in table 3 displays information about the results of training and applies the final network to the holdout sample.

i. Cross entropy error is displayed because the output layer uses the softmax activation function. This is the error function that the network tries to minimize during training.
ii. The percentage of incorrect predictions is taken from the classification table and will be discussed further in that topic.
iii. The estimation algorithm stopped because the maximum number of epochs was reached. Ideally, training should stop because the error has converged. This raises questions about whether something went wrong during training and is something to keep in mind while further inspecting the output.

Table 4: Model Classification

Sample	Observed	Predicted		
		No	Yes	Percent Correct
Training	No	344	16	95.6%
	Yes	48	81	62.8%
	Overall Percent	80.2%	19.8%	86.9%
Holdout	No	142	15	90.4%
	Yes	34	20	37.0%
	Overall Percent	83.4%	16.6%	76.8%

The classification table as in table 4 shows the practical results of using the network. For each case, the predicted response is Yes if that case's predicted pseudo-probability is greater than 0.5. For each sample:

i. Cells on the diagonal of the cross-classification of cases are correct predictions.
ii. Cells off the diagonal of the cross-classification of cases are incorrect predictions.

Of the cases used to create the model, 74 of the 124 people who previously defaulted are classified correctly. 347 of the 375 non-defaulters are classified correctly. Overall, 84.4% of the training cases are classified correctly, corresponding to the 15.6% incorrect shown in the model summary table. A better model should correctly identify a higher percentage of the cases.

Classifications based upon the cases used to create the model tend to be too "optimistic" in the sense that their classification rate is inflated. The holdout sample helps to validate the model; here 74.6% of these cases were correctly classified by the model. This suggests the overall model is correct.

Table 5: Parameter Estimates

Predictor		Predicted					
		Hidden Layer 1				Output Layer	
		H(1:1)	H(1:2)	H(1:3)	H(1:4)	[default=0]	[loan=1]
Input Layer	(Bias)	-2.569	-.654	6.727	2.560		
	[ed=1]	-2.748	1.228	1.041	1.112		
	[ed=2]	4.892	5.531	1.194	5.115		
	[ed=3]	-4.075	-2.235	1.654	-1.685		
	[ed=4]	-1.218	-4.860	1.988	-2.090		
	[ed=5]	.411	-.543	-.085	-.005		
	creddebt	-1.006	3.535	-3.365	-5.798		
	debtinc	.698	1.289	-3.021	5.508		
	age	-1.968	1.087	-.546	1.252		
	income	3.291	-1.831	-.566	2.851		
	othdebt	2.745	.995	-1.209	-3.790		
	address	-.935	-1.195	1.277	.836		
	employ	5.816	.623	6.546	3.019		
Hidden Layer 1	(Bias)					-3.689	3.666
	H(1:1)					5.392	-5.413
	H(1:2)					-5.984	5.961
	H(1:3)					4.019	-4.034
	H(1:4)					5.468	-5.488

Table 5 is the parameter estimates which is the overall summary of the network activity, comprising of twelve input variables, four hidden layers variable and two output processed variables that shows whether to give loan or not. This table shows everything that appear in the network architecture as in figure 13

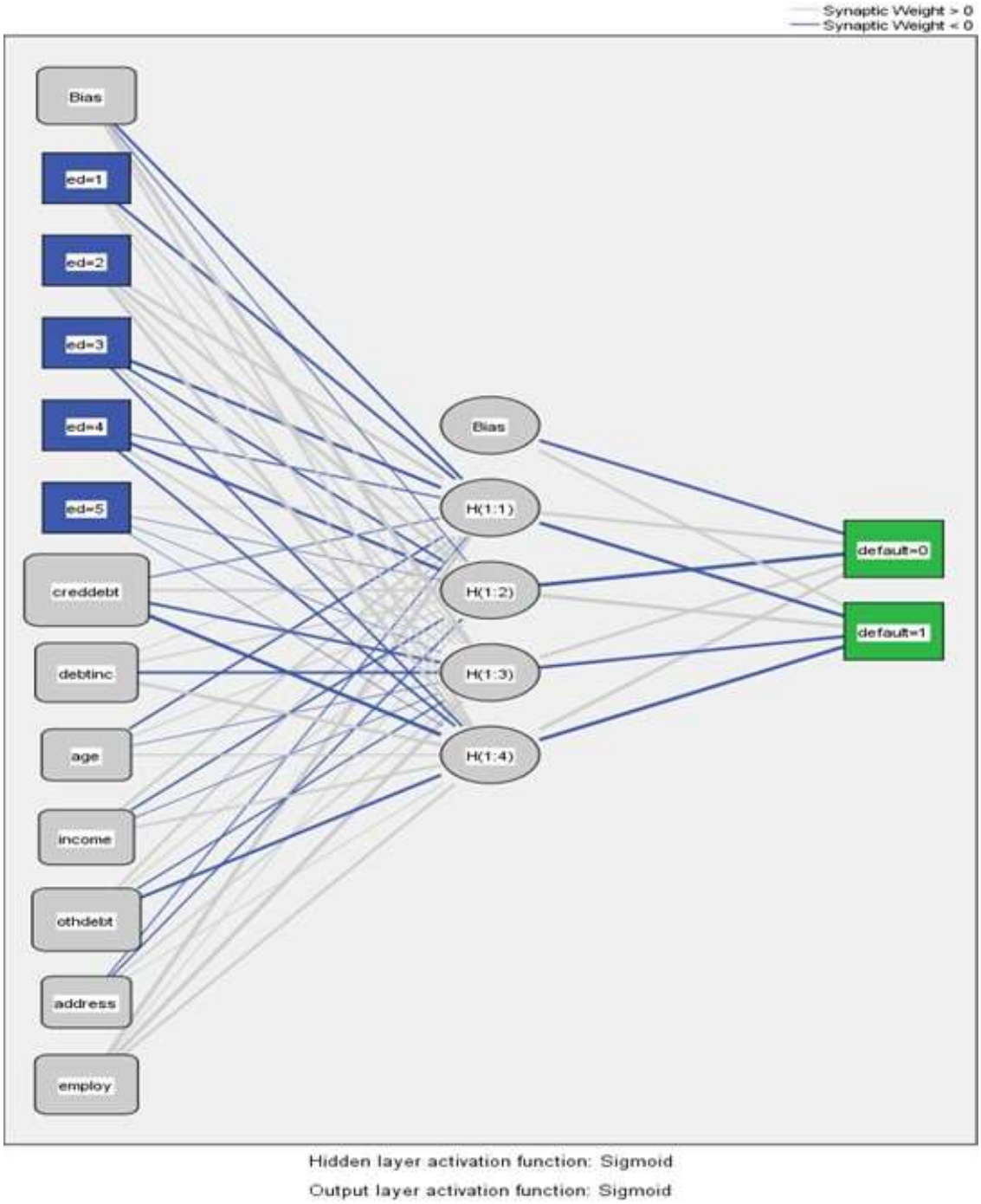


Figure 13: The Network architecture

5. CONCLUSION

The network architecture for the decision support system was generated in figure 13 for the loan decision for commercial banks customers. The architecture will help account officers take automated loan decisions instead of using the conventional method of taking decision concerning loans. This paper developed a model that identifies artificial neural network as an enabling tool for evaluating credit applications to support loan decisions in the commercial banks. A multi-layer feed-forward neural network with backpropagation learning algorithm was used to build up the model. Different representative cases of loan applications were considered based on the guidelines of different banks in Nigeria to generate the neural network model. The system will assist banks in personal credit evaluation in order to reduce the risk of loan default because presently in Nigeria, commercial banks rely on loan officers personal judgement to take credit decisions. Using neural networks in loan evaluation will also reduce any bias or emotional intention that can distort the decision process. This model will also help in reducing the cost of loan processing by personal judgement of these account officers thereby improve the quality of customer services.

REFERENCES

1. Anyaeche C. O. and Ighravwe D. E., (2013). *Predicting performance measures using linear regression and neural network: A comparison.* African Journal of Engineering Research Vol. 1(3), pp. 84-89.
2. Bigus, J.P. (1996). *Data mining with neural networks: Solving business problems from application development to decision support.* McGraw Hill, USA.
3. Cireşan, Dan Claudiu, Meier, Ueli, Gambardella, Luca Maria and Schmidhuber, Jürgen (2010). *Deep, Big, Simple Neural Nets for Handwritten Digit Recognition. Neural Computation.* **22** (12): 3207–3220. arXiv:1003.0358. doi:10.1162/neco_a_00052. ISSN 0899-7667. PMID 20858131
4. Farley, B.G. and Clark W. A. (1954). *Simulation of Self-Organizing Systems by Digital Computer*..IRE Transactions on Information Theory. 4 (4): 76–84. doi:10.1109/TIT.1954.1057468.
5. Glorfeld, L.W. and Hardgrave, B.C. (1996*). An improved method for developing neural networks: The case of evaluating commercial loan creditworthiness.* Computer Operation Research, 23 (10), Pp.: 933944.
6. HandzicMeliha, Tj and Rawibawa Felix and Yeo Julia.(2003). *How Neural Networks Can Help Loan Officers to Make Better Informed Application Decisions, Informing Science In site.* .MITPress.ISBN0-262-63022-2
7. Hebb, Donald (1949). *The Organization of Behavior*. New York: Wiley. ISBN 978-1-135-63190-1
8. Malhorta R. And Malhorta D.K., (2003). Evaluating Consumer Loans Using Neural Networks, Elsevier Science LtD
9. McCulloch, Warren and Walter Pitts (1943). *A Logical Calculus of Ideas Immanent in Nervous Activity*. Bulletin of Mathematical Biophysics. **5**(4): 115–133. doi:10.1007/BF02478259.
10. Minsky, M.; S. Papert (1969).An Introduction to Computational Geometry.MITPress.ISBN0-262-63022-2.
11. Muller G., Steyn-Bruwer and Hamman W. (2009). Predicting Financial Distress of Companies Listed on the JSE-A Comparison Techniques, S.Afr.J. Business Management 40(1).
12. Ndinechi Michael C., Okwu Patrick I, Ezeanyaeji Peter C and Ezekwe Genevra C (2011). *Artificial Intelligence and Expert System for Engineering and Science Students*. Jogene, Onitsha
13. Rumelhart, D.E; James McClelland (1986). Parallel Distributed Processing: Explorations in the Microstructure of Cognition. Cambridge: MITPress.

14. Russell, Stuart J. and Norvig, Peter (2010). Artificial Intelligence A Modern Approach. Prentice Hall. p. 578. ISBN 978-0-13-604259-4
15. ShachmuroveYochanan, (2002). Applying Artificial Neural Networks to Business, Economics and Finance.
16. Sven Behnke (2003). *Hierarchical Neural Networks for Image Interpretation (PDF)*. Lecture Notes in Computer Science. **2766**. Springer
17. Tafti Mohammed H.A. And NikbakhtEhsan. (1993). Neural Networks and Expert Systems: New Horizons in Business Finance Applications, Information Management & Computer Security, and Vol.1no.1.
18. Thomas, L.C. (1998). *A survey of credit and behavioural scoring: Forecasting financial risk of lending to customers.* Retrieved July 5, 2018, from http://www.bus.ed.ac.uk/working_papers/full text/crc9902.pdf.
19. Werbos, P.J. (1975). Beyond Regression: New Tools for Prediction and Analysis in the Behavioral Sciences.
20. Zhang G.Peter. (2004). Neural Networks In Business Forecasting, Idea Group Inc.