CHAPTER ONE

INTRODUCTION

1.1 Background of the Study

The increase in the understanding of technology and its critical role in meeting global challenges and improving the human condition cannot be underestimated in our present and future generations. Presently, on everycontinent, many people are suffering the effects of diminishing health,struggle for appropriate education, and safety of life and property necessitated by natural disasters or man induced activities or both. All over the world, including the most prosperous nations,safe healthcare practice continues to be marred by considerable unprofessionalism and uncontrollable factors leading to improper diagnosis of patients with children and vulnerable senior citizens of the society being the worse hit. The lack of availability of life-saving and life-enhancingtechnologies especially as new cases of diseases appear has been an age long problem plaguing man and threatening its very existence. Infectious diseases remain a leading cause of morbidity and mortality worldwide, with Human Immunodeficiency Virus (HIV), tuberculosis and malaria being the leading causes of most deaths annually. New pathogens continue to emerge, as demonstrated by the Severe Acute Respiratory Syndrome(SARS) epidemic in 2003, the Swine Flu pandemic in 2009,the Middle East Respiratory Syndrome-Coronavirus (MERSCoV) in 2012and the Zika virus in 2016 (Cutler, Fooks,& Poel, 2010; Lee, Haidari, & Lee, 2013; Manheim, 2016;Morse, 1995). The ability of these diseases causing organisms to mutate into different strains further compound the problem of detection and control.

Modern healthcare delivery to citizens has also showed considerable unprofessionalism, negligence and inability to contain emerging and contagious cases of diseases as they make their first appearance in the public domain through clinics and hospitals. For instance, an average scenario of patients trying desperately to see a doctor especially in public hospitals and primary healthcare centres leaves much to be desired; experiences such as undue waste of time, frustration and delays, exploitation and lackadaisical attitude of some of the healthcare providers abound. In a developing country such as Nigeria, where there is consistent trade union and government disagreements over salaries, entitlements, working conditions and health care facilities, life threatening situations and epidemic related conditions may further be compromised by these healthcare institutions. This is further complicated by the fact that diseases outbreaks usually have debilitating consequences, endangering even the healthcare providers' health and the populace if not

detected, isolated and controlled(Friedl & Ceccato, 2010). Even in animal husbandry, farm animals are routinely vaccinated or inoculated against contagious disease such as Contagious Bovine Pleuropneumonia(CBPP), Tuberculosis andRinderpestwith various practices including quarantining and outright slaughtering of infested population. For example, traditional practices such as preventive inoculation of cattle by African cattle owners against CBPP were already in place before the 17th century even when their European counterparts were yet to understand the nature of the contagious disease and depended on drastic measures such as outright slaughtering of infested and in-contact livestock(Blancou, 1996).It is therefore a matter of urgency and necessity for a review of the health institution's modus operandi in tandem with recent developments and best practices available in other climes.

This work recommends a way for healthcare providers to examine patients in emergencies with a view to minimizing contact with yet to be properly diagnosed patients that may be a host to contagious diseases such as influenza (flu), chickenpox and ebola among others. By minimal contact, nurses and other lower cadre healthcare providers who are likely to first receive patients on arrival in the emergency units of hospitals and clinics are allowed to have a general idea of the case they are dealing with based on an expert system comparable to MYCIN(Schneider &Kandel, 1992) such that during the window period prior to when a patient finally sees the doctor, the patient who can be referred to as the index case has not infected the medical personnel on duty or even other patients in the general patient out-ward unit.

The World Health Organization(WHO), through its various programmes have developed framework for diseases' outbreaks, monitoring and evaluation of surveillance and response systems. It has placed emphasis on effective, efficient and early response to contagious and communicable diseases to help mitigate widespread disease occurrence which can reach alarming proportions if not detected early.

On a global scale, WHO continues to prioritize diseases targeted for surveillance and control for a more coordinated response based on availability of funds and resources (World Health Organization, 2006). The international body outlined core functions of surveillance systems to include: case detection, case registration and case confirmation. Other functions outlined by WHO included case reporting, data analysis and interpretation, feedback,epidemic preparedness, response and control.

Table 1.1 Contagious disease affecting humans and primates

SN	Contagious Disease	Mode of Transmission	Vector/Agents (Zoonosis)	Victims
1	Ebola Virus Disease	Contact with blood or fluid of Vector	Rats, Bats	humans and primates
2	Lassa Hemorrhagic Fever	Contact with blood or fluid of Vector	Rats, Bats	humans and primates
3	Marburg Hemorrhagic Fever	Contact with blood or fluid of Vector	Monkeys, Bats	humans and primates
4	Avian flu or Influenza Flu	Contact with blood or fluid of Vector	Poultry, Birds	humans

World Health Organization (2006) also emphasized the usefulness of surveillance data on early warning systems in a surveillance system and asserts that "sensitivity in surveillance refers to the proportion of actual cases in a population that are detected and notified through the (surveillance) system. Sensitivity is particularly important in an early warning system designed to detect outbreaks. It is usually not practical to obtain highly accurate estimates of sensitivity as this requires the true number of cases in the population to be known, something that is almost impossible, and that the diagnosis of reported cases be confirmed to eliminate 'false positives'." The effects of false positive can reduce the sensitivity of surveillance systems as more time and resources may be spent on validation on false alarms thus overwhelming the diagnostic system.

The disease detection and isolation approach adopted by this researchwill help protect everybody involved either directly or indirectly in the event of a disease outbreak. The proposed expert system will help to contain the disease spread, kick-start quarantine procedures where necessary and proffer initial solutions verifiable by trained medical personnel if necessary and thus serve as an electronic form of first aid pending final diagnosis by trained personnel.This system can also serve as a suggestive mechanism for referral of chronic cases to Emergency Diseases Hospitals (EDH) if available within its locality. This work will allow healthcare personnel and even the patients-to-be-diagnosed to respond to basic or simple questions from the developed expert systemthat will be resident in the familiar personal computer and available in the general open ward for out-patients.The expert system could be use to collect symptoms from patients positioned at about one metre away from healthcare personnel for safety reasons and be used to obtain a prediagnosis. It is hoped that by this structured information related approach adopted in this model design, it will thus be

able to help reduce the severity for spreading of contagious diseases bearing in mind that there is normally an ample amount of time wasted while waiting for the doctor or waiting in a queue in hospitals and clinics.

The use of machine learning techniques based on mathematical models are being increasingly used to understand the transmission of infections, monitorpathogenic modes of propagation andidentify causative organisms' targets(Brunton, Proctor & Kutz,2016;Gaebel, Cypko &Lemke, 2016;Peek, Holmes& Sun, 2014). These models also help evaluate the potential impact of control programmes in reducingmorbidity and mortality(Su,Huang, Wu& Zhang, 2006;Dugas et al.,2012). The application of mathematical models such as the Mamdani model for linguistic fuzzy inferencing and Neural Network learning with the capability to detect new epidemic-level contagious disease cases or threats has necessitated this work. Also, this work provides a recommendation for a standard proactive computer-aided clinical procedure aimed at screening every intended patient in real-time whether the patient will be admitted, quarantined, referred or rejected.

1.2 Problem Statement

The need for proactively detecting, diagnosing and predicting the spread of pathogenic disease in real-time has motivated this work as man's imminent future and longevity depends on his ability to contain new cases of contagious diseases and eradicate proven cases of contagious diseases as they occur. Presently on a global scale, the need for people to travel from one part of the world to another has made man the greatest zoonotic (human to human infection) agent. The problem of man's limited memory in retaining precisely,previous accruing data from experience, heuristics (trial and error) and sensor readings and thenprocessingsuch huge collated data effectively without aid has been age-long.The limited capability of humans to attain high data sieving and processing skillsby themselves also made it difficult for man to exist without aid in the form of artificial intelligence embedded ICT tools such as expert systems. These tools will be needed in monitoring systems that report or show physical parameters in real-time,based on sensors and transducers readings for each physical parameter in medicine related fields and others.The modelingprocessin healthcare delivery that utilizes expert system's design is indeed an arduous process especially when non linear systems and inputs are involved.The problem of collecting, processing and analyzing human mostly fuzzy inputs that are made available at instantaneous or dynamic states is indeed an uphill task for today's increasing wireless control world. The need for increasingly complex and distributed automated machinery to be made available and centrally controlled

through the Internet or other communication channels is of paramount importance in today's future systems. This has become extremely important due to the debilitating nature of contagious diseases spread and such a central control must be foolproof, rugged and designed with high integrity and precision.

1.3 Aim and Objectives of the Study

The aim of this work is to develop a fuzzy-neural expert system for contagious diseases detection and isolation. To achieve this aim, the following specific objectives will be pursued:

a. To develop an expert system's simulation of visually obtained patient reported symptoms, compute a fuzzified value of each symptom's contribution and then analyze an appropriate diagnostic result on the patient. The sub objectives for achieving this goal include:
    i. Recommending a specific action to take based on the processed inputs.
    ii. Providing an extrapolative tool fordetecting new cases of emerging diseases in real-time.
b. To obtain a fuzzified value of the level of confidence of diagnosis, compute an error in judgment value based on the number of symptoms processed and the critical action to take as recommended by the expert system.
    i. To reduce or eliminate false alarms and/or errors generated by each diagnostic session by validating with the set of all possible symptoms used in classifying such an ailment.
c. To investigate the performance metrics for various topologies of the artificial neural network (ANN) processing in the developed expert system.
    i. To analyze performance metrics for one hidden layer ANN topology
    ii. To analyze performance metrics for two hidden layer ANN topology
d. To investigate a technique for speeding up the peculiar slow nature of gradient descent optimization of the back propagation algorithm.
e. To validate the developed model with real clinical data obtained from recognized hospitals and research/control institutions.

1.4 Significance of the Study

This research provides an integrated approach to contagious disease detection and isolation by identifying certain signatures or trait patterns of symptoms visible on patients or reported by healthcare personnel that are in minimal contact with suspected patients. These identified patterns wereentered into the developedExpert system and analyzed using mathematical models to obtain a diagnosis of patients' condition,help estimate a risk factor associated with each patient's diagnosisand recommend an action to take. Thus providing a optional way of safeguarding the healthcare provider and other patients around the vicinity of a contagious disease bearing patient. New cases of diseases or emerging diseases present in patients under diagnosis could be further identified by the proposed model, by analyzing the new symptoms with known patterns or signatures of known diseases. Thus, the proposed modelrecommends a standard proactive clinical procedure aimed at silently screening every intended patient on arrival irrespective of whether the patient will be admitted, referred or rejected by host health centre for presence of contagious diseases.

1.5 Scope of the Study

The study is bounded by the following:

I. The use of the Mamdani model to analyze fuzziness in collected patient's symptoms for processing for the presence of contagious diseases using the interval or range of 0.1 to 0.9 inclusive.
II. The pre-processing of patients symptoms by the fuzzy inference unit for input to ANN for further processing.
III. Mathematical analysis of the gradient descent optimization method used by the back propagation algorithm.
IV. Investigation of the various topologies of ANN to obtain best features for implementing the resulting expert system.
V. The scope of this research also covers the use of the euclidean distance function of the k-nearest neighbor algorithm for extrapolating new cases of contagious borne disease.

1.6 Limitations of the Study

The following channels will be used to span through this research:

I. The input data range for analysis in this work is limited to presented symptoms by patient (i.e. without any major pre-testing, puncturing or operation carried out) since this research is intended to obtain a discrete diagnosis.

II. The fuzzification process is limited to the use of the Mamdani or Takagi-Sugeno zero order model and thus cannot be used for approximation of problems represented by n – order polynomials when $n > 0$.

III. The input range for data must be normalized to between 0.1 to 0.9 (inclusive) interval for normal operation of the developed model.

IV. The study is limited to an analysis of a maximum of thirty inputs to both the fuzzy processing and artificial neural network processing to avoid the problem of over parameterization.

V. Due to the peculiar slow nature of gradient descent optimization, the developed back propagation algorithm is limited to a topology of a maximum of two hidden layer artificial neural network.

1.7 Dissertation Outline

The dissertation is organized into five chapters with chapter one giving an introduction and a brief history of diseases spread and epidemic occurrence in the world. It also states the specific objectives this work is aimed at, the direction and work extent. The literature review of various authors' materials in medicine, engineering, education and other areas is in chapter two. The review covers the concept of dependence on machine learning techniques based on artificial intelligence agents like Artificial Neural Nets and Fuzzy Logic for model analysis. Artificial Neuro-Fuzzy Inference System, ANFIS modeling, data acquisition and simulation. Chapter two concludes with an analysis of existing systems. Chapter three outlines the tools and methodologies utilized in this research by enumerating the methods of machine learning techniques used. The use of the Neuro-fuzzy techniques and its inference system was elaborated in this chapter. The system's design approach and the proposed system are also presented. Chapter four shows the preliminary use of the model, the training of the designed ANFIS system and the resulting simulation of fictitious patient symptoms that characterize contagious and non contagious disease cases. It also captures the various tests, data analysis and validation carried out on real data from Federal Teaching Hospital Abakaliki, Ebonyi State, Institute of Lassa Fever Research and Control at Irrua Specialist Teaching Hospital, Edo State and data from WHO Ebola Response Team to validate the results obtained. Chapter five covers the inferences drawn as a result of this work and some recommendations proffered.

CHAPTER TWO

LITERATURE REVIEW

Since the advent of the computer, it has become very clear that man's control of his environment could be better enhanced by the dependence on the increasing processing power, speed and memory capacities of today's computers and its peripherals. The use of ICT based tools to aid diagnosis of health related symptoms in patients especially deployment of medical expert systems was reviewed in this chapter. A quick look into the mathematics used in the modeling of machine learning based control systems was appraised and the rudiments involved in the design of massively parallel processors to handle huge computational task in real-time was also reviewed.

An examination of various literatures on the applications of machine learning techniques on related dynamic systems and classification of already available expert systems into categories was obtained in this chapter.Also, descriptions of some symptoms and/or signs that were obtained from field work to hospitals and research institutions that have dealt with contagious diseases caseswere collated and reproduced in this chapter. The various existing and 'still in use' systems in today's disease contagion-unaware healthcare system were summarized in a flowchart with highlights on the limitations and flaws in the system.

2.1 Use of ICT to Aid Health Diagnosis

The use of computers to aid health care providers and patients have been available as far back as 1970s with the design and deployment of MYCIN, a medical expert system for diagnosing ailment with mostly bacteria origin. MYCIN was designed based on rule base reasoning that was premised on certainty factors of symptoms present in patients(Shortliffe, 1976). Another expert system with diagnostic abilities is the CASNET which was designed to help medical personnel to detect ailments and diseases in patients by considering the most significant result of tests(Wikipedia, 2015). It uses semantic network representation formalism, having nodes which represent disease states with attached weights to determine how they relate under some appropriately defined relationship. An expert system according to Basheer and Hajmeer (2000) is a computer program that mimics the human reasoning process, which relies on logic, belief, rule of thumb, opinion and experience. Various medical expert systems abound and their list continues to increase as new problems and information concerning new treatment or diagnosis continue to emerge. For example, Asabere (2012)developed a Mobile Medical Expert System (mMES) using mobile devices with embedded computing technology that is hosted in a Cloud based system (CBS). They stated

that “Medical Doctors in Ghana can speed up diagnosis, confirm their own diagnosis, provide advice on found diagnosis and provide advice on certain diseases when diagnosed on a patient” while using mMES.A tabulation of some of the expert system in useis shown in Table 2.1.

Table: 2.1 Categories of Expert System. Source: (Wikipedia, 2015).

Category	Problem Addressed	Examples
Analysis	Inferring situation descriptions from sensor data	Hearsay (Speech Recognition), PROSPECTOR
Prediction	Inferring likely consequences of given situations	Preterm Birth Risk Assessment
Diagnosis	Inferring system malfunctions from observables	CADUCEUS, MYCIN, CASNET, PUFF, Mistral, Eydenet, Kaleidos
Design	Configuring objects under constraints	Dendral, Mortgage Loan Advisor, R1 (Dec Vax Configuration)
Planning	Designing actions	Mission Planning for Autonomous Underwater Vehicle
Monitoring	Comparing observations to plan vulnerabilities	REACTOR
Debugging	Providing incremental solutions for complex problems	SAINT, MATHLAB, MACSYMA
Repair	Executing a plan to administer a prescribed remedy	Toxic Spill Crisis Management
Instruction	Diagnosing, assessing, and repairing student behavior	SMH.PAL, Intelligent Clinical Training, STEAMER

Due mainly to the dynamic nature of diseases occurrence (Reddy, 2003), complexity in its spread as diseases can spread rapidly through countries and national borders (Manheim, 2016) and sensitive factors involved in treatment and isolation, coupled with the professional expertise needed for building databases, verifying and deploying tools for patients’ diagnostic purposes,CNN (2014) much has not really been done in the scientific community especially in deployment and use of ICT diagnostic tools for disease isolation in developing countries. This was observed in the 2014 outbreak of the Ebola Viral Disease (EVD) in West Africa which claimed the lives of several healthcare providersand was reported by CNN (2014) to have

escaped detection when it was ‘exported’tothe US, through a patient known as Thomas Eric Duncan in Texas Health Presbyterian Hospital (THPH). Though the nurse on duty at THPH obtained from the patient his travel history of just coming from Ebola ravaged Liberia, according to the report, the information was fully communicated to the medical team and yet was not acted upon. This negligence affirms the lack of structured diagnostic procedures even in developed countries. Clearly, ICT basedstandard health procedures could have helped in preventing the death of the healthcare providers who probably due to the fact that EVD contagion was not rampant then and infected patients showed symptoms similar to fever, malaria andflu which are not contagious. All these would have made the healthcare providers to initiate very low alert levels in dealing with the EVD case. Statistics from the Centers for Disease Control and Prevention (CDC)on the epidemic which claimed several thousands of people worldwideplaced the total cases (suspected, probable and confirmed) at 26,325 and total deaths at 10,905 as at 28th of April, 2015 (Centers for Disease Control, 2015).

An ICT supported healthcare system using expert system’s artificial intelligence and capability to control/analyze huge data in real-time will provide the necessary large database that can contain all possibility of the symptoms of a disease(Asabere, 2012). This system should be able to identify even Emerging Infectious Diseases (EIDs) for which Vrbova et al.(2009) puts the number to be increasing globally over the past 50 years with estimates of the proportion of EIDs that involve pathogens transmitted from animals to humans, or *zoonoses*, ranging from 60% to 75% of contagious diseases. Although the number according to Vrbova et al.(2009) of “Emerging zoonoses can become devastating if they become transmissible from person to person. For example, the complete genetic characterization of the pandemic (in) 1918 (of the) “Spanish Flu” virus suggests it not only originated from an avian influenza virus, but that the pandemic virus was in fact an adapted avian influenza virus.Thesefindings show that zoonotic agents can result in severe impacts with minimal genetic changes, in this case increased severity and facilitated human to human transmission, some of which are already present in the current circulating avian viruses”. EVDs zoonotic agents include bats, monkeys, apes and duikers (African Antelopes). They pointed out that “society would be better prepared to detect and prevent EIDs if we can get ahead of the curve; if we are able to identify risky situations before the first cluster of cases in humans are identified in hospitals”.

The increasing use of ICT and its tools has witnessed an unprecedented growth in almost all areas including security and crime detection (Badiru, Asaolu & Omitaomu, 2006) where uncertain and vague responses of eyewitnesses’ account to a crime was used as inputs

to a machine learning based neuro-fuzzy inference system for a pattern generation which helped law enforcement agents to track criminals and suspects. ICT based machine learning has also been used in teaching and learning (Reamon & Sheppard, 1997).Such a learning scheme provides real world simulation of teachers' intent; for example, a simulated practicalthat allows the performance of surgery on virtual patients using computer aids in ICT supported teaching hospitals. These virtual simulationswhich can be repeated over again allow students' grasp of intended knowledge or training when compared to students depending on books alone.Healthcare though has witnessed a few use of ICT tools over the past few decades will continue to witness deployment of ICT tools as man continue to understand his natural, physical and biological world. Hudson and Cohen (1992) developed a medical expert system based on approximate reasoning from a rule-based system to analyze chest pain in an emergency room environment. The expert system which they called EMERGEprovided rapid decision making support to doctors by utilizing certainty factors to indicate the seriousness of the patients illness. EMERGE according to developers however lacked the ability to record nuance (high distinction) and did not intuitively appear to follow the same processes as the human reasoning process.

Nigel et al. (1992) investigated the difficulties that even experienced clinicians face in diagnosing melanoma, an uncommon skin disease that involves growth of moles or tumors on various parts of the patient's skin.Melanoma occurrence and diagnosis according tothem continued to be difficult to distinguish from non-melanoma pigmented lesions such as basal cell papillomas. This haveresulted in the health system depending on image processing for accurate detection based on the presence of attributes such as asymmetry of the shape, diameter and height of the tumor in the irritated skin of the diagnosed patient.

2.1 Recent Researches in Medical Expert Systems

The proliferation of medical expert systems for better decision supporting in healthcare, continue to be on the increase since the advent of MYCIN owing to the increased dependence on mathematical models (Avci & Dogantekin, 2016;Peek, Holmes& Sun, 2014) and the huge computational power of today's computers(Saha, 2014). Mathematical models that have evolved includes: fuzzy logic based (Ajmalahamed, Nandhini, & Anand, 2014; Madaan & Garg, 2016; Sharma & Choudhary, 2015; Singla, Grover & Bhandari, 2014), knowledge based and web/cloud based (Bourouis, Feham, & Bouchachia, 2014). These models have culminated to even being hosted in handheld devices (Isinkaye, Awosupin & Soyemi, 2017; Singh, 2014). Today's expert systems continue to find deployment in various

delicate areas of medical diagnosis including in infants' related ailments (Naser & Hamed, 2016) and diagnosis of the elderly (Pierleoni, et al. 2014).

2.2 Interactive Models versus Black Box Modeling

Another work worth reviewing is that by Babuska and Verbruggen (2003) in whichthey showed that nonlinear system identification using neuro-fuzzy methods, though captures non linear behavior of control systems and possess high degree of uncertainty or time-varying characteristics over conventional modeling approaches may not necessarily possess improved performance after training especially for dynamic systems. They stated that neuro-fuzzy models described systems by means of fuzzy if-then rules represented in a network structure to which learning algorithms known from artificial neural networks can then be applied.These algorithms are structured such that they are transparent to interpretation and analysis. That is, they can be better used to explain solutions to users than completely black box models such as neural networks only. Their work identified two main fuzzy models: the Mamdani (or linguistic) model, used mainly in knowledge-based expert systems and the Takagi-Sugeno model. According to Babuska and Verbruggen(2003) the Takagi-Sugeno model is able to analyze fuzziness in huge data mining problems or in a collection of huge data that had a fuzzy membership. Thus necessitating new approachesto huge data analysis and mining whichculminated in the introduction of fuzziness into consumer electronics and state of the art applications such as automobiles, airplanes and integrated missile lauching and tracking systems. In these systems,fuzzy data from several sensors readings in real-timearemore keenly monitored, analyzed and a control strategy adopted in the designed.

The Takagi-Sugeno (data-driven identification) model represented by

$$R_i\text{: If } x \text{ is } A_i \text{then } y_i = a_i^T x + b_i; \qquad (2.1)$$

R_i represent the fuzzy proposition been investigated or represented, A_i represents the antecedent part (i.e If-part of the rule), yi is then the consequent part (i.e then-part of the rule), $y_i$is an affine linear function of the inputs variables such that a_i is the consequent parameter vector, b_i is a scalar offset and i= 1,2, … K (Babuska & Verbruggen, 2003). This allows huge database to be mined by seeking a relationship between contributing data pairs in such a way that the then-part of the fuzzy proposition links up all necessary contributing paramenters in returning the result of processing. The Takagi-Sugeno model which can be regarded as a smooth piece-wise linear approximation of a nonlinear function or a parameter scheduling model combines a linguistic description with standard functional regression to output the form

$$y = \frac{\sum_{i=1}^{K} \beta_i(\boldsymbol{x}) y_i}{\sum_{i=1}^{K} \beta_i(\boldsymbol{x})} = \frac{\sum_{i=1}^{K} \beta_i(\boldsymbol{x})(\boldsymbol{a}_i^{\mathrm{T}} \boldsymbol{x} + b_i)}{\sum_{i=1}^{K} \beta_i(\boldsymbol{x})} \qquad (2.2)$$

where $\beta_i(x)$ is the degree of fulfillment of the *i*th rule. This allows the fuzzification and approximation of n – order polynomial with up to n = 2. They combined Equation 2.2 with error propagation of dynamic neural networks to obtain the adaptive network based neuro-fuzzy inference system (ANFIS) models which uses global and local least squares estimation in estimating the parameters of the system. Due to the inherent problem of obtaining a good fit and local behavior of the system with these schemes, constrained estimation techniques were suggested to account for overall system stability. Babuska and Verbruggen(2003)carried out experiments based on their analysis with simulation examples of hybrid learning using the ANFIS function of the MATLAB Fuzzy Logic Toolbox. In these experiments, the membership functions were adjusted by the gradient-descent optimization method and showed that while data training improved for each iteration, data validation became less accurate leading to the problem of over training. Clearly this kind of behavior is difficult to predict for real world dynamic systems such as the modeling of the diagnostic process in healthcare. However, they concluded by asserting that drawbacks of neuro-fuzzy modeling is that current techniques for constructing and fine tuning fuzzy models are rather complex methods requiring specific skills and knowledge of the system under investigation. They suggested that neuro-fuzzy modeling should be used as an interactive method, facilitated by active participation of the user(s) in a computer-assisted modeling session. Presently, methods of fine tuning fuzzy models has evolved from robust tools such as the fuzzy control toolbox in MATLAB that allows users greater flexibility and control of the method.

Babuska(2002)in a research that involved the time series modeling of River Baitarani basin, one of the major rivers in Orissa state of India, observed that ANFIS models outperformed their Artificial Neural Networks (ANN) counterparts while still preserving the capabilities of the ANN.Though they observed that by increasing the number of membership functions assigned to each input to the ANFIS, the model performance did not improve; rather on the contrary, it increased the model complexity and parsimony. However, Babuska, 2002observed that unless carefully trained the ANFIS model's performance may not be satisfactory. In the resulting hydrographs from training the ANFIS model that were previously transformed to time series, an improvement in performance of the model compared to a model with non-transformed data was observed. These results indicated that if the data used for analyzing ANFIS models are normally distributed thenimproved performance will be obtained by the model. The improved performance according to them may have resulted because the mean square error function was used to optimize the parameters of the ANFIS structure. In general,their results were found to be highly promising and a comparative analysis also suggested that the proposed modeling approach outperformed ANNs and other traditional

time series models in terms of computational speed, forecast errors, efficiency and peak flow estimation. A significant improvement was observed for the ANFIS in the peak flow prediction compared to ANN. In addition, although both models underestimated the peak flow due to abrupt fluctuations due to the dynamic nature of the river flow, the ANFIS underestimated it by 11.71% as opposed to 34.26% for the ANN. It was noted that the ANFIS model reached convergence in just 20 epochs, while the ANN model took more than 300 epochs, implying considerable savings in computational time for ANFIS models. Their results further suggested that the model building process can be simplified when an ANFIS model is developed compared to an ANN, as the ANFIS model preserves the full potential of ANN models in its performance and much more.Critics of ANN such as Prasadl et al.(2011) having studied various machine learning algorithms suggested the use of Particle swarm optimization (PSO) as a more promising method to design and develop expert systems for medical diagnosis. They claimed that PSO's implementation is faster and provides better results than the Back Propagation algorithm of ANN which involves adjusting of network weights and building of topological structure.

2.3 Modeling Complex Systems with Several Sub Units

Kavulya et al. (2012) posited that failure diagnosis is a highly technical and challenging endeavor consisting of the process of identifying the fault that had led to an observed failure of a system or its constituent components and the relationship between faults, failures, and their observable symptoms. For example, they stated that in complex systems single faults often produce multiple symptoms in different parts of the system, e.g, a mis-configuration fault in a critical network component such as a Dynamic Host Configuration Protocol (DHCP) server can cause all client computers on the network to fail; conversely, similar symptoms may be caused by many different types of faults, e.g, the failure of a networked computer to receive an IP address may have several causes including, but not limited to, packet loss in the physical network, a client mis-configuration, or a problem with the DHCP server. They pointed out that though most fault diagnosis werepredominantly manually inclined, more automated fault diagnostictechniques are emerging. These automated problem diagnosis techniques localizes the most likely sources of a problem to a set of metrics (e.g, anomalous CPU usage), a set of nodes (e.g, anomalous web server), or a type of problem (e.g, using known problem signatures to identify mis-configuration). Operators can then use the output of automated problem diagnosis to guide root-cause analysis by analyzing source code, or hardware and software settings at the identified culprits terminals. For example, an

examination of the source code at the web server might show that the anomalous CPU activity at the web server was due to an infinite loop in a scheduling function. They provided a broad overview of automated techniques for fault diagnosis ranging from knowledge-based techniques that encode expert knowledge in the form of rules or system models to model-free techniques that rely on statistical correlations, regression, and machine learning to perform some aspects of the diagnosis task without any prior human knowledge. They also provided examples of industrial applications in which automated diagnosis has proven to be a valuable tool for ensuring and evaluating resilience, elaborating on telecommunications and Internet services that have to deal with issues of scale; and automotive and aerospace systems that have to deal with the absence of human expertise when problems occur. As challenge for future research,they identified dynamic problems that occurred due to emergent, unpredictable behaviors inherent in complex nonlinear systems and the need for recovery techniques to automatically act upon the output of diagnosis algorithms.

2.4 Serial versus Parallel Data Processing

Topping et al.(1998) investigated the rudiments of parallel and distributed computing which utilizes neural networks and genetic algorithm processing techniques. They illustrated how a parallel finite element analysis could be undertaken in an efficient manner by preprocessing of the finite element model using a genetic algorithm utilizing a neural network predictor. They partitioned the finite element mesh into sub-domains using Sub-domain Generation Method (SGM) to ensure load balancing and minimum inter processor communication during the parallel finite element analysis on a Multiple Instruction Multiple Data (MIMD) with distributed memory computer. The result of computations using the resulting transputer-based system was then sent to the central or root processor which then handles all the information for the final refined mesh analysis of the finite element coming from the host processors. In parallel genetic algorithm implementations, they identified three models namely, global, island and cellular for selection, crossbreeding, mutating and evaluating of genetic offspring's manipulation; methods which eliminated the bottlenecks of serial or sequential models. Though each method has its merits and demerits in their manner of genetic manipulations and time spent, however lack the portability required for parallel processing as they rely on complex topologies of processors and very expensive hardware.

Closely related to Topping et al.(1998)work is the programming of massive parallel processorsKirk and Hwu(2010) which allowed for routing of jobs or activities through a process of decomposition of a domain problem into properly defined coordinated work units.

These defined units tasks can then be realized with efficient numerical methods and well known algorithms. They stated that “a programmer with strong computational thinking skills not only analyzes but also transforms the structure of a domain problem by determining which parts are inherently serial, which parts are amenable to high-performance parallel executions, and tradeoffs involved in moving parts from the first category to the second”. They establish that;“a strong combination of domain knowledge and computational thinking skills is often needed for creating successful computational solutions to challenging domain problems”.Theyalso listed three goals of parallel programming to include solving a given problem in less time, solving bigger problems within given amount of time. The third goal of using parallel programming is to achieve better solutions for a given complex problem in a given amount of time. Thus increased speed is the primary motive for using parallel processing. A parallel processing process can be divided into these various steps (Kirk & Hwu, 2010):

a. Problem decomposition to identify each unit’s work to be performed by parallel execution.
b. Algorithm selection which involves a step-by-step procedure where each step is precisely stated and can be carried out on a computer. (It involves definiteness, effective computability and finiteness)
c. Implementation in a programming language such as C, C + +, VB and
d. Performance Tuning.

2.5 Industrial Application of Machine Learning

Gao(2003) investigated machine condition monitoring and fault diagnosis using Neural Networks in a research sponsored by US National Science Foundation. The author stated that sensor-based machine condition monitoring provided insight into the manufacturing process enabling effective high-decision making for quality production at lower costs. The lower costs was based on diagnosing and/or prognosing (prediction of the remaining service life of a machine) the health status of a machine to avoid catastrophic breakdown. Gao (2003) identified the major challenge for machine condition monitoring to be differentiating changes in signals reception from sensors measurement of machine defects from that of changing operating conditions and ambient noise as these defects especially at their incipient stage are “fuzzy” and highly complex to identify.

Schneider et al. (1992) developed a fuzzy intelligent autonomous expert system named COMEX. This systemutilized a variety of autonomous external sensors to collect and update channel and network communication parameters so that an operator which initially supplies the learning information for the system to ‘study’ can use COMEX to establish, monitor and control a desired communication in an optimal manner. COMEX which runs on an IBM

compatible PC or any other personal computer uses three separate knowledge bases for three distinct phases of the decision process and can be used by any operator to achieve expert results in communication systems.

Anextensive and elaborate mathematical review of stability of non linear systems and a proposed controller that uses multiple parameter models to control a class of nonlinear adaptive systemswas carried out by (Lee, 2006). He compared the parameter estimation errors of fixed models and inferred the most closest nominal model from the multiple models and by switching the adaptive model to the selected model under Lyapunov stability, performance improving switching was achieved. As investigated by theLee(2006) scheme, the transient response of the system under test was improved upon in view of abrupt parameter variations and was validated by simulating the approach on anti-lock braking system (ABS).

2.6 Structured System Designing and Human-Centric Systems

A structured systems analysis and design method(SSADM) was investigated by Downs et al. (1992)and showed various techniques for identification of users' requirements and their incorporation into function definition. SSADM allowed a means by which processeddata could be mapped through to work out patterns of users, whether they are relatively simple or complex. It also allowed for systems to benefit from logical data flow modeling by identifying the entities involved, their relationship, logical data structure, removing redundancy and validation of the system.

Human-Centric ambient intelligence systems design was investigated by Aghajan et al.(2010) providing a bridge between data processing and intelligent reasoning methods. They suggested an ambitious strategy that involves a shift from algorithmic context detection for users in home health technologies to a user-centric context detection system. In such a system, users are allowed to calibrate the system and its sensors in a real world deployment by themselves. However, cynics of the proposed ubiquitous computing systemargued that such home health technologies are difficult if not impossible to achieve as people defer in their level of intelligence, awareness and technological knowhow. For example, just sensor deployment for the system may be aherculean task for an average user of such a system. Triangulation of investigation (or stepwise replication) which according to (Hays & Singh, 2012) refers to the use of multiple researchers or teams of researchers to collect and/or analyze data, write reports and present findings may be applied in the design of SSADM and human-centric ambient intelligence systems. This approach to human-centric designing will thus ensure the design of highly robust systems especially in consumer devices and healthcare

monitoring systems as processed data can be revalidated by others to produce the same inferences as these devices or systems can keep a log of sensors readings over time.

2.7 Management and Measurement of Patient Information

Global Observatory for eHealth (2012) analyzed the role of individual patientinformation systems over aggregated patient data, the current state of their deployment and the issues resulting from their implementation. The report which advocates privacy laws and policies to protect patients data, reiterated the need for a shift from paper and face-to-face patient consultation to electronic driven consultation. The report stated that "while the conventional way to collect such data are on paper forms and register books, increasingly face-to-face encounters are being captured electronically". This trend will continue as improvements are made in computer hardware, software, telecommunication infrastructure and as countries continue to develop the skills necessary to implement electronic data storage, analysis and transmission systems.A chartof this progress, i.e the evolution from paper-based to electronic records is shown in Figure 2.1.According to Global Observatory for eHealth (2012)Electronic medical records (EMR), Electronic health records (EHR) and Personal health records (PHR) can all serve as rich sources of patient's diagnostic data in a robust health information system (HIS).

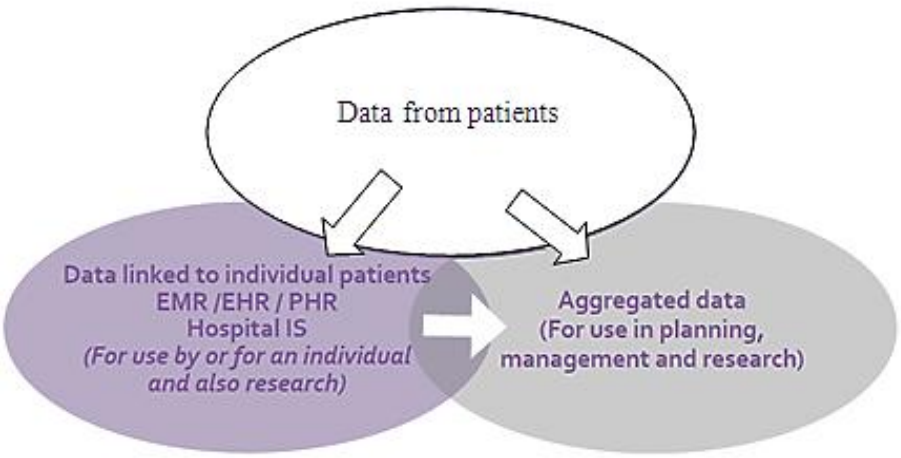


Figure 2.1: The evolution from paper-based records to electronic records

The document however identifies "thatsystems that collect, manage, anddisplay individual patient data can be difficult to implement – particularly in low-resource settings, wherehealth budgets are already strained" in an eHealth scheme.The report which also x-rayed the use of mobile technologies for data collection as part of a growing area of eHealth called mHealth, asserts that standards for their use and interoperability with EMRs and EHRs are still being fine tuned. Presently, these tools are still not available especially for large scale

deployments and active research is still ongoing for fine tuning the methods for a wide range of health concerns.

Nelson(2012) in an extensive profiling of health status of cases of patients from Dartmouth Medical Center, Lebanon, Group Health Cooperative in Seattle and KarolinskaUniversity in Sweden identified the importance of patient-reported measurement systems for measuring, improving and researching health outcomes and health care value.The report stated that the preferred source for some core data required to measure and improve health outcomes are the patients who also are the primary beneficiary of the healthcare system. It concluded by asserting that "a promising way to improve health care value is to embed information provided by patients into the flow of clinical care using patient-reported measurement systems. The health of patients and the population can be improved and the value of care enhanced by taking advantage of patient-reported data within the clinical context".He also outlined some emerging opportunities to include Tele-health that aims to provide a sophisticated and less expensive care to patients directly in their homes by leveraging on technology. The less expensive healthcare been advocated is possible according to Asabere (2012), if mMES are hosted in CBS thus allowing patients to be supported by one control and robust software resident in the cloud system.

Another emerging opportunity worth reviewing is the Computerized Adaptive Test (CAT) which is based on mathematical techniques such as Item Response Theory (IRT) and Rausch methods. CAT enables a computer to select survey questions, deliver to a respondent based on the individual's prior responses so that the respondent's score on a particular area of focus can be measured with high precision and parsimony and then used for medical inferences. CAT method which is tremendously advanced by the National Institute of Health (NIH) was used in the design of the Patient-reported Outcomes Measurement Information System (PROMIS). NIH through PROMIS offers public domain static measures of adult's and children's health status and also make dynamic CAT scoring methods available on its platform using open source for anyone to use. This creates a knowledge based system with roles and mechanisms to acquire or capture knowledge and experience as well as to process and utilize them. According to Leondes (2000), "knowledge management is the formal management of knowledge to facilitate the creation, identification, acquisition, development, distribution, utilization and preservation of an enterprise's knowledge using advanced information technology".

2.8 Health Care Crisis

The crisis witnessed in the healthcare system of today as lamented by (Armstrong,2015;Dugas et al., 2012; Gaebel et al., 2016; Serrano, 2011)elaborates the need to understand diseases better, identify disease signatures and patterns in real-time, proffer solutions collaboratively and develop strategies to treat subsets of particular diseases as one solution cannot solve all health challenges. They pointed outthat to make faster and better progress in developing new pharmaceuticals, a better understanding of diseasecontagion and an increase in the understanding of the basic mechanisms that control their behavior. Simple models for disease(s) detection and simple physiological readouts before treating or admitting patients should be encouraged.Organisms causing diseases continue to evolve and change their modes of growth, attack mechanism and resistance to drugs depending on environment as lamented byArmstrong (2015) that at present, diagnosis or decision making in healthcare using machine learning algorithms are not replacement for clinicians as a good diagnosis must take into account all structured and unstructured data, image data such as x-rays and even subtle visual cues from the patient and their time frame for answering questions. Lee et al. (2013) and Manheim et al.(2016) have noted the present healthcare crisis to also include inability to use and interprete developed models for disease detection and control due mainly to poor clinical practice and infrastructure in hospitals and healthcare centers.

2.9 A Bayesian Network for Diagnostic Analysis

Bayesian networks which constitutes a class of probabilistic models for modeling logic and dependency among variables representing a system was investigated by Ayyub(2003). These Bayesian networks consist of set of variables, graphical structure connecting variables and set of conditional distributions.These networks wereused to represent knowledge structure that models the relationship among possible medical difficulties, their causes and effects, patients' information and diagnostic test results.Using tests results for x-rays, tuberculosis skin test and dyspnea (shortness of breath) and dependence on patients' information and medical difficulties, a marginal probability distribution function was used to update the beliefs attached to each relevant nodes on the network. For example, interviewing a patient may produce the information of visiting Asia, which is then updated in the network and further updates may be obtained if the patient is a smoker and so on. The random variables dependence may then be expressed in probability trees to help localize a treatment based on all the variables obtained by the network.

2.10 Mathematical Modeling of DiseaseInteractions

The role of mathematical modeling of disease mechanism of operation,their metabolism and spread in relation withbiological systems'immunities was investigated by thework of Achebe(2010). The author investigated the various Hamaker coefficients for both HIV infested and non infested human blood using the Lifshifz model(Equation 2.3) that allowed for mathematical derivation of the Van der Waals forces of interaction (attractive and repulsive)between solid bodies in contact. Various behaivoural patterns of the solid bodies in contactwith and without HIV presence was investigated and remarkable differences between infested and non infested particles in the serum were obtained.

$$A_{132} = \frac{3}{4}\pi\eta\int_0^\infty \left[\frac{\varepsilon_1(i\zeta)-\varepsilon_3(i\zeta)}{\varepsilon_1(i\zeta)+\varepsilon_3(i\zeta)}\right]\left[\frac{\varepsilon_2(i\zeta)-\varepsilon_3(i\zeta)}{\varepsilon_2(i\zeta)+\varepsilon_3(i\zeta)}\right]d\zeta \tag{2.3}$$

Where $\varepsilon_1(i\zeta)$ is the dielectric constant of material j, along the imaginary i,frequency axis ($i\zeta$) and $A_{132}$is the absolute combined Hamakercoefficient which the work showed to be negative if blood sample is not infected with HIV and positive if otherwise. Achebe (2010) work was based on the thermodynamic principle of particle-particle interaction in a medium, where the virus and the harmless human lymphocytesconstitutes the particles that are dispersed in a liquid medium of serum or plasma.Achebe (2010)investigation brings to fore the indispensible fact of the inter-relativity of engineering and biological systems which is a vital key in twenty first century research.

2.11 Contagious Diseases Spread

The mechanism of contagious diseases making an appearance in the public has occurred severally with even the healthcare industry losing some of her professionals. The dynamic nature of these diseases has always made it difficult to identify its presence in the human body; this is because of the highly complex and dynamic nature of the human biological system. Availability of data to help prepare against high mortality rates in the event of an occurrence or outbreak of such diseases is still inadequate. Presently, emphasis is on how to reduce the time spent or detection time in the event of an outbreak.

2.11.1 Lassa Hemorrhagic Fever

TheLassa Hemorrhagic Fever(LHF) is a contagious disease with a high mortality rate to its victims and common in most West African countries. The zoonotic disease manifest with symptoms of high fever, general weakness, headache, chest pain, vomiting, diarrhea,

cough, pleural effusion, bleeding from orifices and in late stages sometimes disorientation and coma (Ajayi et al., 2012).A tabulation of these symptoms is shown in Table 2.2. The source however, suggested a proactive response to curtailing the disease by early detection and improving the epidemic preparedness and response to its containment even before the occurrence of an epidemic. Theirinvestigatedearly detection system helped in reducing the time of diagnosis and eventual reduction in mortality to as little as an average of nine days for twenty cases that involved six deaths in 2012 at Federal Teaching Hospital, Abakaliki (FETHA) in Ebonyi State, Southeast Nigeria. This is indeed a commendable achievement owing to the high mortality rate witnessed previously when such contagious diseases surface in pre-colonial African with capacity to wipe out a whole village as carried by some African stories and forklore. This low mortality rate recorded (Ajayi et al.,2012) was achievable due to the strong surveillance system deployed at the hospital (FETHA) where the data was obtained, otherwise, such occurrence would have resulted in several hundreds of deaths or fatalities.

Table 2.2Clinical Presentation of Lassa fever cases.Source: Ajayi et al. (2013)

SN	Clinical Presentation	Frequency (N=20)
1	Fever	20
2	Sore Throat	14
3	Abdominal pain	17
4	Headache	7
5	Vomiting	10
6	Bloody vomiting	3
7	Bloody stool	3
8	Body pains	5
9	Body weakness	5
10	Prolonged menstruation	2
11	Spontaneous abortion	2

Appendix D shows normalized data obtained from such a strong surveillance regime. The role of exploratory laparotomies to either save or endanger patients' lives especially during emergency was explored by Dongo et al.(2013). They showed how patients complaining of acute lower abdominal pains along with others symptoms like fever, headache and weakness were operated upon for apppendicetomy, which later lead to continuous internal bleeding and external bleeding through punctured sites. Though some patients (shownin Table 5.2 of Appendix C) made uneventful recoveries when administered with ribavirin therapy, it was clear that for areas or localities with a history of Lassa fever occurrence, early succeptivityfor Lassa fever could be key to early detection and prompt action. This was evident in the referred

cases to the emergency unit of Irrua Specialist Teaching Hospital(ISTH) in Edo State (Dongo et al., 2013).

2.11.2 The Ebola Virus Disease

A highly contagious disease with no known cure is the Ebola virus disease (EVD). This is also called Ebola haemorrhagic fever (EHF). EVD affects both humans and primates (Gorillas and Chimpanzees; see Appendix B for a historical list).Been a zoonosis, its primary host is still being thought to be from rodents and/or from bats. Muyembe et al. (2012) stated that "the hunting and eating of fruit bats may have resulted in the primary transmission of Ebola virus to humans. Human-to-human transmission is associated with direct contact with body fluids or tissues from an infected subject or contaminated objects". They also lamented that "despite several, often heroic field studies, the epidemiology and ecology of Ebola virus, including identification of its natural reservoir hosts, remains a formidable challenge for public health and (the) scientific communities."

Symptoms and signs that vary with respect to a victim's immune systems' ability to inhibit organ deterioration are the only source of diagnostic data for detecting the highly morbid disease. The most common symptoms reported between symptom onset and case detection included those of "fever (87.1%), fatigue (76.4%), loss of appetite (64.5%), vomiting (67.6%), diarrhea (65.6%), headache (53.4%), and abdominal pain (44.3%)" (Muyembe et al.,2012). The source's data which is replicated in appendix E, also stated that "specific hemorrhagic symptoms were rarely reported (in <1% to 5.7% of patients). 'Unexplained bleeding,' however, was reported in 18.0% of cases". The preponderance factor in diagnosing this disease is the ease at which it mimics malaria and fever which are often common ailments in a locality thus making alert levels of health care providers to be on an 'uneventful' low state even in standard health systems.The dynamic nature of Ebola virus spread owing to its different stages in a patient and the possibility of a standard health system not to detect it remains a serious source of concern to all. In most outbreaks, recognition of the disease was delayed because physicians were not accustomed to this new illness and the symptoms were generally non-specific. Outside the epidemic context, it appears quite impossible to recognize the first Ebola case in an outbreak on clinical grounds only. Suspicion of EHF is only possible later during the aggravation phase".

2.12 Analysis of Existing Systems

Presently, clinical procedures and general methods of admission of new patients into hospitals and other healthcare centers to a large extent are non contagious-disease aware. Patients show up in hospitals and healthcare providers respond to treatment with little or no caution as regards disease spread. This is further compounded by the lackadaisical attitude of some of the healthcare providers to delivery of healthcare to patients especially in developing countries. (Appah and Afrane, 2014;KPJ Healthcare, 2015;McHugh et al., 2011) gave an illustration of the existing protocol in hospitals and helped in creating the flow chart merge of the salient features available in today's healthcare delivery as depicted in Figure 2.2.As can be noticed in the flow chart (Figure 2.2),patients regardless of their medical state have contact with healthcare providers without any serious preventive measures in place by hospitals or clinics to avoid disease contagion. Inpatients and outpatients also have contact with themselves as the same exit point is used. Thispractice has several debilitating consequence on the healthcare system through disease spread and re-infection cumulating in epidemic occurrence and even death of patients and medical personnel.

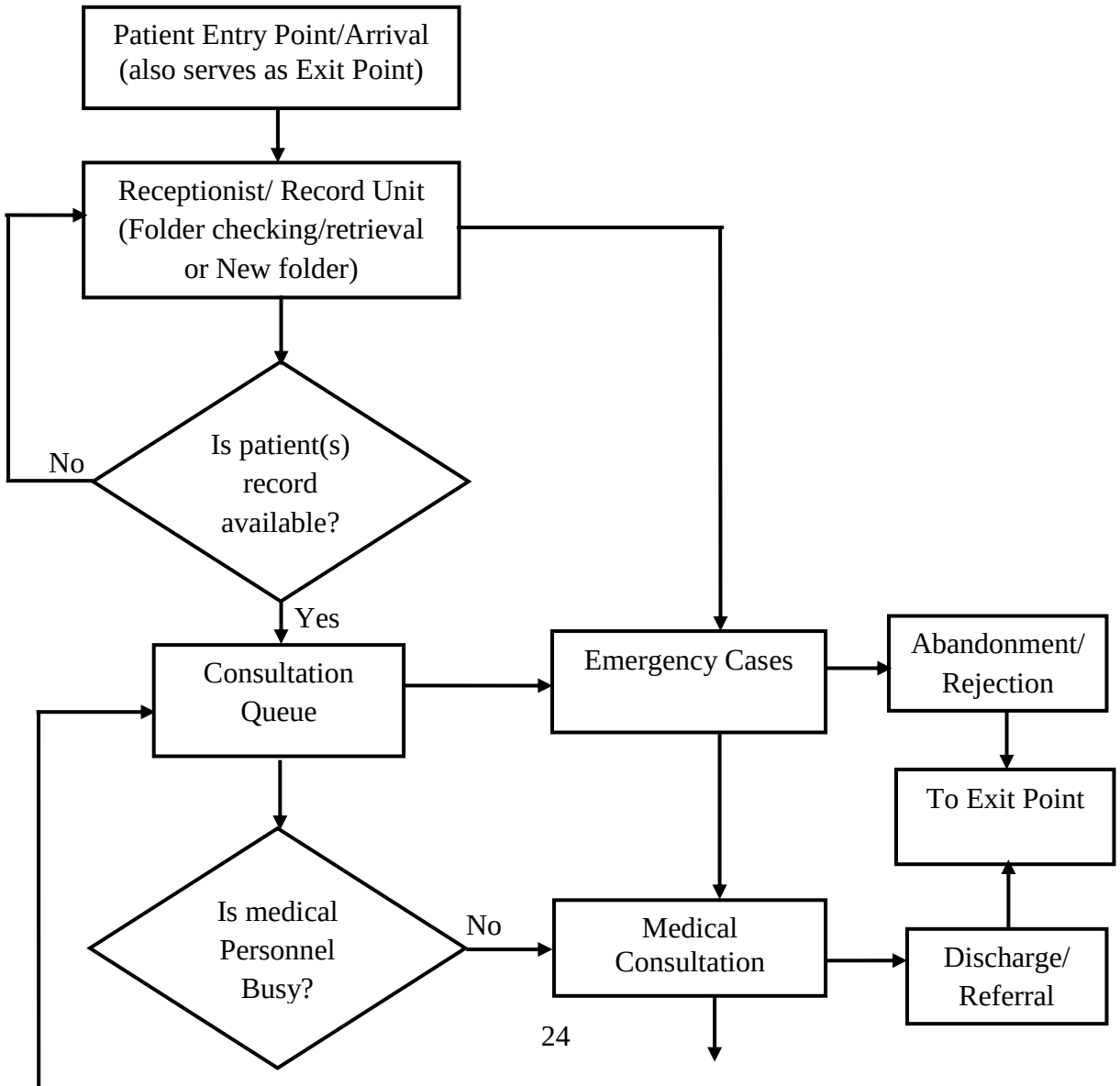


Figure2.2: Flowchart of existing non preventive patient admission process (KPJ Healthcare, 2015;McHugh et al., 2011).

2.13 Limitations of Existing Systems

The limitations of the existing system include the following:

a. Avenues For Contagious Disease Spread

Existing systems allow patients to clog together to wait or queue to see the doctor or other superior healthcare provider, an age long practice that sometimes may require disappointed patients to show up the next day or week, depending on existing issues. Especially in resource constrained Africa healthcare settings with very poor infrastructure, considerable unprofessionalismmanifests even for patients or their relatives making frantic efforts to obtain card/folder from the receptionist/nurse on duty before commencement of treatment. These delays and drags sometimesleads to emergency cases being abandoned, outright rejected or death of patient.This practice if allowed to continue will result in several exposure of present system to vulnerabilities as Emerging Infectious Diseases (EIDs) continue to surface (Vrbova et al., 2009). These EIDs which have become global problems due to globalization and ease of people travelling between countries have their origins from underdeveloped countries struggling with poverty, poor sanitation, lack of education and poor healthcare practice.

b. Avenues for Re-Infesting Patients

Due to the dynamic nature of contagious disease spread especially by airborne diseases, the ability of the existing healthcare systems to serve as a means of eventual epidemic medium is high. This is because especially in third world countries access to healthcare is similar to an uncontrolled or unregulated open market where buyers and sellers walk in and out. Rejected patients who may be hosting contagious vectors or patients undergoing treatment at various stages including suppression of these contagious vectors and even discharged patients who having been given a clean bill of health may become re-infested with a new disease due to poor clinical practice.

c. Clinical Practices Endangering Healthcare Providers

The existing systems currently in practice in most hospitals and clinics do not have any protection for healthcare providers which come into contact with outpatients. This is further compromised by a high patient-to-healthcare provider ratio available in most underdeveloped country's healthcare systems (Agbo & Nwodoh, 2011). These personnel surprising have access to virtually all staff and patients and most times help during emergency situations without protective clothing or apparatus.

d. Equipment and Facilities Contamination.

Inability of the existing system to detect contagious cases promptly could compromise equipments and facilities such as beds, bed sheets, patient cloths and personal effects as laundering in this hospitals and clinics are usually contracted to third parties. These parties sometimes do not understand the importance of sterilization and proper disinfection of laundry to prevent contagious disease residues or agents. A foolproof system will therefore be necessary that helps to reduce or eliminate the need for expensive and microscopic laundry by helping to remove the threat before it manifest.

e. Infesting Household Members.

Healthcare providers under the existing system are likely to infest their household members, or their colleagues and other patients with this poor clinical practice. This is due to the fact that healthcare providers' contact with patients who might be index cases or host to disease vector are not routinely checked with standard alert levels. Conscious healthcare personnel trying to be careful may be considered to be infringing on patient's right to adequate treatment, obstructing patient's desire to see the consultant, e.t.c. This is due to lack of a structured approach to disease control in the present healthcare system.

f. Inability to Detect Index Case.

According to the present system, a patient who could have been identified as an index case or disease vector host and quarantined or other appropriate measures activated may be set lose by this unstructured system. This patient having obtained a clean bill of health may proceed to travel to other regions where the incubation period of the disease may have reached its final or mature stages and cause an epidemic in that area.

g. Delays and Inproper Patient Appointments

Inappropriate appointments by hospitals and clinics could cause delays, drags and patients waiting or queuing to see the medical consultant. This could cause patients to become

exposed to contagious diseases especially airborne, released by contagious vector-carrying patient(s) on the same queue.

2.14 Summary of Review of Related Works

From this review, numerous authors outlined the role of huge data mining to reveal special characteristics upon which further processing could be acted upon to produce in-depth meanings and outcomes. The various methods used to achieve this analysis and mining has been elaborated and applications to various industries including healthcare wasenumerated(Armstrong,2015;Badiru et al.,2006;Reddy,2003;Shortliffe1976). These reviews have pointed out the need of an implementation of an ICT based infrastructure to help mine available data, make sense from the analysis and apply the solution to solve a particular problem in the society. Application of a successful model could thus help for example to curtail a contagious disease epidemic in an area. The nature of epidemics are highly societal problems which havebecome global;for example, the US hospital's low alert levels on the Eric Duncan case at THPHCNN (2014)is an eye opener to the fact that travellers are the greatest zoonotic agents (Manheim, 2016) and developed countriesshould be part of aggressive research into huge data mining for healthcare especially for contagious diseases. This is because even in the face of racism, tribal sentiments and other mundane issues as some literature claim for the mishapCNN (2014), it is clear that an ICT based infrastructure will not partake in racism or any other rather would have helped the US hospital to detect the contagious disease bearing patient.

The advantage of dependence on artificial intelligence based mathematical techniques such as fuzzy logic and artificial neural networksfor data mining, interpretations of data and validation of results was presented succinctly. Practical applications of these tools for solving problems in man's complex world continue to be inadequate or unavailable necessitating the need for creation of more tools and models able to utilize the hidden secrets embedded in artificial intelligience and knowledge. In addition, the future trend of applications of emerging technologies and knowledge management is the design of more robust, intelligent and client oriented approach even in healthcare delivery. Although most of the reviewed literature advocated for the use of ICT based solutions in modern healthcare delivery, they however admitted that there still exists a lack of available tools asthese solutions are still being developed to address emerging diagnostic problems.Availability of data and appropriate filtering and interpretation models are among the problems facing ICT based solutions.These

problems also include real-time modeling of emerging diseases especially contagious ones in today's 'static' and humdrum world.

The non contagious disease aware nature existing in today's healthcare system can embrace the powers embedded in mathematical modeling techniques especially non 'black box' models such as ANFIS models that can utilized parallel processorsfor massive data mining (Babuska & Verbruggen, 2003; Kavulya et al., 2012; Topping et al., 1998). This will lead to harnessing huge databasesfor pointers to future fatality in a structured manner (Downs, Clare& Coe,1992).

The management and measurement of patient's health informationwith emphasis on a shift from paper and face-to-face patient consultation to electronic driven consultation was elaborately reviewed. This paradigm shift would help make huge database available such that global medical expert systems which could be hosted in a CBS could be used to quickly analyze the data and provide a reasonable and accurate health condition of the diagnosed patient.However, the present situation of healthcare creates the need for proper management of patients and their ailments by resourceful and efficient dependence on ICT based tools from the point of patient's first contact with the hospital or clinic, diagnosis, treatment, data management and admission. These tools which can depend on the power embedded in mathematical models and artificial intelligence are hugely inadequate and are still being developed as diseases and their corresponding ailments continue to evolve into strains that are getting resistant to known drugs and cure methods. The role of such methods have proven to be efficacious for dealing with most ailments as in some investigated literature, however, there is need for the use of mathematical models and active interdisciplinary collaboration geared towards modeling and simulation of all possible diseases mechanism through knowledge bases and thus helping the detection, control and isolation of contagious disease in real-time.

Conclusively, the developed fuzzy neural model will solve the problem of identification of contagious diseases discretely by utilizing salient features in vaguely and ambiguous data available easily from patient's reported symptoms and signs. The developed expert system targets the subset of contagious diseases and thus is different from the investigated counterparts such as MYCIN and CASNET which though are used in diagnosing health conditions are definitely not used for contagious diseases. Also, the establishment of the learning rate, η to be equal to 0.3 for a faster, more transparent analysis and non-deteriorative gradient descent optimization's convergence based on the unique inputs to the artificial neural network was investigated and thus is different from some investigated literature that achieve

convergence with completely black box models such as Levenberg-Marquardt Algorithm (LMA) that can achieve convergence in alarming one or few iterations for a huge training dataset and suggestive of the problem of overgeneralization.

CHAPTER THREE

METHODOLOGY AND SYSTEM ANALYSIS

The need for a structured system approach in handling patients in the healthcare system with the overall goal of mitigating the window period of disease spread has necessitated this analysis. The methodology adopted for this research include the use of machine learning techniques for analyzing and diagnosing patient's ailments with bias for contagious disease cases. In this chapter an analysis of the mathematical concepts utilized in this work that include fuzzy reasoning and artificial neural networks.An analysis of the system's design process for implementing the developed model was detailed.A graphical representation of the proposed structured system approach to healthcare delivery which differs from the already depicted existing system'sin chapter twowas produced.

3.1 Methodology

The developed diagnostic expert system in this work is based on the application of the Mamdani model that allows the use of rule-based fuzzy models to obtain a direct mathematical relationship between variables that interact as signs and symptoms to collectively determine the state of a patient. The other part of the methodology is the application of the Gradient descent optimization algorithm on the pre-fuzzified symptoms' values as inputs into the developed artificial neural network part of the expert system.The fuzzy processing involves grouping signs and symptoms into sets of membership for which a direct mathematical value is obtained by the Mamdani model. The expert system represents and operates on these variables (noting that presented variables are not by themselves conclusive of current diagnosis) by means of if–then rules with vague or imprecise (ambiguous) predicates, such as:

If body temperature is High then Malaria susceptibilityis High.

If body pressure is Low then Fever susceptibility is Low.

If Weight Loss is High then Chronic Condition is High.

This defines in a rather qualitative way the relationship between the patient's body temperature, pressure and weight and the disease causative mechanism. However, human reasoning and response to data entry or collation is based on heuristics and ambiguity such that temperature may be reported as high, slightly high or very high. Pressure could be reported as normal or abnormal and weight loss could be interpreted as high, slightly high or severe. Thus to make such model operational, the meaning of the term 'high' or 'low' must be defined more precisely. This is done by using fuzzy sets, i.e., sets where the membership is changing. The Mamdani model was chosen for this work over the Takagi–Sugeno Model due to the ability of the model to deal with linguistic fuzzy since the latter is more adapted for data driven identification. In order to deal with the mathematics involved in linguistic fuzzy, it is necessary to defuzzify the meanings and relationship of vagueness and ambiguity as used in this work. In comparing vagueness and ambiguity, for which Leondes(2000) stated that "vagueness is associated with difficulty making sharp or precise distinctions in the (real) world"; that is, a domain of interest is vague if it cannot be delimited by sharp boundaries.Ambiguity is related to one-to-many relationships, that is, situations in which the choice between two or more alternatives is left unspecified. It is widely thought that fuzzy set theory provides a mathematical foundation for dealing with vagueness, whereas fuzzy measure provides a formal framework for handling ambiguity.

3.1.1 Mamdani Model

In this model, the antecedent (if-part of the rule) and the consequent (then-part of the rule) are fuzzy propositions(Babuska & Verbruggen, 2003;Mamdani &Assilian, 1975):

$$R_i: \quad \text{If } x \text{ is } A_i \text{then } y \text{ is } B_i;\ i = 1,2, \ldots K \tag{3.1}$$

Here $A_i$and $B_i$are the antecedent and consequent linguistic terms of patients' symptoms as present (such as 'Normal', 'Slightly large', 'Very High' etc.), represented by fuzzy sets, $R_i$is the fuzzy proposition(s)and K is the number of rules in the model.

3.1.2 Fuzzy Inference Systems (FIS)

A FIS according to Ardil and Sandhu (2010) "is a way of mapping an input space to an output space using fuzzy logic. A FIS tries to formalize the reasoning process of human language by means of fuzzy logic (that is, by building fuzzy IF-THEN rules)". The Mamdani-type inference expects the output membership functions to be fuzzy sets whose output, after

the aggregation and defuzzification process is a single value or spike rather than a distributed fuzzy set. This output which is a singleton output membership function of the pre-defuzzified fuzzy set corresponds to the efficiency of the defuzzification process and the computation of the centroid of a two-dimensional function. The centroid of such a function could be obtained by using different simple mathematical functions and equations to obtain the upper bounds and lower bounds of the two-dimensional function. For example, a half filled glass cup could be thought of a two dimensional function since further addition or removal of water creates an approach to two boundaries.

For this research, linguistic fuzzy model is useful for representing quantitative and qualitative knowledge based on available or visible symptoms and assuranceor trust that the respondent, that is either the patient or nurse is giving accurate report of the occurring symptoms for the presenting ailment. These symptoms which are reported using uncertain and ambiguous linguistic variables or 'qualifying adjectives' which contain more information than 'crisp' values of true or false, high or low, hot or cold, better capture diagnosis as disease isolation depends more on the degree of causative elements in the patient's body. For example early stages of a disease will manifest fewer symptoms compared to later and final states with each stage having its own unique signatures. By using membership functions the degree or extent of viral, pathogenic and bacterial involvement, their stage of development, location and population present in the patient's immune system (the host) can be more understood. For example, a membership function mapping of body temperature and blood pressure of a patient could be as shown in Figure 3.1.

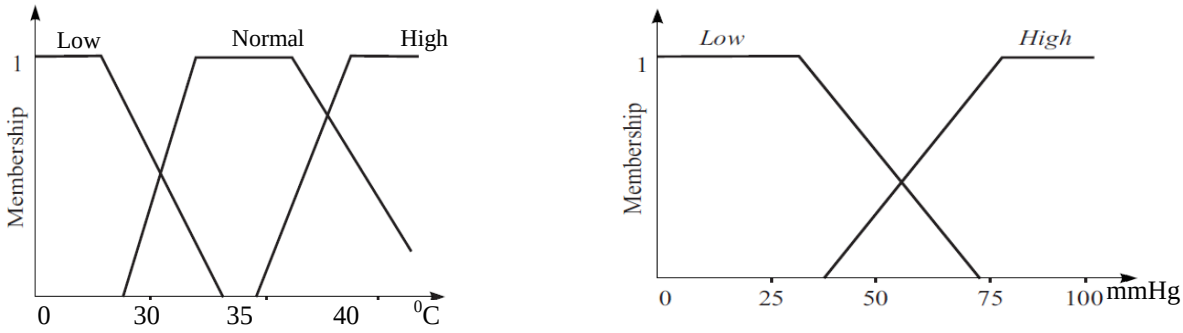


Figure 3.1: Membership Functions of present state of patient's condition.

Since the input–output data of the system under study are available the membership functions can be constructed or adjusted automatically with the stringent rule that the qualitative relationship given by the rules is usually expected to be valid for a range of conditions. The fuzzy membership function could have different forms such as bell-shaped curve, trapezoidal or flattened bell-shaped, triangular, etc.

3.1.2.1 *Fuzzy Expected Interval (FEI)*

To handle the fuzzy population distribution in the proposed inference system Equations 3.2 and 3.3were used to obtain the upper and lower bounds in the computation of the Fuzzy Expected Interval (FEI) from which a Fuzzy Expected value (FEV) was computed. This is based on the definition that if α and β are intervals in the expected interval, then we say that α is higher than β if the upper bound of α is greater that the upper bound of β. Equations 3.2 and 3.3 which were obtained from Schneider andKandel(1992) was used to handle the fuzzy population distribution of the reported symptoms i.e it was used to obtain the upper and lower bounds for each manifesting symptoms and then obtain the fuzzy values of patient's responses into the software GUI.

$$UB_j = \frac{\sum_{i=j}^{n} \mathrm{MAX}(pi1, pi2)}{\sum_{i=j}^{n} \mathrm{MAX}(pi1, pi2) + \sum_{i=j}^{J-1} \mathrm{MIN}(pi1, pi2)} \quad (3.2)$$

$$LB_j = \frac{\sum_{i=j}^{n} \mathrm{MIN}(pi1, pi2)}{\sum_{i=j}^{n} \mathrm{MIN}(pi1, pi2) + \sum_{i=j}^{J-1} \mathrm{MAX}(pi1, pi2)} \quad (3.3)$$

The obtained fuzzy expected values for that considered symptom which like probabilistic theory are between zero and one (i.e between 0 and 1). This FEV then kick starts or initiate the fuzzification process which involves mapping of the inputs into fuzzy set and membership values based on the universe of discourse in the fuzzy logic control system (FLC). The universe of discourse for this work is the set of all known and unknown symptoms manifesting or non manifesting on the patient under investigation in real-time.

3.1.2.2 *Fuzzy Logic Control System (FLC)*

According to Burns (2001), a FLC will be required to minimize the error, e(t) and the rate of change of error de/dt generated by the non linear system under investigation. Burns (2001) pointed out that decisions on number of inputs, size of universe of discourse and number and shape of fuzzy sets need to be made a priori. The FLC as shown in Figure 3.2 contains the Knowledge base, Fuzzy Inference unit, the fuzzification and defuzzification units.

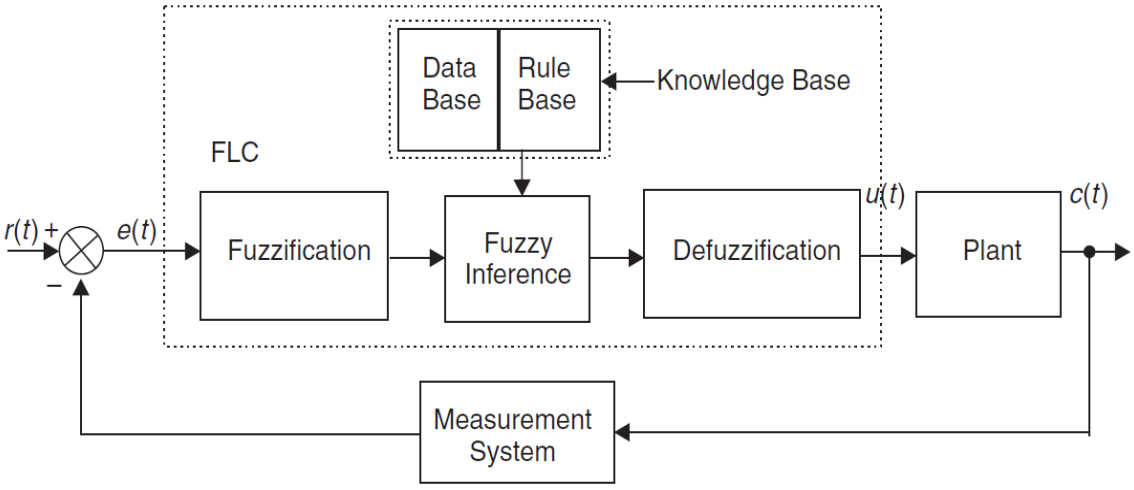


Figure 3.2: The Fuzzy Logic Control System

3.*1.2.3* *The Knowledge Base*

This consists of the sets of data localized in the systems database used to attune, calibrate or operate the system and a set of antecedent-consequent linguistic fuzzy rulebase of the form

If e is PS AND ce is NS then u is PS (3.4)

where PS, NS are linguistic terms and constructed according to Burns(2001), from a priori knowledge from either one or all of the following sources:

a. Physical laws that govern the plant/system dynamics
b. Data from existing controllers/sensors
c. Imprecise heuristic knowledge obtained from experienced experts which do not require knowledge of the plant's mathematical model.

According to Paraskevopoulos (2002), the rule base constitute the heart of the controller and the 'If' part is called the premise (or antecedent or condition) and the 'then' part is called the consequent (or action) then it is sufficient to describe the system under test even without relying on control concepts such as root locus, Nyquist plots, pole placement and stability tests but rather dependence on reliability and integrity of the rule base.

3.*1.2.4* *The Inference Engine*

This deduces from the evidence or data made available from the fuzzy inputs a logical conclusion after processing. The inference engine is essentially a program that uses the rule base and the fuzzy generated data of the controller to draw the conclusion according to the

modus ponens rule in Equation 3.4. This conclusion is then the fuzzy output of the controller and subsequently the input to the defuzzification interface.

3.*1.2.5* *The Defuzzification Process*

This is the procedure for mapping from a set of inferred fuzzy control signals contained within a fuzzy output window to a non-fuzzy control signal known as Crisp output. It involves the fuzzy conclusion being defuzzified and serves as the final output of the FLC. Using the centre of area defuzzification technique, the crisp control signal can be obtained in terms of linguistic terms of the form:

$$\text{Crisp control signal} = \frac{\text{Sum of first moments of area}}{\text{Sum of areas}} \tag{3.5}$$

Which for a continous system, according to (Burns, 2001) becomes

$$u(t) = \frac{\int u\,\mu(u)du}{\int \mu(u)du} \tag{3.6}$$

and for a discrete system,

$$\sum_{i=1}^{n} u\,i\mu(ui) \tag{3.7}$$

where ui is the member of the set and $\mu(u)$ is the associated membership function.

3.1.3 The Takagi–Sugeno Fuzzy Model

The Takagi-Sugeno Fuzzy Model (TSFM) was developed by the Japanese duo for application of fuzzy principles in a manner different from the linguistic based Mamdani approach. The TSFM allowed the analysis of data-driven fuzzification that involves the identification of huge data with highly non linear behaivour and time-varying characteristics. The TSFM can be regarded as a piece-wise linear approximation of a nonlinear function or a parameter-scheduling model with an n – order polynomial(Babuska & Verbruggen, 2003). However, Equation 2.2 can be reduced to the Mamdani model by n = 0 order polynomial approximation to produce the Takagi-Sugeno zero order fuzzy model (TSZOM).

3.2 The Artificial Neural Network

The Artificial Neural Network (ANN)which (Krose & Smagt, 1996) described as a type or variation of the parallel distribution processing (PDP) idea; consists of a pool of simple

processing units which communicates by sending signals to each other over a large number of weighted connections. The implementation of such a network mimicsat least computationally the biological neural network(BNN) in animalsfor which estimates consists of several billions of neurons.For example in the human brain with a densely interconnectednetwork of approximately 10^{11} neurons, each connected, on average, tol0^4 others leading to an awesome 10^{15} synaptic connections.Scientistswere inspired by the interconnection of the brain with the spinal cord andtheirreceptivity to all types of neurons which acts as inputs to the BNN, see Figure 3.3. Following this mimicking, scientists were able to design the ANN to be receptive/transmittable to messages from/to other interconnected neurons.In its operation, the ANNuses a set of processing units called neurons, that receives activation from a set of activation units, $y_k$or nodes. This activation unit isconnectedthrougha forward flow connection that is weighted by a weight w_{jk} which defines the effect of the signal of unit j on unit k. An ANN, uses a propagation rule and an activation function F_k which determines the new activation level based on the effective inputs and the current or updated activation $y_k(t)$. A bias or offset external input and a method for gathering information or a learning rule all constitutes or implements an ANN in an environment within which the system is designed to operate along with input and error signals.

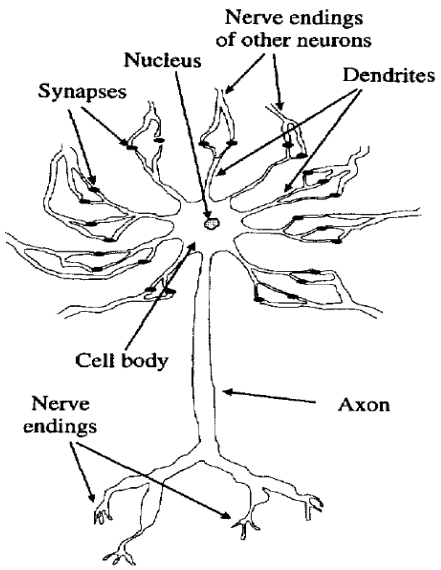


Figure 3.3: An idealized representation of a BNN. Source: (Topping et al., 1998)

The use of Artificial Neural Network (ANN) has witnessed a tremendous growth in the artificial intelligence industry over several decades as scientists and analyst continue to explore the possibility of mimicking the biological human brain in computer based computational tasks. Galushkin (2007) represented a neural network in the form of an equivalent system that adapts to external conditions with a general block scheme of such a

system as shown in Figure 3.4 with x(n) as the multidimensional stochastic process having the form of pattern sequence at the neural network input; where n is a discrete argument.

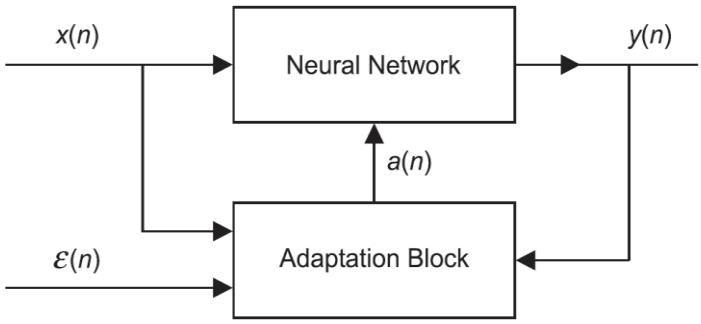


Figure 3.4: A general block scheme of a Neural Network with external conditions

Signal Ɛ(n) is determined by the supervisor instructions belonging to the current pattern at the neural network input to a particular class (Galushkin, 2007). Y(n) is the multidimensional output signal and vector a(n) is determined by the unit of neural network parameter adjustment and information about y(n) transformation structure. [x(n), Ɛ(n)] is the input signal of the neural network. The system is basically a linearized system possessing the properties of homogeneity and superposition. Homogeneity, according to(Dorf & Bishop, 1998) and (Nise, 1995)is the response of a system to a multiplication of a scalar and the input while superposition refers to the principle that the sum of the excitations produces a sum of the responses by the system, i.e. if there exists $x_1(n)$, $x_2(n)$,…. inputs that yields $y_1(n)$, $y_2(n)$,….then the system is possessing the property of superposition that is of the form:

$$\sum_{i=1}^{n} xi(n) \quad = \quad \sum_{i=1}^{n} \quad yi(n)$$

3.2.1 Processing Units

In ANN, inputs into the network and outputs out of the network are preprocessed by the hidden units inside the network. Each unit does this by receiving inputs from its neighbors or from external sources and computes an output signal which is then propagated to other units. The processing units which comprise the input, hidden and output units are also responsible for the adjustment of the weights and these adjustments are essentially parallel in operation allowing for simultaneous computations. Also, during operation, units can be updated using a synchronous or an asynchronous protocol. In synchronous mode, activation updating by all units is done simultaneously while in the asynchronous mode, the activation is allowed to update each unit one at a time based on a probabilistic process.

3.2.2 Connectivity between Units

For each iteration, during operation, ANNs connections between units provide a contribution w_{jk} that is called excitation when the contribution is essentially positive and when the contributionis negative, it is called inhibition.The total input to unit k is then the weighted sum of all separate outputs from each of the connected units plus a bias or offset term Θ_k and is given by the propagation rule;

$$s(t) = \sum_j \text{wjk(t)yi(t)} + \Theta\text{k}(t) \tag{3.8}$$

Though other propagation rules occur, the rule above is popularly used and units utilizing this method of excitation and/or inhibition are called sigma units. A graphical illustration of the connection pattern is depicted in Figure 3.5, showing three inputs that are connected to two output nodes through four hidden units. Each circular node represents an artificial neuron and each arrow represents a connection from the output of one neuron to the input of another.

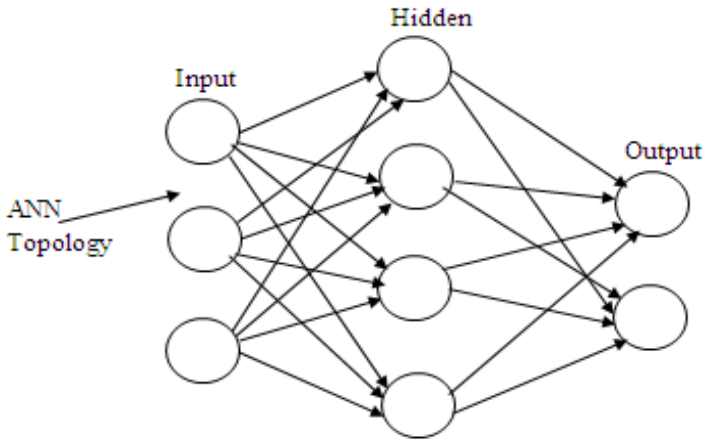


Figure 3.5: Example of a connection pattern in an ANN.

3.2.3 Activation and Output Rules

A rule for computing the output signalis known as the activation function and the outputted value is known as the activation for the unit. The activation constitutes the output to be transmitted to other units or the environmentas the result of computation (Callan, 1999).The activation function, F_k takes the total input $s_k(t)$ and the current activation $y_k(t)$ and produces a new value of the activation of the unit k:

$$y_k\,(t + 1) = F_k\,(y_k(t), s_k(t)) \tag{3.9}$$

and for a nondecreasing function of the total input of the unit, Equation 3.9 yields

$$y_k\,(t+1) = F_k\,(s_k(t)) = y_k\,(t+1) = F_k\Big(\sum_j\ wjk(t)yi(t) + \theta k(t)\Big) \qquad (3.10)$$

The Sigmoid function is normally introduced in the activation function to provide a hard limiting threshold function, a linear function or a smooth limiting threshold. The sigmoid function which allows the output to fall within a continuous range from 0 to 1 or a bipolar sigmoid with an output from – 1 to 1. Since it allows any real-value input to produce an output bounded between two Boolean states, the sigmoid function is also known as a squashing function.

$$y_k\,(t) = F(s_k) = \frac{1}{1\ +\ e^{-sk}} \qquad (3.11)$$

where *e* is base of natural logarithms, equal to about 2.718281828. The Sigmoid squashing function is introduced in the ANN modeling to capture the non linear behaivour of the mimicked BNN. Recall that in BNN, a neurons "fires" when sent or received signals between the neuron and another neuron reaches a particular threshold. According to Larose (2005), "this is a nonlinear behaivour, since the firing response (of BNN) is not necessarily linearly related to the increment in input stimulation" but a nonlinear response or action is necessary for any input to the BNN.

The sigmoid function is the most common activation function because it combines nearly linear behaivour, curvilinear behaivour and nearly constant behaivour with respect to the input as depicted in Figure 3.6.

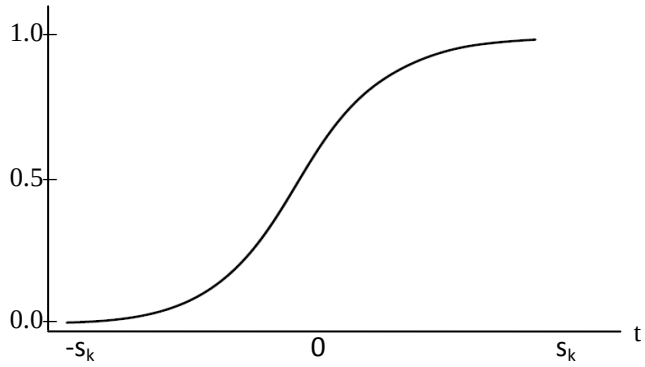


Figure 3.6: Graph of the sigmoid function, $y_k\,(t) = F(s_k) = 1/(1\ +\ e^{-sk})$

The graph shows the sigmoid function been used to "squash" the nonlinear $y_k(t)$ into a nearly linear behaivour for the interval, $-\ s_k < t < s_k$for every increment in t. Thus as t increases moderately, the value of $y_k(t)$ increases varyingly such that at near the centre of $y_k(t)$ as shown in the graph, increments in t produces a nearly linear increment in $y_k(t)$, while near the

extremes, moderate increments in t produces tiny or infinitesimal increments in $y_k(t)$ thus exhibiting nearly constant behaivour.

3.2.4 Back Propagation Learning

ANN is a supervised learning that requires a large number of training set of complete records that serves as input to its architecture from which the network tries to smoothen to a set of output which has been presented to it. The capability of approximating non-linear functions of their inputs to nearly linear function is a major advantage of the use of ANNs over other mathematical models. The Back Propagation learning was adopted over the Feed Forward Learning owing to the ease of adaptation of weights and provision of an algorithm that allows a forward and backward sweeps of the inputs layer to output. Unlike the feed forward learning where propagation is from inputs to output, the BPL propagates error values back to the inputs units for processing in the hidden unit from the output to the input. Thus allowing a method for the weights to be adjusted and adapted during training of the network.Calculation of the derivatives flows backwards through the network, hence the name, Back Propagation.Due to the large data set likely to be encountered in implementing ANNs, computer software that can carry out the rigorous computations abound (Fletcher, 1987). This is because several epochs (an epoch is one complete cycle through the ANN) will be required in other for the solution to converge to a particular tolerable value.The BPL also allows the use of termination criteria to ensure that the system does not loop endlessly and a learning rate coupled with a momentum term to ensure quick convergence. The BPL however suffers from over-fitting of data and lack of ability to generalize especially when the ANN is presented with new dataset that is outside the data used in the training set.Good generalization therefore means that a model fitted to one data set will also perform well on another data set from the same process.Again, however just as in statistical analysis, as stated by Suhir (1997), "the amount of information necessary for carrying out diagnostic analysis is never complete and therefore a probabilistic evaluation should be conducted to draw a sufficiently reliable conclusion."Back Propagation allows for possible assignment of network weights which represents a syntactically distinct hypothesis that in principle can be considered by the learner. In other words, the continuous hypothesis space is the n-dimensional Euclidean space of the n network weights.

3.2.4.1 The Sum of Squared Errors

For a BPL to be meaningful, the output node's error must be fed back into the system's input to know the direction in which the weights will be adjusted to ensure a quick convergence. The sum of squared errors, SSE is a prediction error introduced to measure how well the output predictions fit the actual target values.

$$SSE = \Sigma_{records}\Sigma_{output\ nodes}(actual - output)^2 \quad (3.12)$$

Where the squared prediction errors are summed over all the output nodes and over all the records in the training set.

However, the SSE introduces the need to construct a set of model weights that will minimize the SSE since the weights are analogous to the parameters of a regression model. According to (Larose, 2005), the "true values for the weights that will minimize the SSE are unknown and (the major) task is to estimate them, given the data. However, due to the nonlinear nature of the sigmoid function permeating the network, there exists no closed-form solution for minimizing SSE as exists for least-squares regression".

3.2.4.2 Gradient Descent Optimization

The Gradient Descent Optimization method can be used to know the direction that the weights to the ANN is adjusted or determined in order to decrease the SSE. The gradient of the SSE with respect to the vector of weights **w**at iteration m is the vector derivative:

$$\nabla SSE(\mathbf{w}) = \left[\frac{\partial SSE}{\partial w_0}, \frac{\partial SSE}{\partial w_1}, \ldots, \frac{\partial SSE}{\partial w_m}\right] \quad (3.13)$$

that corresponds to the vector of partial derivatives of SSE with respect to each of the weights,**w** = w_0, w_1, w_2,…., w_m.

The direction of weight adjustment that will ensure a minimized SSE, can be deduced from Figure 3.7which shows a plot of the error SSE against the range of values for weights, w. If the optimal or targeted value of weight w is w* that will obtain the minimumthen a rule that can move our old value of w closer to the optimal w* will be: $w_{new} = w_{old} + \Delta w_{old}$, where Δw_{old} is the change in the current or old location of w.

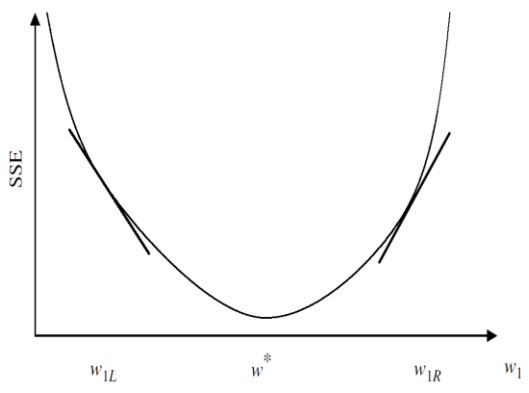


Figure 3.7: Obtaining the weight adjustment direction based onSSE slope versus w.

From Figure 3.7, the slope of the SSE curve at w_1 is the derivative $\partial SSE/\partial w_1$ and for values of wclose to w_{1L} the slope is negative and for values close to w_{1R}, the slope is positive thus making the direction of adjustment of $w_{old}$to be the negative of the sign of the derivative of SSE at w_{old},that is $- \pm(\partial SSE/\partial w_{old})$.$\nabla$ SSE(**w**) is itself a vector in the weight space and its gradient specifies the direction that produces the steepest increase in SSE and it is the negated gradient -∇ SSE(**w**). Finally,the derivative is multiplied with a positive constant term η *(Greek's eta)* known as the learning rate to give the Equation 3.14.

$$w_{new} = -\eta(\partial SSE/\partial w_{old}) \tag{3.14}$$

3.2.4.3 Derivation of the Gradient Descent Optimization Rule

The direction of the steepest descent along the error curve can be obtained according to (Mitchell, 1997) by computing the derivative of E with respect to each component of the vector, **w** from:

$$E(\mathbf{w}) = \frac{1}{2}\sum_{d\epsilon D}(t_d - o_d)^2 \tag{3.15}$$

where D is the set of training examples, t_d is the target output for training example d and o_d is the output of the linear unit for training example d and E(**w**) is the half of the SSE.

The vector of $\partial E/\partial w_i$derivative that form the gradient can be obtained by differentiation of Equation 3.15.

$$\frac{\partial E}{\partial w_i} = \frac{\partial}{\partial w_i}\frac{1}{2}\sum_{d\epsilon D}(t_d - o_d)^2$$

$$= \frac{1}{2}\sum_{d\epsilon D}\frac{\partial (t_d - o_d)^2}{\partial w_i}$$

$$= \frac{1}{2}\sum_{d\epsilon D} 2(t_d - o_d)\frac{\partial (t_d - o_d)}{\partial w_i}$$

$$= \sum_{d\epsilon D}(t_d - o_d)\frac{\partial (t_d - \mathbf{w}.\mathbf{x}_d)}{\partial w_i}$$

$$= \sum_{d\epsilon D}(t_d - o_d)(-x_{id}) \tag{3.16}$$

where x_{id} denotes the single input component x_i for training example d.Substitution of Equation 3.16 into Equation 3.14, yields the weight update rule for gradient descent for a single linear output

$$w_{new} = \eta(t_d - \sum_{d \epsilon D} o_d)\, x_{id} \tag{3.17}$$

For a BPL algorithm in a multilayer network with fixed sets of units and interconnections, the gradient descent update rule for weights tuning that attempt to minimize the squared error between the network output values and the targeted values for the outputs, according to (Mitchell, 1997) is as follows:

For each training d, every weight w_{ji} is updated by adding to it Δw_{ji} such that

$$\Delta w_{ji} = -\eta \frac{\partial E_d}{\partial w_{ji}} \tag{3.18}$$

$$E_d(\mathbf{w}) = \frac{1}{2} \sum_{k \epsilon outputs} (t_k - o_k)^2 \tag{3.19}$$

where E_d is the error on training example d, summed over all output units in the network and *outputs* is the set ofoutput units in the network, t_k is the target value of unit k for training example d, and o_k is the output of unit k given training example d.

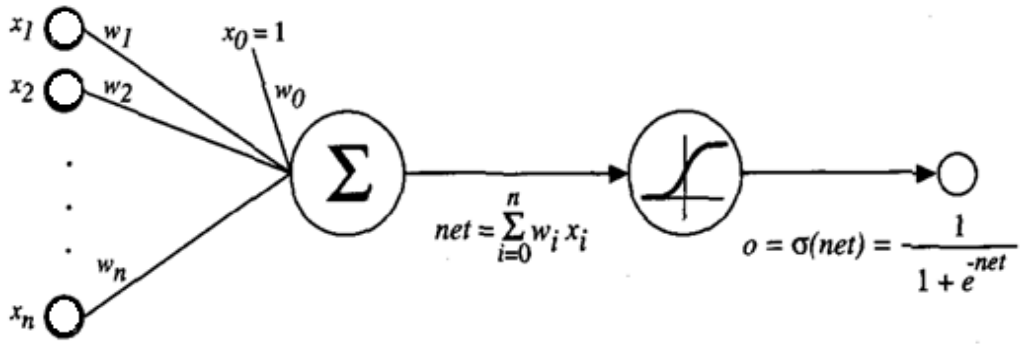


Figure 3.8: A sigmoid unit with the threshold output as a continuous function of its inputs(Mitchell, 1997).

To obtain the stochastic gradient descent rule while following the notation in Figure 3.8, a subscript *j* to denote the *j*thunit of the network was added to obtain the chain rule below;

$$\frac{\partial E_d}{\partial w_{ji}} = \frac{\partial E_d}{\partial net_j} \frac{\partial net_j}{\partial w_{ji}}$$

$$= \frac{\partial E_d}{\partial net_j} x_{ji} \tag{3.20}$$

with:

x_{ji} = the ith input to unit j

w_{ji} = the weight associated with the ith input to unit j

net_j = Σi(the weightedsum of inputsfor unit j)

o_j = the output computed by unit j

t_j = the target output for unit j

From equation 3.20, unit j can be an output for the network and also j can be an internal unit in the network, thus necessitating the derivation of a convenient expression for $\partial E_d/\partial net_j$in terms of the two cases of j.

3.2.4.3.1 Case I: Obtaining Output Unit Weights

The training rule to obtain the output unit weights involves replacing $net_j$with o_j

$$\frac{\partial E_d}{\partial net_j} = \frac{\partial E_d}{\partial o_j}\frac{\partial o_j}{\partial net_j} \tag{3.21}$$

$$\frac{\partial E_d}{\partial o_j} = \frac{\partial}{\partial o_j}\frac{1}{2}\sum_{k \epsilon outputs}(t_k - o_k)^2 \tag{3.22}$$

Since the derivatives of $\partial(t_k - o_k)^2/\partial o_j$ will be zero for all output units k except when k = j, then Equation 3.22 becomes

$$\frac{\partial E_d}{\partial o_j} = \frac{\partial}{\partial o_j}\frac{1}{2}(t_j - o_j)^2$$

$$= \frac{1}{2}2(t_j - o_j)\,\partial(t_j - o_j)/\partial o_j$$

$$= -(t_j - o_j) \tag{3.23}$$

From Equation 3.21, the second term $\partial o_j/\partial net_j$, which is the derivative of the sigmoid function, i.e

$$\frac{\partial o_j}{\partial net_j} \quad \frac{\partial \sigma(net_j)}{\partial net_j}$$

$$= \sigma(net_j)(1 - \sigma(net_j))$$

$$= o_j(1 - o_j) \tag{3.24}$$

where σ = the sigmoid function and plugging Equations3.24 and 3.23 into 3.21 gives

$$\frac{\partial E_d}{\partial net_j} = -(t_j - o_j)\, o_j\,(1 - o_j) \tag{3.25}$$

Combining Equation 3.25 with Equations 3.18 and 3.19, the stochastic gradient descent update rule for the output units is obtain thus

$$\Delta w_{ji} = -\eta\frac{\partial E_d}{\partial w_{ji}} = \eta(t_j - o_j)\, o_j\,(1 - o_j)\, x_{ji} \tag{3.26}$$

3.2.4.3.2 Case II: Obtaining the Hidden Unit Weights

The derivation of the training rule for w_{ji}must take into account the indirect ways in which w_{ji}can influence the network outputs and hence E_d for the case where *j*is an internal, or hidden unit in the network**.** Theset of all units immediately downstream of unit j in the network (i.e., all unitswhose direct inputs include the output of unit j) will be denoted by *dstream(*j). Also, *netj*can influence the network outputs and also$E_d$only through the units in *dstream(j)***.** Thus:

$$\frac{\partial E_d}{\partial net_j} = \sum_{k \epsilon dstream(j)} \frac{\partial E_d}{\partial net_k} \frac{\partial net_k}{\partial net_j}$$

$$= \sum_{k \epsilon dstream(j)} -\delta_k \frac{\partial net_k}{\partial net_j}$$

$$= \sum_{k \epsilon dstream(j)} -\delta_k \frac{\partial net_k}{\partial o_j} \frac{\partial o_j}{\partial net_j}$$

$$= \sum_{k \epsilon dstream(j)} -\delta_k w_{kj} \frac{\partial o_j}{\partial net_j}$$

$$= \sum_{k \epsilon dstream(j)} -\delta_k w_{kj} \; o_j (1 - o_j) \qquad (3.27)$$

$$-\frac{\partial E_d}{\partial net_j} = \delta_j = o_j (1 - o_j) \sum_{k \epsilon dstream(j)} \delta_k w_{kj}$$

and $$\Delta w_{ji} = \eta \delta_j \; x_{ji} \qquad (3.28)$$

where *dstream(j)* = the set of units whose immediate inputs include the output of unit *j*and Equation 3.28 is similar to Equation 3.14 and also referred to as the delta rule.

In general, gradient descent can serve as the basis for learning algorithms that must search through a hypothesis space containing many different types of continuously parameterized hypotheses. However, the error surface for multilayer networks may contain many different local minima, which gradient descent may become trapped in any of them. As a result, Back Propagation over multilayered networks is only guaranteed to converge toward some local minimum in E and not necessarily to the global minimum error.Though in practice BPL have been found to produce excellent results in many real-world applications, its slow iterations to produce a convergence make some to depend on other means of training the ANN especially taking into account the powerful computational abilities of today's computers (Castillo, Berdinas, Romero & Betanzos, 2006; Nayak, Sudheer, Rangan & Ramasastri, 2004; Halawa, 2008; Pislaru, Trandabat & Schreiner, 2006; Stepnowski, Moszynski & Dung,

2003).According to Mitchell (1997)the Back Propagation algorithm(BPA)is the most commonly used ANN learning technique as it is appropriatefor problems with these characteristics:

a. ***Instances are represented by many attribute-value pairs.*** Input attributes may be highly correlated or independent of one another and could be any real values.
b. ***The target function output may be discrete-valued, real-valued, or a vectorof several real- or discrete-valued attributes.***
c. ***The training examples may contain errors.***ANN learning methods are quiterobust to noise in the training data.
d. ***Long training times are acceptable.*** Network training algorithms typicallyrequire longer training times than other methods. Training times can range from a few seconds to many hours, dependingon factors such as the number of weights in the network, the number oftraining examples considered, and the settings of various learning algorithmparameters.
e. ***Fast evaluation of the learned target function may be required.***AlthoughANN learning times are relatively long, evaluating the learned network, inorder to apply it to a subsequent instance, is typically very fast.
f. ***The ability of humans to understand the learned target function is not important.***The weights learned by neural networks are often difficult for humans tointerpret. Learned neural networks are less easily communicated to humansthan learned rules.

3.2.4.4 The Learning Rate, η

The gradient descent method employed by the BPA searches through the space of possible network weights of a neural network, iteratively reducing the error E between the training example target values and the network's outputs. In practice however, the learning rate, η, $0 < \eta < 1$ is an arbitrary constant chosen to help get the neural network weights towards its global minimum for SSE. From Equation 3.14, the learning rate means that the change in the current weight (i.e the new weight) is equal to the negative of a small constant times the slope of the error function at w_{old}.

The learning rate must be chosen in such a way that it is neither too small or two large. If η is chosen to be too small, the neural network will take an unacceptable long time to converge to a solution when initialized and when η is large, the network will take a shorter time to converge to a solution but creates the problem of the algorithm overshooting the optimal solution which is undesirable. One solution or modification to the use of a large η, to avoid unfortunate

oscillation in arriving at the optimal solution during the network training is to allowη to change its values as the training moves forward, that is, at initialization, η is set to a relatively large value to allow the network to quickly approach pre-convergence and then the learning rate, η is gradually reduced to avoid overshooting the global minimum.

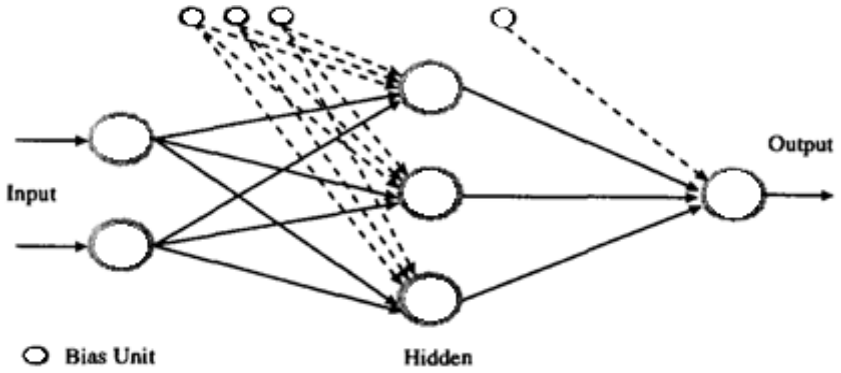


Figure 3.9: A three-layered fully connected neural network.

Source: (Topping et al., 1998)

3.2.4.5 The Momentum Term

From the modification of Equation 3.28 with a momentum term, **α** as shown below, the BPA

$$\Delta w_{ji}(n) = \eta\, \delta_j\, x_{ji} + \boldsymbol{\alpha}\Delta w_{ji}(n-1) \tag{3.29}$$

can be allowed to alter the weight-update rule of the algorithm by making the weight update on the nth iteration partly dependent on the update that occurred during the (n – 1)th iteration. The Δw_{ji} (n) term is the weight update performed during the nth iteration through the main loop of the algorithm. The momentum term or constant, **α,** $0 < \boldsymbol{\alpha} < 1$, acts as a low-pass filter by reducing rapid fluctuationsas the algorithm approach optimality and can be illustrated as a bias to the network's algorithm. According to (Basheer & Hajmeer, 2000), "the added momentum term helps direct thesearch on the error hyperspace to the global minimumby allowing a portion of the previous updating(magnitude and direction) to be added to the currentupdating step".Thus **α** accelerates the weight updates when there is a need to reduce the learning rate to avoid oscillations. The momentum term, **α** can also be likened to an illustration provided by Larose(2005), that tries to depict the function or effect of the size of **α**on the BPA. This analogous illustration is shown in Figure 3.10.

Figure 3.10: Very small **α** may cause BPA to undershoot global minimum

The consequence of a very small momentum term, **α** as symbolized with the ball in Figure 3.10 is that the algorithm could become stuck in the local minimum of point A, after initializing from point I. Thus, the small value of **α**, enables the algorithm to easily locate the first trough but does not allow it to find the global minimum at B which corresponds to the optimal solution of the task. As depicted, the small ball will require a larger size and "momentum" to 'jump out' from the first valley and through the hill.

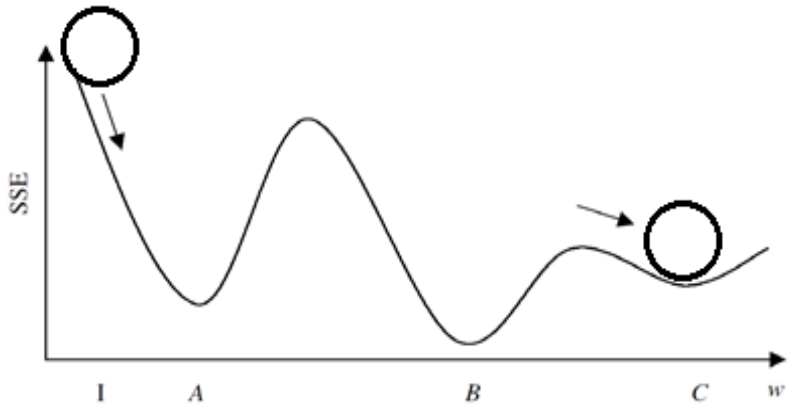


Figure 3.11: Too large **α** may cause BPA to overshoot global minimum.

In Figure 3.11, the momentum term, **α** is chosen to be larger thus possessing the needed momentum to locate the first valley on the error curve and also jump the first hill. However, as can be seen on the illustration large**α**value as depicted by the ball may be too large to allow the ball to rest at the global minimum B. The too large momentum will cause the ball to overshoot or oscillateout of convergence towards the local minimum C or beyond. It is therefore clear that the momentum term, **α** if chosen appropriately helps the algorithm in its early stages (stages close to initialization) by increasing the rate at which the weights approach the neighborhood of optimality and encourage the adjustments to stay in the same direction.

3.2.4.6 Termination Criteria

The role of a termination criterion ensures that the BPA does not loop endlessly towards infinity after convergence has been achieved or the global minimum has been located. The choice of termination criterion is an important one because too fewiterations can fail to reduce error sufficiently and too many can lead to overfittingthe training data. Since the goal of the network is to achieve a generalization of the training set to adapt to new data or testing (or validation data) set, the termination criteria must be chosen carefully. Methods of terminating an iteration includes:

i. Number of Epochs – The number of passes through the data set for training the algorithm could also constitute a termination criterion although too few training could introduce appreciable error.
ii. Time – The amount of real-time the algorithm must consume can be employed as a stopping criteria. However, this could cause a degradation in the model's efficacy as it may take several iterations to approach global minimal.
iii. Threshold level – A threshold value could be set in such a way that when the SSE of the algorithm reaches a particular level, the iteration should stop. This is also prone to overfitting and the network may suffers from memorizing the idiosyncratic patterns in the training set instead of generalizability to new data.
iv. Another method which is computationally more demanding and often requires abundant data is the cross-validation method.

3.2.4.7 Training and Verification Guidelines

The attractiveness of BPANNs comes from their remarkable information processing characteristics pertinent mainly to nonlinearity, high parallelism, fault and noise tolerance, and learning and generalization capabilities.However, according to Larose (2005), "regardless of the stopping criterion used; the neural network is not guaranteedto arrive at the optimal solution, known as the *global minimum* for the SSE. Rather,the algorithm may become stuck in a local minimum, which represents a good, if notoptimal solution".The author further suggested the following cross-validation termination procedure:

1. Retain part of the original data set as a holdout validation set.For example, the set of all known samples is broken into two orthogonal (independent) sets, that includes the Training set and the Testing (or Validation) set belonging to the universal set of all known samples.

►Training sets – A group of samples obtained from all the set and used to train the neural network.

► Testing set – A group of samples obtained from all the set or the remaining set of samples that is used to test the performance of thedesigned neural network.

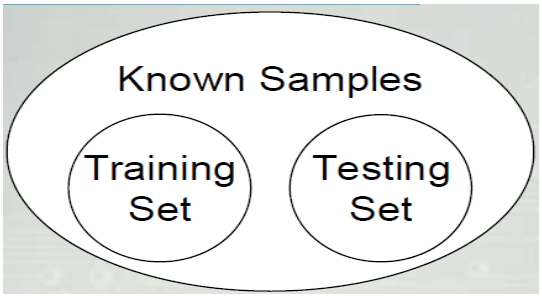


Figure 3.12: Create two orthogonal and independent sets from the known samples

2. Proceed to train the neural network as elaborated above on the remaining training data.

3. Apply the weights learned from the training data on the validation data.

4. Monitor *two sets of weights*, one current set of weights produced by thetraining data, and one best set of weights, as measured by the lowest SSE sofar on the validation data.

5. When the current set of weights has significantly greater SSE than the best setof weights, then terminate the algorithm.

3.2.4.8 Sensitivity of a Trained Network

Though neural networks provide an excellent flexibility that allows modeling of a wide variety of nonlinear behavior, its operation is shrouded in opacity making users not to posses the ability to peek or analyze the interpreted results using easily verifiable process or formulated rules no straightforward procedure exists for translating the weights of a neuralnetwork into any known mathematical or statistical technique.However, the procedure of sensitivity analysis, according to Larose (2005) doesexist. The sensitivity of the designed network allows a measure of the relative influence each attribute has on the output result of the network. It involves:

1. Generating a new observation x_{mean}, with each attribute value in x_{mean} equal tothe mean of the various attribute values for all records in the test set.

2. Then, find the network output for input x_{mean}. Call it $output_{mean}$.

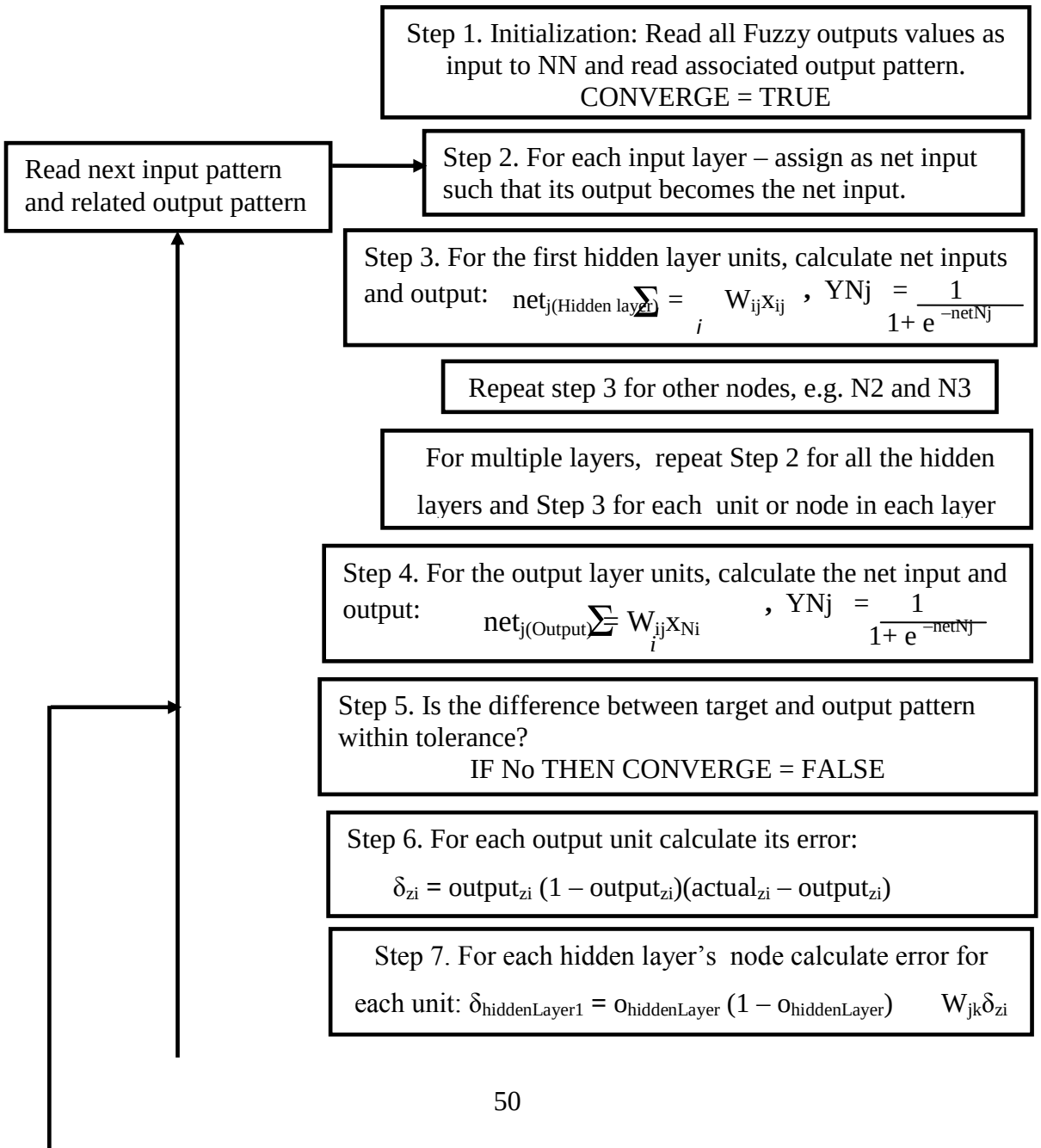
3. Attribute by attribute, vary x_{mean} to reflect the attribute's minimum and maximum values.Then, find the network output for each variation and compare it to $output_{mean}$.

It was observed that the sensitivity analysis showed that varying certain attributes from their minimumto their maximum values while others are kept constant had a greater effect on the resulting network output thanit had for other attributes. The idea of the sensitivity can help prune a network of unnecessary inputs or inputs that have very negligible influence on the overall performance of the ANN. Thus it responds to arbitrary variations in the values or the removalof any of the neurons contributing to theinput. Other analysis such as the calculation of the Confusion (or Error) matrix to visualize the performance of the supervised backpropagation algorithm and the receiver operating characteristics (ROC) graphs that shows

a tradeoff between sensitivity and specificity on data samples during training, validation and testing.

3.2.4.9 The Back Propagation Algorithm

The non opaque nature of the ANN model and the various iteration flows in the design is shown in the Back Propagation Algorithm in Figure 3.13.This non opaque structure of the ANN gives the reason why this method has been chosen for implementing this study due to the critical nature of this research and modeling requirements for complex systems such as the biological human system. The algorithm shows the forward and backward flow for each iteration as a convergence is been sought by the network with input to the algorithm obtained from normalized and fuzzified values from the fuzzy inferencing system.



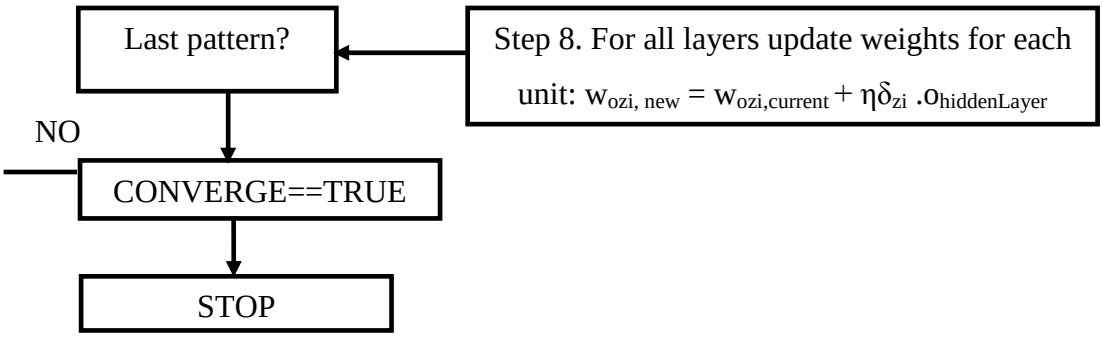


Figure 3.13: The Back Propagation Algorithm for the ANN

3.2.5 K-Nearest Neighbor Algorithm

In data mining methods, the k-nearest neighbor algorithm, k-NN was developed to handle mostly classification and regression tasks by allowing an object to be classified by a majority vote of its neighbors. The k-NN which can also be used for estimation and prediction is a type of instance-based learning since the function is only approximated locally while all its computation is deferred until classification. This algorithm is regarded as one of the simplest of all machine learning algorithms as noted by (Wikipedia, 2015). In k-NN, weights are assigned to the contributions of the neighbors, such that the nearer neighbors contribute more to the average than the more distant neighbors. The neighbors are members of the set of objects for which the class (the k-NN object) is known. The k-NN algorithm is however sensitive to the local structure of data and uses a distance function to locate its objects by utilizing the Euclidean distance function (Theodoridis, Pikrakis, Koutroumbas & Cavouras, 2010)which is approximately the normal way humans compute or perceive distance. The Euclidean distance function is however not helpful for determining high order dimensionality such as neighbors that are represented by vector space. To handle such complex classification and predictive task, another distance function employed by k-NN is the overlap metric or Hamming distance which is useful for text classification.

Most data mining methods such as k-NN, ANN and Decision trees are supervised methods as contrasted with unsupervised methods. Supervised methods involves the network been presented with pre specified target values during a training session for which predictor variables could then be used to classified new targets.In analysis of the k-NN algorithm, to

obtain the distribution of each point with respect to its nearest neighbours,the Euclidean distance function was used, i.e.

$$d(x,y) \quad = \quad \sum_i (\text{xi} - \text{yi}) \tag{3.30}$$

where $x = x_1, x_2, \ldots, x_m$ and $y = y_1, y_2, \ldots, y_m$represent the mth attribute of the two related records.

During implementation, all samples in the training set could be considered when classifying a new instance yielding an algorithm that is termed global method. However, though this method improves the reliability of the classification but possesses the disadvantage of running very slowly. If only the nearest training examples are considered, then *local* method is obtained.

3.2.6 Data Analysis and Uncertainty

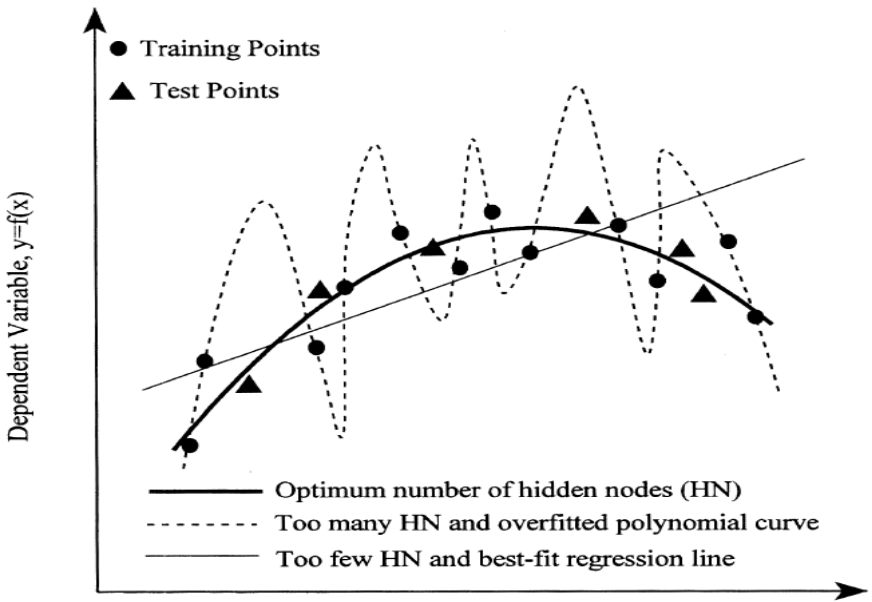
Fundamental to most data mining activities, if not all is the notion of a sample which could be domiciled in a database structure. The omnipresence of uncertainty in sampled data from our databases and even from our daily lives gives us the desire to be able to predict or estimate even with some tolerable errors the uncertain or random future. As pointed out by Hand, Mannila and Smyth(2001), "one of the great achievements of science (is) that we have developed a deep and powerful understanding of uncertainty. The capricious gods that were previously invoked to explain the lack of predictability in the world have been replaced by mathematical, statistical, and computer-based models that allow us to understand and manipulate uncertain events. We can even attempt the seemingly impossible and predict uncertain events, where prediction for a data miner either can mean the prediction of future events (where the notion of uncertainty is very familiar) or prediction in a nontemporal sense of a variable whose true value is somehow hidden from us (for example, diagnosing whether a person has cancer, based on only descriptive symptoms).The basic tool for dealing with uncertainty is probability(Larose, 2006;Sebe, Cohen, Garg & Huang, 2005; The Open University, 2006); this is because the field of data mining often encounters huge data sets, for which the prior distribution is dominated by overwhelming amount of information to be found in the observed data. After observation, the prior information about the distribution of Θ can be updated. This modification leads to the posterior distribution, $p(\Theta|\mathbf{X})$, where x represents the entire array of observable data as was first discovered by Reverend Thomas Bayes and

hence the theory was nicknamed Bayesian theory. In Bayesian statistics, parameters are considered to be random or instance variables and the data is considered known.

3.2.7 The Optimal Number of Hidden Nodes and Layer Size

One of the most critical tasks in ANN designing is the choice of the number of hidden layer, NHL and the number of hidden nodes, NHN. In general, a trade off is usually sorted from all the participating parameters including the average time required for the network convergence, the computational ability and memory of the processing unit, the complexity in the model designed, the critical nature of the solution needed to be obtained, for example, the noise tolerance level, etc.

The NHL is the number of interlayer or gap between the two successive layers (input, output or another layer) that encloses the connection weights. One hidden units is appropriate for most continuous approximation functions while in general, two hidden layers may be appropriate for learning functions with discontinuities. Three or more hidden layers may lead to a messy log jam and overwhelming of computational power. The optimal NHN, is mostly determined from painstaking trial and error approaches, as with increasing number of hidden nodes training becomes excessively time-consuming. The NHN for generalizing the network is a function of the input-ouptut vector sizes, training, validation and testing subsets, the problem of nonlinearity, etc. Unlike the known inputs and outputs units, no prior knowledge of the NHN is provided and has to be determined objectively. With increasing hidden nodes, the network becomes excessively time-consuming. For example, if the hidden nodes are too many the algorithm will follow the noise in the data due to over parameterization leading to a poor generalization for untrained data while if the NHN is too few then the algorithm would be incapable of differentiating between complex patterns leading to only an imprecise linear estimate of the actual trend in the data. A comparison of the various NHN is provided by (Basheer & Hajmeer, 2000) and it is illustrated in the Figure 3.14.



3.3 System Design

The system design approach adopted for the purpose of achieving the objectives of this research involves obtaining the fuzzy inference of entered patient's symptoms into the designed Graphical User Interface(GUI) by the user or healthcare provider using simple clicking of options buttons on the GUI. These option buttons corresponds to fuzzy logic and generates the fuzzy membership function to be used in the analysis. The system then analyzes the inputs and givesout results along with necessary line of action(s) based on the present or real-time processed data.

3.3.1 The Fuzzy GUI

To analyze the largely vague and ambiguous reporting of patient's symptoms that is mostly filled with nuances, the graphical user interface (GUI), shown in Figure 3.15 was developed.With this GUI, healthcare providers and even patients can respond to basic questions of occurring symptoms in patients as they present themselves in the hospital and clinics using the provided GUI which accepts inputs from its user by simply clicking corresponding options buttons of occurring symptoms. The expert system then automatically generates the normalized fuzzy value of the inputted symptomfor each option button.

Figure 3.15: The GUI diagnostic console for inputting patient's symptoms in real-time.

The GUI contains nine classification groupings of patient's symptoms each with three member parts corresponding to twenty seven likely occurring symptoms for a contagious disease patient making its first appearance in the public i.e a possible index case. By clicking on any option button, the fuzzy value corresponding to that option button is 'fired' and thus contributes to the current analysis. The GUI which was designed with the Visual Basic programming language was named SOSIClinic Expert System (SES) and can run in any Microsoft Windows operating system platform. The software also allows for direct input or edit of the inputted symptom's fuzzy values such that the inputted value must be greater than zero and less than one on the text boxes on the right of the symptoms. However, this is for the technically advanced user. The software allows for populating a database with occurring patients' symptoms which can then be used for data mining and related applications.

To further improve the input stage for symptoms, a temperature sensor can be provided close to the diagnostic computer hosting the program and directed towards the patient under diagnosis to obtain the real-time temperature state or changes of the patient. This temperature or rate of change of temperature can be processed to ascertain or collaborate the fuzzy reports been collated by the SES based on normal and abnormal body temperature readings in relation to infectious diseases, their risk factors, mechanism of disease spread, etc. according to (Lee, Hsu & Stasior, 1997; MedGuidance, 2015;Thibodeau & Patton, 2010).

Bearing in mind that higher temperatures in a patient could be indicative of the presence of mild or high fever, it is also important to know that a patient's high temperature (up to 39^{o}C – a mild fever) could help the patient's immune system to get rid of an infection (BetterHealth, 2015) while according to the source, a body temperature equal to or greater that 42.4^{o}C could cause irreversible damage to the brain of an elderly person. Thus the dependence in temperature is relative and not necessary conclusive enough to confirm a contagious disease presence in the patient, rather the medical professionals can rely on their experience in using ICT solutions such as this.

Table 3.1 Analysis of normal body temperatures (NBT) in humans: babies, infants, teenagers, adults and the elderly. Source (MedGuidance, 2015).

°F	0 - 2 years	3 - 10 years	11 - 65 years	> 65 years
Oral	—	95.9 99.5	97.6 99.6	96.4 98.5
Rectal	97.9 100.4	97.9 100.4	98.6 100.6	97.1 99.2
Axillary	94.5 99.1	96.6 98.0	95.3 98.4	96.0 97.4
Ear	97.5 100.4	97.0 100.0	96.6 99.7	96.4 99.5
Core	97.5 100.0	97.5 100.0	98.2 100.2	96.6 98.8

After initial input into the SES GUI, a result of the fuzzy inference system could be obtained by the user which may be sufficient enough to ascertain the state of the patient under diagnosis. However, if the healthcare giver or user of SES, get a premonition or hunch of the presence of a disease with a new characteristics or feature in the occurring symptoms on the patient in real-time, then provided textboxes will be used by the user to list up to three of such identified symptoms. This new symptoms will be used for classification and as extra input to the neural network. In fact, the new symptoms from the last three inputs into the neural network. From the fuzzy obtained symptoms and the new symptoms, the neural network while using the fuzzy symptoms as the result or output of the ANN, will then try to obtain the level of culpability of the identified symptoms as inputted by the healthcare personnel. The k-nearest neighbors algorithm becomes active in placing the new case to any of the already existent and diagnosed diseases such as Ebola, Marburg, Lassa fever, etc. by using the Euclidean distance based clustering technique. Theuse of this advance function and expert system's contribution has been left to the professional healthcare providers alone to use since according to Cho (2012), only about "36% of U.S. adults may have limited or basic health literary; that is, literacy and problem-solving skills that restrict their ability to navigate a variety of health situations" and support solutions.

The role of the healthcare professional is essential even in the use of ICT based tools as was evident in the rapid response of several healthcare providers in the fight against the 2014 ebola outbreak in Nigeria according to Office of the Special Adviser to the President on Research, Documentation and Strategy (2014).The role of empathy, patience, politeness, etc. rather than apathy from healthcare providers as a prerogative for getting information from patients according to (Harvey & Koteyko, 2013) will help ICT solutions such as this to be more precise and effective in analysis of diagnosis. The acquired information by the healthcare giver can be verified by the visible temperature marker on the top right hand corner of the diagnostic GUI or a separate temperature sensor gadget. A very high or changing (increasing) temperature could contribute to indicate a possibility of a contagious case and help reduce false alarms. Also, to curtail spread of contagious diseases, even in the hospitality industry hidden temperature sensors gadgets could be focused on persons during check in and collated data could be stored or transferred to other sources during the tracking of an index patient who may have brought a contagious disease to a locality using Electronic Health Records(HER)(Davis & LaCour, 2007; Sloan, Legrand & Chen, 2013). Manifesting symptoms of diseases were aggregated by the summation of all symptoms used in diagnosis of such an ailment according to (Osigbemeh, Ogunwolu, Omoare &Inyiama, 2014). Also, the level of confidence of resulting analysis and EIJ values are obtained from the algorithm below:

3.3.1.1 Fuzzification Algorithm

Obtain all possible symptoms and signs (Ω)

1. Initialization:

Read all manifesting symptoms from patient (Ψ)

Set diagcalc = 0

For all manifesting symptoms Ψ ε Ω add to diagcalc

Update:

$t_i \leftarrow t_i + 1$ (Counter to be used for calculating convergence values)

2. Fuzzification

For i = LBound To UBound of each symptom value

diagcalc = diagcalc + Val(i)

Next i

t_k = diagcalc / Val(t_i) (obtaining Confidence level values)

EIJ = 1/t_i* 100 (EIJ is constant if all or most symptoms are participating)

If $t_k >= x_1$ then

Display message i_1

Else if $t_k >= x_2$ then

Display message i_2

.

.

Else if $t_k >= x_n$ then

Display message i_n

End if

UpdateSession:

$t_K \leftarrow t_K + 1$

3. Send pre-processed update to neural network

4. If Update requirement is satisfied

Then Stop

Else Goto 2

3.3.2 The Designed Neural Network

To handle the possibility of a new case in the diagnosis process or to revalidate the result of the fuzzy inference, the fuzzified values from the GUI's processing were used as inputs to the artificial neural network architecture shown in Figure 3.16. The ANN consists of thirty inputs units of twenty seven fuzzified already known inputs and three "new" symptoms inputs as identified by trained personnel.This constitutes a supervised learning since the result of the ANN will have to converge to the inputted solution. It also consists of three hidden units to handle the preprocessing of the inputs in the interlayer between the inputs and the outputs units. As placed succinctly by Basheer and Hajmeer (2000), this network is an example of a Feedforward Error-BackPropagation Learning Algorithm(FEBPLA) famous for training ANNs. The FEBPLA operates by searching an error surface using gradient descent for points with minimum error. For the duration of each iteration, the network generates two sweeps: a forward sweep that produces a solution and a backward propagation of the computed error to modify the weights. During initialization, the ANN is fed with assumed initial weights as one training example such that from the input layer, each input node transmits the value received forward to each hidden node in the hidden layer. The error when compared to the target output is determined and the activation of that node is obtained using the sigmoidal transfer function to obtain an output value of between 0 and +1 (non inclusive). The values of 0 and 1 are not inclusive in the sigmoidal output because according to Mitchell

(1997), “the reason for avoiding target values of 0 and 1 is that sigmoid units cannot produce these output values given finite weights. If we attempt to train the network to fit target values of exactly 0 and 1, gradient descent will force the weights to grow without bound. On the other hand, values of 0.1 and 0.9 are achievable using a sigmoid unit with finite weights. The same procedure was then repeated for each hidden node and for all hidden layers. The net effect of all these procedures is then transformed into an activation function at the output nodes and thus represents the ANN solution of the fed samples. This output is most likely to deviate from the target solution due to the randomly initialized interconnected weights for which the probability of obtaining a converged solution by just the initial iteration is extremely infinitesimal.

However, in the backward sweep, the error difference from the targeted output and the recent output from the ANN is used to adjust the interconnected weights. Repeated forward and backward sweeps will eventually allow the outputted solution or target value to converge or agree with the FEBPLA of the ANN within allowable tolerance which could be prespecified. From the transparent process above though may be opaque at a first glance considering the long execution time and several epochs or iterations needed to be run to ensure a convergence, neural network methods are not “black box” models.Sometimes, as pointed out by Bhadeshia (1999), neural networks are incorrectly described as “opaque” systems. A lot depends on the computational ability, strength and efficiency of the host computer to handle huge data and on the analysis provided by the implementing models to complete the convergence in record time(Zhou, Chawla, Jin & Williams, 2015). The work of Topping et al. (1998) suggested either a single (mostly used method) or batch pattern training for weights adaptation with the latter more appropriate for parallel processing for weight adaptation and update. Since the ANN are opaque to direct validation using other methods of data mining and are increasing been used in artificial intelligence for modeling complex systems that involves global optimization care must be ensured that errors during model designing, uncertainty in human inputs and the environment are eliminated as much as possible (Gopal, 2009;Jakeman, Voinov, Rizzoli & Chen, 2008).

The superiority of ANN over other popular methods of AI such as knowledge based systems (KBS) is the ANN’s ability to learn from examples. This is similar to a human learning from experience and learning through trial and error. The basic difference between ANNs and KBSs is that ANNs do not require a knowledge base or its equivalent during problem solving. ANNs only require a number of solved problems (input and output sets) in order to train the network and produce a validational set of weights that can quickly be used to sort other new data. However, KBSs and ANNs can be combined together so as to take

advantages of their strengths to solve specific problems (Krishnamoorthy & Rajeev, 1996). This reason motivated the use of ANN modeling over knowledge or case base reasoning (CBR) in this work.

A summary of the various aspects of implementing a ANN is listed on Table 3.2.

Table 3.2 Analysis of the effects of various values of design parameters on ANN training convergence and generalization.

Design parameter	Too high or too large	Too low or too small
Number of hidden nodes (NHN)	Overfitting ANN (no generalization)	Underfitting (ANN unable to obtain the underlying rules embedded in the data)
Learning rate (η)	Unstable ANN (weights) that oscillates about the optimal solution	Slow training but stable optimal solution convergence
Momentum coefficient (μ)	Reduces risk of local minima. Speeds up training. Increased risk of overshooting the solution	Suppressed effect of momentum leading to increased risk of potential entrapment in local minima. Slows training
Number of training cycles	Good recalling ANN (i.e., ANN memorization of data) and bad generalization to untrained data	Produces ANN that is incapable of representing the data
Size of training subset (N_{TRN})	ANN with good recalling and generalization	ANN unable to fully explain the problem and also generalize

Size of test subset (N_{TST})	Ability to confirm ANN generalization capabilty	Inadequate confirmation of ANN generalization capabilty

Figure 3.16 shows the connection patterns for the inputs, weights and outputs of the implemented ANN for this work. The ANN is completely connected, meaning that every node in a given layer is connected through a weight (w_{ij}) to every node in the next layer, although not to other nodes in the same layer. At initialization, these weights are randomly set to between 0 and 1. The network contains thirty inputs units in the input layer and three nodes in the hidden layer where weighting is implemented and five output nodes which corresponds to the outputs of the fuzzy inferred diseases. For a full illustration of the designed ANN with all its weights, nodes and bias see Figure 5.1 in Appendix A.

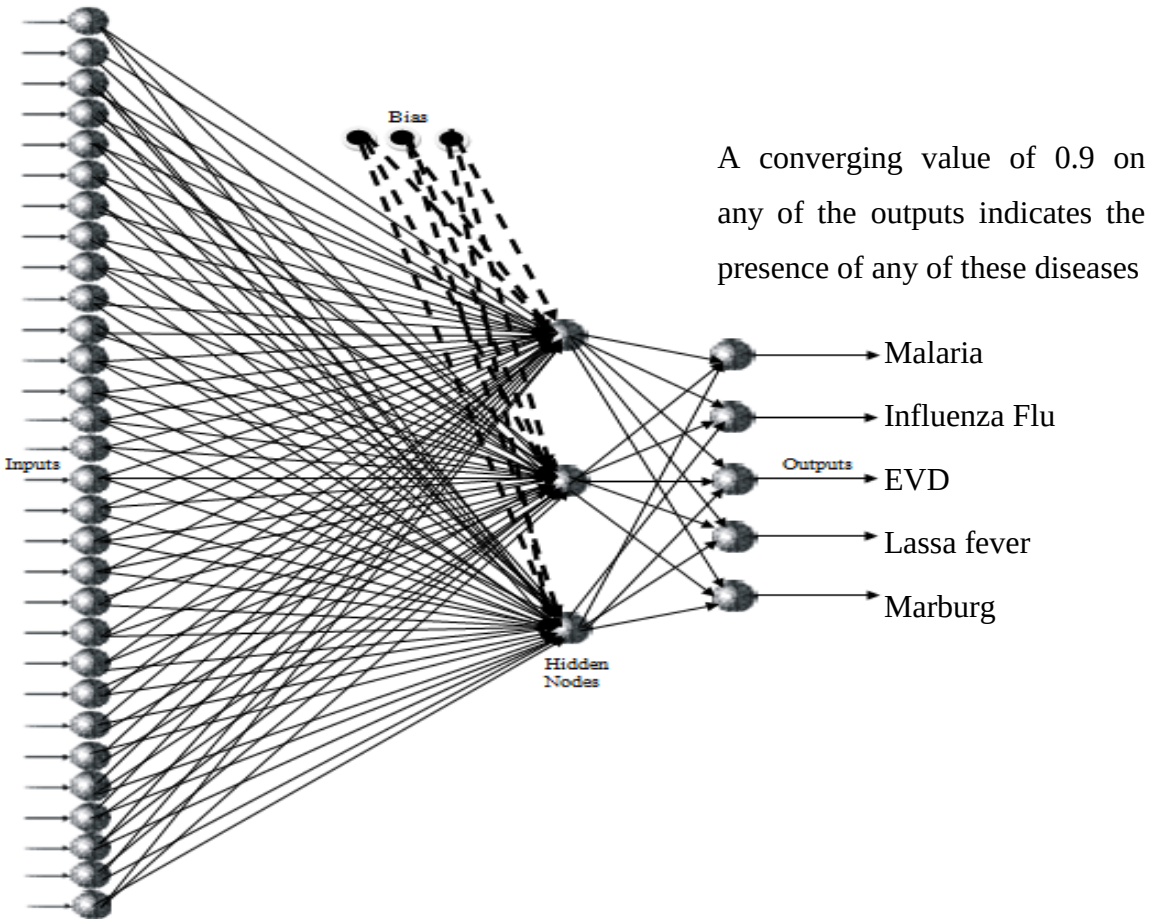


Figure 3.16: Connection patterns for input neurons, bias and output for the ANN

Further analysis can help show the effect of new symptom(s) on the present diagnosis and the similarity with already existing and diagnosed conditions. This is done by means of the k-nearest neighbor algorithm after normalization of the outputs, thus allowing for classification of new disease strains or new symptoms.

Table 3.3 shows a sample of the normalized initial inputs for training the ANN for a case of Influenza flu, i.e. without new symptoms.

1	0.1	6	0.9	11	0	16	0	21	0.7	26	0.8
2	0.75	7	0	12	0.75	17	0	22	0	27	0.5
3	0.1	8	0.25	13	0.9	18	0.5	23	0.9	28	0
4	0.5	9	0	14	0	19	0.25	24	0.5	29	0
5	0	10	0.5	15	0.75	20	0	25	0	30	0

The GUI for the FEBPLA showing preliminary execution of the ANN is shown in Figure 3.17. An adjustable learning rate of 0.1 was initially chosen with randomized weight's values of less than one, i.e. $0 < W < 1$ (0 and 1 non inclusive).

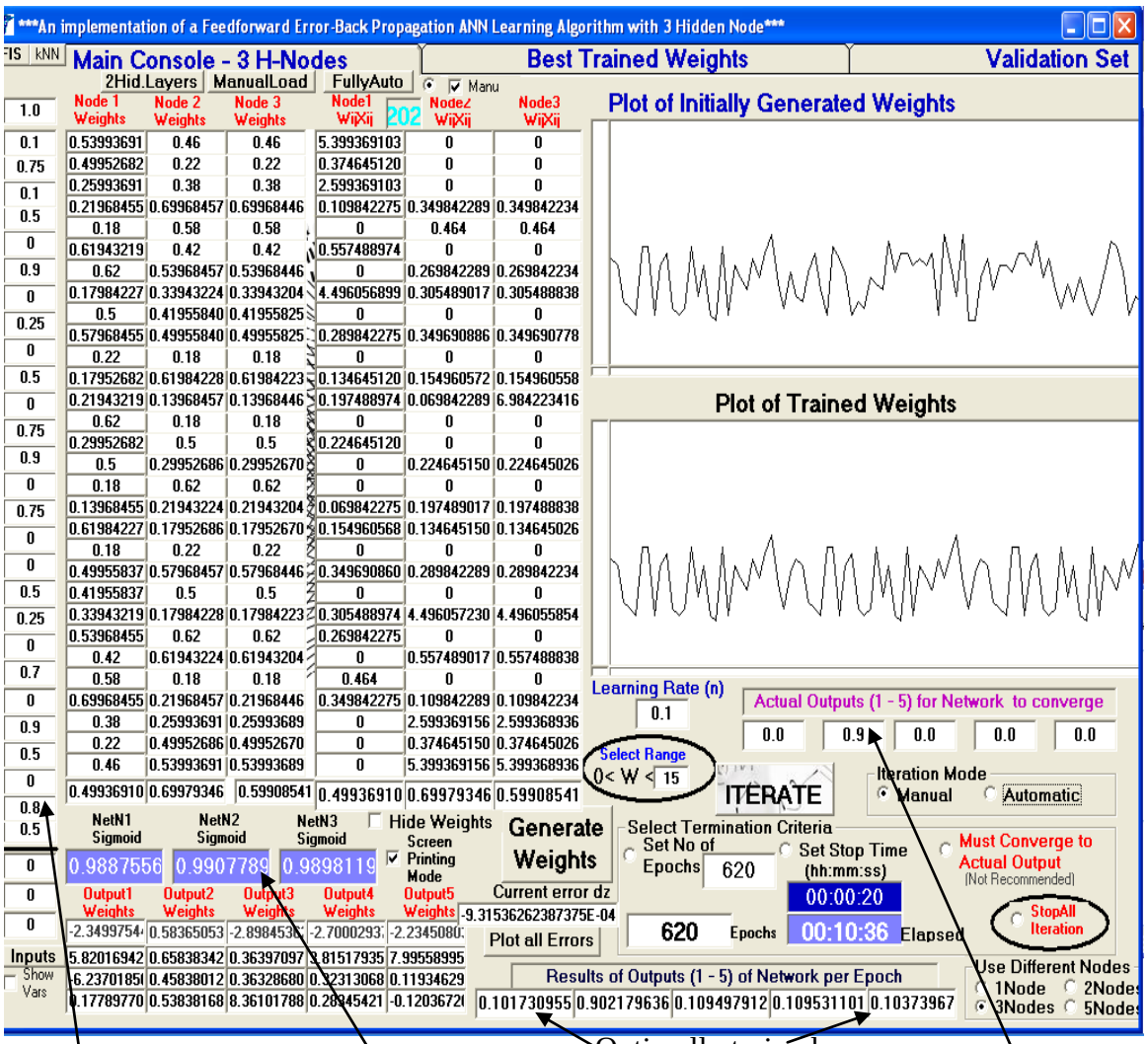


Figure 3.17: A Screenshot of the designed ANN showing the results of weights during the 620th epoch and the initial training inputs.

Though the ANN cannot in practice converge to the actual values of taught or learned outputs, it continues to fine tune its convergence solution to be closer to the actual outputs in each iteration until an optimal solution is attained after which the iteration will overshoot optimality and produce errors. The resultant optimal convergence point is critical in the generalization of the ANN to new data. New symptoms input as shown in Table 3.4 can be fed to the designed ANN and their effect on the present diagnosis investigated.

Table 3.4 A sample of the normalized initial inputs for training the ANN for a case of Influenza flu with the last three inputs supplied (i.e inputs 28 – 30).

1	0.1	6	0.9	11	0	16	0	21	0.7	26	0.8
2	0.75	7	0	12	0.75	17	0	22	0	27	0.5
3	0.1	8	0.25	13	0.9	18	0.5	23	0.9	28	0.9
4	0.5	9	0	14	0	19	0.25	24	0.5	29	0.6
5	0	10	0.5	15	0.75	20	0	25	0	30	0.3

3.3.3 The Neural Network Architecture

The type of designed topology or architecture of ANNs brings out their transparent processing that is devoid of the "black box" model in that the working and learning of the networks follow a straight forward and re-verifiable process which makes ANNs highly resourceful tools for analysis, data mining and prediction. A single and double hidden layer topology was considered.

3.3.3.1 One Hidden Layer Topology

Choosing the right number of nodes in the hidden layer of the ANN is crucial for a good analysis. The various sweeps for such a network is analyzed below.

3.3.3.1.1 First Network Sweep or Iteration

The implemented hidden layer topology(HLT) of the ANN for this research involves identifying a constant input x_{cj} with a value of 1 allocated to it by convention and thirty inputs $x_0, x_1, \ldots, x_{29}$ which are the normalized or pre-fuzzified upstream values of between 0.1 to 0.9 inclusive. These values combined linearly with the generated equivalent weights W_{ij} using the summation, Σ combination function to form the composite value called net_j. W_{ij} represents the

weights associated with the ith input to node j forming a total of 30 + 1 inputs to node j. Hence, each hidden or output layer node j has an "extra" input of $W_{cj}x_{cj} = W_{cj,}$ this forms the first component of the net_j computation.

$$net_{j(\text{Hidden layer})} = \quad W_{ij}x_{ij} \tag{3.31}$$

$$= W_{cj}x_{cj} + W_{0j}x_{0j} + W_{1j}x_{1j} + \ldots + W_{29j}x_{29j}$$

$$= W_{cj} + W_{0j}x_{0j} + W_{1j}x_{1j} + \ldots + W_{29j}x_{29j} \tag{3.32}$$

The resulting computation yielded net_{N1} which after sigmoidation yielded the value for X_{ij} into the output layer. The squashing sigmoid function is then

$$Y_{N1} = \frac{1}{1 + e^{-netN1}} \tag{3.33}$$

also, for node 2 and 3, i.e. net_{N2} and net_{N3} respectively, Y_{N2} and Y_{N3} was computed with the resulting squashed values after application of the sigmoid function serving as inputs to the output layers.

To obtain net_j for the output layer, the randomly generated output weights were combined using the summation combination function with the computed net_{Ni} to obtain the final output for the first forward sweep or iteration of the network. That is from Equation 3.8;

$$net_{j(\text{Output})} = \quad \sum_i W_{ij}x_{Ni}$$

$$= W_{cj}x_{cj} + W_{0j}x_{0j} + W_{1j}x_{1j} + W_{2j}x_{2j}$$

$$= W_{cj}x_{cj} + W_{0j}net_{N1} + W_{1j}\, net_{N2} + W_{2j}net_{N3} \tag{3.34}$$

Since the constant term is of the value 1.0, Equation 3.34 became

$$= W_{cj} + W_{0j}net_{N1} + W_{1j}net_{N2} + W_{2j}net_{N3}$$

This output values from the hidden layer nodes was compared with the expected output to determine the error value and the degree of weight adjustment for the next iteration of the network. This was done for j = 0, 1 … 4 for five outputs.

3.3.3.1.2 Second Network Sweep or Iteration

For this iteration, the error responsibilities of the output and the hidden layer were computed as follows. The output layer's error responsibility, δ_z is given as:

$$\delta_{zi} = output_{zi}\,(1 - output_{zi})(actual_{zi} - output_{zi}) \tag{3.35}$$

so that for the first of the five outputs,

$$\delta_{z1} = \text{output}_{zi}\,(1 - \text{output}_{z1})(\text{actual}_{z1} - \text{output}_{z1}) \text{ and so on, i.e. } \delta_{z1}, \delta_{z2}, \ldots \delta_{z5}.$$

The change in weight is then computed from Equation 3.26 as

$$\Delta W_{ozi} = \eta\delta_{zi}(1)$$

and the new weight for populating each output node is obtained from

$$w_{ozi,\,new} = w_{ozi,current} + \Delta W_{ozi} \tag{3.36}$$

Also, the hidden layer's error responsibility for each node is given as

$$\delta_{hiddenLayer} = \text{output}_{\,hiddenLayer}\,(1 - \text{output}_{hiddenLayer}) \sum_{Downstream\ i} W_{jk}\delta_{zi} \tag{3.37}$$

The new update rule for that hidden layer's new weight is then,

$$\Delta W_{hiddenLayer} = \eta\delta_{zi}\,.\text{output}_{\,hiddenLayer} \tag{3.38}$$

The new weight for populating each hidden node's updatein the layer is then

$$w_{ozi,\,new} = w_{ozi,current} + \Delta W_{ozi} \tag{3.39}$$

This update rule is maintained for each subsequent iteration until eventual optimal convergence of the network to the required solution is attained.

3.3.3.2 Two Hidden Layers Topology

The behavior of multiple hidden layers on ANN's output was also explored using the two HLT. To obtain more intrinsic features of the implemented ANN, the output from the first hidden layer (FHL), y_1 was coupled to another hidden layer just immediately right to the FHL when moving feed-forwardly in a completely connected network. The architecture which is depicted in Figure 3.18 and in Appendix L, shows the fed inputs being weighted by the second hidden layer (SHL) weights to form the input into the output layer from where gradient descent produces an output for that iteration. During the second iteration, the error obtained is then used to adapt the weights into the FHL in the direction of reducing the generated error from the last iteration. This process is repeated for each subsequent iteration until the convergence criteria is attained.

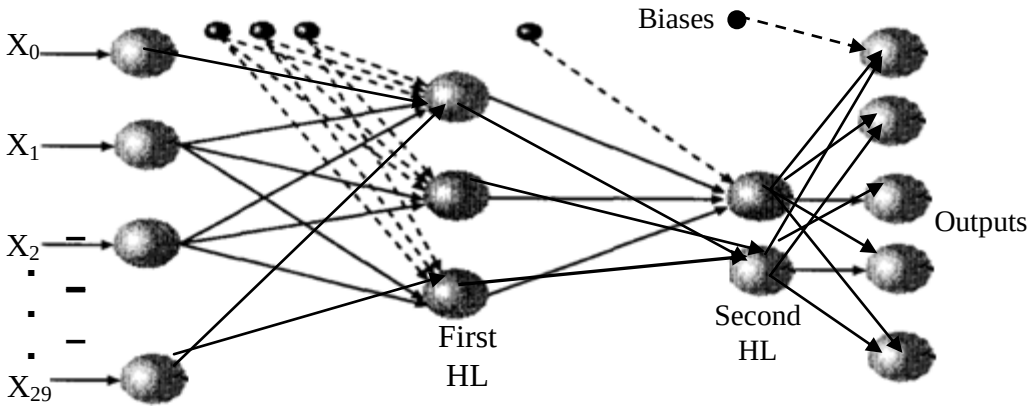


Figure 3.18: Architecture for the two hidden layer ANN

3.3.3.2.1 First Network Sweep

For the two HLT, instead of coupling the resulting solution from Equation 3.34 to the outputs directly, the resulting solution values from the first hidden layer nodes ($net_{j(Output)}$) was used as input into the SHL. A summation of these values with the generated weights of the second layer nodal values was then used to feed the output layer after squashing with the sigmoid function. The generated weights w_{in} served as the weights associated with the SHL to which inputs from the FHLnodes was summed to obtain net_s below.

$$net_{s(\text{second HL})} = \sum_{Downstream\ i} W_{in}y_{in} \tag{3.40}$$

$$= W_{cn}y_{cn} + W_{0n}y_{0n} + W_{1n}y_{1n} + W_{2n}y_{2n} \text{ and since } y_{cn} = 1;$$

$$= W_{cn} + W_{0n}y_{0n} + W_{1n}y_{1n} + W_{2n}y_{2n} \tag{3.41}$$

where W_{cn} is a constant introduced by convention in other for the ANN not to disappointly damp to zero and $y_{in} = net_{j(Output)}$ from each first hidden node.

The net_s from each first hidden node are then computed and the resulting computation yielded net_s and for flexibility a reciprocal function on net_syeilded

$$net_S = 1 / net_{s(\text{second HL})} \tag{3.42}$$

The introduced reciprocal function by this research helped to reduce the effect of the squashing operation by the sigmoid function that was used in the upstream hidden layer and also to improve the peculiar slow convergence for steepest or gradient descent optimization of multiple hidden layers. The resulting net_S for node 1 and 2 of the SHL obtained from Equation 3.42 was sent for sigmoidation which yielded

$$Y_{In} = 1 / (1 + e^{-netNs1})$$

The resultant or actual output for the iteration was obtained from

$$net_{s(SecLayerOutput)} = \sum_{i} W_{in} y_{in} \quad (3.43)$$

$$= W_{cn} y_{cn} + W_{0n} y_{0n} + W_{1n} y_{1n} \quad \text{and since } y_{cn} = 1.0$$

$$= W_{cn} + W_{0n} y_{0n} + W_{1n} y_{1n} \quad (3.44)$$

3.3.3.2.2 Second Network Sweep

The error responsibility of the second forward sweep's output is given by

$$\delta_{hi} = output_{hi} (1 - output_{hi})(actual_{hi} - output_{hi}) \quad (3.45)$$

i.e. for i = 1, $\delta_{h1} = output_{h1} (1 - output_{h1})(actual_{h1} - output_{h1})$

and so on, i.e. $\delta_{h2}, \delta_{h3, \ldots} \delta_{h5}$ and the change in weight is then computed from

$\Delta W_{oh1} = \eta \delta_{hi}(1)$ and the new weight for populating each output node is

$$w_{ohi,\, new} = w_{ohi,current} + \Delta W_{ohi} \quad (3.46)$$

Also, from Equation 3.37, the first hidden layer's error responsibility is

$$\delta_{hiddenLayer1} = output_{hiddenLayer1}(1 - output_{hiddenLayer1}) \sum_{Downstream\ i} W_{jk} \delta_{hi}$$

The new update rule for that hidden layer's new weight is then,

$$\Delta W_{hiddenLayer1} = \eta \delta_{hi} .output_{\,hiddenLayer1} \quad (3.47)$$

and the new weight for populating each hidden node in the FHL is

$$w_{ohi,\, new} = w_{ohi,current} + \Delta W_{ohi} \quad (3.48)$$

The second hidden layer's error responsibility is computed as follows

$$\delta_{hiddenLayer2} = output_{hiddenLayer2}(1 - output_{hiddenLayer2}) \sum_{Downstream\ i} W_{jk} \delta_{hi} \quad (3.49)$$

The new weights are then updated and the weights for populating the second hidden layer's nodes was obtained from Equations 3.47 and 3.48 with i set to 2.

3.3.4 Resulting Fuzzy-Neural Model

The ANFIS model developed is both knowledge and data driven (Nauck, 1997)and results to representing the function with the following:

R: **IF** x_1 is μ_1**and**x_2 is μ_2,**and** … **and** x_n is μ_n

then pattern (x_1, x_2, …. X_n) belonging to Class Z

where $\mu_{1, \ldots} \mu_n$ are fuzzy sets which described the presented symptoms pattern. This conclusion must be supported or validated by the ANN processing in a particular convergence epoch to be empirically determined thus leading to a cooperative ANN and FIS system as against the concurrent model which allows a continuous often difficult to coordinate inter unit communication or handshake between ANN and the FIS unit (Abraham, 2002). The developed model is also a hybrid system since it implements the ANFIS hybrid model (Viharos & Kis, 2014; Abraham, 2005). Due to the pre-fuzzification of the inputs to the ANN by the FIS unit the familiar problem of gradient descent technique been trapped in local optima is significantly reduced since the error surface have been considerably reduced by the preprocessing. Without the fuzzy preprocessing the ANN processing of the fed inputs would have been completely a black box model with no clues to pattern of execution and making the iterative session to iterate endlessly.However, depending on the preprocessing of the fuzzy logic unit based on a priori gotten rule bases from expert knowledge as in this research from symptoms of contagious disease patients the fuzzy inferencing fine tunes the fed input to the ANN in a cooperative scheme. Thus the fuzzy inference uses a mapping of the qualitative relationship of the fuzzy inferred patients' symptoms rules (that are expected to be valid for the accurate condition of each patient) and the apparent increasing partial transparency of the ANN architecture to achieve nonlinear system identification by inferring the unknown condition or contagious state of the patient from the known or presented data samples in the form of symptoms. The ANN architecture then based on the input-output data mapping produces a set of weights that corresponds to the stored information of the sensitivity in the mapped data. The resulting ANFIS model is partially transparent providing a hybrid that harnesses the accuracy with complexity of the problem or nonlinear identification task.

3.3.4.1 Analysis of the Resulting Fuzzy-Neural Model

The Adaptive fuzzy-neural Expert system was analyzed by the used of the fuzzification algorithm (Section 3.3.1.1) to obtain the fuzzified symptom's values and the programming codes as domiciled from page 160 working on the generated fuzzy values of the GUI in Figure 3.15. Some result of analysis for the fuzzification unit can be found from Section 4.1.

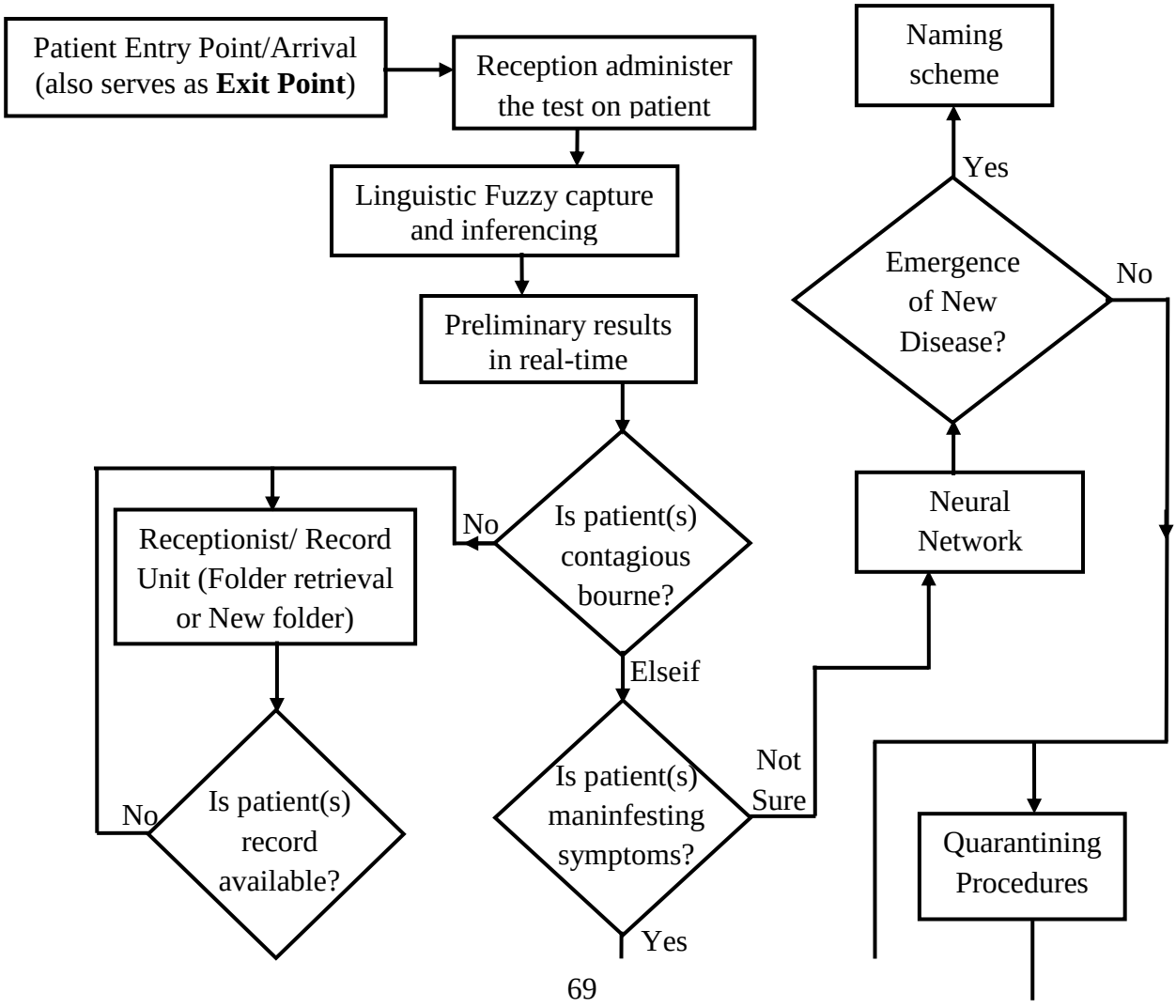
The Back Propagation Algorithm for the developed ANN in Figure 3.13 was used to implement the gradient descent optimization technique for the development of both the one hidden layer and the two hidden layer topologies. The programming code for ANN analysis is available from page 141 and the results of analysis are domiciled in Section 4.2 with the results of investigated topologies to achieve best sensitivity of the ANN in the Section 4.6 of this work. The resulting model/design is depicted in Figure 3.20.

3.3.4.2 Optimization of the Resulting Fuzzy-Neural Model

The resulting ANFIS model was optimized for network efficiency by using the three nodes single hidden layer topology structure for the considered range of input-output data pairs of contagious diseases sampled data. A training sample of a total of 1185 data samples was used in the training and validation of the network.

3.4 Proposed System

The proposed system allows for efficient mitigation of contagious disease spread by equipping healthcare providers with an insight into the nature, risk or state of the patient they are dealing with. The proposed system is depicted inFigure 3.19.



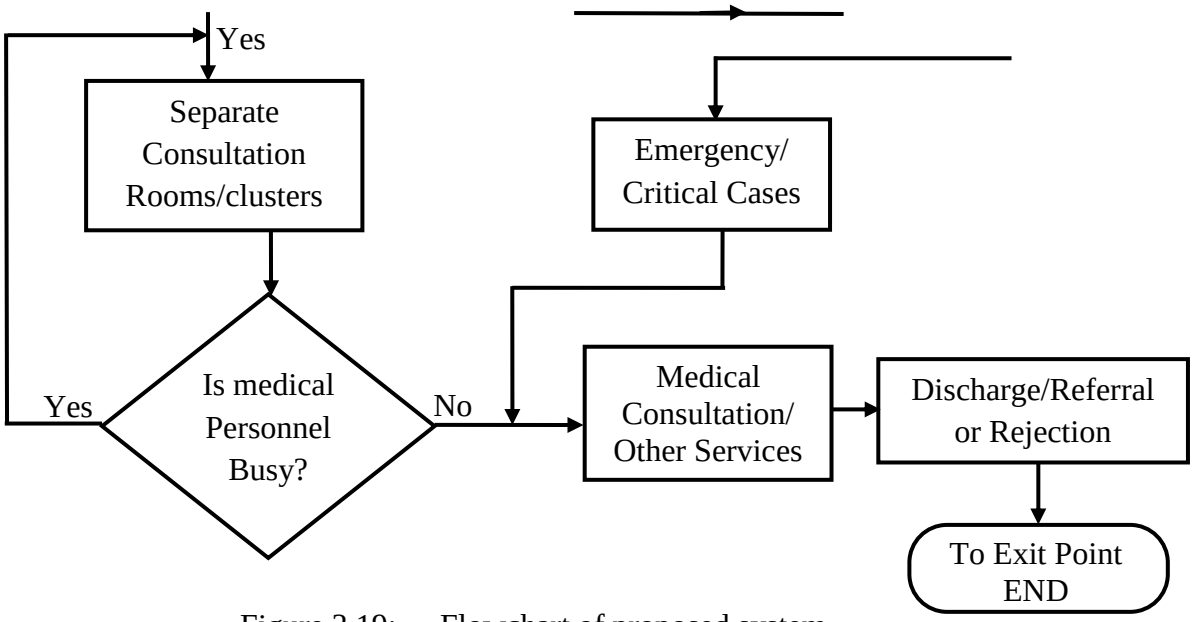


Figure 3.19: Flowchart of proposed system

In this analysis, the most important characteristic of ANNwhich is the ability to learn has been exploited by adapting it to new scenarios in medical diagnosis based on their fault tolerant pedigree and ability to deal with fuzzy data. Though training time of ANN and the need for a large training sample sets for efficient results have been its greatest disadvantagesthis researchutilized gradient descent optimizationmethod of error backpropagation over stochastic gradient descent to improve training time.Presently, methods which are predominantly manual with high error bounds are already in place for quick detection of contagious diseases such as the use of flowcharts and pictures(An example from the major extrances to wards in University of Lagos Teaching Hospital, LUTH is shown in Appendix I).

From the vast number of networks such as Hopfield (Hopfield, 1984, Van et al.,1996); Adaptive Resonance Theory(ART) (Carpenter & Grossberg, 1988); Radial Basis Function(RBF) (Haykin, 1994); Kohenen (Kohenen, 1989), Recurrent (Pham, 1994) networks, etc, the BPANN (Hassoun, 1995) was used for this research. This is owing to the BPANN's transparency and versatile application to data modeling, classification, forecasting, control, image compression and pattern recognition problems. In the BPANN, the First Network Sweep (FNS) or Iteration (FNI) of the preprocessed fuzzy inputs fed to the ANN produced the first output of the network (Equation 3.31 and 3.33and software implementation on page 141) and the corresponding error from the actual outputs. The second iteration (based on Equation 3.35 with implementation code on page 184) attempts toreduce the produced error across the network by considering the hidden layers error responsibilitiesobtained from (Equation 3.37 and 3.39) and repeats the process for each successive iteration until the

convergence criterion is attained.The criterion was based on the last observed iteration just before introduction of errors.

3.5 Expert System Model/Design

In the developedmodel shown below, fuzzy inferencing supplies pre-fuzzified inputs to ANN for further processing by sharing a globalknowledge base and memory while the k-NN providesan instancebaselearning for new sessions based on Euclidean distance to neighbours.

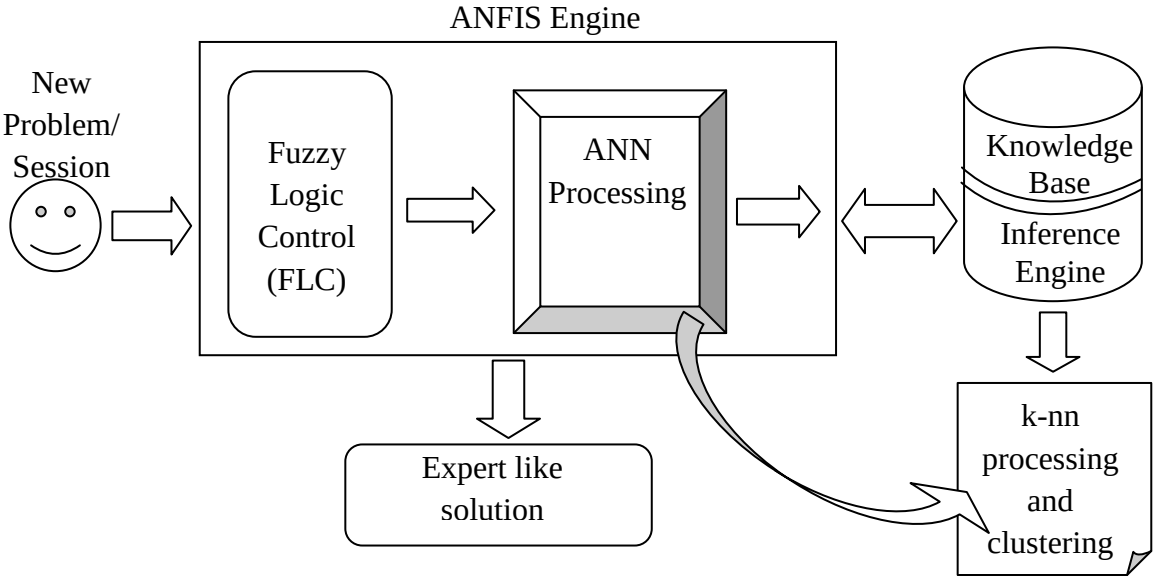


Figure 3.20: Expert System Model/Design

CHAPTER FOUR

IMPLEMENTATION, TESTING AND RESULTS

4.0 Preamble

Having outlined the various conceptual backgrounds to aid in the description and presentation of this work, the following sections described the designed Expert system implementation, testing and results obtained. Several tests where carried out to ensure the optimal functionality of the model and that the obtained results do not deviate from conceived or expected implementation during execution or service life. The results of implementation and testing of the Fuzzy Inferencing and the implemented Feedforward Error-Back Propagation Learning Algorithm to validate the model was also domiciled in this chapter.

4.1 Implementation

The resulting model from this work seek to obtain fuzzy inferred conclusions from fuzzy analysis and inferencing of inputted patients' symptoms that are occurring in real-time by trained healthcare providers using the designed GUI(Figure 3.12). Based on vague and

ambiguous reporting of patient's symptoms, fuzzy membership functions are generated and processed to obtain a degree of seriousness and appropriate actions which include quarantining could be initiated. The temperature sensor designed to take instantaneous reading and placed in from of administering healthcare provider provide further checks to avoid unnecessary alarms and fictitious actions based on the resulting nuances or noise in patient reporting of symptoms. The figures below shows results of processing of possible or instances of patients' reported symptoms with the seriousness of the ailment or disease, confidence level of the analysis and suggested line of action to be taken.

4.1.1 Results for Identification of Malaria

Based on inputted symptoms into the GUI of the model shown in the screenshot of the window in Figure 3.12, the following diagnosis were obtained by the proposed model. Whennot too severe symptoms that are susceptible to the presence of the malaria parasite were inputted into the model, the FIS returned the following result.The inputted symptoms are: slightly high body temperature, slightly high fever, headache, slight weakness, loss of appetite and feverishness; conditions that suggests the presence of malaria**(**Body and Health Home, 2015). The result of the model is as shown by the screenshot in Figure 4.1. It should be pointed out that the designed GUI for deployment of this software will require training of the healthcare provider to ensure ease of use, proper understanding of the workings, analysis and how to implement the suggestions from the model's analysis. Above all, the healthcare provider must be able to manage unnecessary apprehension, tension or "false-alarms" due to misuse of the expert system's advice.

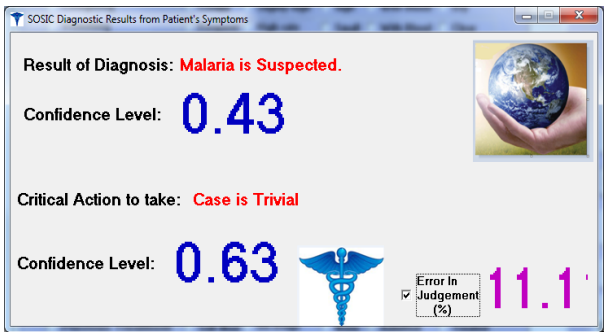


Figure 4.1: Screenshot shows results of diagnosis of a mild or trivial Malaria case.

In the screenshot based on the symptoms inputted, malaria was detected and the query result of diagnosis was returned as 'Malaria is Suspected' and the confidence level of the returned query is based on the values of the normalized fuzzy generated values which are low suggesting a trivial case. Though the presence of malaria present symptoms that are similar or at early stages of most contagious diseases such as ebolavirus disease(EVD) and lassa fever.(Bernhard Nocht Institute for Tropical Medicine, 2014), the model is sure that the

diagnosed case is a trivial case when compared to contagious diseases and returns a high confidence level of 0.63 for a malaria diagnosis.

As a further check to ensure that errors are not propagated by wrong analysis, the error in judgment (EIJ) is computed. The EIJ gives an estimate of the percentage of the number of symptoms inputted and participating in the model's analysis to the user or healthcare personnel. In this case 11.1% shows that only few of the symptoms are present. However, the EIJ of 14.2% in the diagnosis below shows that more symptoms where entered to obtain higher confidence levels with the model suggesting that patient shouldn't be admitted. This assessment or inference is in line with the overall aim of detecting contagious cases as they make their first appearance and is not the final deciding factor as the patient will still have to see the doctor or healthcare personnel on duty.The value of EIJ can converge to a constant value if all the symptoms needed to diagnose a particular disease have been inputted while the model will only respond to refining computed confidence level for any further adjustments.

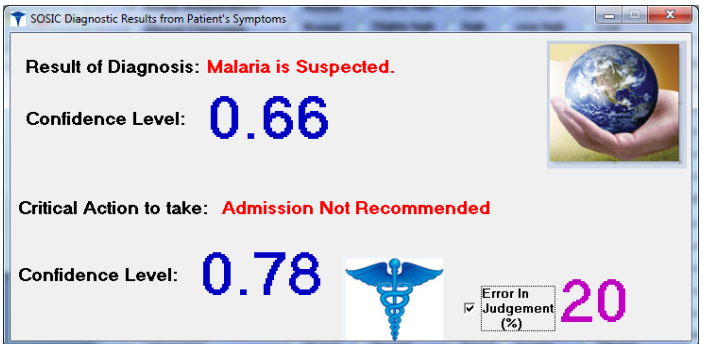


Figure 4.2: Screenshot showing confirmation of Malaria case.

4.1.2 Results for Identification of Dengue fever and Yellow fever

The sensitivity of the model to identify special features and closely related symptoms was used to identify Dengue fever and Yellow fever of which both exhibits mostly symptoms of malaria but different from malaria with high presence of nausea for Dengue fever and yellow skin for Yellow fever.

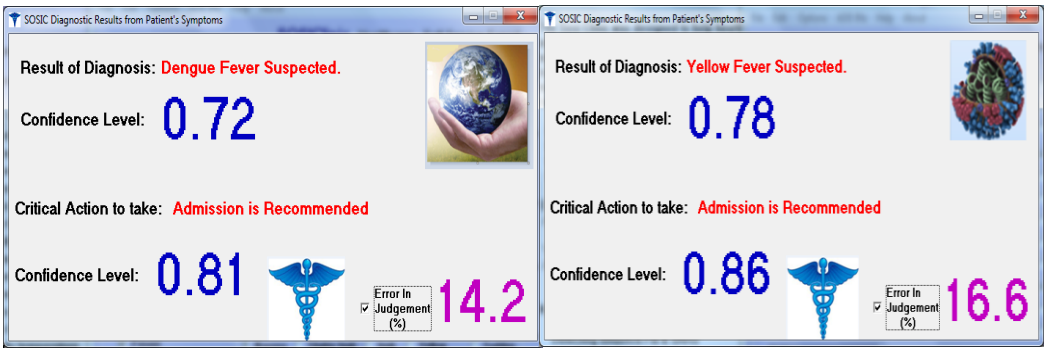


Figure 4.3: Screenshots shows results of diagnosis of Dengue and Yellow Fever.

Dependingon presentedsymptoms by patient under diagnosis, obtained normalized fuzzy membership functions and computed confidence levels which themselves are fuzzy values

between 0 and 1 will be generated. The above analysis can also be made to return a patient with typhoid fever based on the inputted symptoms which are normally vague and ambiguous.

4.1.3 Results for Identification of Cholera

The detection of cholera in a patient reporting symptoms suggestive of cholera is as depicted in the Figure 4.4. Again two instances of different fuzzy symptoms where used to obtain the results with different suggestions to take and different computed EIJ values. When symptoms such as mild diarrhea, high fluid loss, slightly high blood pressure, constant coughing, mild gastrointestinal pain and mild weakness were selected, a low 0.36 confidence level was obtained and with further refinement a 0.77 confidence level was attained when 'severe' was selected as against 'mild' with the same symptoms; thus the weight on the adjectives triggers the fuzziness in the symptoms hence implementing a linguistic fuzzy inference system.

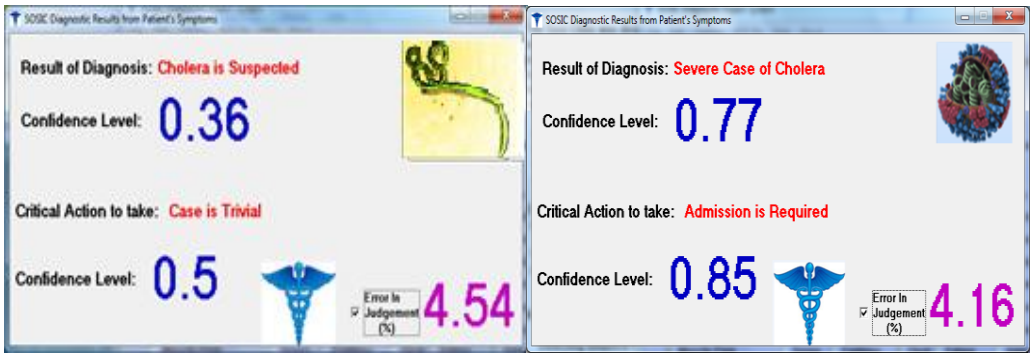


Figure 4.4: Screenshots shows results of diagnosis of cholera and severe cholera.

This model uses or exploits the thin line existing between severe diseases signatures to draw its conclusion and suggests a particular line of action. In theFigure 4.4, based on features of patient's symptoms provided two classification of the same ailment was diagnosis but the confidence level and the computed EIJ was used to separate the cases into trivial and serious cases.

4.1.4 Results for Identification of EVD

Figure 4.5 shows the results obtained when symptoms suggestive of the presence of EVD or Ebola hemorrhagic fever (EHF) was entered into the model from the GUI. When the model was presented with symptoms suggestive of Malaria as above and mild sore throat, infrequent vomiting, red eyes, mild diarrhea, slight muscle pain, mild gastrointestinal pain, mild skin rashes, mild external and internal bleeding the figures below where obtained suggesting either EVD or Lassa fever. Due to the contagious nature of the EHF and Lassa fever, alert levels in the hospital or clinic where the software is be localized and where the

patient is being diagnosed will have to be placed on red alert of a possible index case when results of diagnosis return ‘EVD is suspected!’

Figure 4.5: Screenshots shows results of diagnosis of EVD.

As shown in Figure 4.5the value of the confidence level of 0.46 is indicative of the model not sure of the output of the diagnosis either due to the symptoms not enough or due to the fact that the contagious diseases classes have very thin signatures pattern differences with interwoven properties. Thus the returned query is split between the early signs of EVD and the presence of Lassa Hemorrhagic Fever. This is in line with literature on the possible similarities in occurring symptoms of the two disease(Bray & Chertow, 2014; Centers for Disease Control, 2015; Planet Health, 2014). However, with fine-tuning of inputted symptoms with adjustment or inclusion of more critical symptoms using ‘severe’ to replace the adjective ‘mild’ in the symptoms, a better confidence level is attained and the model is now sure of its diagnosis and now recommends admission of patient. Of course, admission is based on the assumption that the hospital or clinic is a CDC or DCU. Others can initiate processes to transfer such patients for admission into such units or containment facilities elsewhere. Further refinement in the inputted symptoms such as ‘slight’ been replaced by ‘high’ or ‘chronic’ a better or higher judgment of the state of the patient under investigation will be obtained as shown below. This is because at higher values of membership fuzzy sets with the fact that the model trust inputs from the healthcare provider, the model is able to compute better and surer values of confidence level. Also for further refinement to be attained the prior location option of the patient under diagnosis which queries the travel history of the patient must be stated to indicate if the patient recently travel to/from an infected area or if the patient is a locale which could suggestan index case. If a patient answers in the affirmative, it suggests that the patient may be a carrier of such contagious virus (Centers for Disease Control, 2015; CNN, 2014). The EIJ returns a constant valuein the state that most (if not all) symptoms necessary to classify the disease had been entered during a diagnostic session. It is important to reiterate at this point that this result or any other result obtained in this work is not intended nor recommended as a substitute for expert medical advice, diagnosis, or

treatment. It is an optional suggestion based on analysis obtained from the application of mathematical principles obtained from Machine Learning through AI and cannot replace the advice or suggestion of a trained physician or other qualified healthcare professional regarding any medical state of a person.

Figure 4.6: Screenshot shows results of diagnosis of severe EVD.

4.1.5 Results for Identification of LHF and Marburg

The presence of Lassa Hemorrhagic Fever and/or MARBURG, in patients based on inputted patient symptoms can be shown in Figure 4.7. LHF have infected people in Nigeria while Marburg Virus Disease (MVD) formerly known as Marburg Hemorrhagic Fever(MHF) was first diagnosed in Marburg, Germany when laboratory workers developed symptoms while working on the development of a polio vaccine and experimenting on some monkeys from Uganda (MedicineNet., 2015). The LHF and MVD are relations of EVD since they belong to the same viral infection family and exhibit sighs that are consistently similar but with very thin unique differences which was exploited in the characterization of the diseases in the developed model.When the model was presented with symptoms consistent with Malaria, EVD, Internal pain, Mucosal, Chest pain, swellings and seizures LHF was obtained and when the model was presented with the following symptoms: those for Malaria, for EVD, rashes, Back Chest pain, flu-like symptoms MVD was diagnosed. The Figure 4.7 shows the screenshots of the various diagnoses.

Figure 4.7: Screenshot shows results of diagnosis of LHF and MVD.

4.1.6 Results for Identification of Acute Respiratory Infection

When the symptoms presented to the model was consisting of severe muscle, chest pain, flu-like symptoms, common cold, fever and dangerous coughing; Acute Respiratory Infection (ARI) or Influenza Flu was returned as result of diagnosis; as shown in Figure 4.8.

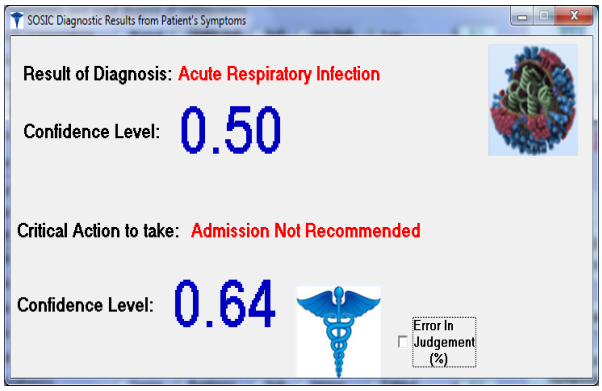


Figure 4.8: Screenshot shows results of diagnosis of Influenza flu.

4.2 Implementation of the Neural Network

The developed ANN was presented with the results of the fuzzy diagnoses for each contagious disease case as outputs such that the ANN was able to revalidate the FIS used in defining or providing the diagnosis. The network was also presented with inputs representing various combinations of symptoms that can manifest in a patient being diagnosed and the new symptoms that were not captured in the fuzzy inference system, i.e the new symptoms as discovered by the administering healthcare provider. The source of the data was from a randomization code that generates likely but fictitious values within the normalized range of 0.1 to 0.9 fuzzy values. The presented inputs were then used to train the network for convergence. After training the network was presented with new samples obtained from likely but fictitiously randomized values generated by software codes for validation and testing the ANN model to ascertain the generalization and sensitivity for new cases.Having obtained a working model, the network was then presented with real data from recognized hospitals.The different or unbiased data must test or verify the quality of the designed neural network. This is because validating with the same testing samples will make the network to obviously perform well on them since the network was optimized on those same samples. This practice which was avoided in this work doesn't give any indication as to how well the network will be able to classify new data inputs that weren't in the training set and that will be presented to the network.The relationship of the training set and the validation/testing set as obtained from implementation and testing of the designed model is as shown in the Figure 4.9.

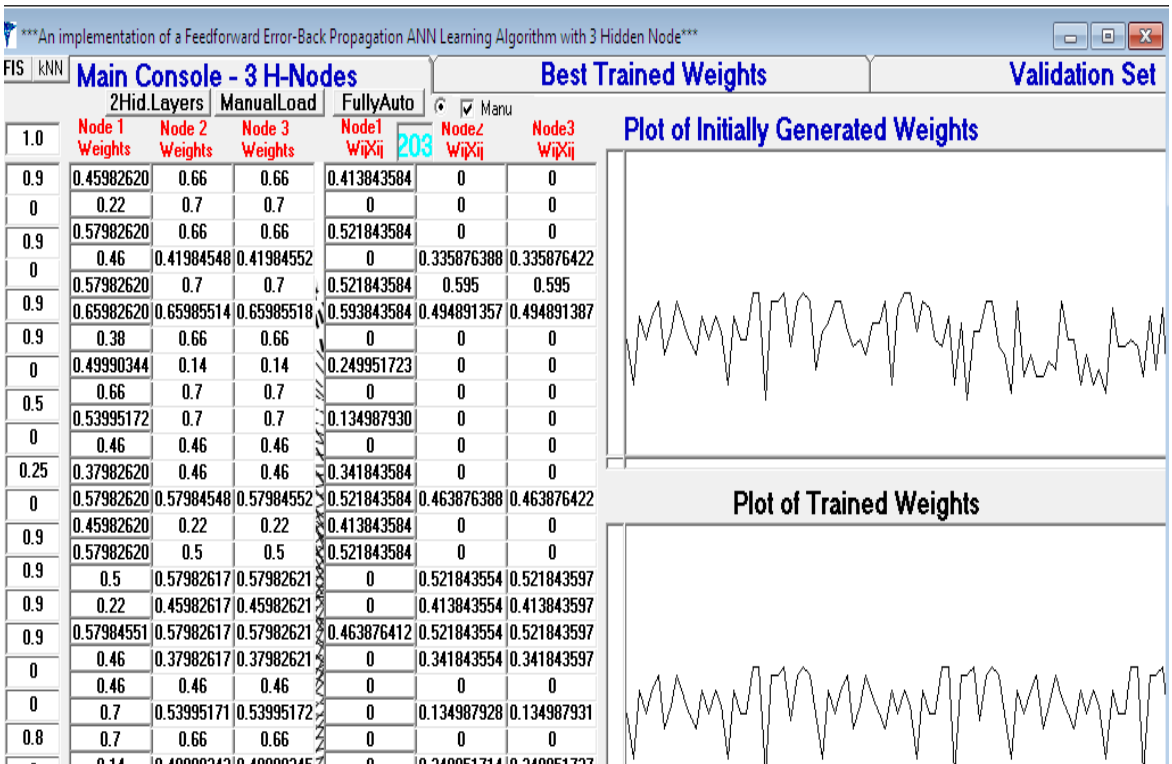
Figure 4.9: Relationship of the training set and validation/testing set.

4.3 Neural Network Implementation Results

For the purpose of validation and testing of the ANN training the following real-data was utilized. Data (Appendix C) obtained from the Institute of Lassa Fever Research and Control (ILFRC) at Irrua Specialist Teaching Hospital (ISTH) in Edo State and data samples (Appendix D) from Federal Teaching Hospital, Abakaliki (FETHA) in Ebonyi State was used to validate the Lassa fever diagnosis. The model's validation for EVD diagnosis was analyzed from diagnostic data (Appendix E) obtained from the WHO Ebola Response Team. (2014)

4.3.1 Lassa Fever Diagnosis Validation

In other to validate the ANN processing of Lassa fever signs and symptoms real data from the ILFRCin Edo Statewhere used. The output from the fuzzy inferencing was used as inputs to the ANN and also provided the guided output for the ANN to converge to as shown with arrows in Figure 4.10.



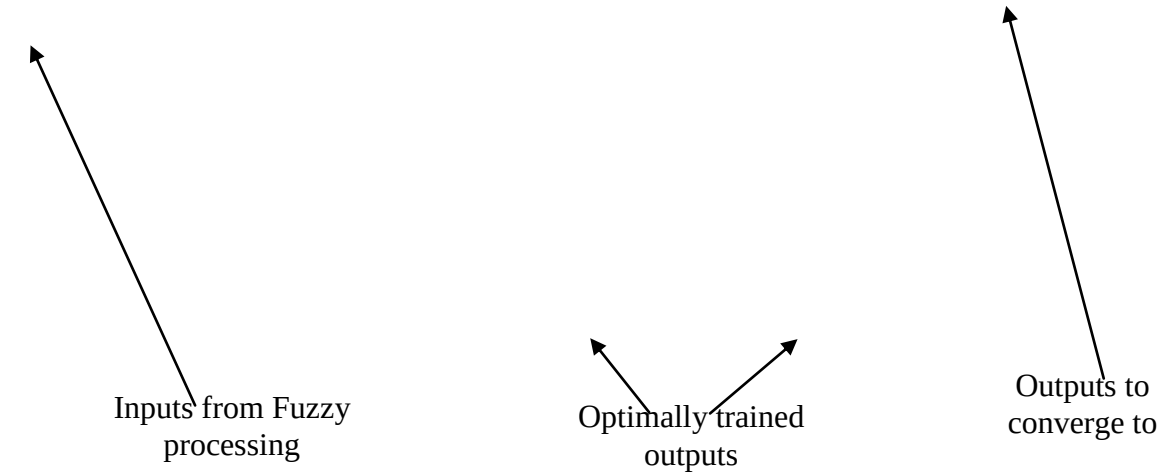


Figure 4.10: A Screenshot of the designed ANN response to Lassa fever symptoms

Further testing of the ANN with new data bearing new symptoms that are consistent with that of Lassa fever characteristics, the following processing in Figure 4.11 was obtained by the network.

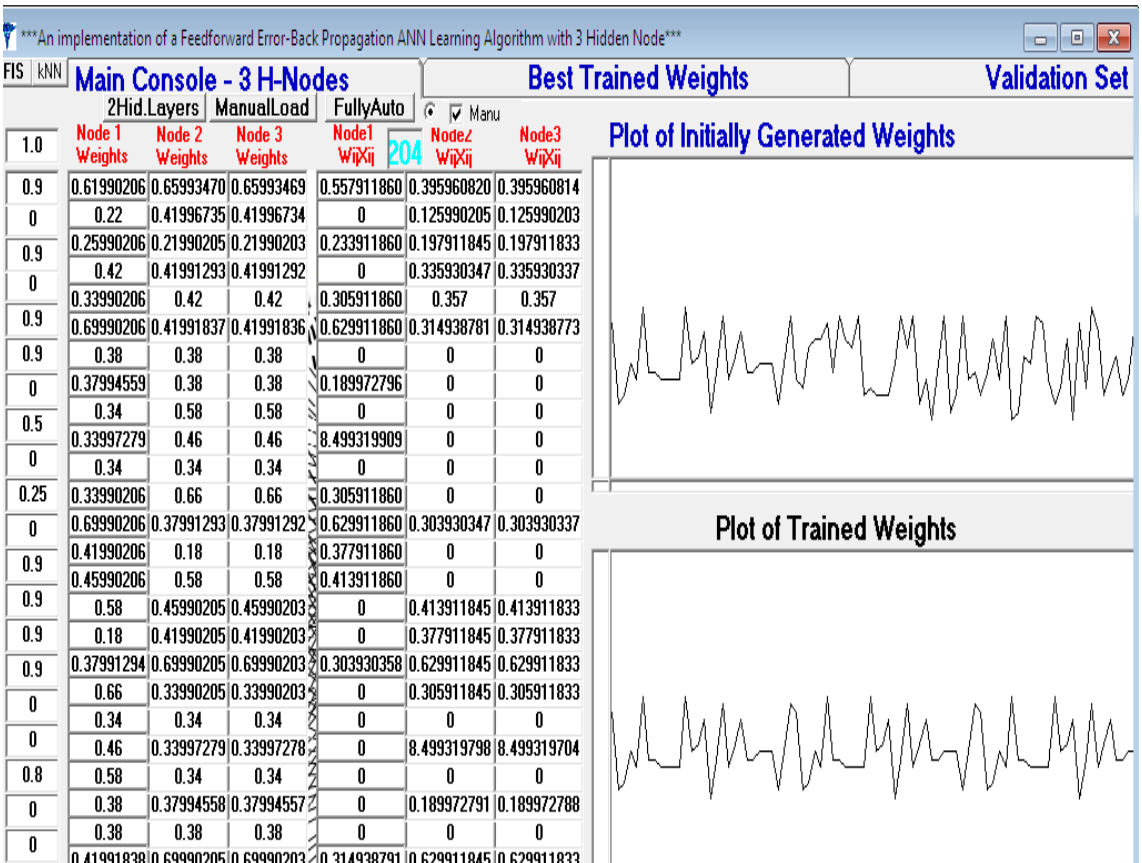


Figure 4.11: A Screenshot of the designed ANN response to more data or new Lassa fever symptoms.

When testing was done with data from FETHA in Ebonyi state, Nigeria, the network also revealed consistency of the results obtained above. A sample of the similarly obtained result is depicted in Appendix F.

The ANN iterations toll on the computer system that was used for this analysis recorded heavy activity and processing in its CPU usage history and physical memory usage history. This was observed in the Windows Task Manager when simulation was carried out on a 2.0GHz Intel(R) Pentium(R) M Processor with 2GB installed RAM. Figure 4.12 shows screenshots of the 32-bit system'sCPU activity reaching as high as 93% during program execution.

Figure 4.12: Screenshot of the Windows Task Manager for the successive iteration's toll on computer resources.

4.3.2 Validation of EVD Diagnosis

The EVD diagnosis validation was performed using real data from the WHO Ebola Response Team collation of EVD in Guinea, Liberia, Nigeria, and Sierra Leoneand the following results was obtained by the designed ANN and is shown inFigure 4.13.The accompanied fuzzy data and result of processing of the FIS are depicted in Appendix G.

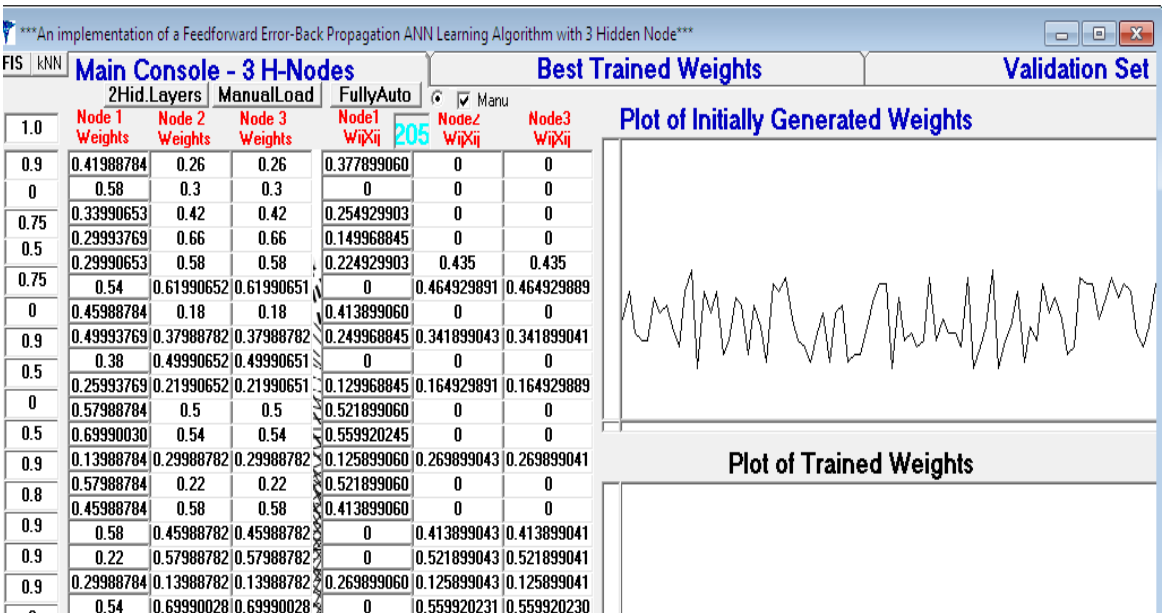


Figure 4.13: A Screenshot of the designed ANN response to more data or new EVD symptoms.

The model after validation tests have been performed at a general epoch of 620 iterationswas found to be generalizable and sensible to presented data for training and new data samples of symptoms for Influenza flu, Lassa fever, Ebola virus disease and by extrapolation Marburg Virus Disease(MVD).However, it must be pointed out that the choice of a 620 epochs as a termination criterion is chosen only on a purely heuristic manner and based on the data being processed with the weights generated by the network. For example, it was observed that convergence can be achieved before or still continue even after the 620th epoch.

4.4 Presenting New Data to the Trained ANN

From the developed ANN algorithm and its implementation, it was observed that when data that was dissimilar and inconsistent with the data used in training the ANN was presented with the optimal weights realized from a particular training session, the validation iteration failed to converge to optimality giving very disparate outputs. The figures below show this phenomenon when data consistent for EVD diagnosis was compared or validated with dissimilar data.

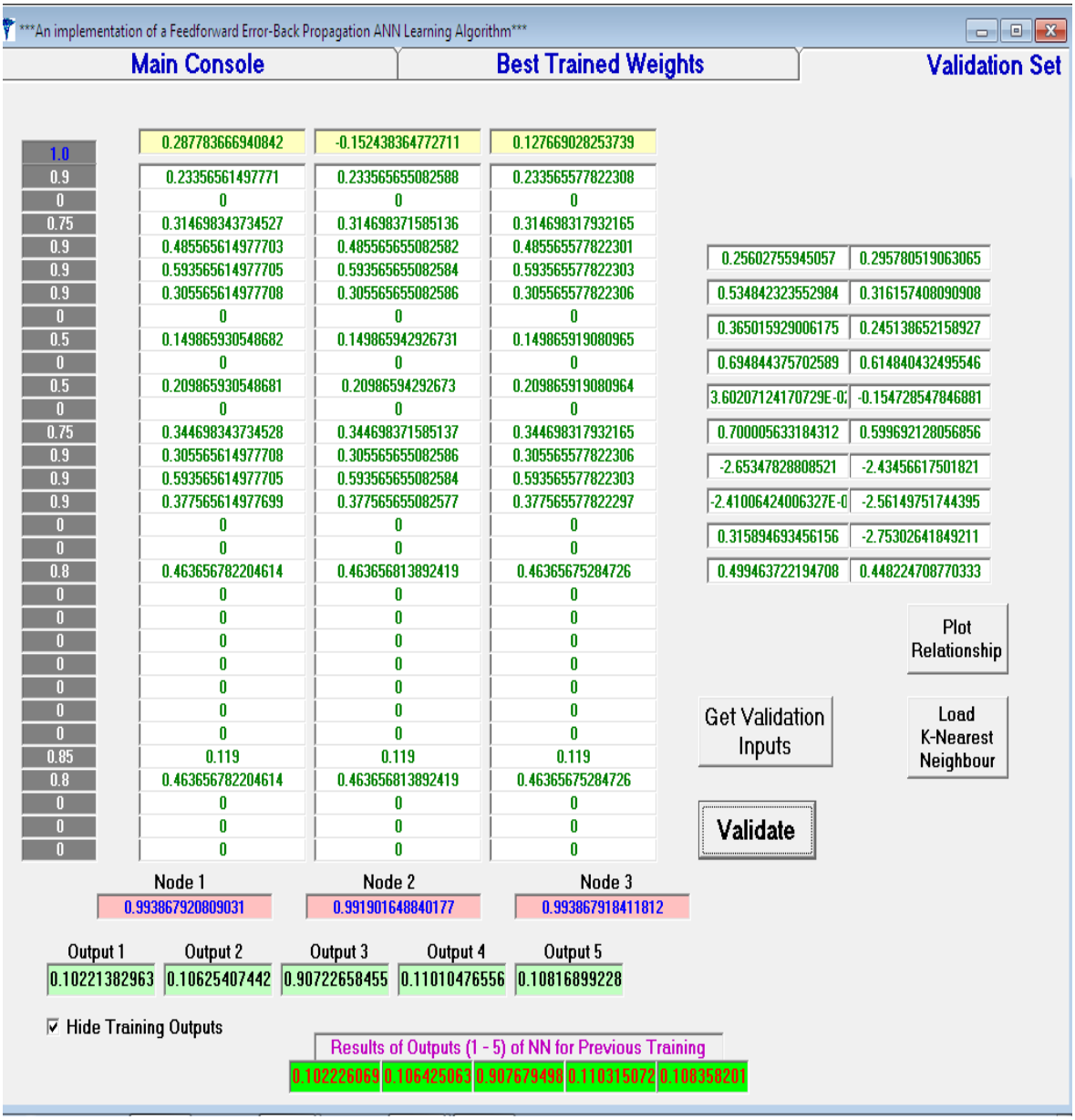


Figure 4.14: A Screenshot of a trained session of ANN when presented with data consistent with a contagious disease's symptoms (EVD).

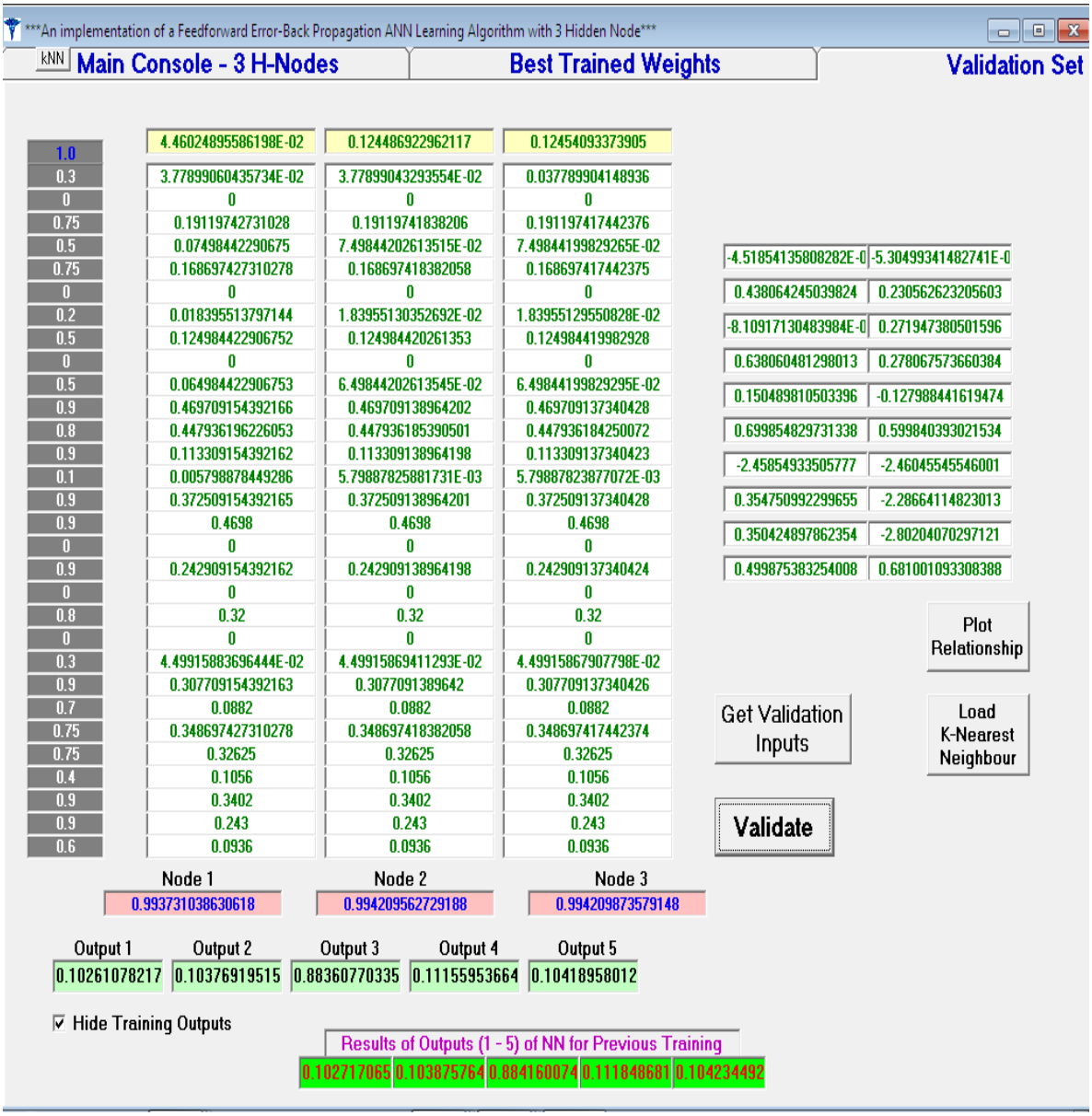


Figure 4.15: A Screenshot of a trained session of ANN when presented with inconsistent and unrelated data.

4.5 Processing of the k-Nearest Neighbors

In other to obtain a consistent method of diagnosis especially for reference purposes, each actual output of the neural network results of processing was stored in the resident computer system such that during execution of the k-nearest neighbor algorithm the various outputted records were retrieved and used to further categorize a new diagnosis. The algorithm involves mapping the present diagnosis in an Euclidean distance space to visualize its difference in terms of distance to other already categorized diagnosis (Sharma, 2007).

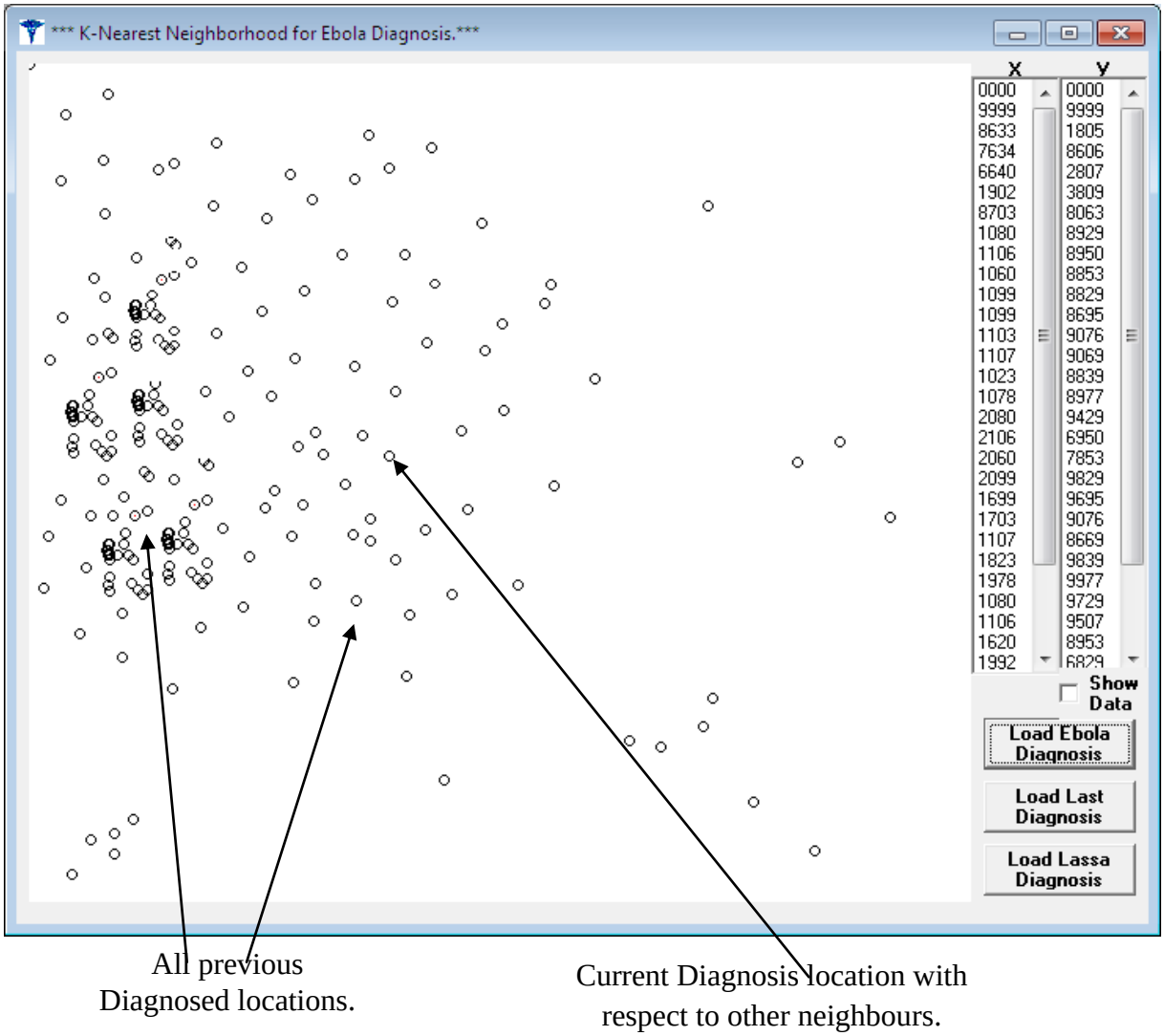


Figure 4.16: A Screenshot of the k-Nearest Neighbors distribution of the results of diagnosis of patient inputted symptoms.

The k-Nearest Neighbors algorithm gives a visual validation of a current diagnosis with respect to all the other results of diagnosed patients after processing is obtained from the ANN. From above, the Euclidean distance between the current analysis and the previous ones can be easily obtained by noting that the new diagnosis falls within the cluster of majority of the confirmed EVD cases and not within the false alarm region.

4.6 Performance Metrics for various Topologies of ANN

To obtain the best topologyas described in (Yegnanarayana, 1994, 1999), (Rad, Khadivi & Hasler, 2010) and (Simpson, 1992) for the implementing neural network with respect to the NHN in a hidden layer and the NHLs several topologies were investigated. Amongst them include the one hidden node, two hidden nodes, three hidden nodes and five

hidden nodes'different responses to thirty pre-fuzzified and fed inputs to their input unitsin a single hidden layer topology (SHLT), (See figures in Appendices I, J and K). Also examined was the multiple hidden layer topology (MHLT), (See Figure 5.11 in Appendices L).The various ANNs which were fed with the same inputs in other to compare their convergence characteristics, sensitivity, execution timeand generalization over training and validation data obtained the results analyzed below.It should be noted that the fed inputs into the SHLT or MHLT corresponds to a single patient manifested symptoms' output from the fuzzy pre-processing of that disease, i.e.either Lassa fever, EVD, Influenza flu or Marburg disease.

4.6.1 Performance Metrics on Lassa fever Data

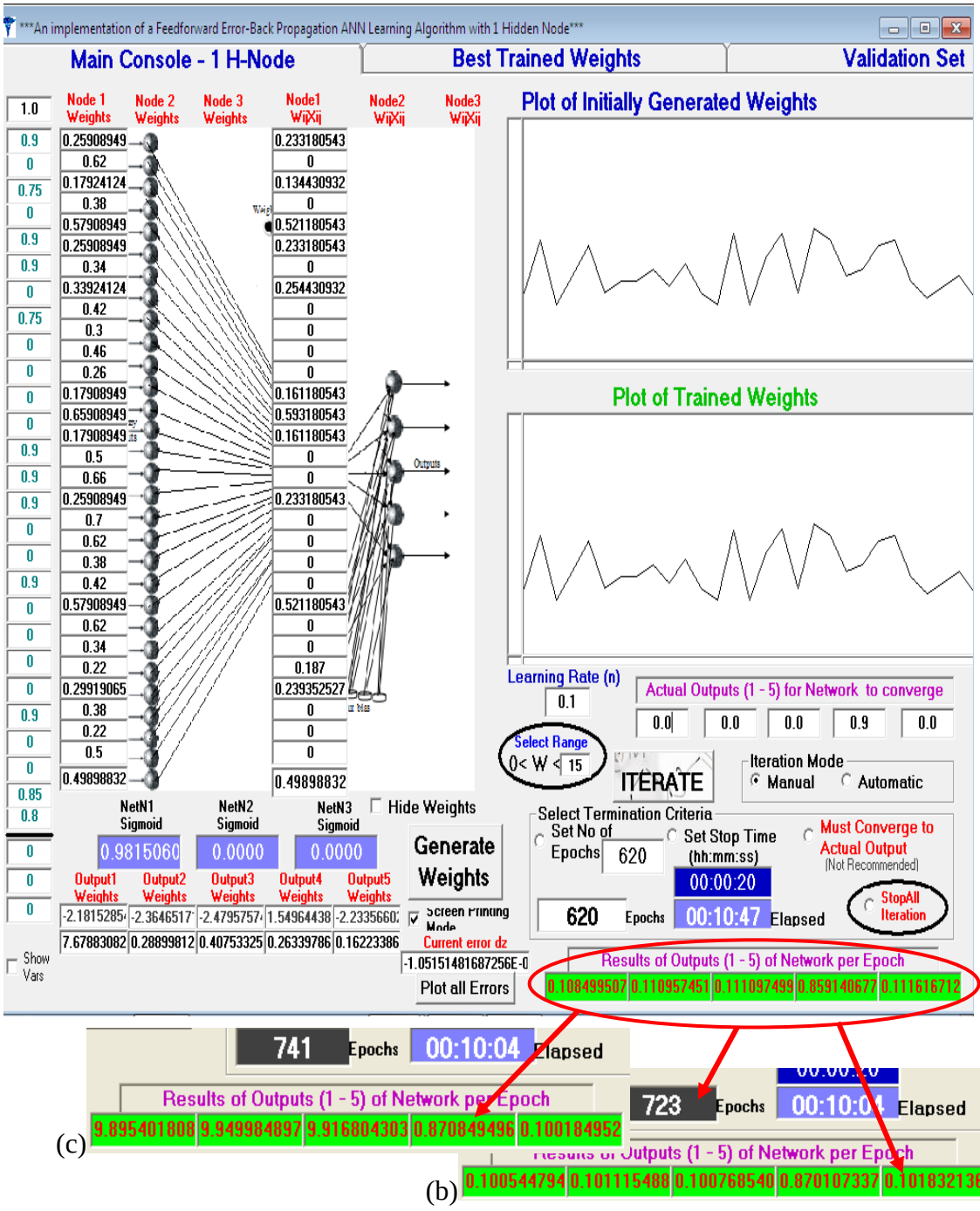
The various designed ANN topologies where investigated with both the 20 cases (10 confirmed and 10 suspected) of validation data on Lassa fever obtained from Federal Teaching Hospital, Abakaliki in Ebonyi State, Southeast Nigeria and 14 cases (7 confirmed and 7 probable) from the Institute of Lassa Fever Research and Control at Irrua Specialist Teaching Hospital in Edo State, Nigeria. Due to the specific manifestation of symptoms of patients with the disease and their locality or contact with probable infested victims or zoonotic agents, ANN processing of already fuzzified inputs (from fuzzy processing of suspected or confirmed Lassa fever patient's symptoms) yielded the following.

4.6.1.1 Performance Metrics for One Hidden Node

The performance metrics for one node in the hidden layerof a fully connected ANN with thirty inputs and five output units showed that convergence was attained around the 620th epoch but further fine-tuning resulted in significant errors.As shown in Figure 4.17(a), the result of the ANN processing using a single node in the hidden layer when trained with Lassa fever desired-output recognition data (0.0, 0.0, 0.0, 0.9, 0.0) resulted in significant convergence at the 620th epoch. Though convergence had started even around the 200th epoch, continued iteration was allowed to ensure the closest marched to the desired-output as stated above. Further iterations beyond the 620th epoch produced more refinement towards the output data as seen in (b) of Figure 4.17 (at 723rd epoch) but eventually resulted in introduction of significant errors at the 741st epoch (c). These errors continue to increase or propagate within the outputted actual-outputs with further iterations thus making that ANN session null and void.The goal therefore was to identify the most tolerable or significant convergence during successive iterations just before errors are generated by gradient descent optimization and then terminating all iterations at that cycle.

The insensitivity of the one node hidden layer can be seen from (b) in which the ANN continued to produce desired output far beyond the 620th epoch and becoming unstable in the 741st epoch at (c). Clearly, utilizing the one node architecture hasresulted in significant time consumption and reduced sensitivity to new data.

(a)



(c)

(b)

Figure 4.17: Screenshots of the results of iteration of a one-node hidden layer ANN (a) 620th Epoch (b) 723rd Epoch and (c) 741st Epoch

4.6.1.2 Performance Metrics for Two Hidden Nodes

The performance metrics for atwo-nodehidden layer of a fully connected Feedforward Error-Back Propagation Learning ANN with thirty inputs and five output units (using the Lassa fever desired-output recognition data) showed that adequate convergence was

attained around the 620th epoch Figure 4.18(a).Further fine-tuning resulted in significant errors at around 680th epoch(b). An appreciable sensitivity was observed when these results werecompared with the one-node hidden layerarchitecture's result. As shown in Figure 4.18, the ANN's processing using two hidden nodes and training with Lassa fever output recognition data (0.0, 0.0, 0.0, 0.9, 0.0) resulted in significant convergence at the 620th epoch even though convergence had started around the 200th epoch. Further iterations beyond the 620th epoch produced more refinement towards the output data but eventually resulted in introduction of significant errors. These errors just as noticed in one node ANN continue to increase within the outputs with further iterations. Time for the session was increased to 00:10:41 over the one-node hidden layer's 00:10:04. Thus increased sensitivity required more time for network convergence; however this time dependence is not linear as was observed in the three hidden nodes with 00:10:36 as shown below in Figure 4.19.

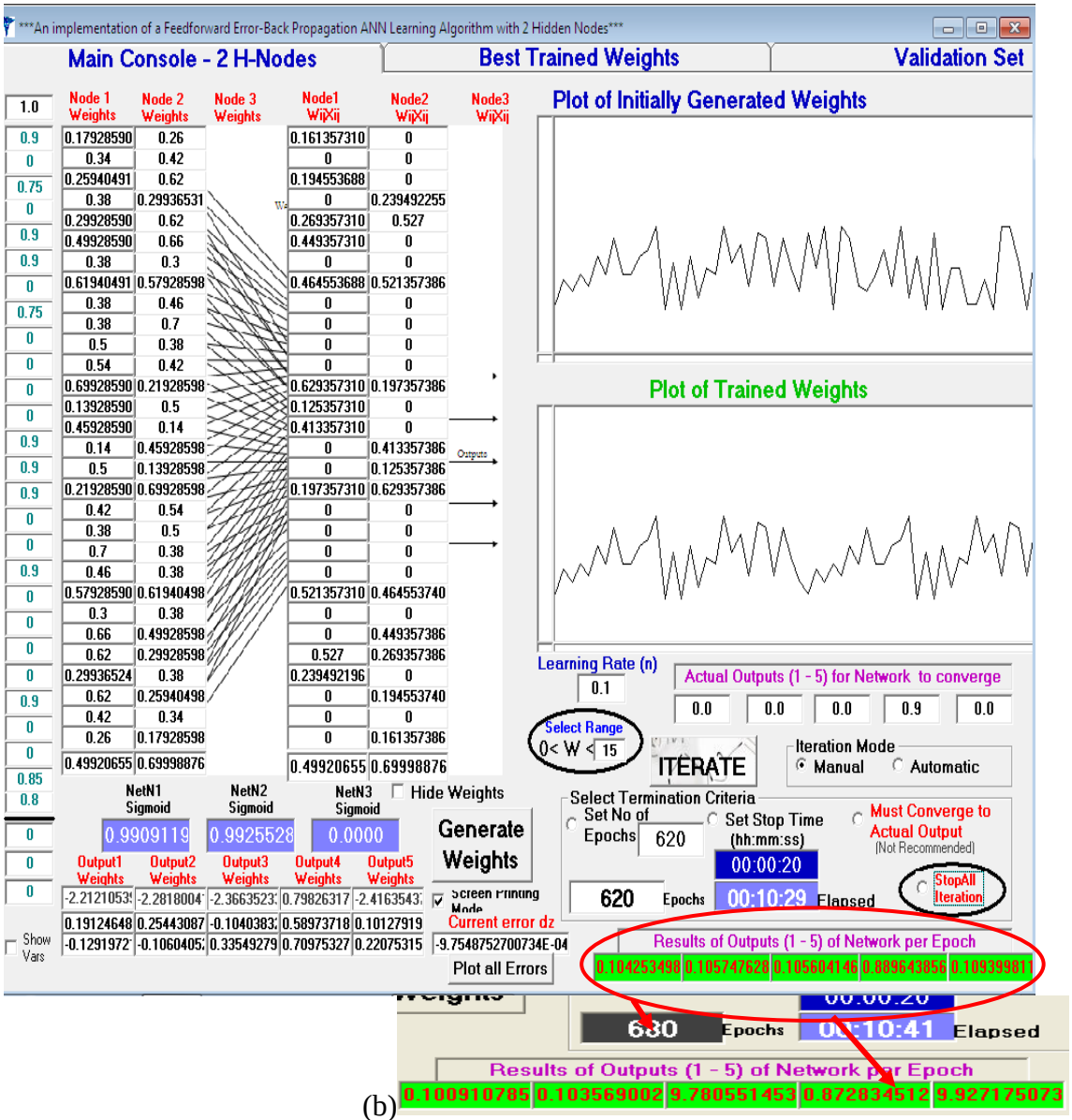
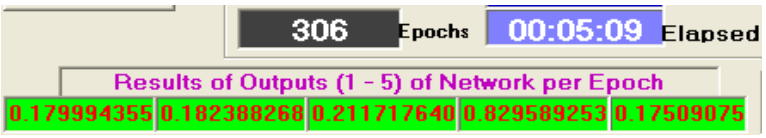


Figure 4.18: Screenshots of the results of iteration of a two-node hidden layer ANN (a) 620th Epoch and (b) 680th Epoch

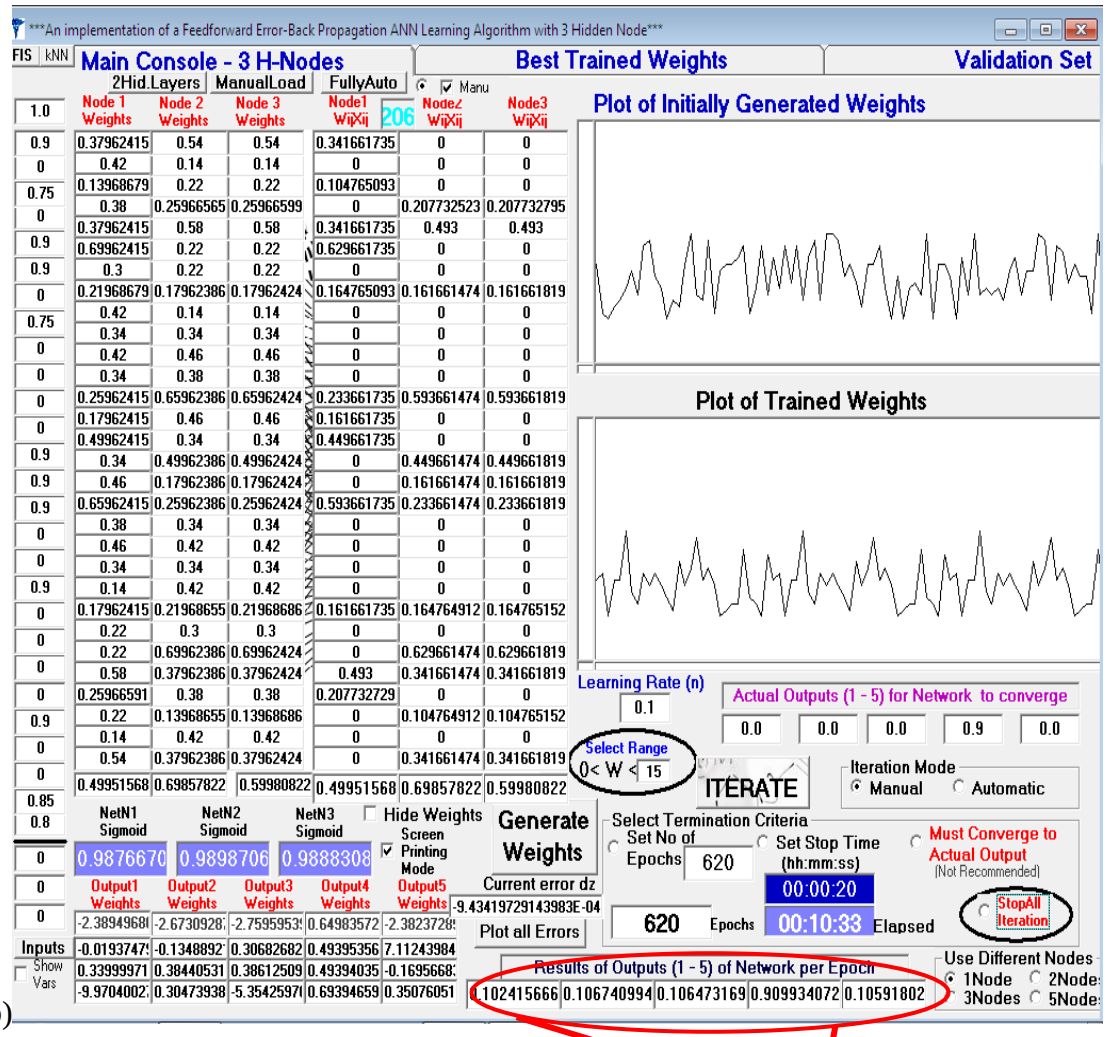
4.6.1.3 Performance Metrics for Three Hidden Nodes

In the performance metrics for a three-node hidden layer implementation of the Feedforward Error-Back Propagation Learning ANN with thirty inputs and five output units (using the Lassa fever desired-output recognition data) convergence was already occurring at about five minutes into the session and around 306th epoch (Figure 4.19(a)). Adequate convergence was againattained around the 620th epoch (Figure 4.19(b)). Further fine-tuning resulted in significant errors at around 663rd epoch (c). An appreciable sensitivity was observed when these results were comparedwith the one-node, two-node and five-node (below) sessions' results since the introduction of errors occurred at a 663rd epoch and session time was at an optimum of 00:10:36.

(a)



(b)



(c)

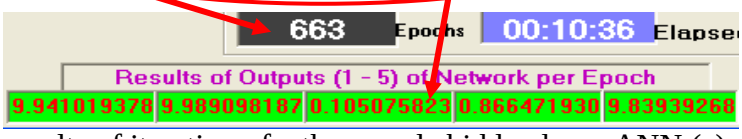


Figure 4.19: Screenshots of the results of iteration of a three-node hidden layer ANN (a) 306th Epoch (b) 620th Epoch and (c) 663rd Epoch

Performance of the three-node ANN on validation data showed good generalization and the result when fed with completely unconnected data showed significant errors at the 620th epoch; thus was chosen for the expert system based on these salient features and sensitivity.

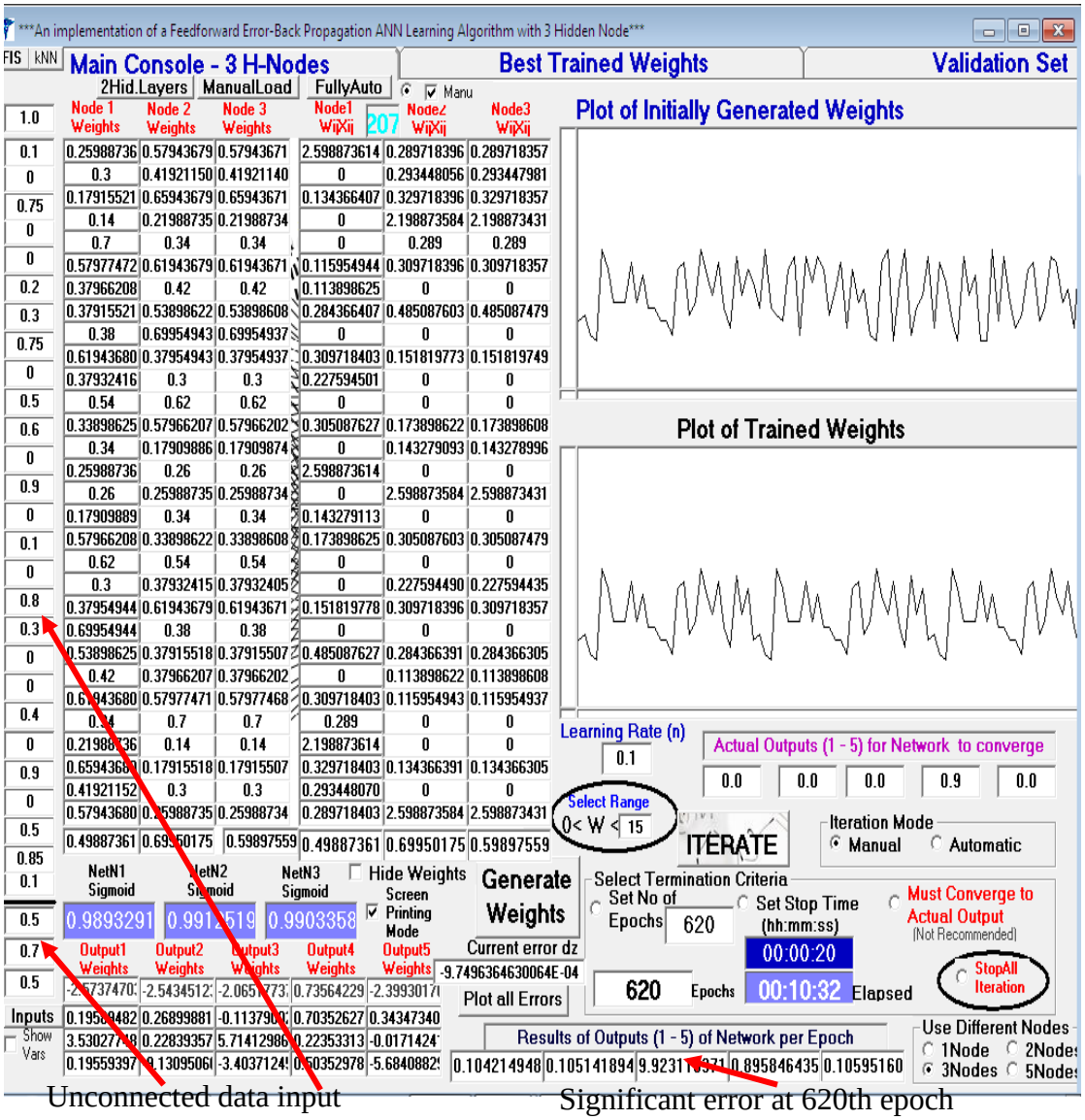
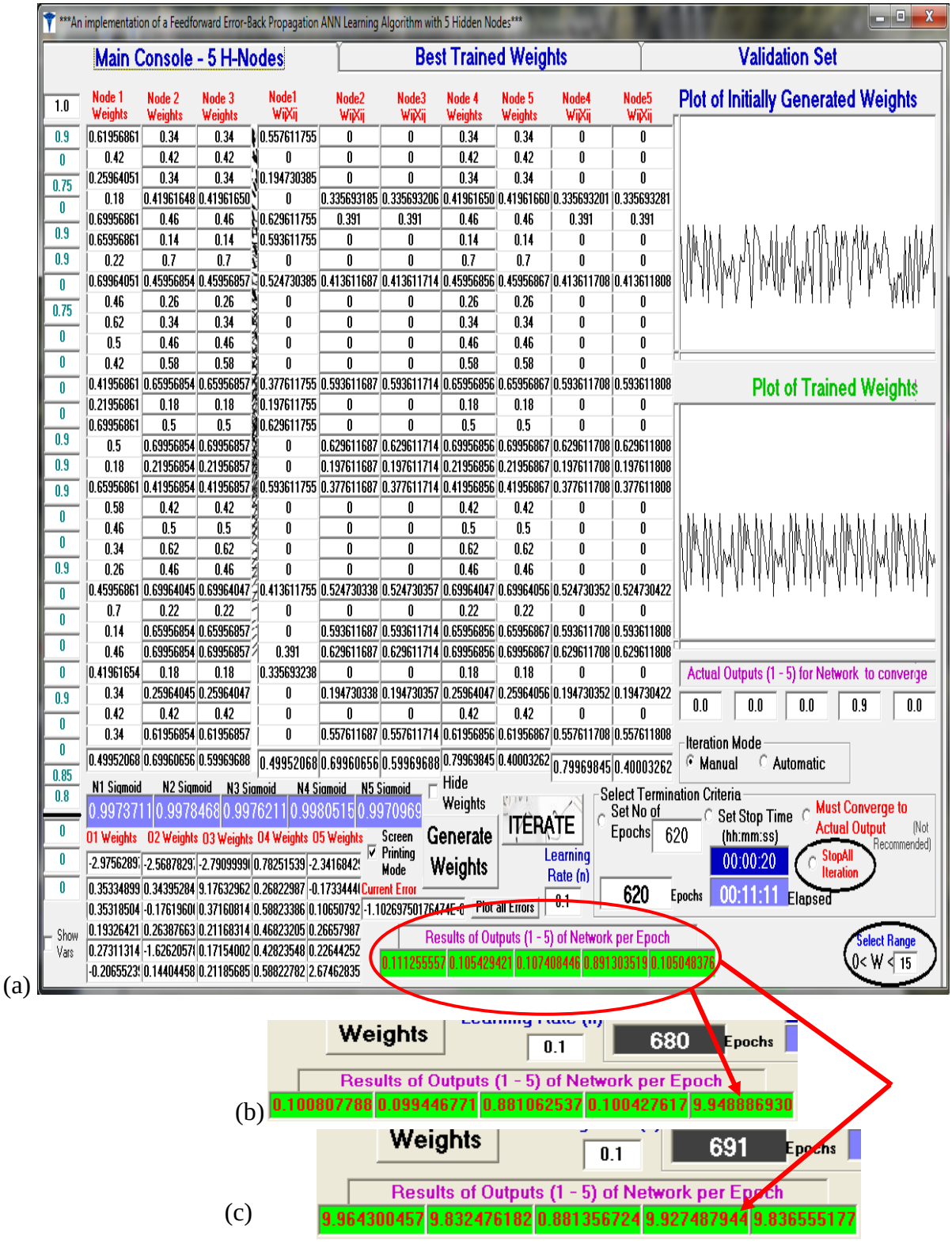


Figure 4.20: Screenshots of the results of iteration of a three-node hidden layer ANN processing when fed with unconnected Lassa fever data.

4.6.1.4 Performance Metrics for Five Hidden Nodes

In the five-node topology, it was observed that execution time for up to the 620th benchmark iteration of the ANN was at 00:11:09 (Figure 4.21) and the sensitivity was appreciable when compared to other nodal topologies as shown in (b) and (c) of Figure 4.21. However, generalization of the five-node ANN when presented with new data was observed to be unpredictable around and beyond the 620th epoch. The result of new unconnected data fed to the ANN processing inputs was also found to be unstable leading to convergence before or around the 620th epoch as shown in Figure 4.22. This proves that the ANN has begun memorizing the input data samples.



(a)

(b)

(c)

Figure 4.21: Screenshots of the results of iteration of a five-node hidden layer ANN (a) 620th Epoch (b) 680th Epoch and (c) 691st Epoch

The response of the five-node hidden layer ANN to new inputs suggests memorization of the network as observed at the 620th epoch. This also suggeststhat if the number of hidden layers continue to be increased, then by extrapolation the network will continue to memorize training data instead of been able to generalize to unseen data.

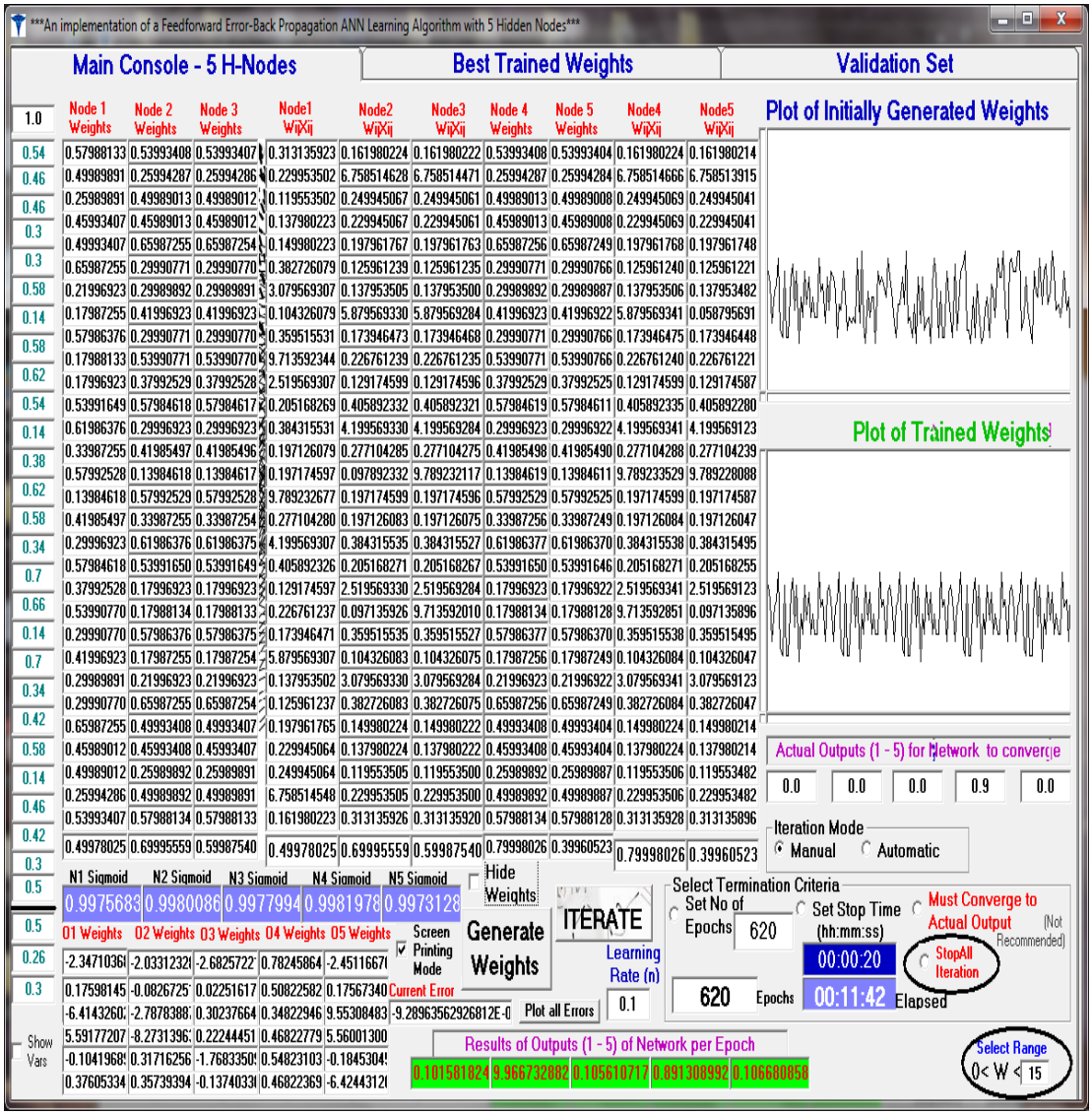


Figure 4.22: Screenshotof the results of iteration of a five-node hidden layer ANN processing when fed with unconnected Lassa fever data.

4.6.1.5 Performance Metrics for Two Hidden Layers

Produced below are the results of processing of inputs from the SHLT's outputs that was fed into the second hidden layer (SHL) of the MHLT, according to the scheme presented in section 3.3.3.2.1 of this desertation and as stated by (Hagan, Demuth& DeJesus,2002) in "for multilayer networks the output of one layer becomes the input tothe following (or second) layer". The performance metrics of the three hidden nodes in the first hidden layer (FHL) coupled into the two hidden nodes in the second layer showed appreciable conformity with thetrend of results of iteration sessions recorded in the SHLT for Lassa fever presented data. This came as a surprise as it was expected that discontinuities would abound as a result of the computed error responsibility coming from the FHL's outputs and not from the SHL. As recorded during iteration sessions, the deep learning ability of the MHLT made iterations sessions longer as the ANN took longer time to achieve convergence. A benchmark iteration

of 680 epochs of appreciable convergence was established from our experiments as shown below in Figure 4.23. This benchmark was high when compared with the SHLT which was set at 620 epochs and shows better characteristics and deeper features of the MHLT when compared with SHLT. Further iteration of the ANN beyond the 680th epoch i.e. around the 700th epoch produced significant errors as shown in (b) of Figure 4.23. The Parsimonious graph of the computed error or difference between the actual error from the expected error at various iterations is shown below in (c) of Figure 4.23.

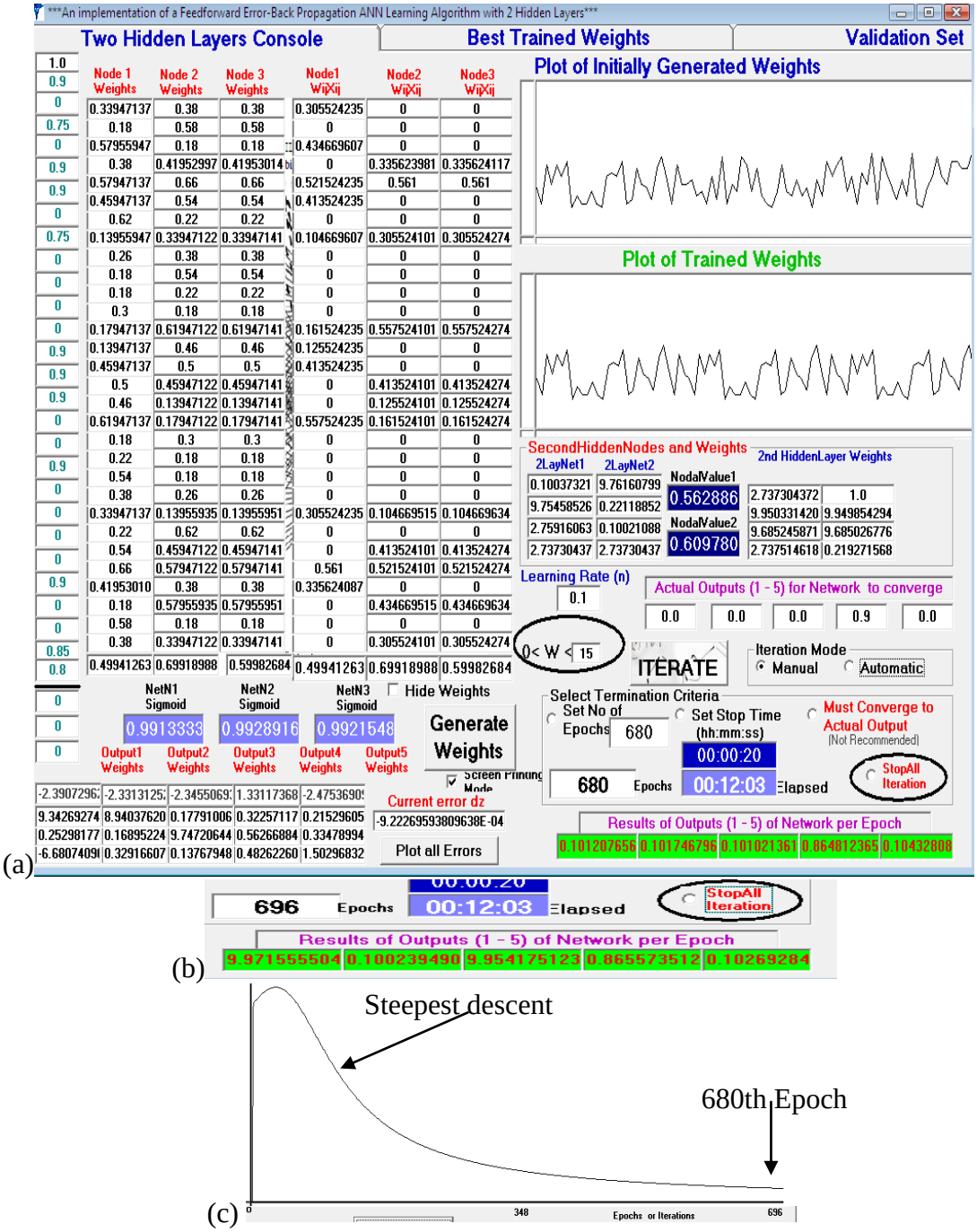


Figure 4.23: Screenshot of the results of iteration of a two hidden layers ANN processing when fed with Lassa fever data at (a) 680th epoch, (b) 696th epoch and (c) graph of computed errors against number of iterations

The effect of random data on training with Lassa fever expected output data showed that abrupt convergence to the inputs was obtained even before the benchmarked 680th epoch for Lassa-fever-consistent inputs showing the ANN's sensitivity to indiscriminate data as shown in Figure 4.24. This result outcome was remarkable in that it deviates from the trend created by the SHLT's execution of extraneous data which produced convergence well after the established benchmark of 620th epoch.Here convergence of indiscriminate data occurred before the established benchmark of 680 epochs for Lassa fever diagnosisand generated significant errors at the 680 epoch as shown below.

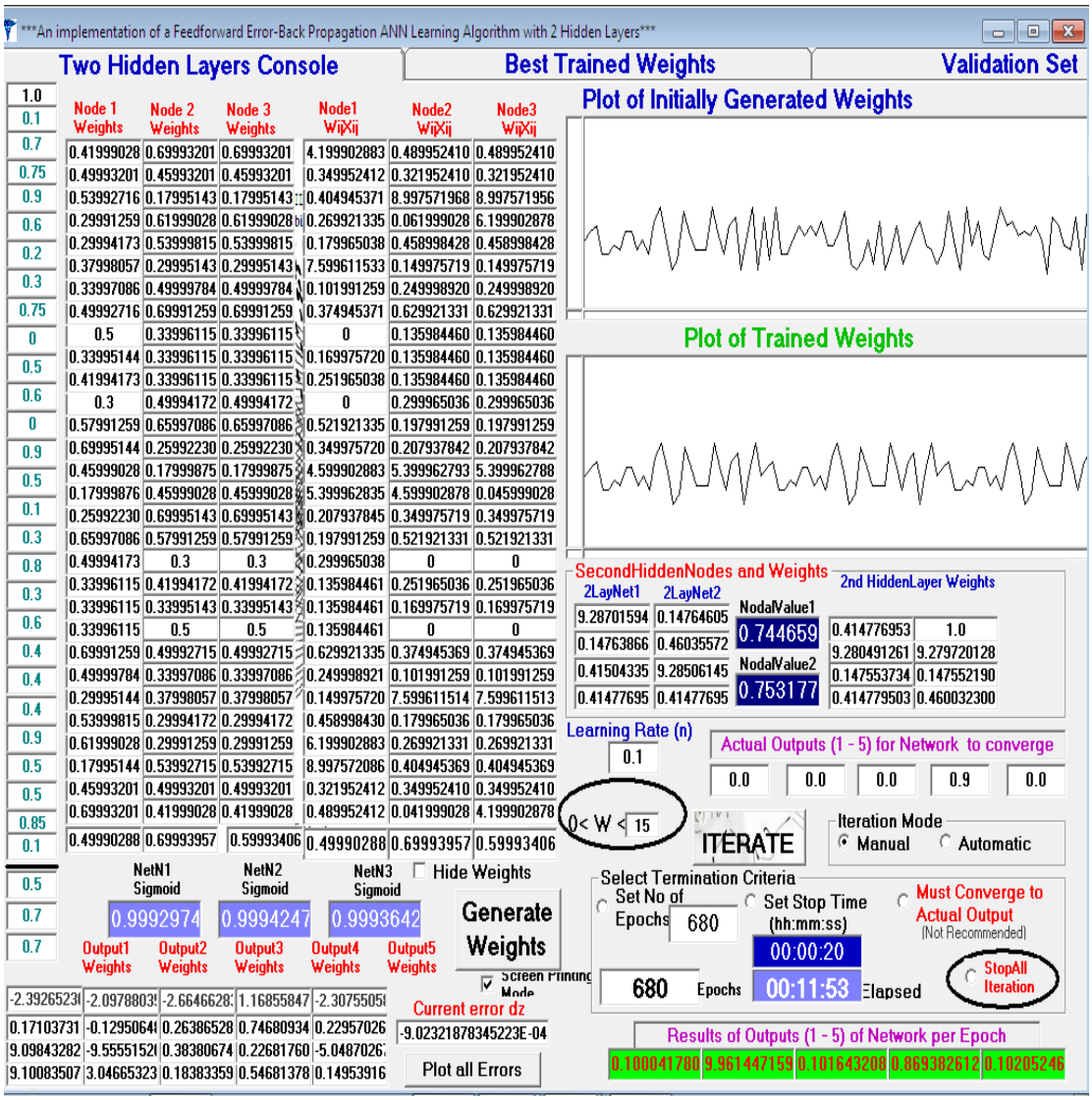


Figure 4.24: Screenshot of the results of iteration of a two hidden layers ANN processing when fed with unconnected Lassa fever data.

4.6.2 Performance Metrics on EVD Data

When the different topologies of the designed ANN were fed with demographic data on symptoms in confirmed and probable Ebola cases of patients from Guinea, Liberia, Nigeria, and Sierra Leone obtained from the WHO Ebola Response Team (Appendix E) the following results were obtained for each topology.

4.6.2.1 Performance Metrics for One Hidden Node

The result of the ANN processing using a single node in the hidden layer with EVD desired-output recognition data of (0.0, 0.0, 0.9, 0.0, 0.0) resulted in significant convergence at around the 620th epoch as shown in Figure 4.25(a) with that section enlarged in (b).The one node hidden layer which wasa fully connected ANN with thirty inputs and five output units showed that convergence continued to be fine-tuned immediately after the 620th epoch before the introduction of significant errors at 702nd epoch (c) and 712nd epoch (d). Further iteration beyond these points continued to propagate the generated errors by the ANN thus making the session null and void. The validation with new data showed that similar data yielded the same convergence results when iterated with the best trained weights while dissimilar data generated appreciable errors thus making generalization to be high with the one node hidden layer. However, the sensitivity of the network was not appreciable due to the long time in processing.

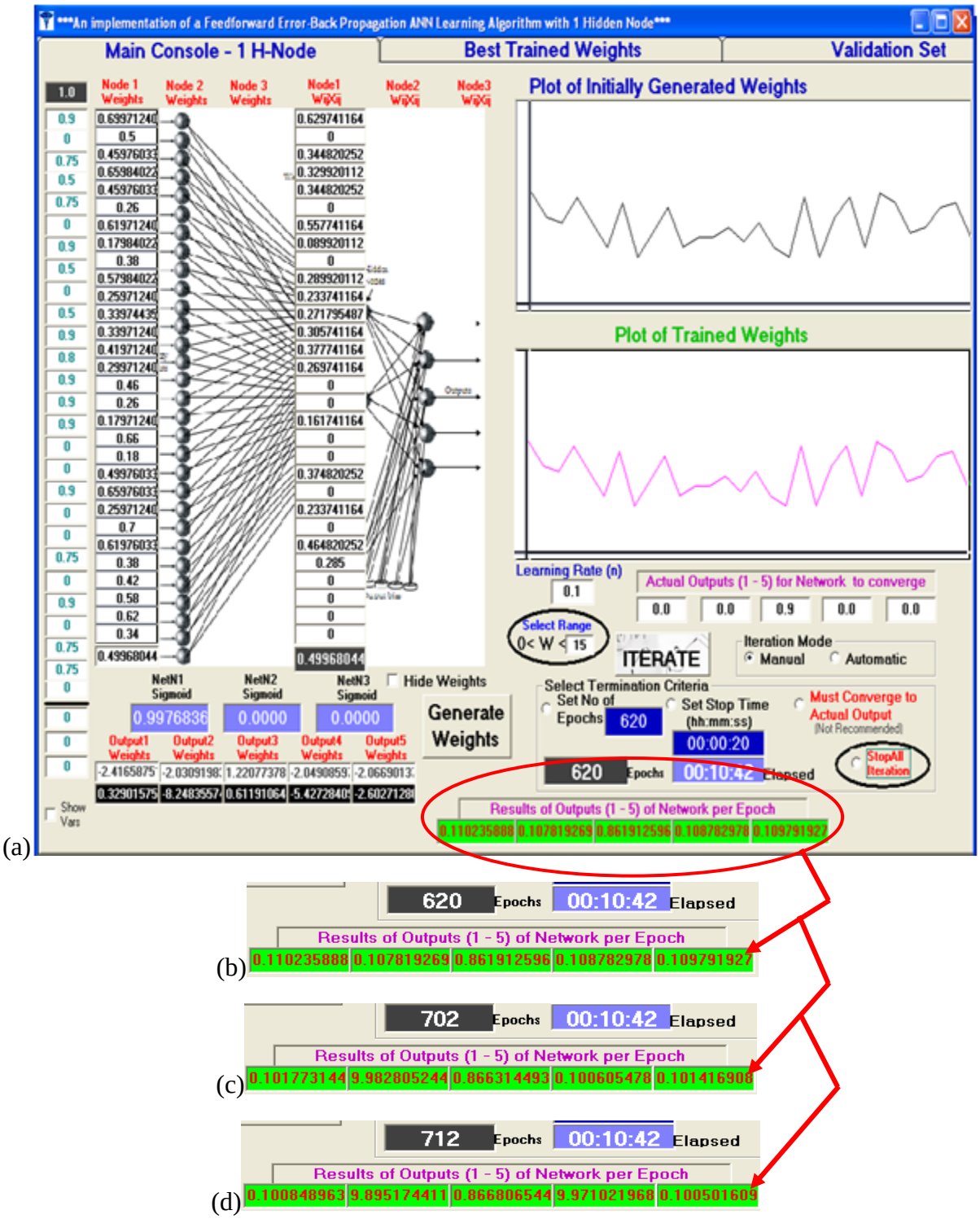
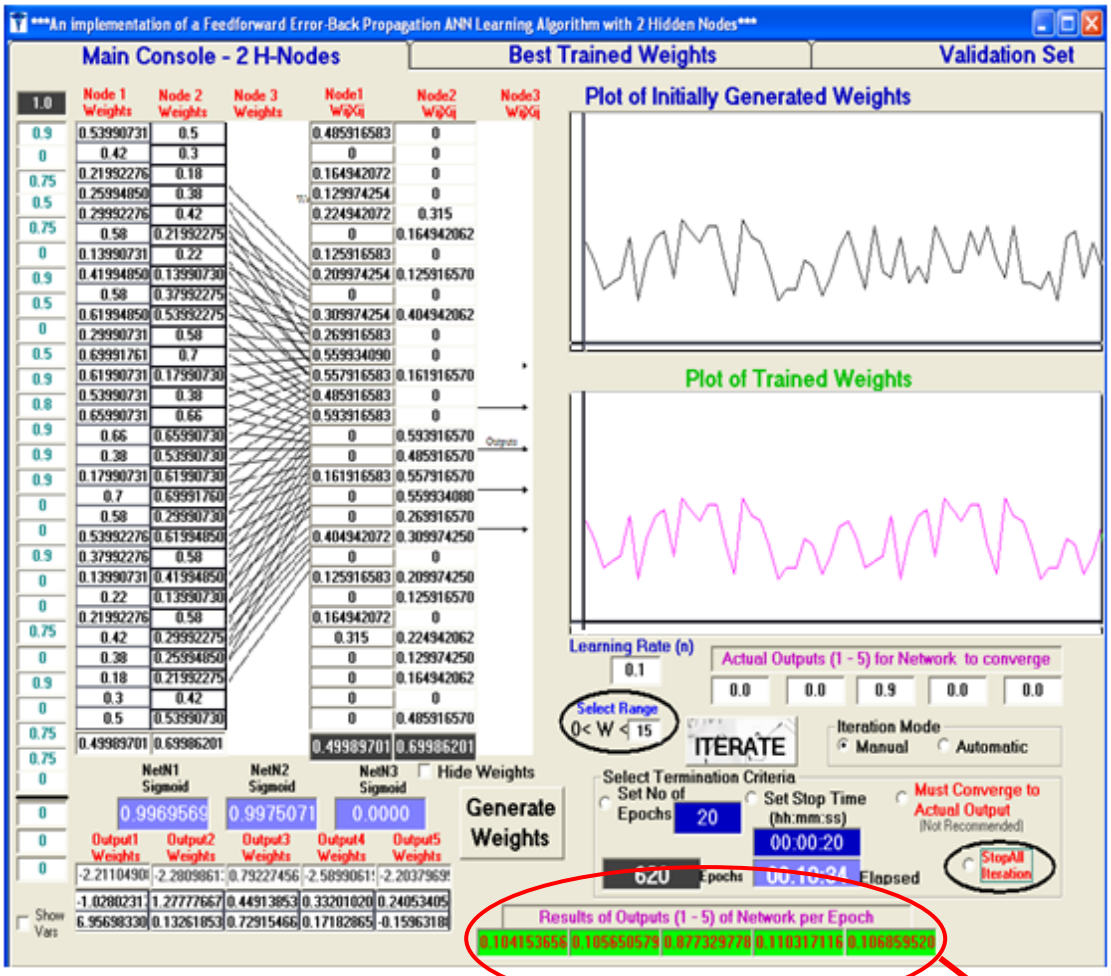


Figure 4.25: Screenshots of the results of iteration of a one-node hidden layer ANN processing of EVD data.

4.6.2.2 Performance Metrics for Two Hidden Nodes

The performance of a two hidden node ANN when fed with EVD consistent pre-fuzzified data showed that significant convergence was attained at around the 620th epoch during training and further training beyond this point showed appreciable finetunement but eventually resulted to the introduction of significant errors at the 675th epoch, Figure 4.26 (b).

(a)



(b)

Figure 4.26: Screenshots of the results of iteration of a two-node hidden layer ANN processing of EVD data

The best trained weights from the training session when iterated with similar input data for validation showed significant convergence but generated slight errors with disimilar input data signifying appreciable generalization.

4.6.2.3 Performance Metrics for Three Hidden Nodes

The performance of a three-node hidden layer ANN architecture showed that convergence had started at around the 200th iteration with significant convergence at around 620th iteration and generated errors at 660th epoch.

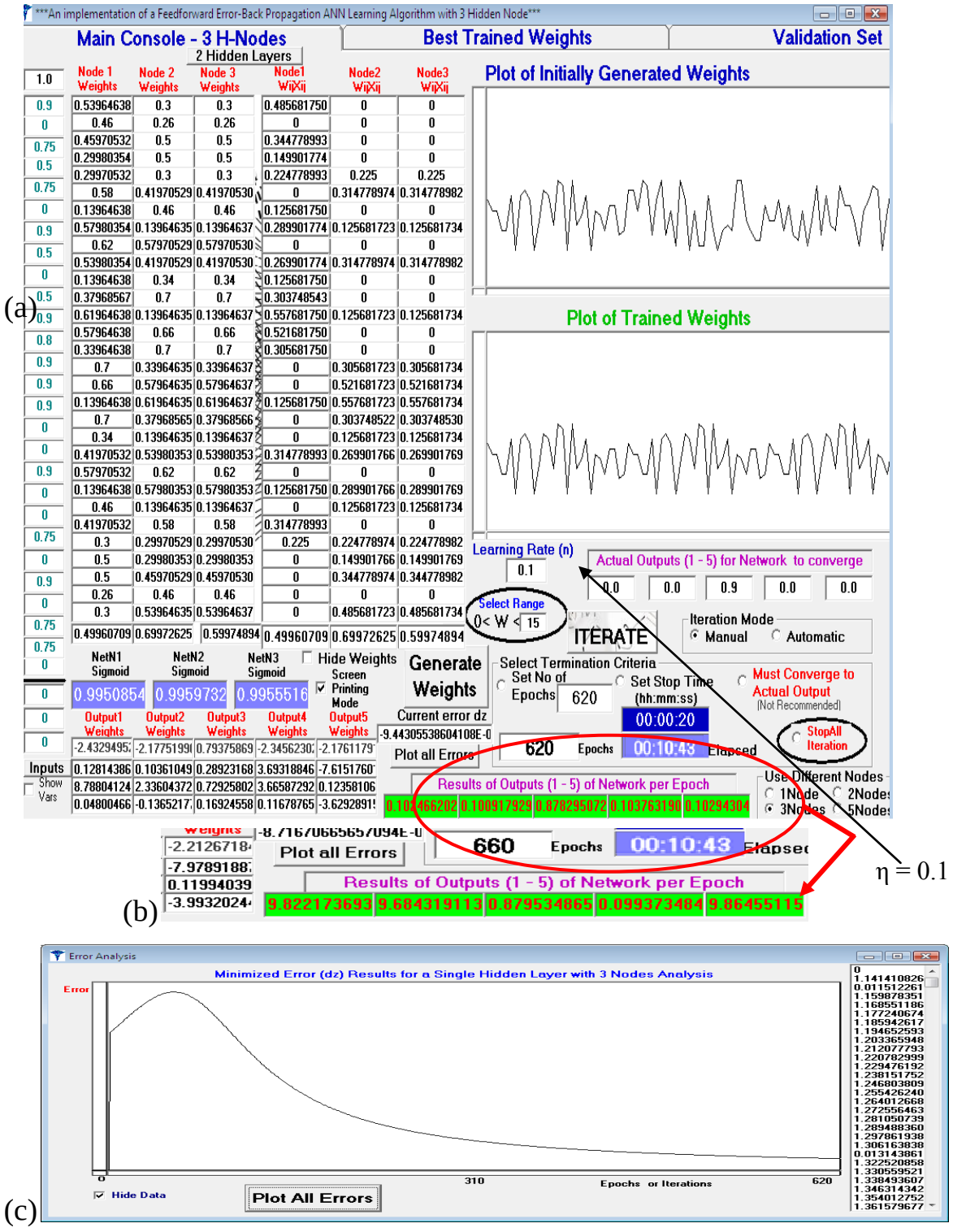


Figure 4.27:Screenshots of the results of iteration of a three-node hidden layer ANN processing of EVD data at (a) 620th epoch, (b) 660th epoch and (c) error graph

Further iteration of the three-node ANN resulted in errors as was noted in Figure 4.27(b) making the session null and void. The generated errors occurring at the 660th epoch showed that the network's sensitivity was high and appreciable generalization was obtained when trained weights were validated with similar input data. Figure 4.27(c) shows the error distribution and how the computed error was used to find a convergent solution and Figure 4.28 shows the response of the three hidden node ANN on dissimilar data. These features of the three hidden node architecture allowed for its adoption for the designed of the Expert systemowing to its sensitivity and generalization on new data.

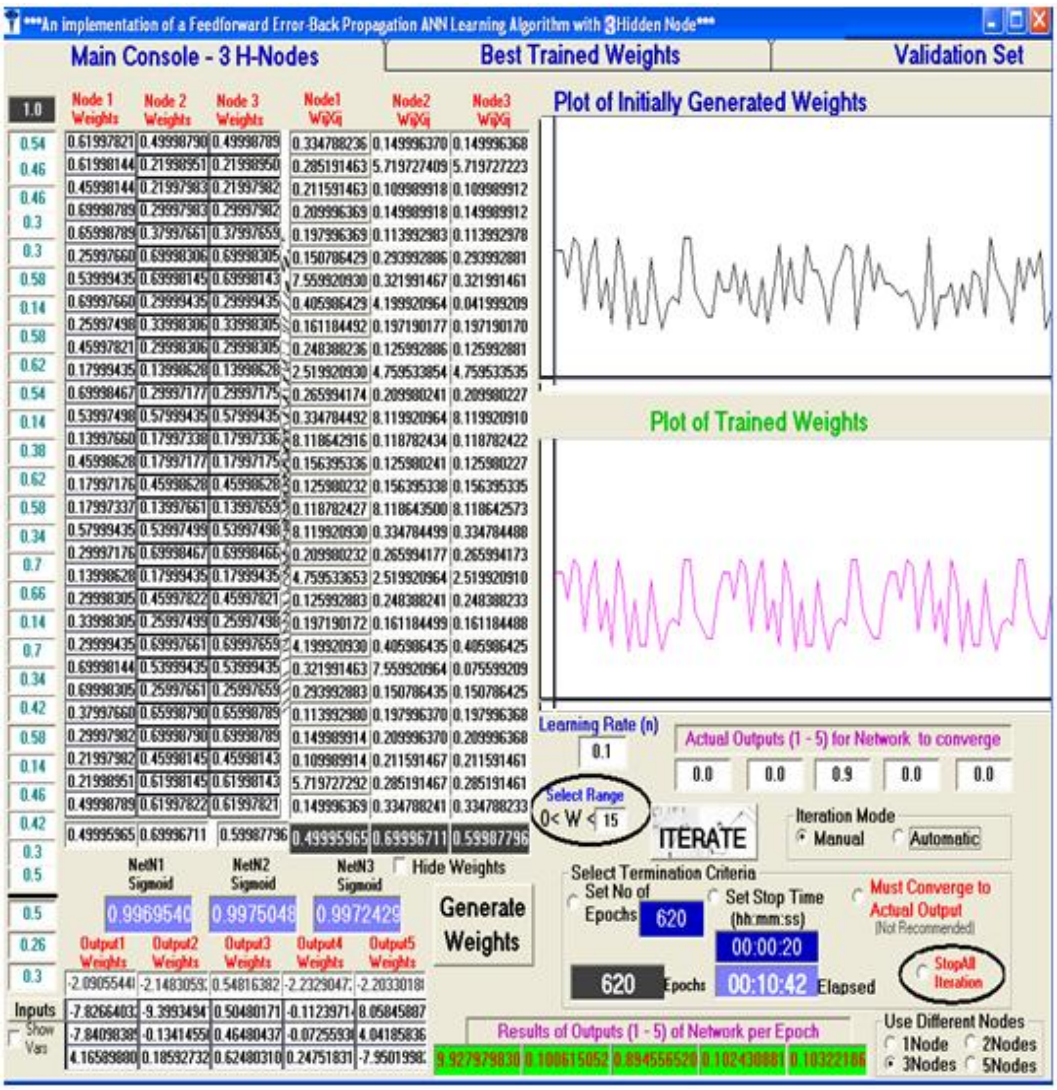


Figure 4.28: Screenshots of the results of iteration of a three-node hidden layer ANN processing of EVD data on random or different fed data.

4.6.2.4 Performance Metrics for Five Hidden Nodes

On investigation with various sessions trainings, the five-node hidden layerANN performed well on training data fed to it by producing convergence around theexperimentally determined benchmark epoch (EDBE)of 620for one hidden layer ANN.However, on further

iteration beyond this benchmark epoch (shown in Figure 4.29) errors were generated as seen in (b) and these errors continued and became significant around the691st epoch (c). Thus the five-node hidden layer'sresponse was high sensitive to inputted data. Its response to unconnected EVD data or random input data also showed that generated errors had begun at the 620th epoch as shown in Figure 4.30.

Hence from above, though with a convergence of input data at the 620th epoch suggesting an EVD analysis but when the iteration on the network was continued significant errors were generated at around the 680th epoch (note around the 660th epoch for the three hidden layer analyzed above). Thus with increasing number of hidden nodes, the ease of memorization of data by the ANN became pronouced; though its sensitivity appreciated considerably, its generalization on new data became unpredictable as presented validation data showed long convergence sessions. This shows that the ANN had begun memorizing the input data sequences instead of training itself to learn and identify the inputted data sequence within an optimal time or period.

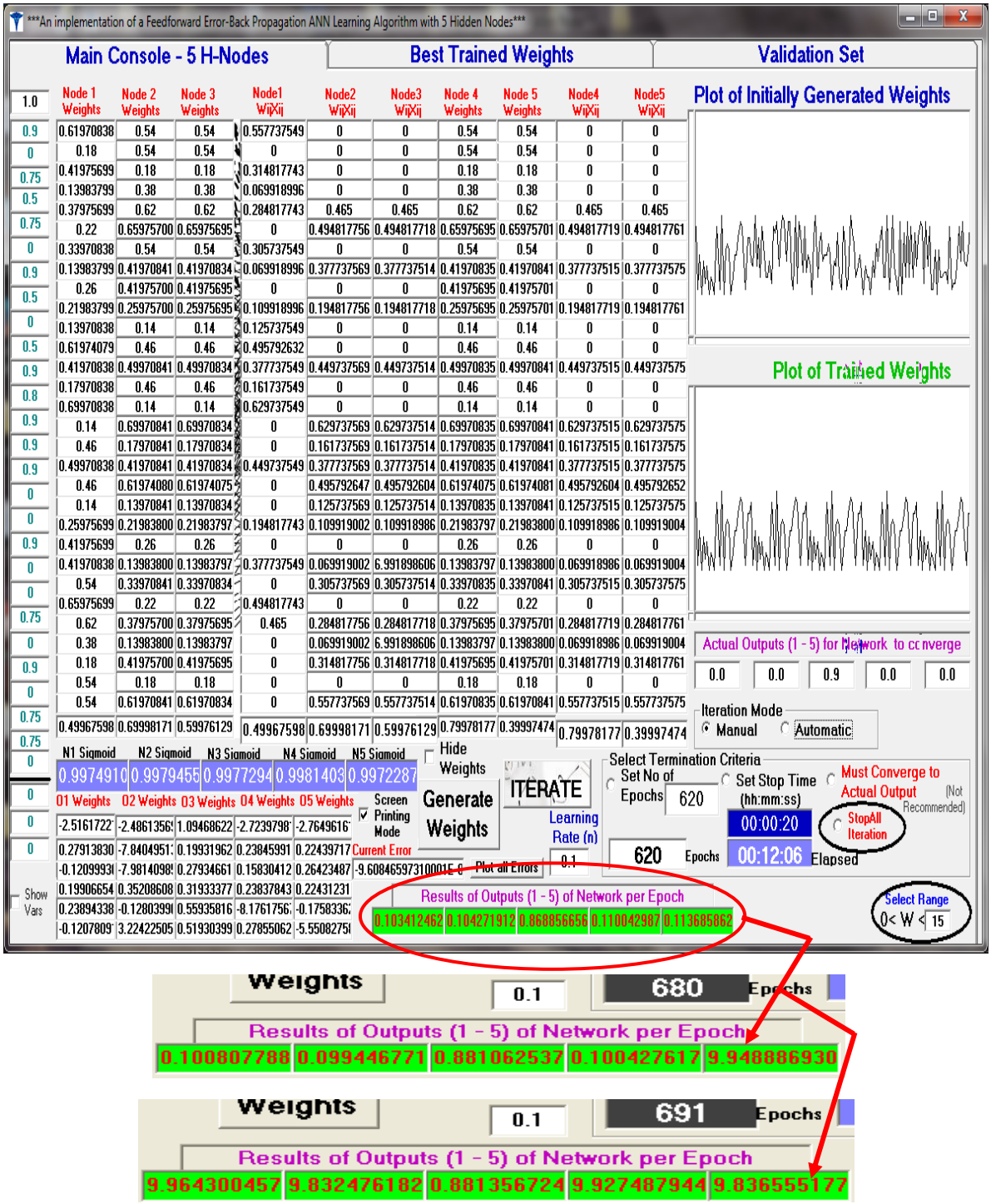


Figure 4.29: Screenshots of the results of iteration of a five-node hidden layer ANN processing of EVD data

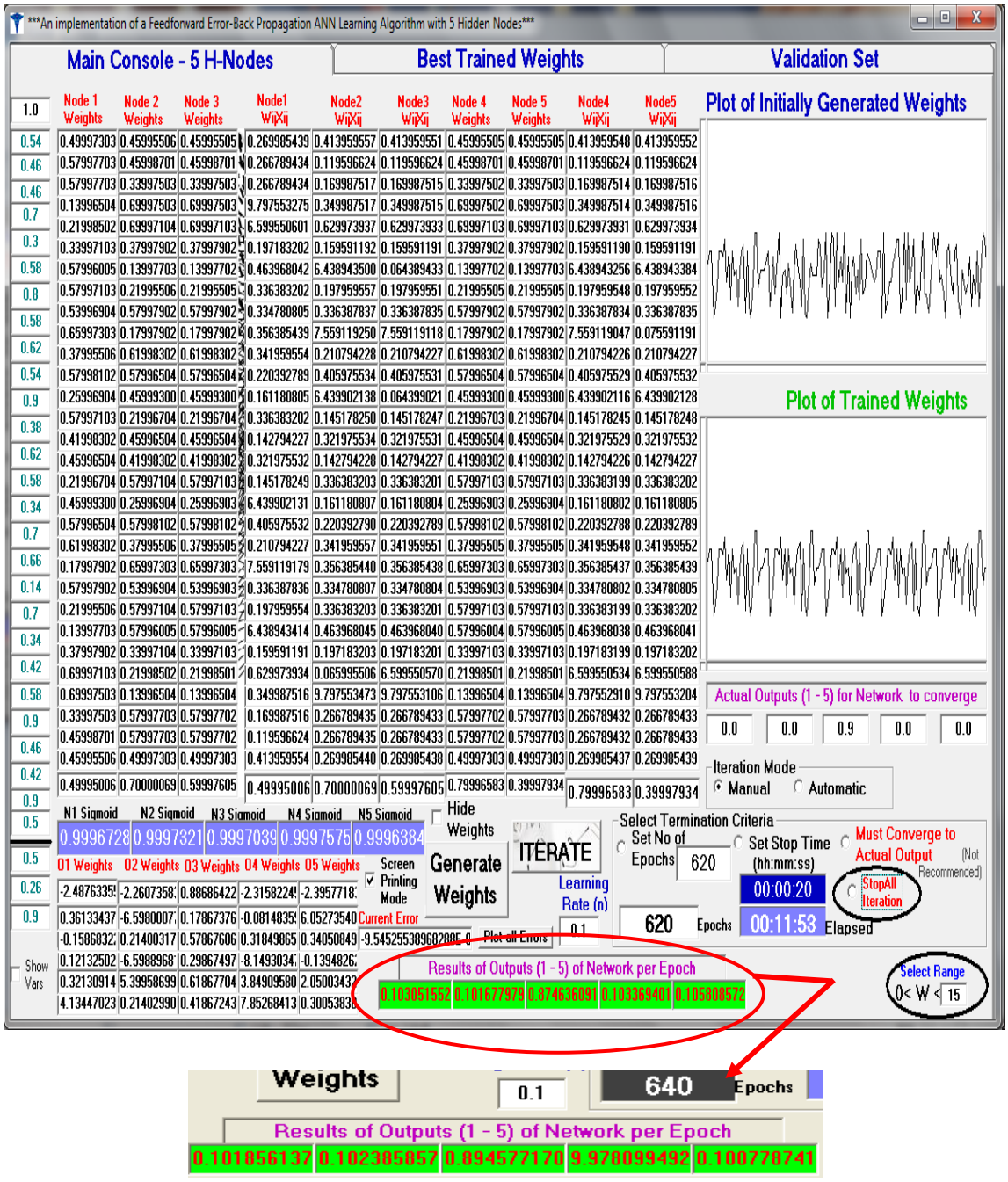


Figure 4.30: Screenshots of the results of iteration of a five-node hidden layer ANN processing of unconnected EVD data or random input data showing data memorization by the ANN.

4.6.2.5 Performance Metrics for Two Hidden Layers

When the two hidden layer (THL) networkwas fed with inputs consistent with EVD pre-fuzzified symptom's data, the obtained performance was as follows. At the 680th epoch (Figure 4.31), appreciable convergence had been attained by the ANN which conforms to the results of analysis of the SHLT already investigated above and proved that the results are consistent with those of the universal set of contagious diseases been investigated though with a higher convergence of 680 epoch. However, beyond this convergence point, significant errors was observed by the ANN at around the 697th epoch as seen in (b).

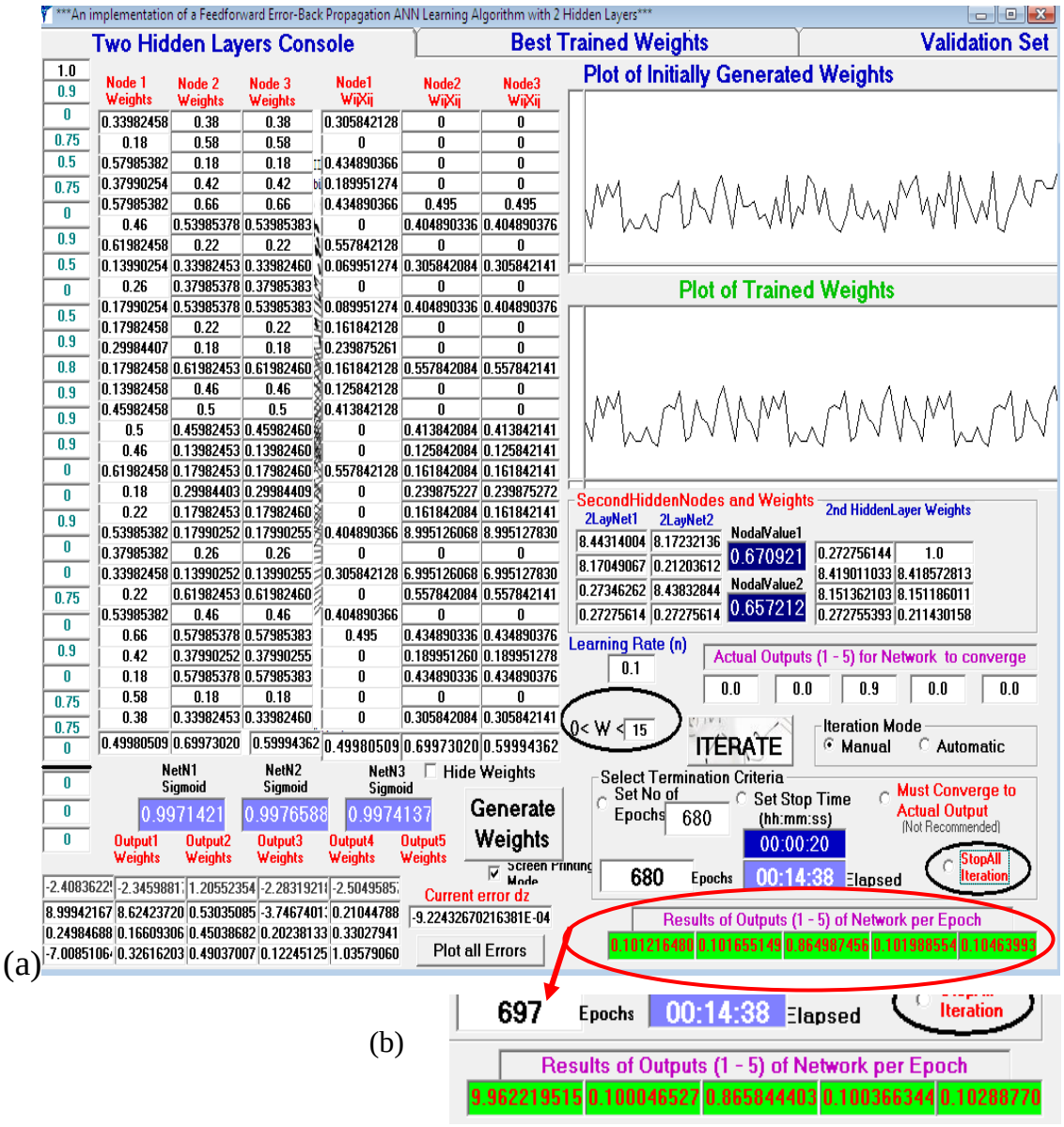


Figure 4.31: Screenshots of the results of iteration of a two hidden layer ANN processing of EVD data at (a) 680th epoch and (a) 697th epoch.

4.6.3 Summary of Performance Metrics on Various Topologies

The analysis above have shown the performance of various ANN topologies on fed inputs by comparing their adaptive tendencies to a desired output with an overall aim of reducing computed errors. Their ability to generalize to new data and their sensitivity to input data have been investigated.The importance of error-prevention and error-reduction in medical software applications as advocated by (Ramadan & Al-Saleh, 2013), Iwasokun et al. (2015) and (Kumar, Sathyadevi & Sivanesh, 2013) have been brought afore in all experimentsand showed that the ANN can be deployed for realtime operation.

The one-node hidden layer topology though possesses good generalizability have not been chosen due to its inability to be sensitive to a range of inputs that are similar to the training inputs, i.e. causing significant underfitting of intended data range. A significant improvement in sensitivity was obtained by the two-node ANN whilethe three-node architecture demonstrated significant sensitivity and generalizability to validation data at the 620th epoch. The five-node ANN architecture demonstrated elements of overfitting on some training and validation sets making it error prone if deployed for the medical expert system for contagious disease detection. In the two hidden layers topology investigated it was observed that deep learning ability was attained by the ANN but the number of iterations (680 epochs) necessary for convergence to occur was two much for a time conscious expert system.Based on the results of experiments abovethe three hidden node architecture was chosen for the expert system since it uses just 620 epochs as against the 680 epochs for the MHLT to reach convergence when presented with the fuzzified data for contagious diseases. This also supports the conclusion by Suzuki et al. (2005), that "a three-layer structure was used as the structure of each MTANN because any continuous mapping can be realized approximately by three-layer ANNs". Also supporting this conclusion is (Funahashi, 1989 and Baron, 1993). Hagan et al. (2013) had also stated that "for a network to be able to generalize, it should have fewer parameters than there are data points in the training set. In neural networks, as in all modeling problems, we want to use the simplest network that can adequately represent the training set. Don't use a bigger network when a smaller network will work (a concept often referred to as Ockham's Razor)".

The learning rate, η has been left at the lowest value of 0.1 in all the scrutinized ANN topologies this is because at a high value close to or equal to 1, the local minimum may be constantly overstepped, resulting in oscillations and low (or no) convergence to the lower error state. However, this has made the number of iterations to become large at a benchmark of 620 epochs for one hidden layer and 680 epochs for two hidden layer based on experiments

recorded during several iterative sessions. The small η also required an average iteration time of about eleven minutes to accomplish for all considered single hidden layer topologies and around fourteen minutes for the THL topology.η has also been deliberated kept low (0.1) to enable the ANN to have the best opportunity of learning and lowest opportunity to memorize. The tabulation below shows a summary of the performance metrices for all considered topologies.

Table 4.1 Performance Metrics of Various ANN topologies showing Sensitivity and Generalization on data.

Hidden-Layer Topology	Average Time(s)	Sensitivity	Generalization	Memorization
One-Node ANN	10:23	Poor	Very Good	Very low
Two-Node ANN	10:36	Fair	Good	Low
Three-Node ANN	10:40	Good	Good	Normal
Five-Node ANN	10:58	Good	Poor	Highest
Two-Hidden Layers	14:30	Very high	Very poor	High

4.6.4 Effect of Increasing Learning Rate,η on Performance Metrics

The three-node hidden layer ANN was used in analyzing the effect on network performance of adjustment in learning rate.The learning rate, η which was chosen to be a default of η= 0.1 to allow very minimal oscillations during gradient descent optimization of the ANN was increase to η = 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8 and 0.9. In each increase of η, the performance of the ANN was observed and iterations were stopped just when the network's prediction accuracy is about to deteriorate or become error bound.

4.6.4.1 Effect of Increasing Learning Rate,η on EVD Data

As state above, the learning rate, η, $0 < \eta < 1$ is an arbitrary constant chosen to help get the neural network weights towards its global minimum for SSE and from Equation 3.14, the learning rate must be chosen in such a way that it is neither too small or two large. At η = 0.2, it was observed that convergence of the ANN when fed with EVD data was faster at 245 epochs(Figure 4.32) when compared with convergence at η = 0.1 (Figure 4.27). This is shown below and it was also observed that the iterations for convergence to occur continued to be

reduced as η increased to a maximum of 0.9 (Table 4.2). However, at this maximum value of η or close to maximum, serious oscillations of the network occurred (shown in Figure 4.34) thus not guarantying that the network would not overshoot its optimal solution when 'fitting the data'if such high values for learning rate was used. Based on conducted tests and results shown inFigure 4.32, a learning rate of η = 0.3 wasadopted as it provided a balance or trade-offwhen considering all the factors of iterations and a quick convergence without significant overshooting of desired optimal solution.

Table 4.2 Performance Metrics of variation of Learning rate, η with EVD data

η	**No of Epochs/Iterations**	**Time**
0.1	620	10.06
0.2	245	04:29
0.3	144	02:29
0.4	89	01:33
0.5	65	01:00
0.6	47	00:51
0.7	38	00:42
0.8	29	00:27
0.9	23	00:27

From Table 4.2, it can be observed that appreciable number of iterations had been performed by the ANN when learning rate has been set to η = 0.3. The result of convergence achieved at this learning rate is still very close to the optimality sort for by the BPA and also helps to improve the peculiar slow nature of gradient descent optimization of BPA. However, further increase to a learning rate of η = 0.4 and above will improve the time to a few seconds and a few iterations but will introduce significant error and distortion or coarseness of the error curve.

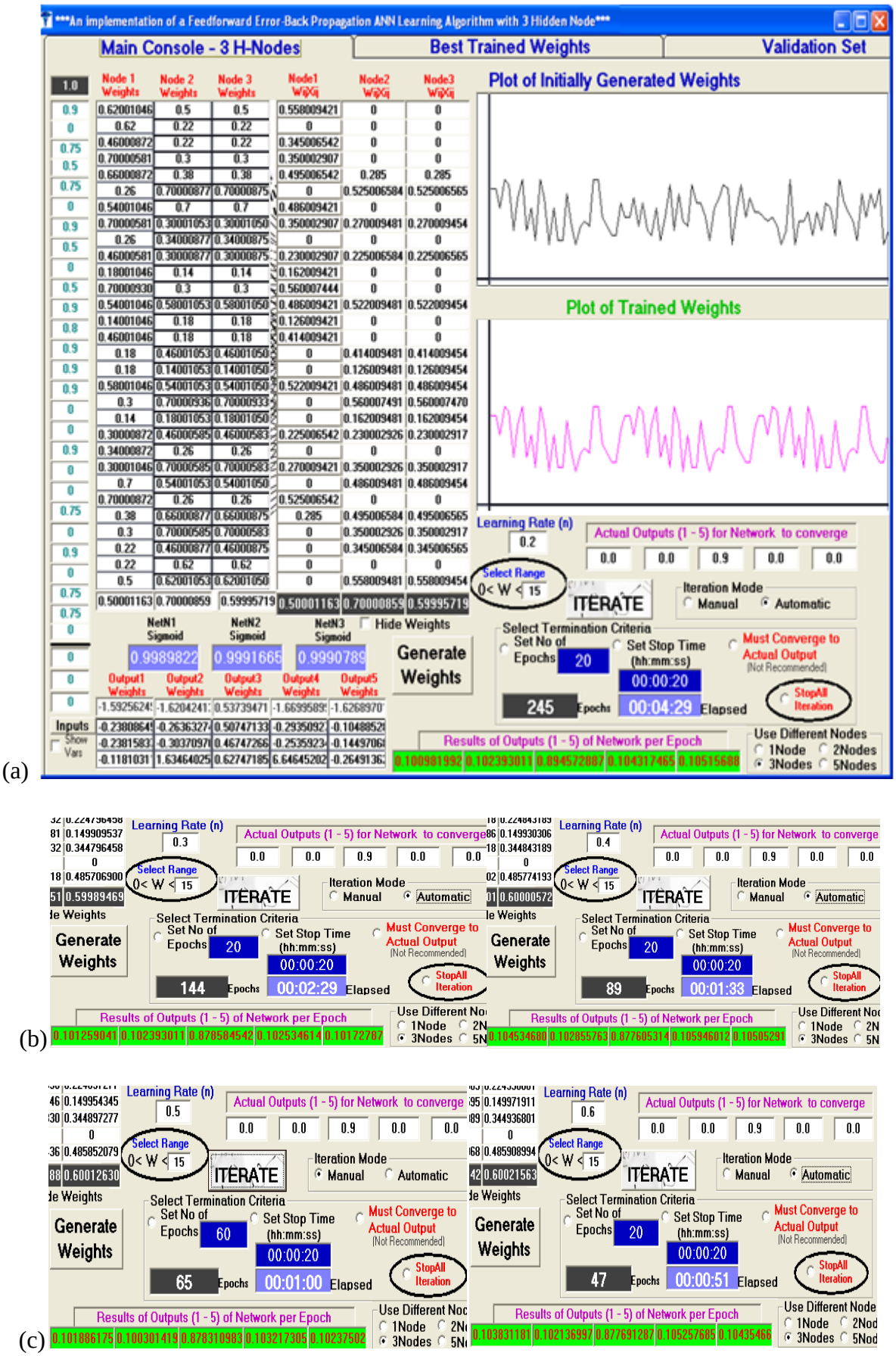


Figure 4.32: Screenshots of the results of increasing learning rate, η, (a) η = 0.2, (b) η = 0.3 and η = 0.4, (c) η = 0.5 and η = 0.6 on EVD data.

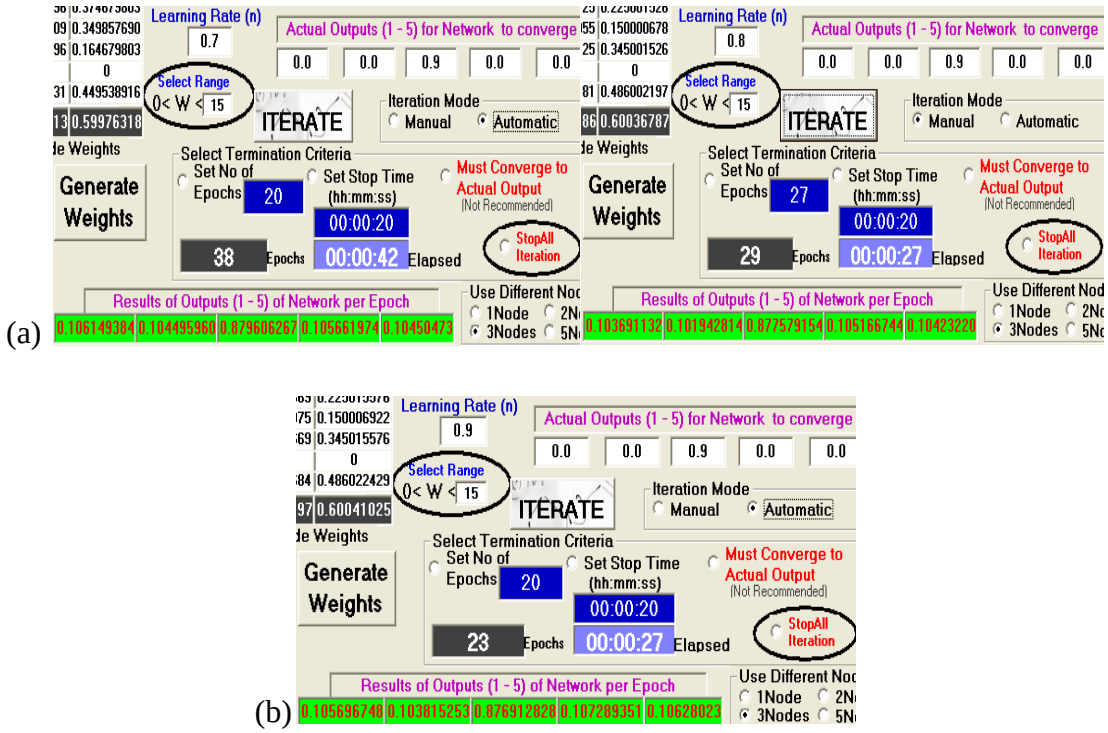


Figure 4.33: Screenshots of the results of increasing learning rate, η at (a) η = 0.7 and η = 0.8, (b) η = 0.9 on EVD data.

A graphical illustration of the effect of the learning rate, η on presented data with respect to the experimentally determined benchmark epoch, EDBE of the EVD symptom's presented data showed that increase in learning rate, η caused the symmetry in the graph of Figure 4.27 (c) for a η = 0.1 to become coarse or unfitting as η continue to increase. This coarseness of the computed errors generated during iterations session became significant at η = 0.4 reaching a climax at the maximum allowable learning rate of η = 0.9. Figure 4.34 shows the effect of each increase in learning rate: whenη was set to 0.2, the symmetry in the graph was maintained showing faster convergence when compared with η = 0.1. Also, at η = 0.3 appreciable speed of convergence by the ANN was attained with tolerable errors and semi-coarsed graph symmetry. At η = 0.4 the appreciable speed has started given rise to intolerable coarseness in the iteration graph and at η = 0.5, 0.6 and 0.8 respectively, from the graphs it was observed that though the ANN speed of processing continue to increase steadily the coarseness of the graphs became intolerable resulting to a poor amenability of the approximation to real nonlinear problems. The worst coarseness in the graphs was observed when η was set to 0.9 showing abnormality behavior. Thus it can be deduced that a compromise in speed or rate of convergence is also a significant concession to a poor accuracy of thesimulation.

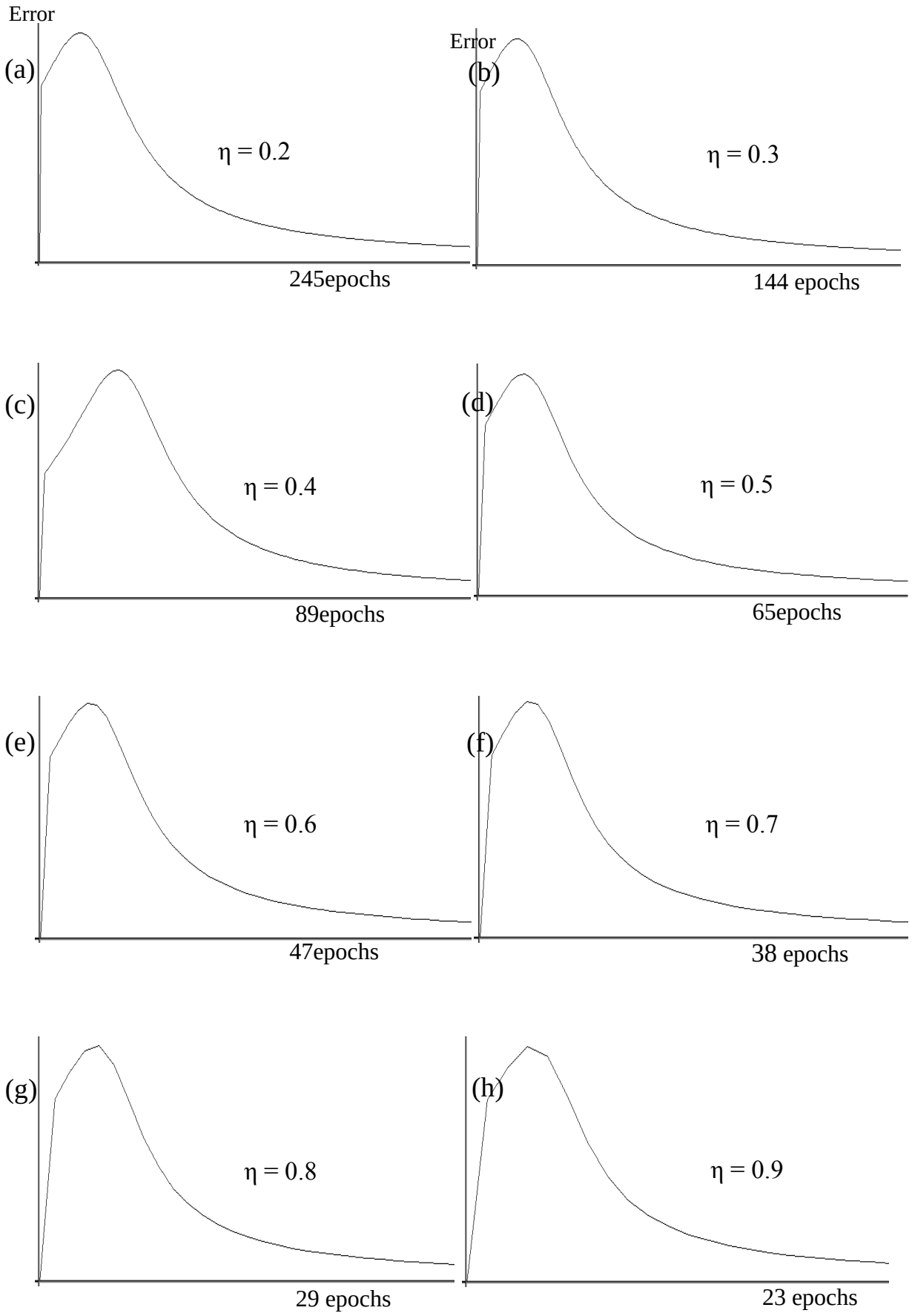


Figure 4.34: Screenshots of the symmetry of the graphs generated during increasing learning rate, η at (a) η = 0.2, (b) η = 0.3,(c) η = 0.4, (d) η = 0.5, (e) η = 0.6, (f) η = 0.7, (g) η = 0.8 and (h) η = 0.9 on inputted EVD data.

4.6.4.2 Effect of Increasing Learning Rate,η on Lassa Fever Data

The effect of increasing learning rate, η on training of Lassa fever data was also investigated by this research in which it was found that appreciable convergence occurred as η continued to be increase from 0.1 reaching a maximum convergence rate at η = 0.9 as shown below. Since Lassa fever diagnosis belong to the set of contagious diseases with similar symptoms to EVD, results were generally similar to the ones shown in Figure 3.34 above. Table 4.3 shows a summary of the various responses of the ANN to increasing η with their accompanying graphs in Figure 4.37 and 4.38.

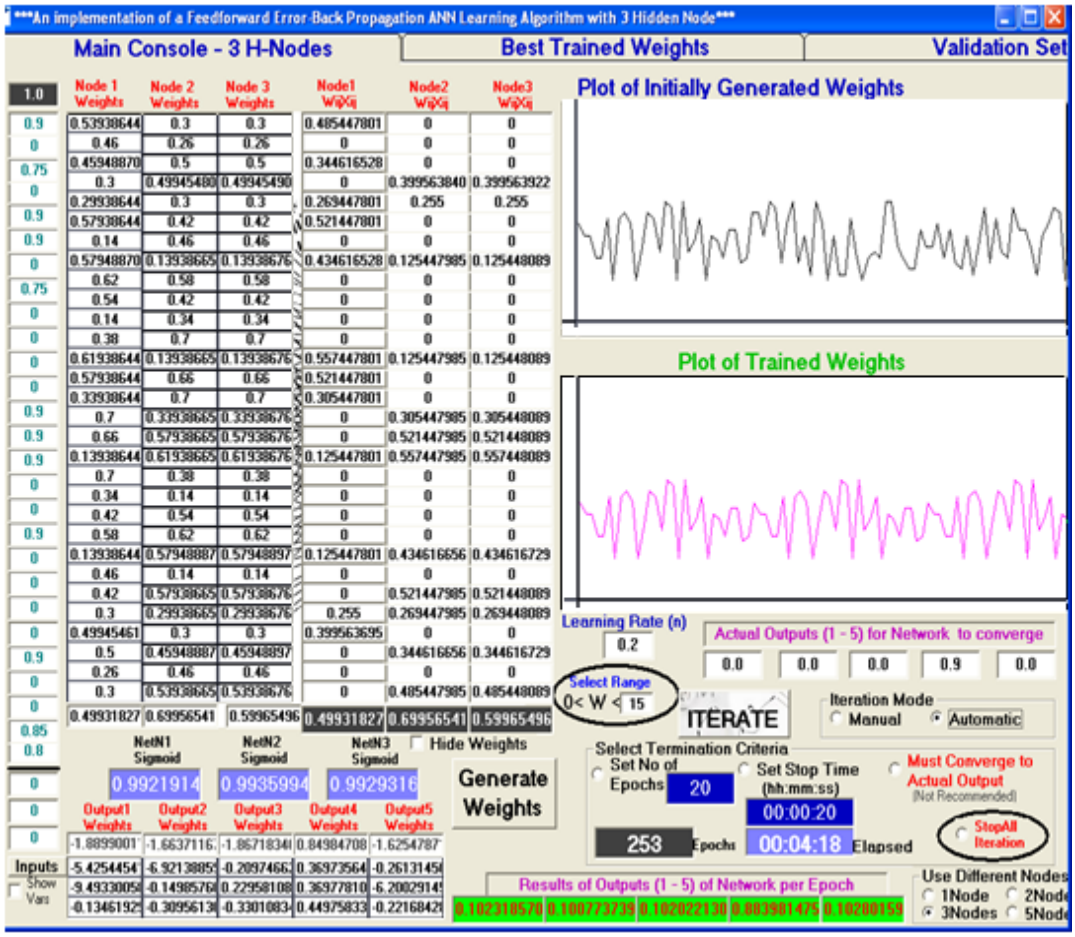
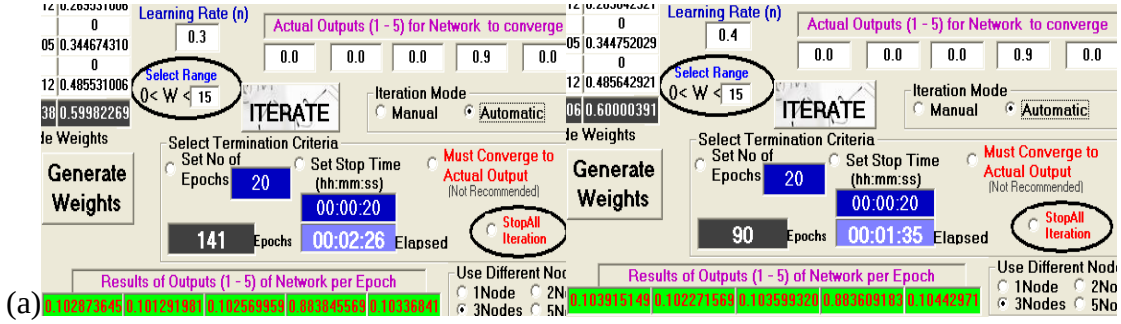
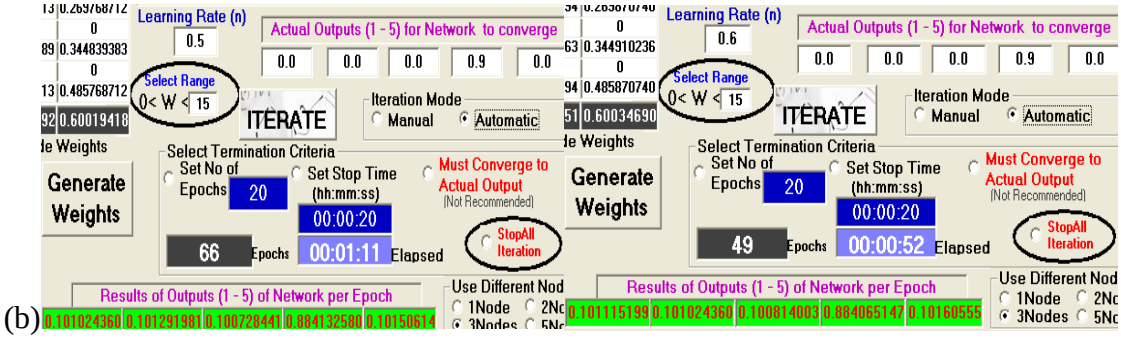


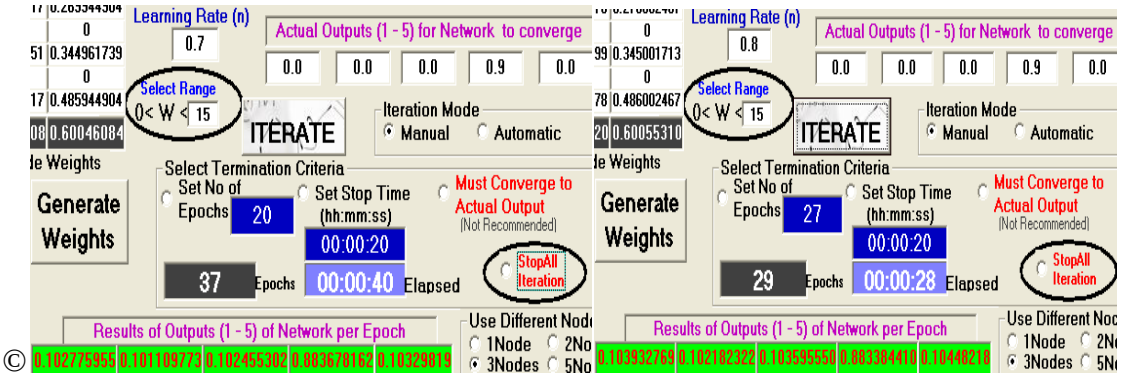
Figure 4.35: Screenshots of the results of increasing learning rate, η to η = 0.2 on presented Lassa fever data on ANN.



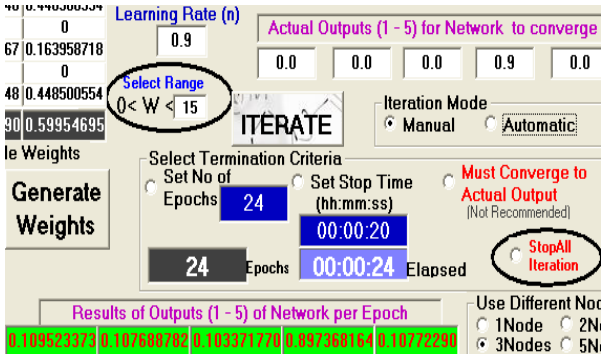
(a)



(b)



©



(d)

Figure 4.36: Screenshots of the results of increasing learning rate, η at (a) η = 0.3 and η = 0.4, (b) η = 0.5 and η = 0.6, (c) η = 0.7 and η = 0.8 and (d) η = 0.9 on presented Lassa fever data on ANN.

Table 4.3 Performance Metrics of Variation of Learning rate, η with Lassa Fever data

η	No of Epochs/Iterations	Time
0.1	620	10:36
0.2	253	04:18
0.3	141	02:26
0.4	90	01:35
0.5	66	01:11
0.6	49	00:52
0.7	37	00:40
0.8	29	00:28
0.9	24	00:24

Based on the analysis above, the value of η should not become exceedingly too high so that the implementing ANN does not become unstable and oscillate out of its optimum solution, thus η was kept at 0.3 for both EVD and Lassa fever diagnosis.

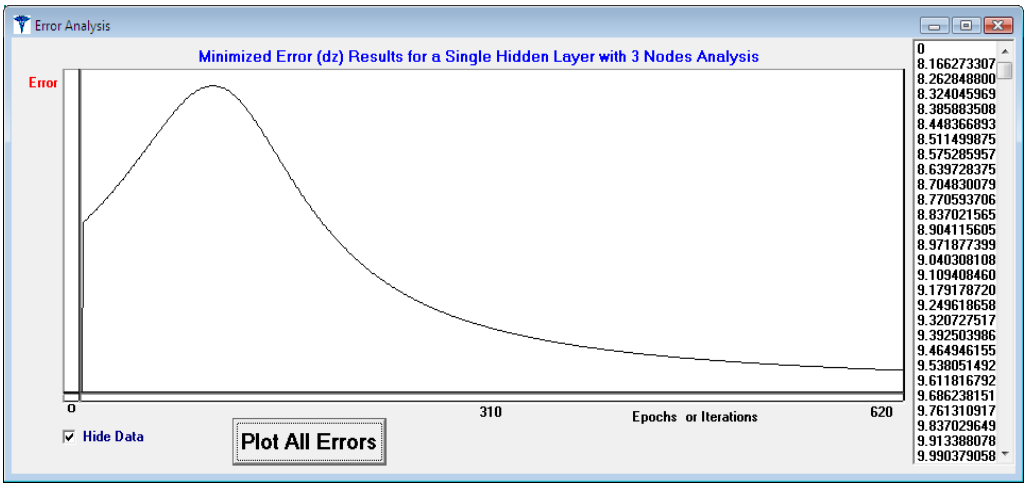


Figure 4.37: Screenshot of the generated graph with the initiallearning rate of η = 0.1 for Lassa fever data.

Also, from Figure 3.38, when η was set to 0.2 appreciable speed was demonstrated by the ANN with an appreciable graph symmetry at (a). The speed of convergence continued to increase when η = 0.3 and 0.4 with appreciable graph symmetries. However, at η = 0.5 the graph symmetry has become intolerably coarse in the convergence graphs and the coarseness continued at η = 0.6 to 0.9, i.e. (d) through (h).

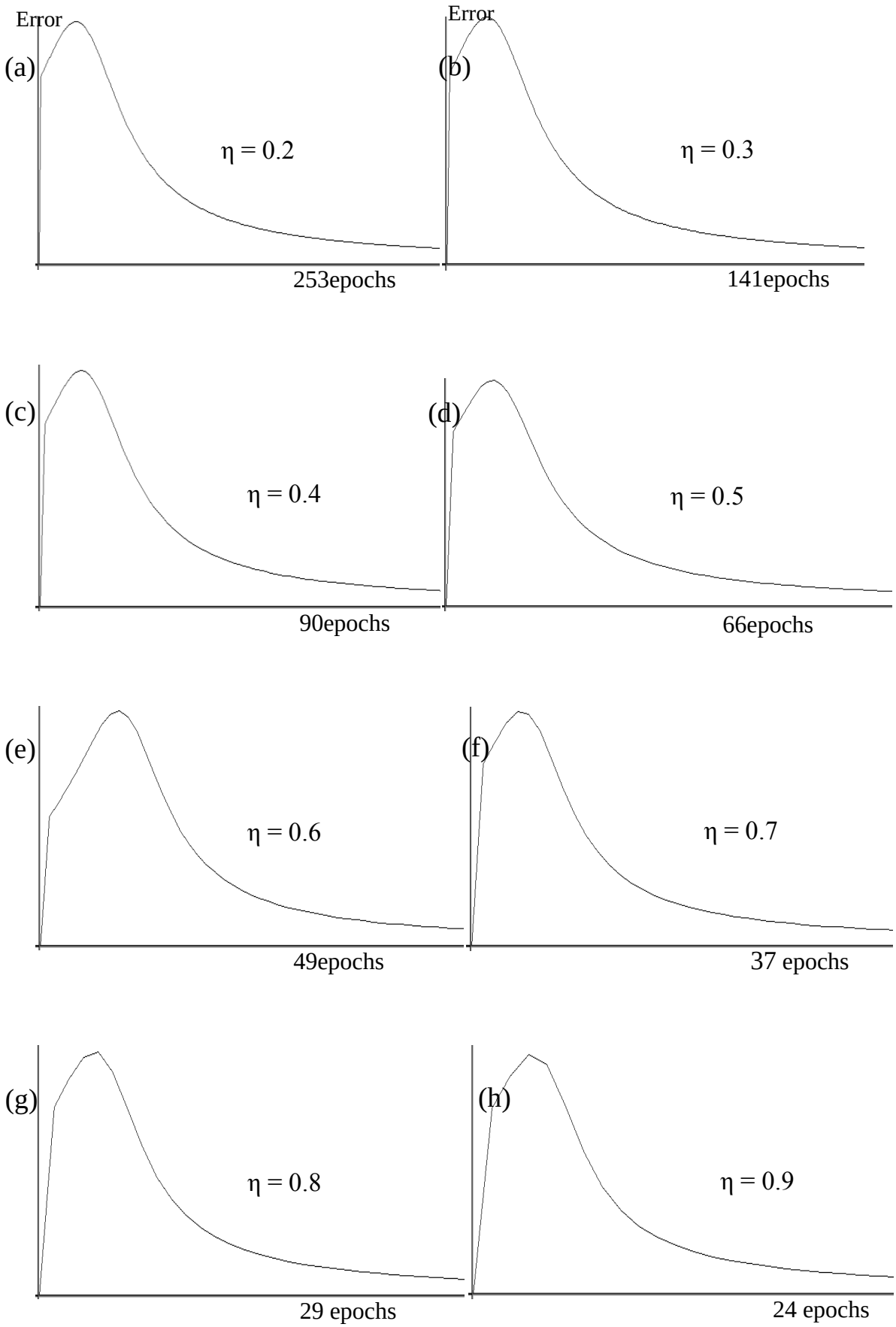


Figure 4.38: Screenshots of the symmetry of the graphs generated during increasing learning rate, η at (a) η = 0.2, (b) η = 0.3, (c) η = 0.4, (d) η = 0.5, (e) η = 0.6, (f) η = 0.7, (g) η = 0.8 and (h) η = 0.9 on inputted Lassa fever data.

4.6.5 Analysis of Results of Processing

The concept of Ockham's Razor (Hagan et al. 2013) which involves not using a bigger network size for solving modeling problems when a smaller network can do the job has been used in this work. The ability to interpret decision rules in terms of linguistic variables in fuzzy-neural inferencing (Kulkarni, 1999) have also been used in this study to identify important classes or sets from mostly vague reporting of symptoms by patients and the power embedded in ANN for data mining and classification have been used to create a pattern recognition or classification system.The improvement in speed as shown in Table 4.2 and 4.3 above of the designed ANN with increasing η makes the work to approach the processing speed with the MATLAB neural network tool (MNNT) that is based on the Levenberg-Marquardt Algorithm (LMA) which is known to be the fastest for ANN simulation as pointed out by(Badri, 2010) and *(*Demuth & Beale, 2002). (Hagan & Menhaj, 1999) also stated that the LMA deployed in the MNNT "trains neural networks at a rate 10 to 100 times faster than the usual gradient descent backpropagation method" provided that the data is within the tolerable range of the LMA.

A confusion matrix, which (Wikipedia, 2016) stated as "in the field of machine learning and specifically the problem of statistical classification, a confusion matrix, (also known as an error matrix) is a specific table layout that allows visualization of the performance of an algorithm, typically a supervised learning one" was used in validation of the designed ANN based on Lassa fever diagnostic data obtained from recognized hospitals. The validational data which was from the Federal Teaching Hospital, Abakaliki (FETHA) in Ebonyi State, Southeast Nigeria and from the Institute of Lassa Fever Research and Control (ILFRC) at Irrua Specialist Teaching Hospital in Edo State, Nigeria showed the following results. These data consisted of 20 cases (10 confirmed and 10 suspected) from FETHAand 14 cases (7 confirmed and 7 probable) from ILFRC making a total of 34 cases of presented Lassa fever data to the designed ANN. The true positives obtained with the confusion matrix analysis (Fawcett, 2003, 2006) and (Swets, 1988, 2000) showed that the ANN's analysis correctly classifiedover94% of both data on Lassa fever as shown in Table 4.4.

The Sensistivity or True Positive Rate (TPR) which is also equivalent to the Hit rate or Recall according to (Wikipedia, 2016) and (Powers, 2011) was computed from

$$TPR = \frac{TP}{TP + TN} \quad (4.1)$$

Also, False Positive Rate (FPR), False Alarm Rate (FAR) or Fall-out and False Negative Rate (FNR) or Miss rate are respectively given as

$$FPR = \frac{FP}{FP + TN} \quad (4.2)$$

$$FNR = \frac{FN}{TP + FN} \quad (4.3)$$

Also, the Precision or Positive Predictive Value (PPV) was obtained from

$$PPV = \frac{TP}{TP + FP} \quad (4.4)$$

and the F-Score which is the harmonic mean of precision and sensitivity was computed from

$$F\text{-}Score = \frac{2TP}{2TP + FP + FN} \quad (4.5)$$

Where FP = false positive, TN = true positive, FN = false negative and TP = true positive.

Table 4.4 Computation of the F-Score values and Precision of Lassa fever validation data

SN	Learning rate, η	Total Records or Symptoms	TP	FP	TN	Precision	Recall	F-Score
1	0.1	34	33	1	0	0.970588235	1	0.985075
2	0.2	34	33	1	0	0.970588235	1	0.985075
3	0.3	34	32	2	0	0.941176471	1	0.969697
4	0.4	34	32	2	0	0.941176471	1	0.969697
5	0.5	34	32	2	0	0.941176471	1	0.969697
6	0.6	34	31	3	0	0.911764706	1	0.953846
7	0.7	34	30	4	0	0.882352941	1	0.937500
8	0.8	34	30	4	0	0.882352941	1	0.937500
9	0.9	34	29	5	0	0.852941176	1	0.920635

Also, validational test carried out on EVD data showed the following results tabulated in Table 4.5.

Table 4.5 Computation of the F-Score values and Precision of EVD validation data.

SN	Learning rate, η	Total Records or Symptoms	TP	FP	TN	Precision	Recall	F-Score
1	0.1	1151	1136	15	0	0.986968	1	0.993441
2	0.2	1151	1134	17	0	0.985230	1	0.992560
3	0.3	1151	1132	19	0	0.983493	1	0.991678
4	0.4	1151	1064	87	0	0.924414	1	0.960722
5	0.5	1151	1060	91	0	0.920938	1	0.958842
6	0.6	1151	1002	149	0	0.870547	1	0.930794
7	0.7	1151	986	165	0	0.856646	1	0.922789
8	0.8	1151	957	194	0	0.831451	1	0.907970
9	0.9	1151	923	228	0	0.801911	1	0.890068

From above, the computation of the F-Score test for Lassa fever yielded a mean of 95.87% while that of the EVD cases yielded a mean of 95%. This results were expected since the data used for validation was real data known to sort patients manifesting such diseases with the same symptoms. However, a huge database is paramount for training and validating ANN as a good network is as powerful as the amount of training and validational data used in its design for a quaranteed testing regime. A tabulation of the averages of false and true positives and their corresponding negatives based on all investigated learning rate for the designed ANNis shown below. The Confusion matrix table which was obtained from the formulation table (Wikipedia, 2016) and shown in Appendix M consists of the training sets, validation sets and testing sets.Table 4.6 shows the relationship for the Target class and Output class for network sessions' performance and evaluation data on EVD.

Table 4.6 Confusion matrix table for the results of pattern classification

		Predicted class		
	N = 1151	True	False	
Actual Class	True	TP 1036	TN 0	Recall 1
	False	FN 0	FP 15	Fall-out 0
		Precision 0.986968	0	

Also, the confusion matrix of the result of training and validation on Lassa fever data also showed considerable precision and appreciable F-Score for the ANN iterations. The Lassa fever results for computation of the confusion matrix of the data is shown in Table 4.7.

Table 4.7 Confusion matrix table for the results of pattern classification for Lassa fever diagnostic data fed to designed ANN.

		Predicted class		
	N = 34	True	False	
Actual Class	True	FP 33	TN 0	Recall 1
	False	FN 0	FP 1	Fall-out 0
		Precision 0.985075	0	

4.6.6 Comparative Analysis Results of Some Medical Expert Systems

A comparative assessement of the performance characteristics of MYCIN, CASNET and the developed expert system was carried out in this study and the results shown in Table 4.8 below. The advantage of the use of the GUI approach for data capture from patients and outputting was observed from the table. Also, the possibility of identifying a new pattern/strains in presenting symptoms of diseases was also elaborated.

Table 4.8 Comparative analysis results of some medical expert systems

	Proposed Expert System	**MYCIN**	**CASNET**
Nature/Origin	Viral	Bacteria	Bacteria and Viral
Nature of Contact	Contagious	Non contagious	Non contagious
Operation Mode	Fuzzy logic reasoning/ANN	Rule base reasoning	Semantic Network Representation
Type of Data Input	Firing of fuzzy membership sets	Certainty factors	Most significant results of tests
Output display	Graphical User display	Teletext display	Teletext display
Detecting New Strain of Disease	Possible	Not possible	Not possible
Programme Host	Computer system	Computer system	Computer system

CHAPTER FIVE

CONCLUSION AND RECOMMENDATION

5.1 Conclusion

Supervised learning was used to provide ICT support to the detection and isolation of contagious diseases in real-time as a way of protecting healthcare providers and other patients in the event of an exposure to a disease of contagious nature. In the resulting Expert system, the fuzzy inferencing system provided the needed data filtering and enhancement to eliminate nuances and vagueness in patients reported symptoms. These fuzzified inputs then serves as training and validation/testing sets to the designed ANN in real-time. Outputs from the resulting fuzzy neural processing will aid healthcare providers to categorize patients with high susceptibility of contracting disease with contagious characteristics. This categorization which must be devoid of apathy towards the index patient (as many people have been known to survive such diseases) is however not a final analysis for the patient's state but just a suggestion which may be acted upon or ignored at the hospital's risk.

Since the results of computation of the ubiquitous Expert system can lead to apathy to patients and scare from healthcare providers, the result of the pre-diagnostic information should be made available to only properly trained personnel. Personnel manning the Expert system must be trained to be confidential with the result of processing as unnecessary flags or alarms could endanger a patient's life who may out of panic of contracting such a disease die of psychological trauma. Also, proper chain of communication must be established before deployment of the Expert system so that unnecessary panic is not generated. Outside this drawback of mismanagement of pre-diagnosed results, flags from the implementation of the Expert system should be taken seriously. The k-NN algorithm provides a storage and classification scheme to allow for classifying new cases. The advantage of the use of a non "black box" or apparent transparent training BPANN algorithm have shown its versatility in the control and management of patient fuzzy data and diagnosis with minimal contribution from human professionals in real-time. Also, an investigation of various topologies that included the one hidden layer topology and two hidden layer topology. The three-nodes one hidden layer topology was chosen for further use in the developed medical Expert system mainly due to speed, appreciable accuracy and good generalizability. The two hidden layer topology which provided features of deeper learning was however not chosen for further analysis due to the large iterations (poor speed) involved in its processing.

In this research, the window period of disease spread especially when index cases become available in the public domain(in hospitals and clinics) can be reduced considerably to a few minutes by the deployment of this research's diagnostic Expert system which is based on the machine learning capabilities of fuzzy processing and artificial neural network. The Fuzzy neural expert system can help protect healthcare providers and others by pre-diagnosing or sorting out patients for emergency consultation or quarantining even before proper diagnosis is carried out in the hospital or clinic it is being deployed.

5.2 Recommendation

The following recommendations were generated as a result of this research:

I. Deployment of standalone interface that runs the designed Expert system in clinics and hospitals. By this deployment, a nurse on duty could administer the diagnostic system and provide a clue to the nature of casualty at the hospital's hands and also provide a quick response aimed at saving a life or providing protection to others.

II. Development of huge data mining algorithms to make sense of increasing data collected by ICT tools. As advancement in database collation systems, feedback systems and increasing memory capacities proliferate, it would be necessary to create algorithms that could mine these increasing avalanche of data and make sense of them to aid healthcare. This will eventually make data miners depend less on nonlinear equations for modeling and problem solving.

III. In areas with high epidemic characteristics for contagious diseases such as the ones investigated in this research, emergency laparotomy or appendicectomy to get to the root cause of ailment should be done with caution and more advance protection schemes as it was observed that some surgeons became exposed to noscomial infections during surgery.

IV. For nonlinear identification tasks involving complex membership functions, the deeper learning provided for by the two hidden layer ANN investigated topology should be used in optimizing such problems with appreciable speed for gradient descent obtained when learning rate is equal to 0.3.

V. The developed Expert system can be modified for use in other areas of engineering analysis such as data mining tasks in image processing where pixel values of images can serve as training inputs to the ANN; security surveillance where events can be modeled and predicted by training the ANN to recognize certain events pattern, etc.

5.3 Contribution to Knowledge

In the novel research work, the following contributions to the body of knowledge have been brought afore:

I. The window period that involves contagious disease appearing in public through a reservoir or host to the time it becomes a full blown epidemic can be reduced to a few minutes by the dependence on fuzzy neural processing to mitigate the period.

II. The development of a completely connected artificial neural network with thirty inputs, three hidden layers for optimization and five outputs in visual studio programming language is also a contribution to the body of knowledge.

III. The pre-fuzzified inputs from the fuzzy inference engine that serves as training inputs into the ANN algorithm in the expert system is also discovered by this research especially in the field of medical diagnosis.

IV. The k-nearest neighbor algorithm has also been used in this research to elucidate clustering and neighborhood membership of diagnosed data with respect to previous data.

V. This research also provides an interactive and transparent model though tailored to analyze a specific data range yet very flexible to data manipulation and normalized input data ranges.

VI. Deeper learning features was accomplished with the two hidden layer ANN's topology when implemented with a reciprocal function on the second hidden layer iteration processing to improve the iterated values.

REFERENCES

Abraham, A. (2002). Intelligent Systems: Architectures and Perspectives, Recent Advances in Intelligent Paradigms and Applications. (A. Abraham, L. Jain, & J. Kacprzyk, Eds.) *Studies in Fuzziness and Soft Computing,* 1(1), 1 – 35.

Abraham, A. (2005). Adaptation of Fuzzy Inference System Using Neural Learning.*StudFuzz*181, 53 – 83.

Achebe, C.H. (2010). Human Immunodeficiency Virus (HIV) – Blood Interactions: Surface Thermodynamics Approach. *A Ph.D. dissertation presented to the Department of Mechanical Engineering, Nnamdi Azikiwe University.* Awka, Nigeria.

Agbo, A.E.,& Nwodoh, T.A. (2011). An Expert System for Diagnosis and Treatment of Tropical Diseases. *International Journal of Electrical & Telecommunication System Research*, 5(5), 29 – 37.

Aghajan, H., Delgado, R. L., & Augusto, J. C. (2010). *Human-Centric Interfaces for Ambient Intelligence.* Elsevier Inc.

Ajayi, N.A., Nwigwe, C.G., Azuogu, B.N., Onyire, B.N., Nwonwu, E.U., Ogbonnaya, L.U., Onwe, F.I., Ekaete T., Gunter, S.,& Ukwaja, K.N. (2013). Containing a Lassa Fever Epidemic in a Resource-Limited Setting: Outbreak Description and Lessons Learned from Abakaliki, Nigeria (January – March 2012). *International Journal of Infectious Diseases,* 17(11), 1011 – 1016.

Ajmalahamed, A., Nandhini, K.M. & Anand, K.S (2014). Designing a Rule Based Fuzzy Expert Controller for Early Detection and Diagnosis of Diabetes. *ARPN Journal of Engineering and Applied Sciences*, 9(5), 819 – 827.

Appah, A.,& Afrane, S. (2014). Queuing Theory and the Management of Waiting-Time in Hospitals: The Case of Anglo Gold Ashanti Hospital in Ghana. *International Journal of Academic Research in Business and Social Sciences*, 4(2), 34 – 44.

Ardil, E.,& Sandhu, P.S. (2010). A Soft Computing Approach for Modeling of Severity of Faults in Software Systems. *Int'l Journal of Physical Sciences,* 5(2), 74–85.

Armstrong, H. (2015). *Machines that Learn in the Wild: Machine Learning Capabilities, Limitations and Implications.* London: Nesta.

Asabere, N.Y. (2012). mMES: A Mobile Medical Expert System for Health Institutions in Ghana. *International Journal of Science and Technology,* 2(6), 333 – 344.

Avci, D.& Dogantekin, A. (2016). An Expert Diagnosis System for Parkinson Disease Based onGenetic Algorithm-Wavelet Kernel-Extreme Learning Machine. *Hindawi Publishing Corporation*, 5264743, 1 – 9. http://dx.doi.org/10.1155/2016/5264743

Ayyub, B.M. (2003). *Risk Analysis in Engineering and Economics*. Chapman and Hall/CRC Press.

Babuska, R., & Verbruggen, H. (2003). Neuro – Fuzzy Methods for Nonlinear System Identification. *Elsevier Science Ltd; Annual Reviews in Control, 27*(1), 73 – 85.

Babuska, R. (2002). Neuro – Fuzzy Methods for Modeling and Identification. *Advances in Intelligent Paradigms and Applications*, pp. 161-186.

Badiru, B.B., Asaolu, O.S.,& Omitaomu, A.O. (2006). Eyewitness Information Management System using Neuro – Fuzzy Classification Schemes. *Journal of Information Science and Technology (JIST),* 2(3), 49 – 64.

Badri, B. (2010). Development of Neural Networks for Noise Reduction. *The International Arab Journal of Information Technology*, 7(3), 289 – 294.

Barron, A. R. (1993). Universal Approximation Bounds for Superpositions of a Sigmoidal Function. *IEEE Transaction on Information Theory*, 39(1) 930–945.

Basheer, I.A.,& Hajmeer, M. (2000). Artificial Neural Network: Fundamentals, Computing, Design and Application. *Elsevier Journal of Microbiological Methods*, 43(1), 3 – 31.

Bernhard Nocht Institute for Tropical Medicine. (2014). *Recommendations Regarding Malaria Testing and Interpretation of Malaria Testing Results in Patients Who Returned from Ebola Outbreak-Affected Countries Within the Past 21 Days*. Retrieved November 29th, 2014, from http://www.rki.de/EN/ Content/Prevention/Ebola_virus_disease/malaria-testing_Ebola-outbreak.pdf

BetterHealth. (2015). *Symptoms of Fever*. Retrieved July 18th, 2015, from m.betterhealth.vic.gov.au/bhcv2/bhcarticles.nsf/mskpages/fever?open

Bhadeshia, H.K. (1999). Neural Networks in Materials Science. *ISIJ International*. 39(10), 966 – 979.

Blancou, J. (1996). *Early Methods of Surveillance and Control for Contagious Bovine Pleuropneumonia*. Retrieved October 17th, 2014, from http://www.oie.int/doc/ged/d9097.pdf

Body and Health Home. (2015). *Malaria: Tropical Disease, Parasitic Infection*. Retrieved February 7th, 2015, from http://bodyandhealth.canada.com/channel_condition_info_details.asp?channel_id=1020&disease_id=85

Bourouis, A., Feham, M.,& Bouchachia, A. (2014). *A New Architecture of a Ubiquitous Health Monitoring System: A Prototype of Cloud Mobile Health Monitoring System*. Retrieved February 12th, 2015,from http://arxiv.org/ftp/arxiv/papers/1205/1205.6910.pdf

Bray, M.,& Chertow, D.S. (2014). *Clinical Manifestations and Diagnosis of Ebola Virus Disease*. Retrieved November 27th, 2014, from http://www.uptodate.com/contents/clinical-manifestations-and-diagnosis-of-ebola-virus-disease

Brunton, S.L., Proctor, J.L.,& Kutz, J.N. (2016). Discovering Governing Equations from Data by Sparse Identification of Nonlinear Dynamical Systems. *PNAS*, 113(15), 3932 – 3937.

Burns, R. S. (2001). *Advanced Control Engineering.* Oxford, Great Britain: Butterworth-Heinemann.

Callan, R. (1999). *The Essence of Neural Networks: Essence of Computing*. Hertfordshire, Great Britain: Prentice Hall Europe.

Carpenter, G. A., & Grossberg, S. (1998). The ART of Adaptive Recognition by a Self-Organizing Neural Network. *Computer March*, pp. 77 – 88.

Castillo, E., Berdinas, B., Romero, O.,& Betanzos, A. (2006). A Very Fast Learning Method for Neural Networks Based on Sensitivity Analysis. *Journal of Machine Learning Research*, 7(1), 1159 – 1182.

Centers for Disease Control and Prevention. (2015). *2014 Ebola Outbreak in West Africa*. Retrieved May 1st, 2015, from www.cdc.gov/vhf/ebola/outbreaks/2014-west-africa/?mobile=nocontent

Cho H. (2012). *Health Communication Message Design: Theory and Practice*. Oaks, California, USA: Sage Publications, Inc.

CNN. (2014). *Hospital 'dropped the ball' with Ebola patient's travel history.* Retrieved October 2nd, 2014, from www.cnn.com/2014/10/01/health /ebola-us/

Cutler, S.J., Fooks, A.R.,& Poel, W.H. (2010). Public Health Threat of New, Reemerging, and Neglected Zoonoses in the Industrialized World. *Emerging Infectious Diseases.*1(16), 1 – 7.

Davis, N., & LaCour, M. (2007). *Health Information Technology*. USA: Elsevier Inc.

Demuth, H.,& Beale, M. (2002). *Neural Network Toolbox-for Use with MATLAB User's Guide.* Massachusetts, USA: The Mathworks.

Dongo, E.A., Kesieme, E.B., Christopher, E.I., Okokhere, P.O., Odigie, C. A.,& Akpede, O.G. (2013). Lassa Fever Presenting as Acute Abdomen: A Case Series. *Virology Journal*, 10(123), 1 – 7.

Dorf, C.D.,& Bishop, H.R. (1998). *Modern Control System*. USA: Addison Wesley Longman Inc.

Downs, E., Clare, P.,& Coe, I. (1992). *Structured Systems Analysis and Design Method*(2nd ed.). United Kingdom: Prentice Hall International Ltd.

Dugas, A.F., Hsieh, Y., Levin, S.R., Pines, J.M., Mareiniss, D.P., Mohareb, A., Gaydos, C.A., Perl, T.M.,& Rothman, R.E. (2012). Google Flu Trends: Correlation with Emergency Department Influenza Rates and Crowding Metrics. *Clinical Infectious Diseases,*54(4), 463 – 469.

Fawcett, T. (2003). *ROC Graphs: Notes and Practical Considerations for Data Mining Researchers.*USA: Hewlett-Packard Company.

Fawcett, T.(2005). An Introduction to ROC Analysis. *ElsevierPattern Recognition Letters*, 27(8), 861 – 874.

Fletcher, R. (1987). *Practical Methods of Optimization*(2nd ed.).New Jersey, USA: John Wiley & Sons, Ltd.

Friedl, L.,& Ceccato, P. (2010). *Critical Earth Observations Priorities: Human Health Infectious Diseases.* SBA Report.

Funahashi, K. (1989). On the Approximate Realization of Continuous Mappings by Neural Networks. *Neural Networks*, 2, 183–192.

Gaebel, J., Cypko, M. A., & Lemke, H. U. (2016). Accessing Patient Information for Probabilistic Patient Models Using Existing Standards. In G. Schreier (Ed.), *Proceedings of the 10th eHealth2016 Conference.223*, pp. 107 - 112. Health Informatics Meets eHealth: Predictive Modeling in Healthcare - From Prediction to Prevention.

Galushkin, A.I. (2007). *Neural Networks Theory*. Heidelberg, Berlin, Germany: Springer-Verlag.

Gao, X.R. (2003). *Neural Networks for Machine Condition Monitoring and Fault Diagnosis.* Retrieved July 8th, 2014, from Department of Mechanical and Industrial

Engineering, University of Massachusetts, USA: http://crema.di.unimi.it/fscott/nn/8-gao-formatted.pdf

Global Observatory for eHealth series - Volume 6, (2012). *Management of Patient Information: Trends and Challenges in Member States.* RetrievedApril 28th, 2015, from http://www.who.int/iris/bitstream/10665/76794/1/9789241504645_eng.pdf

Gopal, M. (2009). *Digital Control and State Variable Methods: Conventional and Intelligent Control Systems* (3rd ed.). Tata McGraw Hill Education Private Limited.

Hagan, M.T, Demuth, H.B, Beale, M.H.,& DeJesus, O. (2002). An Introduction to the Use of Neural Networks in Control Systems. *International Journal of Robust and Nonlinear Control,* 12, 959 – 985.

Hagan, M.T, Demuth, H.B, Beale, M.H.,& DeJesus, O. (2013). *Neural Network Design*(2nd ed.).

Hagan, M.T, Demuth, H.B.,& Beale, M.H. (1996). *Neural Network Design*. Boston, USA: PWS Publishing Company.

Hagan, M.T.,& Menhaj, M. (1999). Training Feed-Forward Networks with the Marquardt Algorithm. *IEEE Transactions on Neural Networks,* 5(6), 989 – 993.

Halawa, K. (2008). Determining the Weights of a Fourier Series Neural Network on the Basis of the Multidimensional Discrete Fourier Transform.*Intl. Journal of Applied Mathematics and Computer Science.* 18(3), 369 –375.

Hand, D., Mannila, H.,& Smyth, P. (2001). *Principles of Data Mining.* Cambridge, MA,Great Britain: The MIT Press.

Harvey, K.,& Koteyko, N. (2013). *Exploring Health Communication Language in Action*. New York, USA: Routledge Publishing.

Hassoun, M.H. (1995). *Fundamentals of Artificial Neural Networks.*Cambridge, MA, Great Britain: MIT Press.

Haykin, S. (1994). *Neural Networks: A Comprehensive Foundation.* New York, USA: Maccillan.

Hays, D. G.,& Singh, A. A. (2012). *Qualitative Inquiry in Clinical and Educational Settings*. New York, USA: Guilford Publications, Inc.

Hopfield, J.J. (1984). Neurons with Graded Response have Collective Computational Properties like Those of Two-state Neurons. *Proceedings of National Academy of Science*, 81, 3088 – 3092.

Hudson, D.L., & Cohen, M.E. (1992). The Role of Approximate Reasoning in a Medical Expert System. (A. Kandel, Ed.) *Fuzzy Expert Systems,* pp. 165 – 179.

Isinkaye, F.O., Awosupin, S.O. & Soyemi, J. (2017). A Mobile based expert system for disease diagnosis and medical advice provisioning. *International Journal of Computer Science and Information Security (IJCSIS)*,15(1), 5149 – 5158.

Iwasokun, G. B., Egwuche, O. S.,& Gabriel, J. A. (2015). Neural Network-Based Health Personnel Monitoring System.*African Jour. of Computing & ICTs*, 8(1), 79 – 87.

Jakeman, A.J.; Voinov, A.A.; Rizzoli, A.E.,& Chen, S. H. (2008). *Environmental Modelling, Software and Decision Support: State of the Art and New Perspectives*. Elsevier Publishing.

Kavulya, S.P., Joshi K., Giandomenico, F.D.,& Narasimhan, P. (2012). *Failure Diagnosis of Complex Systems*. Springer Verlag.

Kirk, D.B.,& Hwu, W.W.H. (2010). *Programming Massively Parallel Processors: A Hands-on Approach*. Elsevier Inc.

Kohenen, T. (1989). *Self-organization and Associative Memory*,(3rd Ed.). New York, USA: Springer.

Kumar, R., Madhura, P., Babu, R., Ramesh, P., & Padmavathamma, M. (2013). Medical Diagnosis Expert System as Service in Cloud. *International Journal of Computer and Communication Engineering*, 2(4), 390 – 392

KPJ Healthcare. (2015). *Admission process Flow Chart*. Retrieved April 24th, 2015, from www.ksh.kpjhealth.com.my/admission-discharge.php

Krishnamoorthy, C.S.,& Rajeev, S. (1996). *Artificial Intelligence and Expert Systems for Engineers*. CRC Press LLC.

Krose, B.,& Smagt, P. (1996). *An Introduction to Neural Network*(8th Ed.). Netherlands:The University of Amsterdam.

Kulkarni, A. D.,& Lulla, K. (1999). Fuzzy Neural Network Based Logic Models for Supervised Classification: Multispectral Image Analysis. *Geocarto International,Hong Kong*, 14(4), 43 – 50.

Kumar, D.S., Sathyadevi, G.,& Sivanesh, S. (2011). Decision Support System for Medical Diagnosis Using Data Mining.*International Journal of Computer Science Issues*, 8(3), 147 – 153.

Larose, D.T. (2005). *Discovering Knowledge in Data: An Introduction to Data Mining*. New Jersey, USA: John Wiley & Sons, Inc.

Larose, D.T. (2006). *Data Mining Methods and Models*.New Jersey, USA: John Wiley & Sons, Inc.

Lee, B., Haidari, L.,& Lee, M. (2013). Modelling during an Emergency: The 2009 H1N1 Influenza Pandemic. *Clinical Microbiology and Infection*, 19(11), 1014 –1022.

Lee, B.W., Hsu, S.I.,& Stasior, D.S. (1997). *Quick Consult Manual Evidence-Based Medicine*. New York, USA: Lippincott-Raven Publishers.

Lee, Y.C. (2006). Adaptive Control of a Class of Nonlinear Systems Using Multiple Parameter Models. *International Journal of Control, Automation, and Systems*, 4(4), 428 – 437.

Leondes, C.T. (2000). *Knowledge – Based Systems: Techniques and Applications*. USA: Academic press.

Madaan, V.& Garg, A. (2016). DSSBD: Fuzzy Rule Based Decision Support System for Blood Diseases Diagnosis. *International Journal of Control Theory and Applications* (*IJCTA*), 9(11), 5149 –5158.

Mamdani, E.H. &Assilian, S. (1975). An Experiment in Linguistic Synthesis with aFuzzy Logic Controller.*International Journal of Man-Machine Studies*, 7(1) 1–13.

Manheim, D., Chamberlin, M., Osoba, O.A., Vardavas, R. & Moore, M. (2016). Improving Decision Support for Infectious Disease Prevention and Control: Aligning Models and other Tools with Policymakers' Needs, California, USA: RAND Corporation.

McHugh, M., Dyke, V., McClelland, M.,& Moss, D. (2011). Improving Patient Flow and Reducing Emergency Department Crowding: A Guide for Hospitals. AHRQ Publication.

MedGuidance. (2015). *Normal Body Temperature*. Retrieved June 24th, 2015, from www.medguidance.com/thread/Normal-Body-Temperature.

MedicineNet. (2015). *Marburg Virus Disease History Symptoms and Treatment.*RetrievedFebruary 7th, 2015, from http://www.medicinenet.com/marburg_virus_history_symptoms_and_treatment/views.htm

Mitchell, M.T. (1997). *Machine Learning*. USA: McGraw-Hill Science/Engineering/Math Publishing.

Morse, S.S. (1995). Factors in the Emergence of Infectious Diseases. *Emerging Infectious Diseases,* 1, 7 – 15.

Muyembe, J.J.T, Mulangu, S., Masumu, J., Kayembe, J.M., Kemp, A.,& Paweska, J.T. (2012). Ebola virus outbreaks in Africa: Past and present. *Onderstepoort Journal of Veterinary Research,* 79(2), 1 – 8.

Nauck, D. (1997). Neuro-Fuzzy Systems: Review and Prospects. *Proceedings of the fifth European Congress on Intelligent Techniques and Soft Computing, EUFIT'97,*Aachen.

Nayak, P.C., Sudheer, K.P., Rangan, D.M.,& Ramasastri, K.S. (2004). A Neuro-Fuzzy Computing Technique for Modeling Hydrological Time Series. *Elsevier B.V; Journal of Hydrology*, 291(1), 52 – 66.

Naser, S.A. & Hamed, M.A. (2016). An Expert System for Mouth Problems in Infants and Children. *Journal of Multidisciplinary Engineering Science Studies* (JMESS), 2(4), 468 – 476.

Nelson, E.C. (2012). *Using Patient-Reported Information to Improve Health Outcomes and Health Care Value: Case Studies from Dartmouth, Karolinska and Group Health*. Retrieved April 28th, 2015, from http://tdi.dartmouth.edu/images/uploads/tdi_tr_pri_ia_sm.pdf

Nigel, K., Dennis, W.K.C., Roger, C.L., John, E.W.,& David, R. (1992). *Diagnosis and Management of Melanoma in Clinical Practice*). London, UK: Springer – Verlag.

Nise, N.S. (1995). *Control System Engineering*(2nd Ed.). USA: Addison – Wesley Publishing

Office of the Special Adviser to the President on Research, Documentation and Strategy. (2014). *Ebola: How Nigerians Conquered Adversity with Unity*. Abuja.

Osigbemeh, M.S., Ogunwolu, F.O., Omoare, A.A, Inyiama, H.C. (2014). A Linguistic Fuzzy Expert System for Contagious Diseases Detection and Isolation". *UNILAG Journal of Medicine, Science & Technology (UJMST),*2(1 & 2), 1 – 10.

Paraskevopoulos, P.N. (2002). *Modern Control Engineering*. New York, USA: Mercel Dekker, Inc.

Peek, N., Holmes, J.,& Sun, J. (2014). Technical Challenges for Big Data in Biomedicine & Health: Data Sources, Infrastructure, and Analytics. *IMIA Yearbook,* 9(1), 42 – 47

Pham, D. T. (1994). Neural Networks in Engineering. In G. Rzevski et al (Ed.), *Applications of Artificial Intelligence in Engineering IX, AIENG* (pp. 3-36). Southeamptom: Proc. of the 9th International Conf. on Computational Mechanics Publications.

Pierleoni, P., Pernini, L., Belli, A. & Palma, L. (2014). An Android-Based Heart Monitoring System for the Elderly and for Patients with Heart Disease. *Hindawi Publishing*

*CorporationInternational Journal of Telemedicine and Applications. ID*625156, 1 – 11. http://dx.doi.org/10.1155/2014/625156

Pislaru, M., Trandabat, A.F.,& Schreiner, C. (2006). Neuro-Fuzzy Surveillance for Industrial Process Fault Detection.*8th International Conference on Development and Application Systems.* Suceava, Romania.

Planet Health. (2014). *Transmission, Symptoms of Lassa Fever Similar to Ebola (EVD).* Retrieved September 16th, 2014, from http://planehealth.blogspot.com/2014/08/transmission-symptoms-of-lassa-fever.html

Powers, D.M. (2011).Evaluation: From Precision, Recall and F-Measure to ROC, Informedness, Markedness & Correlation. *Journal of Machine Learning Technologies,* 2(1), 37 – 63.

Prasadl, B., Prasad, P.K.,& Sagar, Y. (2011). An Approach to Develop Expert Systems in Medical Diagnosis using Machine Learning Algorithms and a Performance Study.*International Journal on Soft Computing (IJSC),* 2(1), 23 – 33.

Rad, A.A., Khadivi, A.,& Hasler, M. (2010). Information Processing in Complex Networks. *IEEE Circuits and Systems Magazine,* 26 – 37. DOI 10.1109/mcas.2010.947881

Ramadan, M.,& Al-Saleh, K. (2013). Development of an Expert System for Reducing Medical Errors. *International Journal of Software Engineering & Applications (IJSEA),* 4(6), 29 – 38.

Reamon, D.T.,& Sheppard, S.D. (1997). The Role of Simulation Software in an Ideal Learning Environment. *Proceedings of DETC'97 ASME Design Engineering Technical Conference.*Sacramento, California.

Reddy, K.S. (2003). *Prevention and Control of Non-Communicable Diseases: Status and Strategies*. Retrieved July 2nd, 2014, from www.icrier.org/pdf/wp104.pdf

Rudin, C.,& Wagstaff, K.L. (2013). *Machine Learning for Science and Society*. Springer. DOI 10.1007/s10994-013-5425-9

Rumelhart, D.,& McClelland, J. (1986). *Parallel Distributed Processing*. Cambridge, Great Britain: MIT Press

Saha, B. (2014). Green Computing. *International Journal of Computer Trends and Technology (IJCTT),* 14(2), 46 – 50.

Schneider, M., & Kandel, A. (1992). General Purpose Fuzzy Expert Systems. (A. Kandel, Ed.) *Fuzzy Expert Systems,* pp. 23 – 41.

Schneider, M., Perl, J.M.,& Kandel, A. (1992). Comex – An Autonomous Fuzzy Expert System for Tactical Communications Networks. (A. Kandel, Ed.) *Fuzzy Expert Systems*, pp 291 – 304.

Sebe, N., Cohen, I., Garg, A.,& Huang, T.S. (2005); *Machine Learning in Computer Vision: Computational Imaging & Vision*. Dordrecht, Netherlands: Springer

Serrano, I.S. (2011). *The World's Health Care Crisis*. USA: Elsevier Inc.

Sharma, C. & Choudhary, T. (2015). A Web Based Fuzzy Expert System for Epistaxis Diagnosis. *International Journal of Computer Science and Information Technologies*, 6(4), 4062 – 4068.

Sharma, G. (2007). *Data Mining, Data warehousing and OLAP*. New Delhi, India: S.K. Kataria and Sons.

Shortliffe, E.H. (1976). *Computer-Based Medical Consultation: MYCIN*. North-Holland, Netherlands: Elsevier Publishing.

Simpson, K. P. (1992). *Foundations of neural networks in artificial neural networks* (eds) Edgar Sanchez-Sinencio, Clifford Lau, IEEE Press New York. 3 – 24.

Singh, S., Khadamkar, P., Kumar, M. & Maramwar, V. (2014). Healthcare Services Using Android Devices. *The International Journal of Engineering and Science (IJES)*, 3(4), 41 – 45.

Singla, J., Grover, D. & Bhandari, A. (2014). Medical Expert Systems for Diagnosis of various Diseases. *International Journal of Computer Applications, 93(7), 36 – 43.*

Sloan, P., Legrand, W.,&Chen, J.S. (2013). *Sustainability in the Hospitality Industry: Principles of Sustainable Operations*(2nd Ed.). New York, USA: NY Publishing.

Stepnowski, A., Moszynski, M.,& Dung, T.V. (2003). Adaptive Neuro-Fuzzy Decision Tree Classifier as Applied to Seafloor Characterization. *Journal of Acoustic Physics,* 49(2), 193 – 202.

Su, Huang, Wu & Zhang (2006). A Logical Framework for Identifying Quality Knowledge from Gifferent Gata sources. *Decision Support Systems*, 42(3), 1673 – 1683.

Suhir, E. (1997). *Applied Probability for Engineers and Scientists*. USA: McGraw Hill.

Suzuki, K., Shiraishi, J., Abe, H., MacMahon, H.,& Doi, K. (2005). False-positive Reduction in Computer-aided Diagnostic Scheme for Detecting Nodules in Chest Radiographs by Means of Massive Training Artificial Neural Network.*Academic Radiology*, 12(2), 191 – 201.

Swets, J. (1988). Measuring the Accuracy of Diagnostic Systems. *Science*, 240, 1285 – 1293.

Swets, J. A., Dawes, R. M.,& Monahan, J. (2000). Better Decisions Through Science. *Scientific American*, 283, 82 – 87.

The Open University. (2006). *Modelling Uncertainty*. Walton Hall, Hilton Keynes, UK: The Open University.

Theodoridis, S., Pikrakis, A., Koutroumbas, K.,& Cavouras, D. (2010). *Introduction to Pattern Recognition: A Matlab Approach*. Kidlinton, Oxford, UK: Elsevier Inc.

Thibodeau, G.A.,& Patton, K.T. (2010). The Human Body in Health and Disease (5th Ed.). Mosby: Elsevier Inc.

Thomas G. T. (2016). Interpreting Diagnostic Tests: Plotting and Intrepretating an ROC Curve. Retrieved June 4th, 2016, from http://gim.unmc.edu/dxtests/roc2.htm

Topping, B.H., Sziveri, J., Bahreinejad, A., Leite, J.P.,& Cheng, B. (1998). Parallel Processing, Neural Networks and Genetic Algorithms. *Elsevier Science Ltd: Advances in Engineering Software*, 29(10), 763 – 786.

Van, R.A., Jain, L.,& Johnson, R. (1996). *Neural Network Training Using Genetic Algorithms*. Singapore: World Scientific.

Viharos, J. Z., & Kis, B. K. (2014). Survey on Neuro-Fuzzy Systems and their Applications in Technical Diagnostics. *Advanced measurement tools in technical diagnostics for systems' reliability and safety*. Warsaw, Poland: 13th IMEKO TC10 Workshop on Technical Diagnostics.

Vrbova, L., Stephen, C., Kasman, N., Boehnke, R., Gibson, B., Brauer, M.,& Patrick D. (2009). *Systematic Review of Surveillance Systems for Emerging Zoonotic Diseases.*Retrieved July 25th, 2014, from www.ncceh.ca/sites/default/files/Zoonoses_Surveillance_ May_2009.pdf

WHO Ebola Response Team. (2014). Ebola Virus Disease in West Africa – The First 9 Months of the Epidemic and Forward Projections.*The New England Journal of Medicine*.DOI:10.1056/NEJMoa1411100

Wikipedia. (2015). *Artificial Neural Network.* Retrieved on February 2nd, 2015, from https://en.wikipedia.org/wiki/Artificial_neural_network.

Wikipedia. (2015). *Expert system*. Retrieved February 24th, 2015, from https://en.wikipedia.org/wiki/Expert_system#cite_note-38

Wikipedia. (2015). *K-Nearest Neighbor Algorithm*. Retrieved April 17th, 2015, from https://en.wikipedia.org/wiki/k-nearest_neighbor_ algorithm

Wikipedia. (2016). *Confusion matrix*. Retrieved on April 28th, 2016, from http://en.wikipedia.org/wiki/Confusion_matrix

World Health Organization. (2006). *Communicable Disease Surveillance and Response Systems: Guide to monitoring and evaluating*.Retrieved October 3rd, 2014,

from http://www.who.int/csr/resources/publications/surveillance/WHO_CDS_EPR _LYO_2006_2.pdf

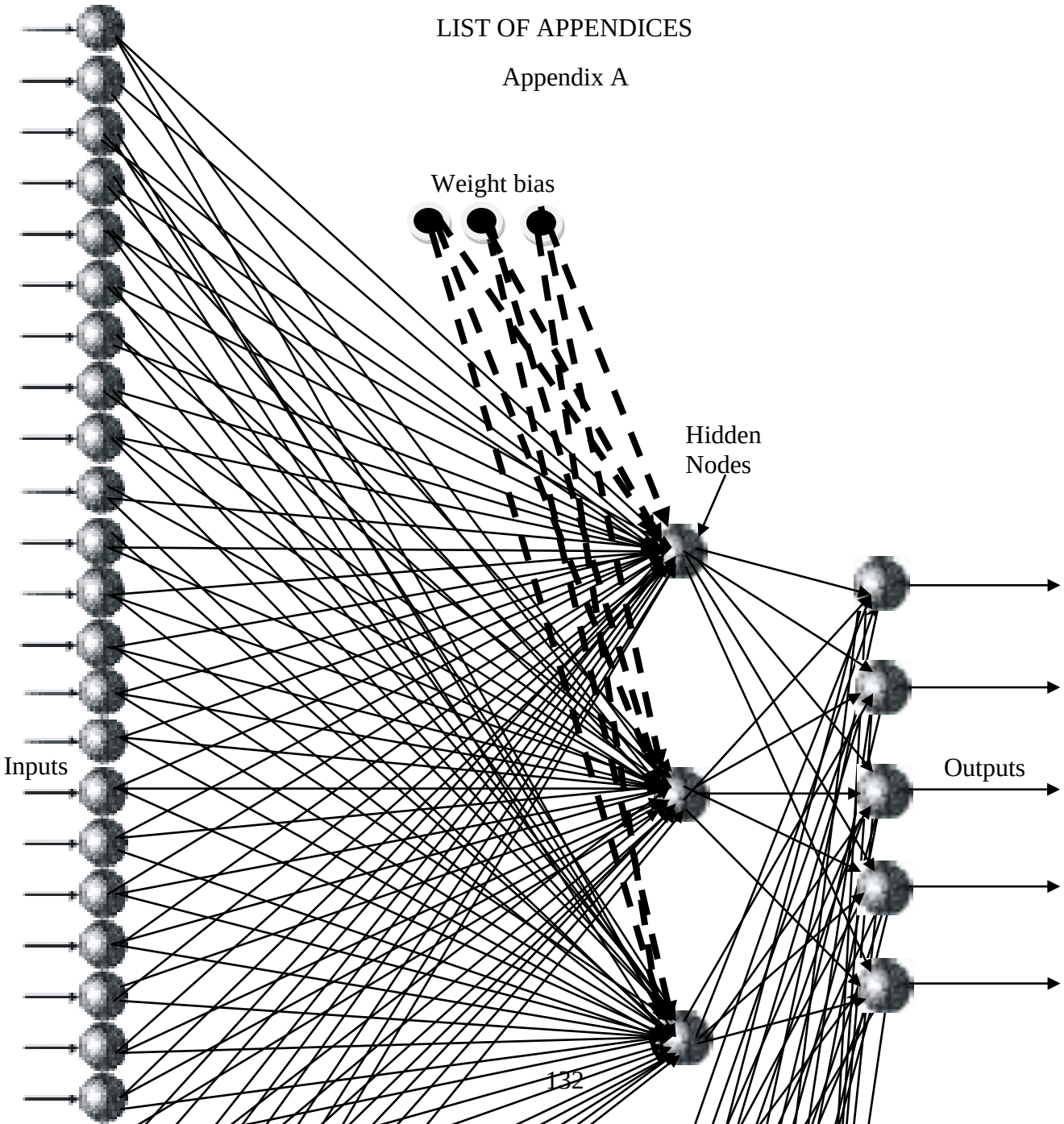
Yegnanarayana, B. (1994). Artificial Neural Networks for Pattern Recognition.*Sadhana,* 19(2), 189 – 238.

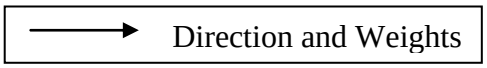
Yegnanarayana, B. (1999). *Artificial Neural Networks*. India: PHI Learning Private Limited.

Zhou, Z., Chawla, N.V., Jin, Y.,& Williams, G.J. (2015). Big Data Opportunities and Challenges: Discussions from Data Analytics Perspectives. *IEEE Computational Intelligence Magazine.* 9(4), 62 – 73.

LIST OF APPENDICES

Appendix A




Direction and Weights

Appendix B

Table 5.1 Documented outbreaks of Ebola virus amongst non-human primates and swine (1980–2005).

Source:(Muyembe et al., 2012)

Year	Non-human primates or swine	Location	Comments
1989–1990	Macaques	Reston, VA, USA	Discovery of a new specie(REBOV)
1992	Macaques	Sienna, Italy	
1996	Macaques	Alice, TX, USA	
2009	Swine	Philippines	
1994	Chimpanzees	Tai Forest, Côte dôlvoire	Discovery of a new specie(CIEBOV)
1996	Chimpanzees	Mayibout 2, Gabon	Ebola Zaire (ZEBOV)
2001–2002	Gorillas	Mendemba, Gabon	Ebola Zaire (ZEBOV)
2003	Gorillas	Mbomo, Republic of the Congo	Ebola Zaire (ZEBOV)
2005	Gorillas	Mbomo-Kellé, Republic of the Congo	Ebola Zaire (ZEBOV)
		Etumbi, Republic of the Congo	Ebola Zaire (ZEBOV)

Appendix C

Table 5.2 Pre-fuzzified data from the Institute of Lassa Fever Research and Control at Irrua Specialist Teaching Hospital, Edo State.

Source: Dongo et al., (2013)

	Age	Sex	Fever	Temp (^{o}C)	Abd. Pain	Chills/ Rigors	Malaria/ Typhoid	Locale	Appearance	Vomiting	Bleeding	Append-icetomy	Status	Ribavirin
1	28	M	High	38.9	Present	Present	Present	NA	Headache	Present	Prolonged	Yes	Recovered	Yes
2	27	F	High	39	Present	Present	Present	Infested	Headache	Present	Prolonged	Yes	Recovered	Yes
3	16	M	Mild	39	Present	Present	Present	NA	Weak&Febrile	Present	Prolonged	Yes	Died	No
4	<2	M	High	38.1	Present	Present	Present	NA	Pale+Bl.Stool	Present	Prolonged	Yes	Died	No
5	25	F	High	38.3	Present	Present	Present	NA	Pregnant	Present	Continued	Yes	Died	Yes
6	40	M	Mild	39.2	Present	Present	Present	NA	Weak	Present	Prolonged	Yes	Died	Yes
7	24	F	High	39.2	Present	Present	Present	Endermic	Ill & Pale	Present	normal	Yes	Recovered	Yes
8	Adult	NA	High	high	Present	Present	Present	Endermic	Weak&Febrile	Present	NA	NIL	Recovered	Yes
9	Adult	NA	High	high	Present	Present	Present	Endermic	Weak&Febrile	Present	NA	NIL	Recovered	Yes
10	Adult	NA	High	high	Present	Present	Present	Endermic	Weak&Febrile	Present	NA	NIL	Died	Yes
11	Adult	NA	High	high	Present	Present	Present	Endermic	Weak&Febrile	Present	NA	NIL	Recovered	Yes
12	Adult	NA	High	high	Present	Present	Present	Endermic	Weak&Febrile	Present	NA	NIL	Recovered	Yes
13	Adult	NA	High	high	Present	Present	Present	Endermic	Weak&Febrile	Present	NA	NIL	Recovered	Yes
14	Adult	NA	High	high	Present	Present	Present	Endermic	Weak&Febrile	Present	NA	NIL	Recovered	Yes

Key: Bl.Stool – Bloody Stool; Abd. Pain – Abdominal Pain.

Appendix D

Table 5.3 Pre-fuzzified data on clinical presentation of Lassa fever cases at Federal Teaching Hospital Abakailiki, Nigeria.

Source: Ajayi et al. (2013).

	Patient	**Presented Symptoms**												
1	A	Fever	H.Temp	S.Throat	Abd. pain	Headache	Vom.				B.weakn.		Sp.abortion	Locale
2	B	Fever	H.Temp	S.Throat	Abd. pain	Headache	Vom.				B.weakn.			Locale
3	C	Fever	H.Temp	S.Throat	Abd. pain	Headache	Vom.				B.weakn.			Locale
4	D	Fever	H.Temp	S.Throat	Abd. pain	Headache	Vom.				B.weakn.			Locale
5	E	Fever	H.Temp	S.Throat	Abd. pain	Headache	Vom.				B.weakn.			Locale
6	F	Fever	H.Temp	S.Throat	Abd. pain	Headache	Vom.							Locale
7	G	Fever	H.Temp	S.Throat	Abd. pain	Headache	Vom.					Prol.Mens		Locale
8	H	Fever	H.Temp	S.Throat	Abd. pain		Vom.							Locale
9	I	Fever	H.Temp	S.Throat	Abd. pain		Vom.							Locale
10	J	Fever	H.Temp	S.Throat	Abd. pain		Vom.							Locale
11	K	Fever	H.Temp	S.Throat	Abd. pain			Bl.Vom.						Locale
12	L	Fever	H.Temp	S.Throat	Abd. pain			Bl.Vom.						Locale
13	M	Fever	H.Temp	S.Throat	Abd. pain			Bl. Vom.						Locale
14	N	Fever	H.Temp	S.Throat	Abd. pain									Locale
15	O	Fever	H.Temp		Abd. pain							Prol.Mens		Locale
16	P	Fever	H.Temp		Abd. pain					B. pains				Locale
17	Q	Fever	H.Temp		Abd. pain					B. pains				Locale
18	R	Fever	H.Temp		Abd. pain				Bl. stool	B. pains			Sp.abortion	Locale
19	S	Fever	H.Temp		Abd. pain				Bl. stool	B. pains				Locale
20	T	Fever	H.Temp		Abd. pain				Bl. stool	B. pains				Locale

Key: H.Temp – High Body Temperature; S – Sore; Abd. – Abdominal; Vom. – Vomiting; Prol.Mens. – Prolonged Menstruation; Bl. – Bloody; B.weakn. – Body weakness; Sp. – Spontaneous.

Appendix E

Table 5.4 Demographic Characteristics and Signs and Symptoms in Confirmed and Probable Ebola Case Patients with a Definitive Clinical Outcome in Guinea, Liberia, Nigeria, and Sierra Leone. Source:WHO Ebola Response Team. (2014).

Variable	All Patients	Patients Who Died	Patients Who Recovered	Odds Ratio (95% CI)†
		no./total no. (%)		
Demographic characteristics				
Male sex	685/1415 (48.4)	515/1056 (48.8)	170/359 (47.4)	0.93 (0.73–1.19)
Age group				
<15 yr	190/1378 (13.8)	145/1021 (14.2)	45/357 (12.6)	1.18 (0.83–1.71)
15–44 yr	838/1378 (60.8)	577/1021 (56.5)	261/357 (73.1)	0.48 (0.36–0.62)
≥45 yr	350/1378 (25.4)	299/1021 (29.3)	51/357 (14.3)	2.47 (1.79–3.46)
Health care worker	158/1429 (11.1)	112/1067 (10.5)	46/362 (12.7)	0.86 (0.60–1.27)
Signs and symptoms				
General symptoms				
Fever	1002/1151 (87.1)	746/846 (88.2)	256/305 (83.9)	1.34 (0.92–1.95)
Fatigue	866/1133 (76.4)	633/829 (76.4)	233/304 (76.6)	0.94 (0.68–1.28)
Loss of appetite	681/1055 (64.5)	498/778 (64.0)	183/277 (66.1)	0.92 (0.69–1.23)
Vomiting	753/1114 (67.6)	566/816 (69.4)	187/298 (62.8)	1.19 (0.89–1.59)
Diarrhea	721/1099 (65.6)	555/813 (68.3)	166/286 (58.0)	1.42 (1.06–1.89)
Headache	553/1035 (53.4)	407/757 (53.8)	146/278 (52.5)	1.03 (0.78–1.36)
Abdominal pain	439/992 (44.3)	311/715 (43.5)	128/277 (46.2)	0.85 (0.64–1.13)
Muscle pain	385/990 (38.9)	293/728 (40.2)	92/262 (35.1)	1.24 (0.92–1.67)
Joint pain	374/950 (39.4)	283/695 (40.7)	91/255 (35.7)	1.32 (0.98–1.80)
Chest pain	254/686 (37.0)	196/488 (40.2)	58/198 (29.3)	1.53 (1.07–2.20)
Cough	194/655 (29.6)	150/462 (32.5)	44/193 (22.8)	1.74 (1.18–2.61)
Difficulty breathing	155/665 (23.3)	123/472 (26.1)	32/193 (16.6)	1.68 (1.10–2.63)
Difficulty swallowing	169/514 (32.9)	138/375 (36.8)	31/139 (22.3)	2.22 (1.41–3.59)
Conjunctivitis	137/658 (20.8)	109/465 (23.4)	28/193 (14.5)	2.03 (1.29–3.29)
Sore throat	102/467 (21.8)	82/339 (24.2)	20/128 (15.6)	1.94 (1.13–3.46)
Confusion	84/631 (13.3)	68/446 (15.2)	16/185 (8.6)	2.00 (1.14–3.71)
Hiccups	108/947 (11.4)	91/699 (13.0)	17/248 (6.9)	2.15 (1.27–3.82)
Jaundice	65/627 (10.4)	52/443 (11.7)	13/184 (7.1)	1.83 (0.99–3.63)
Eye pain	48/622 (7.7)	39/438 (8.9)	9/184 (4.9)	1.95 (0.95–4.40)
Rash	37/642 (5.8)	30/453 (6.6)	7/189 (3.7)	1.90 (0.86–4.83)
Coma or unconsciousness	37/627 (5.9)	34/445 (7.6)	3/182 (1.6)	4.59 (1.61–19.34)

Table 5.4 Continued

Variable	All Patients	Patients Who Died	Patients Who Recovered	Odds Ratio (95% CI)†
		no./total no. (%)		
Unexplained bleeding	168/932 (18.0)	140/693 (20.2)	28/239 (11.7)	1.83 (1.20–2.90)
Hematemesis	26/670 (3.9)	20/503 (4.0)	6/167 (3.6)	1.07 (0.44–3.01)
Blood in stool	48/843 (5.7)	35/614 (5.7)	13/229 (5.7)	0.98 (0.52–1.96)
Bleeding gums	19/837 (2.3)	18/608 (3.0)	1/229 (0.4)	6.69 (1.35–121.32)
Bloody nose	16/836 (1.9)	15/610 (2.5)	1/226 (0.4)	8.02 (1.54–148.62)
Bloody cough	20/831 (2.4)	16/605 (2.6)	4/226 (1.8)	1.63 (0.58–5.82)
Other bleeding	8/657 (1.2)	5/493 (1.0)	3/164 (1.8)	0.45 (0.11–2.23)
Bleeding at injection site	20/833 (2.4)	19/605 (3.1)	1/228 (0.4)	6.51 (1.32–118.04)
Blood from vagina§	14/431 (3.2)	13/290 (4.5)	1/126 (0.8)	6.0 (1.11–112.4)
Blood in urine	10/827 (1.2)	9/601 (1.5)	1/226 (0.4)	5.14 (0.90–98.73)
Bleeding under skin	5/827 (0.6)	5/604 (0.8)	0/223	NA

Appendix F

Figure 5.2: A Screenshot of the results of fuzzy processing of FETHA in Ebonyi State data.

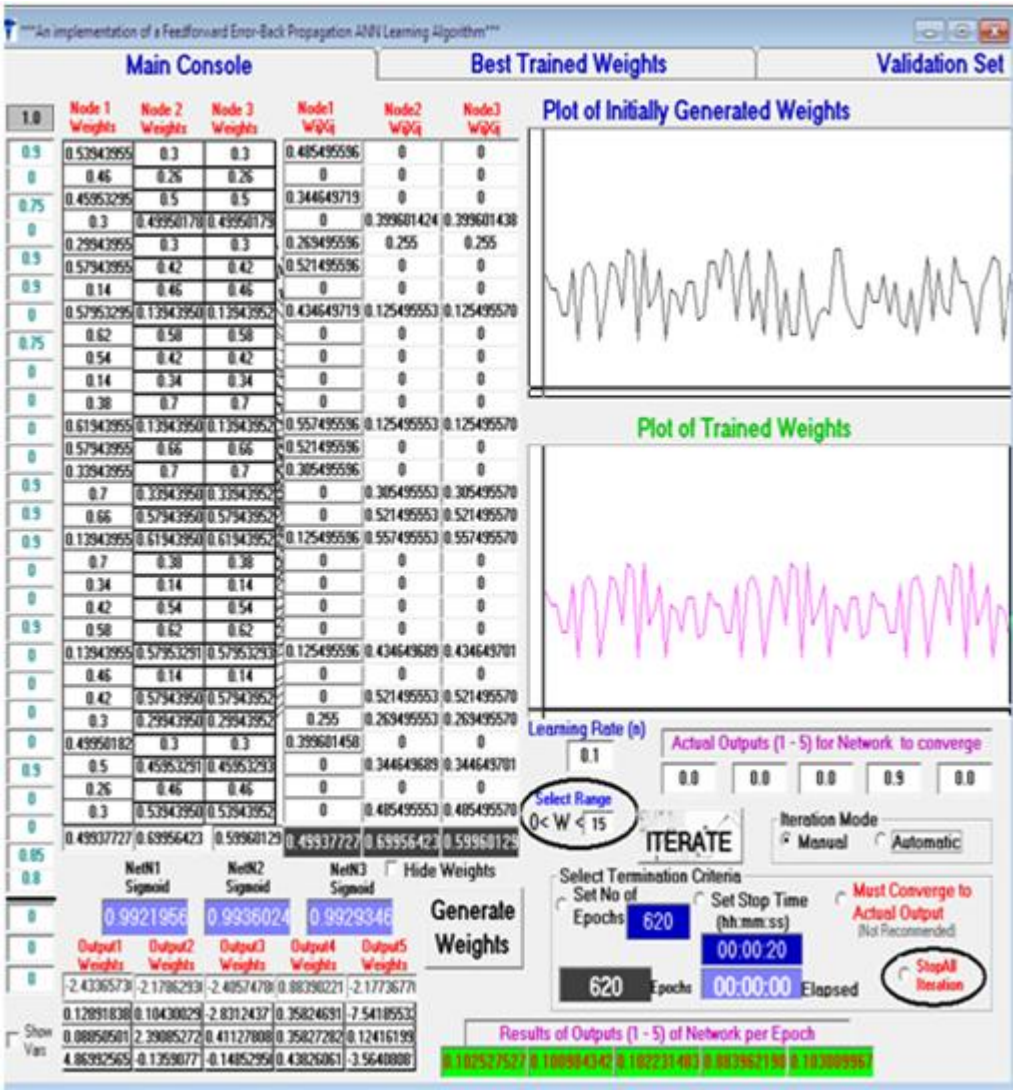


Figure 5.3: A Screenshot of the results of ANN processing of FETHA data.

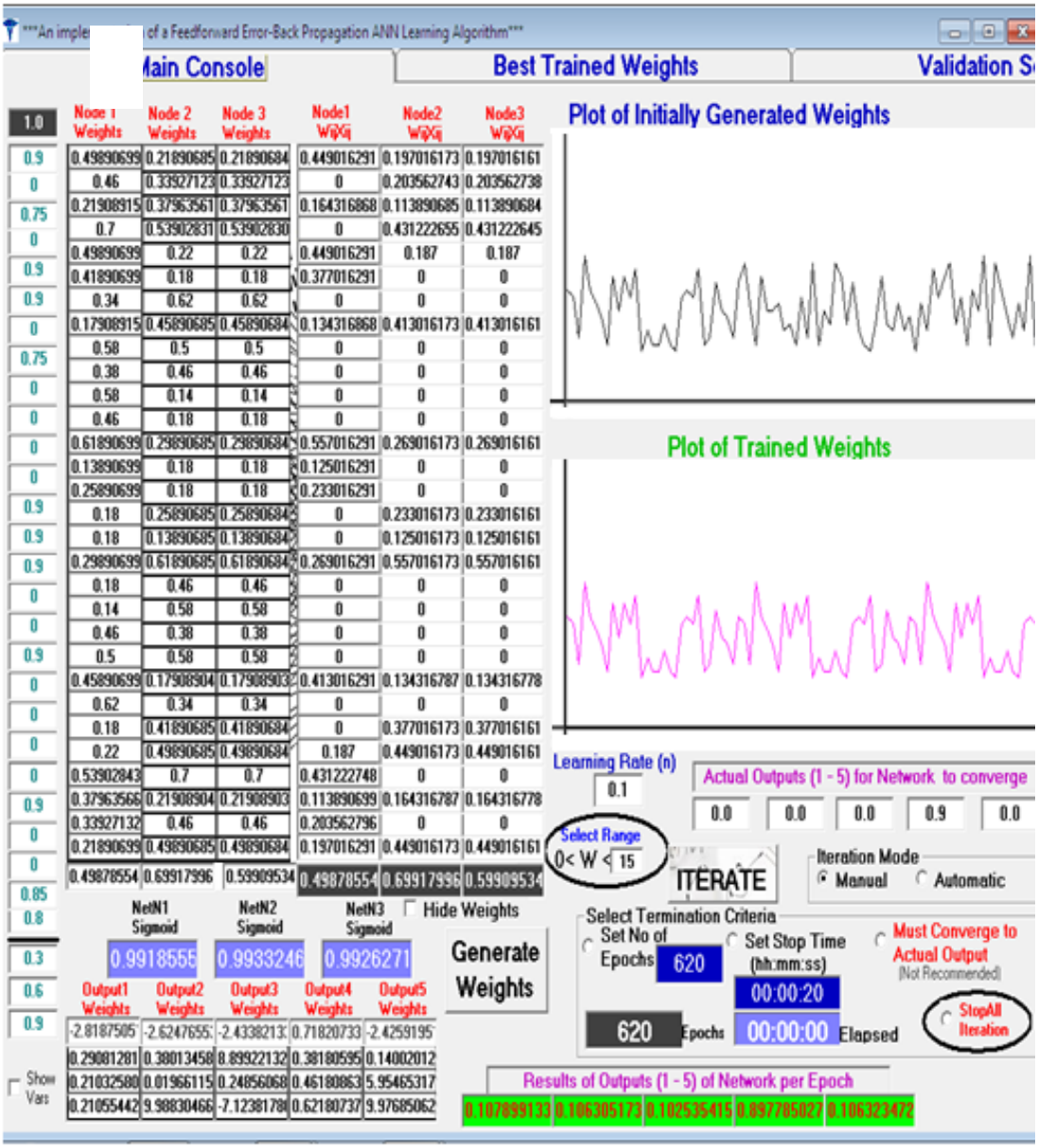


Figure 5.4: A Screenshot of the results of the behavior of ANN processing of FETHA data with new symptoms that are consistent with those of Lassa fever.

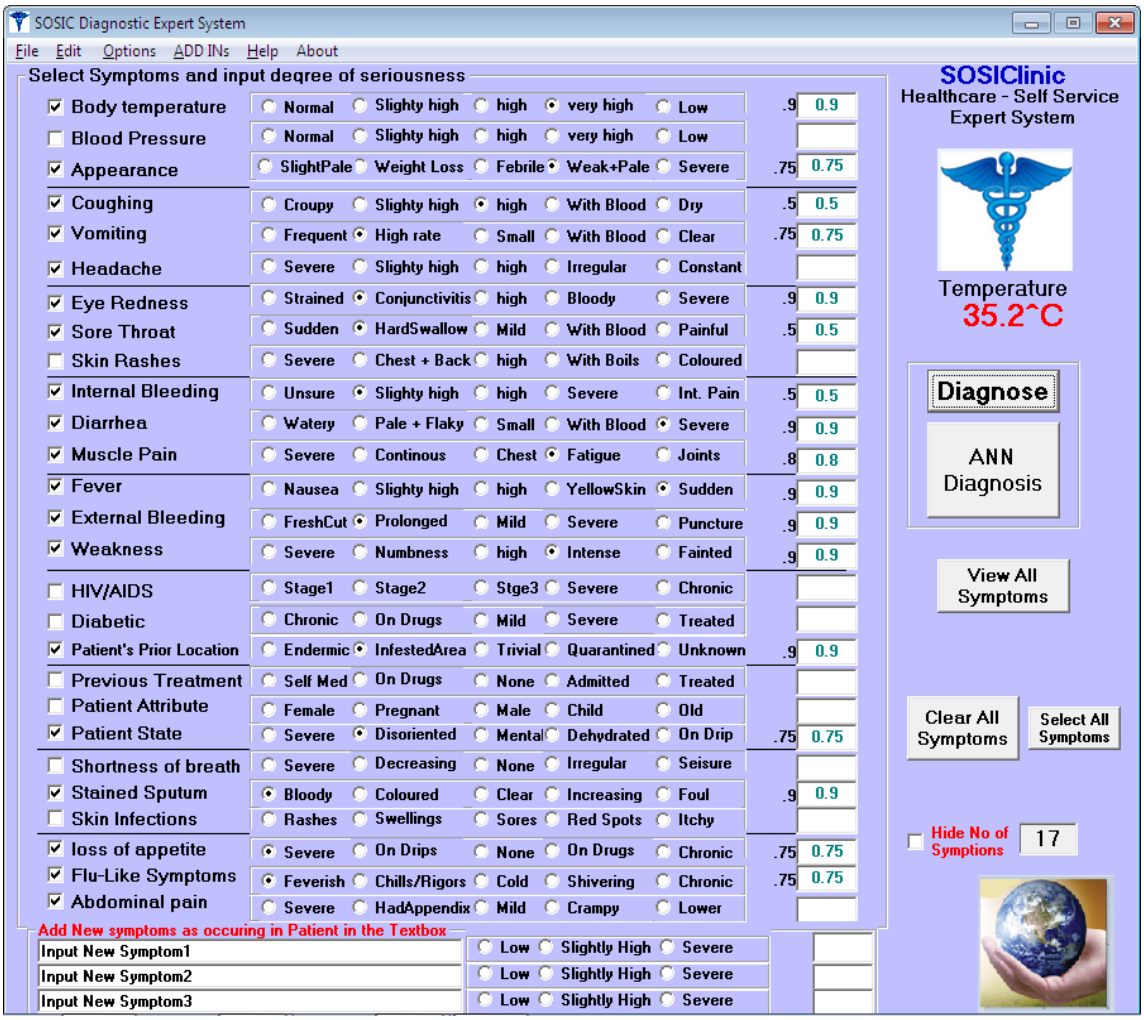


Figure 5.5: A Screenshot of inputted symptom's inputs obtained from the demographic data on EVD by WHO Ebola Response Team on Guinea, Liberia, Nigeria, and Sierra Leone.

Figure 5.6: A Screenshot of the result of fuzzy processing of the EVD symptoms in Figure 5.5.

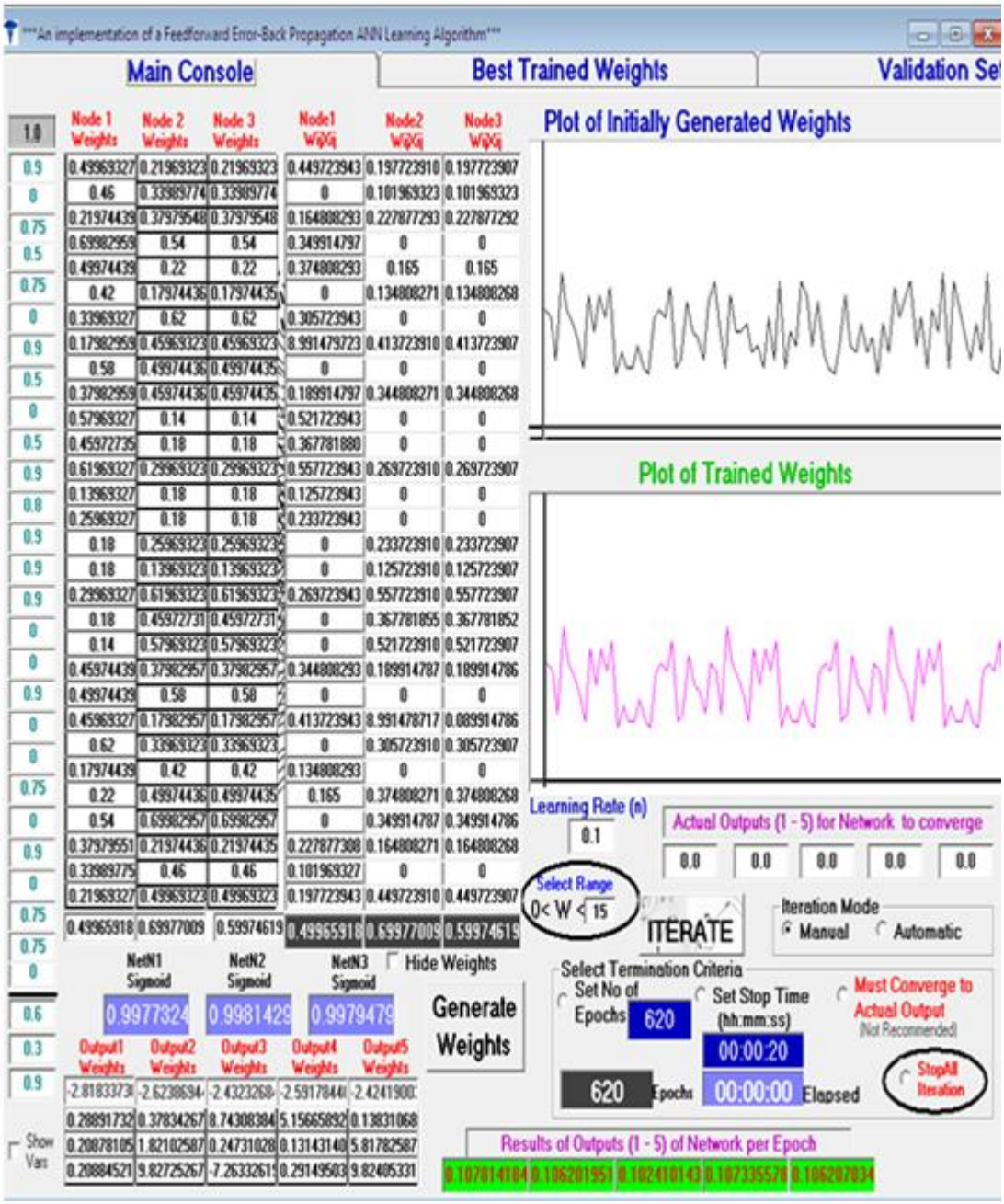


Figure 5.7: A Screenshot of the results of the behavior of ANN processing onWHO Ebola Response Teamdata with new symptoms that are consistent with those of EVD.

Appendix H

The Visual Basic Studio 6.0 programming code

```
Private Sub Iterate()
Dim i As Integer
'wieght adjustments for input layers (first forward sweep)
If lblOutput.Caption = "0" Then
'plot initial weight
Call PlotWeight
'On Err GoTo errIterate:
'node1
For i = 0 To 29
 txtEpochWeight(i).Text = Val(txtWeights(i).Text) * Val(txtQuantity(i).Text)
Next i
txtEpochWeight(92).Text = Val(txtWeights(105).Text) * Val(txtQuantity(30).Text)
txtNode(0).Text = Val(txtEpochWeight(92).Text) + Val(txtEpochWeight(0).Text) +
Val(txtEpochWeight(1).Text) + Val(txtEpochWeight(2).Text) +
Val(txtEpochWeight(3).Text) + Val(txtEpochWeight(4).Text) +
Val(txtEpochWeight(5).Text) + Val(txtEpochWeight(6).Text) +
Val(txtEpochWeight(7).Text) + Val(txtEpochWeight(8).Text) +
Val(txtEpochWeight(9).Text) + Val(txtEpochWeight(10).Text) +
Val(txtEpochWeight(11).Text) + Val(txtEpochWeight(12).Text) +
Val(txtEpochWeight(13).Text) + Val(txtEpochWeight(14).Text) +
Val(txtEpochWeight(15).Text) + Val(txtEpochWeight(16).Text) +
Val(txtEpochWeight(17).Text) + Val(txtEpochWeight(18).Text) +
Val(txtEpochWeight(19).Text) + Val(txtEpochWeight(20).Text) +
Val(txtEpochWeight(21).Text) + Val(txtEpochWeight(22).Text) +
Val(txtEpochWeight(23).Text) + Val(txtEpochWeight(24).Text) +
Val(txtEpochWeight(25).Text) + Val(txtEpochWeight(26).Text) +
Val(txtEpochWeight(27).Text) + Val(txtEpochWeight(28).Text) +
Val(txtEpochWeight(29).Text)
txtNode(0).Text = 1 / (1 + Exp(-(txtNode(0).Text)))
'node2
txtEpochWeight(30).Text = Val(txtWeights(30).Text) * Val(txtQuantity(0).Text)
txtEpochWeight(31).Text = Val(txtWeights(31).Text) * Val(txtQuantity(1).Text)
txtEpochWeight(32).Text = Val(txtWeights(32).Text) * Val(txtQuantity(2).Text)
txtEpochWeight(33).Text = Val(txtWeights(33).Text) * Val(txtQuantity(3).Text)
txtEpochWeight(34).Text = Val(txtWeights(34).Text) * Val(txtQuantity(4).Text)
txtEpochWeight(35).Text = Val(txtWeights(35).Text) * Val(txtQuantity(5).Text)
txtEpochWeight(36).Text = Val(txtWeights(36).Text) * Val(txtQuantity(6).Text)
txtEpochWeight(37).Text = Val(txtWeights(37).Text) * Val(txtQuantity(7).Text)
txtEpochWeight(38).Text = Val(txtWeights(38).Text) * Val(txtQuantity(8).Text)
txtEpochWeight(39).Text = Val(txtWeights(39).Text) * Val(txtQuantity(9).Text)
txtEpochWeight(40).Text = Val(txtWeights(40).Text) * Val(txtQuantity(10).Text)
txtEpochWeight(41).Text = Val(txtWeights(41).Text) * Val(txtQuantity(11).Text)
txtEpochWeight(42).Text = Val(txtWeights(42).Text) * Val(txtQuantity(12).Text)
txtEpochWeight(43).Text = Val(txtWeights(43).Text) * Val(txtQuantity(13).Text)
txtEpochWeight(44).Text = Val(txtWeights(44).Text) * Val(txtQuantity(14).Text)
txtEpochWeight(45).Text = Val(txtWeights(45).Text) * Val(txtQuantity(15).Text)
txtEpochWeight(46).Text = Val(txtWeights(46).Text) * Val(txtQuantity(16).Text)
txtEpochWeight(47).Text = Val(txtWeights(47).Text) * Val(txtQuantity(17).Text)
txtEpochWeight(48).Text = Val(txtWeights(48).Text) * Val(txtQuantity(18).Text)
```

```
txtEpochWeight(49).Text = Val(txtWeights(49).Text) * Val(txtQuantity(19).Text)
txtEpochWeight(50).Text = Val(txtWeights(50).Text) * Val(txtQuantity(20).Text)
txtEpochWeight(51).Text = Val(txtWeights(51).Text) * Val(txtQuantity(21).Text)
txtEpochWeight(52).Text = Val(txtWeights(52).Text) * Val(txtQuantity(22).Text)
txtEpochWeight(53).Text = Val(txtWeights(53).Text) * Val(txtQuantity(23).Text)
txtEpochWeight(54).Text = Val(txtWeights(54).Text) * Val(txtQuantity(24).Text)
txtEpochWeight(55).Text = Val(txtWeights(55).Text) * Val(txtQuantity(25).Text)
txtEpochWeight(56).Text = Val(txtWeights(56).Text) * Val(txtQuantity(26).Text)
txtEpochWeight(57).Text = Val(txtWeights(57).Text) * Val(txtQuantity(27).Text)
txtEpochWeight(58).Text = Val(txtWeights(58).Text) * Val(txtQuantity(28).Text)
txtEpochWeight(59).Text = Val(txtWeights(59).Text) * Val(txtQuantity(29).Text)
'compute sigmoid for node1
txtEpochWeight(91).Text = Val(txtWeights(106).Text) * Val(txtQuantity(30).Text)
txtNode(1).Text = Val(txtEpochWeight(91).Text) + Val(txtEpochWeight(30).Text) +
Val(txtEpochWeight(31).Text) + Val(txtEpochWeight(32).Text) +
Val(txtEpochWeight(33).Text) + Val(txtEpochWeight(34).Text) +
Val(txtEpochWeight(35).Text) + Val(txtEpochWeight(36).Text) +
Val(txtEpochWeight(37).Text) + Val(txtEpochWeight(38).Text) +
Val(txtEpochWeight(39).Text) + Val(txtEpochWeight(40).Text) +
Val(txtEpochWeight(41).Text) + Val(txtEpochWeight(42).Text) +
Val(txtEpochWeight(43).Text) + Val(txtEpochWeight(44).Text) +
Val(txtEpochWeight(45).Text) + Val(txtEpochWeight(46).Text) +
Val(txtEpochWeight(47).Text) + Val(txtEpochWeight(48).Text) +
Val(txtEpochWeight(49).Text) + Val(txtEpochWeight(50).Text) +
Val(txtEpochWeight(51).Text) + Val(txtEpochWeight(52).Text) +
Val(txtEpochWeight(53).Text) + Val(txtEpochWeight(54).Text) +
Val(txtEpochWeight(55).Text) + Val(txtEpochWeight(56).Text) +
Val(txtEpochWeight(57).Text) + Val(txtEpochWeight(58).Text) +
Val(txtEpochWeight(59).Text)
txtNode(1).Text = 1 / (1 + Exp(-(txtNode(1).Text)))
'node 2
txtEpochWeight(60).Text = Val(txtWeights(60).Text) * Val(txtQuantity(0).Text)
txtEpochWeight(61).Text = Val(txtWeights(61).Text) * Val(txtQuantity(1).Text)
txtEpochWeight(62).Text = Val(txtWeights(62).Text) * Val(txtQuantity(2).Text)
txtEpochWeight(63).Text = Val(txtWeights(63).Text) * Val(txtQuantity(3).Text)
txtEpochWeight(64).Text = Val(txtWeights(64).Text) * Val(txtQuantity(4).Text)
txtEpochWeight(65).Text = Val(txtWeights(65).Text) * Val(txtQuantity(5).Text)
txtEpochWeight(66).Text = Val(txtWeights(66).Text) * Val(txtQuantity(6).Text)
txtEpochWeight(67).Text = Val(txtWeights(67).Text) * Val(txtQuantity(7).Text)
txtEpochWeight(68).Text = Val(txtWeights(68).Text) * Val(txtQuantity(8).Text)
txtEpochWeight(69).Text = Val(txtWeights(69).Text) * Val(txtQuantity(9).Text)
txtEpochWeight(70).Text = Val(txtWeights(70).Text) * Val(txtQuantity(10).Text)
txtEpochWeight(71).Text = Val(txtWeights(71).Text) * Val(txtQuantity(11).Text)
txtEpochWeight(72).Text = Val(txtWeights(72).Text) * Val(txtQuantity(12).Text)
txtEpochWeight(73).Text = Val(txtWeights(73).Text) * Val(txtQuantity(13).Text)
txtEpochWeight(74).Text = Val(txtWeights(74).Text) * Val(txtQuantity(14).Text)
txtEpochWeight(75).Text = Val(txtWeights(75).Text) * Val(txtQuantity(15).Text)
txtEpochWeight(76).Text = Val(txtWeights(76).Text) * Val(txtQuantity(16).Text)
txtEpochWeight(77).Text = Val(txtWeights(77).Text) * Val(txtQuantity(17).Text)
txtEpochWeight(78).Text = Val(txtWeights(78).Text) * Val(txtQuantity(18).Text)
txtEpochWeight(79).Text = Val(txtWeights(79).Text) * Val(txtQuantity(19).Text)
txtEpochWeight(80).Text = Val(txtWeights(80).Text) * Val(txtQuantity(20).Text)
```

```
txtEpochWeight(81).Text = Val(txtWeights(81).Text) * Val(txtQuantity(21).Text)
txtEpochWeight(82).Text = Val(txtWeights(82).Text) * Val(txtQuantity(22).Text)
txtEpochWeight(83).Text = Val(txtWeights(83).Text) * Val(txtQuantity(23).Text)
txtEpochWeight(84).Text = Val(txtWeights(84).Text) * Val(txtQuantity(24).Text)
txtEpochWeight(85).Text = Val(txtWeights(85).Text) * Val(txtQuantity(25).Text)
txtEpochWeight(86).Text = Val(txtWeights(86).Text) * Val(txtQuantity(26).Text)
txtEpochWeight(87).Text = Val(txtWeights(87).Text) * Val(txtQuantity(27).Text)
txtEpochWeight(88).Text = Val(txtWeights(88).Text) * Val(txtQuantity(28).Text)
txtEpochWeight(89).Text = Val(txtWeights(89).Text) * Val(txtQuantity(29).Text)
'compute sigmoid for node2
txtEpochWeight(90).Text = Val(txtWeights(107).Text) * Val(txtQuantity(30).Text)
txtNode(2).Text = Val(txtEpochWeight(90).Text) + Val(txtEpochWeight(60).Text) +
Val(txtEpochWeight(61).Text) + Val(txtEpochWeight(62).Text) +
Val(txtEpochWeight(63).Text) + Val(txtEpochWeight(64).Text) +
Val(txtEpochWeight(65).Text) + Val(txtEpochWeight(66).Text) +
Val(txtEpochWeight(67).Text) + Val(txtEpochWeight(68).Text) +
Val(txtEpochWeight(69).Text) + Val(txtEpochWeight(70).Text) +
Val(txtEpochWeight(71).Text) + Val(txtEpochWeight(72).Text) +
Val(txtEpochWeight(73).Text) + Val(txtEpochWeight(74).Text) +
Val(txtEpochWeight(75).Text) + Val(txtEpochWeight(76).Text) +
Val(txtEpochWeight(77).Text) + Val(txtEpochWeight(78).Text) +
Val(txtEpochWeight(79).Text) + Val(txtEpochWeight(80).Text) +
Val(txtEpochWeight(81).Text) + Val(txtEpochWeight(82).Text) +
Val(txtEpochWeight(83).Text) + Val(txtEpochWeight(84).Text) +
Val(txtEpochWeight(85).Text) + Val(txtEpochWeight(86).Text) +
Val(txtEpochWeight(87).Text) + Val(txtEpochWeight(88).Text) +
Val(txtEpochWeight(89).Text)
txtNode(2).Text = 1 / (1 + Exp(-(txtNode(2).Text)))
'wieght adjustments for hidden layers (first forward sweep)
txtOutput1.Text = Val(txtWeights(112).Text) + Val(txtNode(0).Text) *
Val(txtWeights(90).Text) + Val(txtNode(1).Text) * Val(txtWeights(91).Text) +
Val(txtNode(2).Text) * Val(txtWeights(92).Text)
txtOutput2.Text = Val(txtWeights(111).Text) + Val(txtNode(0).Text) *
Val(txtWeights(93).Text) + Val(txtNode(1).Text) * Val(txtWeights(94).Text) +
Val(txtNode(2).Text) * Val(txtWeights(95).Text)
txtOutput3.Text = Val(txtWeights(110).Text) + Val(txtNode(0).Text) *
Val(txtWeights(96).Text) + Val(txtNode(1).Text) * Val(txtWeights(97).Text) +
Val(txtNode(2).Text) * Val(txtWeights(98).Text)
txtOutput4.Text = Val(txtWeights(109).Text) + Val(txtNode(0).Text) *
Val(txtWeights(99).Text) + Val(txtNode(1).Text) * Val(txtWeights(100).Text) +
Val(txtNode(2).Text) * Val(txtWeights(101).Text)
txtOutput5.Text = Val(txtWeights(108).Text) + Val(txtNode(0).Text) *
Val(txtWeights(102).Text) + Val(txtNode(1).Text) * Val(txtWeights(103).Text) +
Val(txtNode(2).Text) * Val(txtWeights(104).Text)
txtOutput1.Text = 1 / (1 + Exp(-(txtOutput1.Text)))
txtOutput2.Text = 1 / (1 + Exp(-(txtOutput2.Text)))
txtOutput3.Text = 1 / (1 + Exp(-(txtOutput3.Text)))
txtOutput4.Text = 1 / (1 + Exp(-(txtOutput4.Text)))
txtOutput5.Text = 1 / (1 + Exp(-(txtOutput5.Text)))
Call counter
Else
End If
```

```
Call WeightList
Exit Sub
'errIterate:
'msg = "Oops! You must select New Weights first! Click on Generate Weights"
'title = "SOSIC Weights Check!"
'Ans = MsgBox(msg, vbExclamation + vbOKOnly, title)
'Exit Sub
End Sub
Private Sub counter()
Static counter As Integer
counter = counter + 1
lblOutput.Caption = counter
End Sub
Private Sub IterateTwo()
Dim i As Integer
'wieght adjustments for output layers (Next (i) forward sweep)
'error responsibility dz
       'dz = outputz(1 - outputz)(actualz - outputz)
   'dz-------Val(txtOutput1.Text) * (1 - Val(txtOutput1.Text)) * (Val(txtActual.Text) -
Val(txtOutput1.Text)) * Val(txtQuantity(30).Text)
   'then     'compute Change in Woz = ndz(1)
   'Woz,new = Woz,current + Change in Woz
dz1 = (Val(txtLearnR_n.Text) * Val(txtOutput1.Text) * (1 - Val(txtOutput1.Text)) *
(Val(txtActual(0).Text) - Val(txtOutput1.Text)) * Val(txtQuantity(30).Text))
dz2 = (Val(txtLearnR_n.Text) * Val(txtOutput2.Text) * (1 - Val(txtOutput2.Text)) *
(Val(txtActual(1).Text) - Val(txtOutput2.Text)) * Val(txtQuantity(30).Text))
dz3 = (Val(txtLearnR_n.Text) * Val(txtOutput3.Text) * (1 - Val(txtOutput3.Text)) *
(Val(txtActual(2).Text) - Val(txtOutput3.Text)) * Val(txtQuantity(30).Text))
dz4 = (Val(txtLearnR_n.Text) * Val(txtOutput4.Text) * (1 - Val(txtOutput4.Text)) *
(Val(txtActual(3).Text) - Val(txtOutput4.Text)) * Val(txtQuantity(30).Text))
dz5 = (Val(txtLearnR_n.Text) * Val(txtOutput5.Text) * (1 - Val(txtOutput5.Text)) *
(Val(txtActual(4).Text) - Val(txtOutput5.Text)) * Val(txtQuantity(30).Text))
'using dz(i) combo size
txtWeights(112).Text = Val(txtWeights(112).Text) + dz1
txtWeights(111).Text = Val(txtWeights(111).Text) + dz2
txtWeights(110).Text = Val(txtWeights(110).Text) + dz3
txtWeights(109).Text = Val(txtWeights(109).Text) + dz4
txtWeights(108).Text = Val(txtWeights(108).Text) + dz5
'wieght adjustments for hidden layers (Next (i) forward sweep)
'error responsibility dA
       'dA = outputNA(1 - outputNA)E(downstream)WjkDj
dA1 = Val(txtNode(0).Text) * (1 - Val(txtNode(0).Text)) * Val(txtWeights(90).Text) * dz1
        'then     'compute Change in WAz = ndz(1).OutputA
   CAz1 = Val(txtLearnR_n.Text) * dz1 * Val(txtNode(0).Text)
   'WAz,new = WAz,current + Change in WAz
   txtWeights(90).Text = Val(txtWeights(90).Text) + CAz1
  'next hidden layer2
dA2 = Val(txtNode(0).Text) * (1 - Val(txtNode(0).Text)) * Val(txtWeights(93).Text) * dz2
        'then     'compute Change in WAz = ndz(1).OutputA
   CAz2 = Val(txtLearnR_n.Text) * dz2 * Val(txtNode(0).Text)
   'WAz,new = WAz,current + Change in WAz
   txtWeights(93).Text = Val(txtWeights(93).Text) + CAz2
```

```
    'next hidden layer3
dA3 = Val(txtNode(0).Text) * (1 - Val(txtNode(0).Text)) * Val(txtWeights(96).Text) * dz3
        'then     'compute Change in WAz = ndz(1).OutputA
    CAz3 = Val(txtLearnR_n.Text) * dz3 * Val(txtNode(0).Text)
    'WAz,new = WAz,current + Change in WAz
    txtWeights(96).Text = Val(txtWeights(96).Text) + CAz3
'next hidden layer4
dA4 = Val(txtNode(0).Text) * (1 - Val(txtNode(0).Text)) * Val(txtWeights(99).Text) * dz4
        'then    'compute Change in WAz = ndz(1).OutputA
    CAz4 = Val(txtLearnR_n.Text) * dz4 * Val(txtNode(0).Text)
    'WAz,new = WAz,current + Change in WAz
    txtWeights(99).Text = Val(txtWeights(99).Text) + CAz4
'next hidden layer5
dA5 = Val(txtNode(0).Text) * (1 - Val(txtNode(0).Text)) * Val(txtWeights(102).Text) * dz5
        'then    'compute Change in WAz = ndz(1).OutputA
    CAz5 = Val(txtLearnR_n.Text) * dz5 * Val(txtNode(0).Text)
    'WAz,new = WAz,current + Change in WAz
    txtWeights(102).Text = Val(txtWeights(102).Text) + CAz5
       'error responsibility dB
       'dB = outputNB(1 - outputNB)E(downstream)WjkDj
dB1 = Val(txtNode(1).Text) * (1 - Val(txtNode(1).Text)) * Val(txtWeights(91).Text) * dz1
        'then    'compute Change in WBz = ndz(1).OutputB
    CBz1 = Val(txtLearnR_n.Text) * dz1 * Val(txtNode(1).Text)
    'WBz,new = WBz,current + Change in WBz
    txtWeights(91).Text = Val(txtWeights(91).Text) + CBz1
'Next hidden layer
       'dB = outputNB(1 - outputNB)E(downstream)WjkDj
dB2 = Val(txtNode(1).Text) * (1 - Val(txtNode(1).Text)) * Val(txtWeights(94).Text) * dz2
        'then    'compute Change in WBz = ndz(1).OutputB
    CBz2 = Val(txtLearnR_n.Text) * dz2 * Val(txtNode(1).Text)
    'WBz,new = WBz,current + Change in WBz
    txtWeights(94).Text = Val(txtWeights(94).Text) + CBz2
'Next hidden layer
       'dB = outputNB(1 - outputNB)E(downstream)WjkDj
dB3 = Val(txtNode(1).Text) * (1 - Val(txtNode(1).Text)) * Val(txtWeights(97).Text) * dz3
        'then    'compute Change in WBz = ndz(1).OutputB
    CBz3 = Val(txtLearnR_n.Text) * dz3 * Val(txtNode(1).Text)
    'WBz,new = WBz,current + Change in WBz
    txtWeights(97).Text = Val(txtWeights(97).Text) + CBz3
    'Next hidden layer
       'dB = outputNB(1 - outputNB)E(downstream)WjkDj
dB4 = Val(txtNode(1).Text) * (1 - Val(txtNode(1).Text)) * Val(txtWeights(100).Text) * dz4
        'then    'compute Change in WBz = ndz(1).OutputB
    CBz4 = Val(txtLearnR_n.Text) * dz4 * Val(txtNode(1).Text)
    'WBz,new = WBz,current + Change in WBz
    txtWeights(100).Text = Val(txtWeights(100).Text) + CBz4
   'Next hidden layer
       'dB = outputNB(1 - outputNB)E(downstream)WjkDj
dB5 = Val(txtNode(1).Text) * (1 - Val(txtNode(1).Text)) * Val(txtWeights(103).Text) * dz5
        'then    'compute Change in WBz = ndz(1).OutputB
    CBz5 = Val(txtLearnR_n.Text) * dz5 * Val(txtNode(1).Text)
    'WBz,new = WBz,current + Change in WBz
```

```
    txtWeights(103).Text = Val(txtWeights(103).Text) + CBz5
    'error responsibility dC
        'dC = outputNC(1 - outputNC)E(downstream)WjkDj
dC1 = Val(txtNode(2).Text) * (1 - Val(txtNode(2).Text)) * Val(txtWeights(92).Text) * dz1
         'then    'compute Change in WCz = ndz(1).OutputC
    CCz1 = Val(txtLearnR_n.Text) * dz1 * Val(txtNode(2).Text)
    'WCz,new = WCz,current + Change in WCz
    txtWeights(92).Text = Val(txtWeights(92).Text) + CCz1
'Next hidden layer
        'dC = outputNC(1 - outputNC)E(downstream)WjkDj
dC2 = Val(txtNode(2).Text) * (1 - Val(txtNode(2).Text)) * Val(txtWeights(95).Text) * dz2
         'then 'compute Change in WCz = ndz(1).OutputC
    CCz2 = Val(txtLearnR_n.Text) * dz2 * Val(txtNode(2).Text)
    'WCz,new = WCz,current + Change in WCz
    txtWeights(95).Text = Val(txtWeights(95).Text) + CCz2
'Next hidden layer
        'dC = outputNC(1 - outputNC)E(downstream)WjkDj
dC3 = Val(txtNode(2).Text) * (1 - Val(txtNode(2).Text)) * Val(txtWeights(98).Text) * dz3
         'then   'compute Change in WCz = ndz(1).OutputC
    CCz3 = Val(txtLearnR_n.Text) * dz3 * Val(txtNode(2).Text)
    'WCz,new = WCz,current + Change in WCz
    txtWeights(98).Text = Val(txtWeights(98).Text) + CCz3
'Next hidden layer
        'dC = outputNC(1 - outputNC)E(downstream)WjkDj
dC4 = Val(txtNode(2).Text) * (1 - Val(txtNode(2).Text)) * Val(txtWeights(101).Text) * dz4
         'then    'compute Change in WCz = ndz(1).OutputC
    CCz4 = Val(txtLearnR_n.Text) * dz4 * Val(txtNode(2).Text)
    'WCz,new = WCz,current + Change in WCz
    txtWeights(101).Text = Val(txtWeights(101).Text) + CCz4
'Next hidden layer
        'dC = outputNC(1 - outputNC)E(downstream)WjkDj
dC5 = Val(txtNode(2).Text) * (1 - Val(txtNode(2).Text)) * Val(txtWeights(104).Text) * dz5
         'then  'compute Change in WCz = ndz(1).OutputC
    CCz5 = Val(txtLearnR_n.Text) * dz5 * Val(txtNode(2).Text)
    'WCz,new = WCz,current + Change in WCz
    txtWeights(104).Text = Val(txtWeights(104).Text) + CCz5
'wieght adjustments for inputs (Next (i) forward sweep)
        'compute Change in WiA = ndAXi
        'WiA,new = WiA,current + Change in WiA
    CW1A = Val(txtLearnR_n.Text) * dA1 * Val(txtQuantity(0).Text)
    txtWeights(0).Text = Val(txtWeights(0).Text) + CW1A
    CW2A = Val(txtLearnR_n.Text) * dA1 * Val(txtQuantity(1).Text)
    txtWeights(1).Text = Val(txtWeights(1).Text) + CW2A
    CW3A = Val(txtLearnR_n.Text) * dA1 * Val(txtQuantity(2).Text)
    txtWeights(2).Text = Val(txtWeights(2).Text) + CW3A
    CW4A = Val(txtLearnR_n.Text) * dA1 * Val(txtQuantity(3).Text)
    txtWeights(3).Text = Val(txtWeights(3).Text) + CW4A
    CW5A = Val(txtLearnR_n.Text) * dA1 * Val(txtQuantity(4).Text)
    txtWeights(4).Text = Val(txtWeights(4).Text) + CW5A
    CW6A = Val(txtLearnR_n.Text) * dA1 * Val(txtQuantity(5).Text)
    txtWeights(5).Text = Val(txtWeights(5).Text) + CW6A
    CW7A = Val(txtLearnR_n.Text) * dA1 * Val(txtQuantity(6).Text)
```

```
txtWeights(6).Text = Val(txtWeights(6).Text) + CW7A
CW8A = Val(txtLearnR_n.Text) * dA1 * Val(txtQuantity(7).Text)
txtWeights(7).Text = Val(txtWeights(7).Text) + CW8A
CW9A = Val(txtLearnR_n.Text) * dA1 * Val(txtQuantity(8).Text)
txtWeights(8).Text = Val(txtWeights(8).Text) + CW9A
CW10A = Val(txtLearnR_n.Text) * dA1 * Val(txtQuantity(9).Text)
txtWeights(9).Text = Val(txtWeights(9).Text) + CW10A
CW11A = Val(txtLearnR_n.Text) * dA1 * Val(txtQuantity(10).Text)
txtWeights(10).Text = Val(txtWeights(10).Text) + CW11A
CW12A = Val(txtLearnR_n.Text) * dA1 * Val(txtQuantity(11).Text)
txtWeights(11).Text = Val(txtWeights(11).Text) + CW12A
CW13A = Val(txtLearnR_n.Text) * dA1 * Val(txtQuantity(12).Text)
txtWeights(12).Text = Val(txtWeights(12).Text) + CW13A
CW14A = Val(txtLearnR_n.Text) * dA1 * Val(txtQuantity(13).Text)
txtWeights(13).Text = Val(txtWeights(13).Text) + CW14A
CW15A = Val(txtLearnR_n.Text) * dA1 * Val(txtQuantity(14).Text)
txtWeights(14).Text = Val(txtWeights(14).Text) + CW15A
CW16A = Val(txtLearnR_n.Text) * dA1 * Val(txtQuantity(15).Text)
txtWeights(15).Text = Val(txtWeights(15).Text) + CW16A
CW17A = Val(txtLearnR_n.Text) * dA1 * Val(txtQuantity(16).Text)
txtWeights(16).Text = Val(txtWeights(16).Text) + CW17A
CW18A = Val(txtLearnR_n.Text) * dA1 * Val(txtQuantity(17).Text)
txtWeights(17).Text = Val(txtWeights(17).Text) + CW18A
CW19A = Val(txtLearnR_n.Text) * dA1 * Val(txtQuantity(18).Text)
txtWeights(18).Text = Val(txtWeights(18).Text) + CW19A
CW20A = Val(txtLearnR_n.Text) * dA1 * Val(txtQuantity(19).Text)
txtWeights(19).Text = Val(txtWeights(19).Text) + CW20A
CW21A = Val(txtLearnR_n.Text) * dA1 * Val(txtQuantity(20).Text)
txtWeights(20).Text = Val(txtWeights(20).Text) + CW21A
CW22A = Val(txtLearnR_n.Text) * dA1 * Val(txtQuantity(21).Text)
txtWeights(21).Text = Val(txtWeights(21).Text) + CW21A
CW23A = Val(txtLearnR_n.Text) * dA1 * Val(txtQuantity(22).Text)
txtWeights(22).Text = Val(txtWeights(22).Text) + CW23A
CW24A = Val(txtLearnR_n.Text) * dA1 * Val(txtQuantity(23).Text)
txtWeights(23).Text = Val(txtWeights(23).Text) + CW24A
CW25A = Val(txtLearnR_n.Text) * dA1 * Val(txtQuantity(24).Text)
txtWeights(24).Text = Val(txtWeights(24).Text) + CW25A
CW26A = Val(txtLearnR_n.Text) * dA1 * Val(txtQuantity(25).Text)
txtWeights(25).Text = Val(txtWeights(25).Text) + CW22A
CW27A = Val(txtLearnR_n.Text) * dA1 * Val(txtQuantity(26).Text)
txtWeights(26).Text = Val(txtWeights(26).Text) + CW27A
CW28A = Val(txtLearnR_n.Text) * dA1 * Val(txtQuantity(27).Text)
txtWeights(27).Text = Val(txtWeights(27).Text) + CW28A
CW29A = Val(txtLearnR_n.Text) * dA1 * Val(txtQuantity(28).Text)
txtWeights(28).Text = Val(txtWeights(28).Text) + CW29A
CW30A = Val(txtLearnR_n.Text) * dA1 * Val(txtQuantity(29).Text)
txtWeights(29).Text = Val(txtWeights(29).Text) + CW30A
CW0A = Val(txtLearnR_n.Text) * dA1 * Val(txtQuantity(30).Text)
txtWeights(105).Text = Val(txtWeights(105).Text) + CW0A
'compute Change in WiB = ndBXi
   'WiB,new = WiB,current + Change in WiB
CW1B = Val(txtLearnR_n.Text) * dB1 * Val(txtQuantity(0).Text)
```

```
txtWeights(30).Text = Val(txtWeights(0).Text) + CW1B
CW2B = Val(txtLearnR_n.Text) * dB1 * Val(txtQuantity(1).Text)
txtWeights(31).Text = Val(txtWeights(1).Text) + CW2B
CW3B = Val(txtLearnR_n.Text) * dB1 * Val(txtQuantity(2).Text)
txtWeights(32).Text = Val(txtWeights(2).Text) + CW3B
CW4B = Val(txtLearnR_n.Text) * dB1 * Val(txtQuantity(3).Text)
txtWeights(33).Text = Val(txtWeights(3).Text) + CW4B
CW5B = Val(txtLearnR_n.Text) * dB1 * Val(txtQuantity(4).Text)
txtWeights(34).Text = Val(txtWeights(4).Text) + CW5B
CW6B = Val(txtLearnR_n.Text) * dB1 * Val(txtQuantity(5).Text)
txtWeights(35).Text = Val(txtWeights(5).Text) + CW6B
CW7B = Val(txtLearnR_n.Text) * dB1 * Val(txtQuantity(6).Text)
txtWeights(36).Text = Val(txtWeights(6).Text) + CW7B
CW8B = Val(txtLearnR_n.Text) * dB1 * Val(txtQuantity(7).Text)
txtWeights(37).Text = Val(txtWeights(7).Text) + CW8B
CW9B = Val(txtLearnR_n.Text) * dB1 * Val(txtQuantity(8).Text)
txtWeights(38).Text = Val(txtWeights(8).Text) + CW9B
CW10B = Val(txtLearnR_n.Text) * dB1 * Val(txtQuantity(9).Text)
txtWeights(39).Text = Val(txtWeights(9).Text) + CW10B
CW11B = Val(txtLearnR_n.Text) * dB1 * Val(txtQuantity(10).Text)
txtWeights(40).Text = Val(txtWeights(10).Text) + CW11B
CW12B = Val(txtLearnR_n.Text) * dB1 * Val(txtQuantity(11).Text)
txtWeights(41).Text = Val(txtWeights(11).Text) + CW12B
CW13B = Val(txtLearnR_n.Text) * dB1 * Val(txtQuantity(12).Text)
txtWeights(42).Text = Val(txtWeights(12).Text) + CW13B
CW14B = Val(txtLearnR_n.Text) * dB1 * Val(txtQuantity(13).Text)
txtWeights(43).Text = Val(txtWeights(13).Text) + CW14B
CW15B = Val(txtLearnR_n.Text) * dB1 * Val(txtQuantity(14).Text)
txtWeights(44).Text = Val(txtWeights(14).Text) + CW15B
CW16B = Val(txtLearnR_n.Text) * dB1 * Val(txtQuantity(15).Text)
txtWeights(45).Text = Val(txtWeights(15).Text) + CW16B
CW17B = Val(txtLearnR_n.Text) * dB1 * Val(txtQuantity(16).Text)
txtWeights(46).Text = Val(txtWeights(16).Text) + CW17B
CW18B = Val(txtLearnR_n.Text) * dB1 * Val(txtQuantity(17).Text)
txtWeights(47).Text = Val(txtWeights(17).Text) + CW18B
CW19B = Val(txtLearnR_n.Text) * dB1 * Val(txtQuantity(18).Text)
txtWeights(48).Text = Val(txtWeights(18).Text) + CW19B
CW20B = Val(txtLearnR_n.Text) * dB1 * Val(txtQuantity(19).Text)
txtWeights(49).Text = Val(txtWeights(19).Text) + CW20B
CW21B = Val(txtLearnR_n.Text) * dB1 * Val(txtQuantity(20).Text)
txtWeights(50).Text = Val(txtWeights(20).Text) + CW21B
CW22B = Val(txtLearnR_n.Text) * dB1 * Val(txtQuantity(21).Text)
txtWeights(51).Text = Val(txtWeights(21).Text) + CW21B
CW23B = Val(txtLearnR_n.Text) * dB1 * Val(txtQuantity(22).Text)
txtWeights(52).Text = Val(txtWeights(22).Text) + CW23B
CW24B = Val(txtLearnR_n.Text) * dB1 * Val(txtQuantity(23).Text)
txtWeights(53).Text = Val(txtWeights(23).Text) + CW24B
CW25B = Val(txtLearnR_n.Text) * dB1 * Val(txtQuantity(24).Text)
txtWeights(54).Text = Val(txtWeights(24).Text) + CW25B
CW26B = Val(txtLearnR_n.Text) * dB1 * Val(txtQuantity(25).Text)
txtWeights(55).Text = Val(txtWeights(25).Text) + CW22B
CW27B = Val(txtLearnR_n.Text) * dB1 * Val(txtQuantity(26).Text)
```

```
txtWeights(56).Text = Val(txtWeights(26).Text) + CW27B
CW28B = Val(txtLearnR_n.Text) * dB1 * Val(txtQuantity(27).Text)
txtWeights(57).Text = Val(txtWeights(27).Text) + CW28B
CW29B = Val(txtLearnR_n.Text) * dB1 * Val(txtQuantity(28).Text)
txtWeights(58).Text = Val(txtWeights(28).Text) + CW29B
CW30B = Val(txtLearnR_n.Text) * dB1 * Val(txtQuantity(29).Text)
txtWeights(59).Text = Val(txtWeights(29).Text) + CW30B
CW0B = Val(txtLearnR_n.Text) * dB1 * Val(txtQuantity(30).Text)
txtWeights(106).Text = Val(txtWeights(106).Text) + CW0B
     'compute Change in WiC = ndBXi
   'WiC,new = WiC,current + Change in WiC
CW1C = Val(txtLearnR_n.Text) * dC1 * Val(txtQuantity(0).Text)
txtWeights(60).Text = Val(txtWeights(0).Text) + CW1C
CW2C = Val(txtLearnR_n.Text) * dC1 * Val(txtQuantity(1).Text)
txtWeights(61).Text = Val(txtWeights(1).Text) + CW2C
CW3C = Val(txtLearnR_n.Text) * dC1 * Val(txtQuantity(2).Text)
txtWeights(62).Text = Val(txtWeights(2).Text) + CW3C
CW4C = Val(txtLearnR_n.Text) * dC1 * Val(txtQuantity(3).Text)
txtWeights(63).Text = Val(txtWeights(3).Text) + CW4C
CW5C = Val(txtLearnR_n.Text) * dC1 * Val(txtQuantity(4).Text)
txtWeights(64).Text = Val(txtWeights(4).Text) + CW5C
CW6C = Val(txtLearnR_n.Text) * dC1 * Val(txtQuantity(5).Text)
txtWeights(65).Text = Val(txtWeights(5).Text) + CW6C
CW7C = Val(txtLearnR_n.Text) * dC1 * Val(txtQuantity(6).Text)
txtWeights(66).Text = Val(txtWeights(6).Text) + CW7C
CW8C = Val(txtLearnR_n.Text) * dC1 * Val(txtQuantity(7).Text)
txtWeights(67).Text = Val(txtWeights(7).Text) + CW8C
CW9C = Val(txtLearnR_n.Text) * dC1 * Val(txtQuantity(8).Text)
txtWeights(68).Text = Val(txtWeights(8).Text) + CW9C
CW10C = Val(txtLearnR_n.Text) * dC1 * Val(txtQuantity(9).Text)
txtWeights(69).Text = Val(txtWeights(9).Text) + CW10C
CW11C = Val(txtLearnR_n.Text) * dC1 * Val(txtQuantity(10).Text)
txtWeights(70).Text = Val(txtWeights(10).Text) + CW11C
CW12C = Val(txtLearnR_n.Text) * dC1 * Val(txtQuantity(11).Text)
txtWeights(71).Text = Val(txtWeights(11).Text) + CW12C
CW13C = Val(txtLearnR_n.Text) * dC1 * Val(txtQuantity(12).Text)
txtWeights(72).Text = Val(txtWeights(12).Text) + CW13C
CW14C = Val(txtLearnR_n.Text) * dC1 * Val(txtQuantity(13).Text)
txtWeights(73).Text = Val(txtWeights(13).Text) + CW14C
CW15C = Val(txtLearnR_n.Text) * dC1 * Val(txtQuantity(14).Text)
txtWeights(74).Text = Val(txtWeights(14).Text) + CW15C
CW16C = Val(txtLearnR_n.Text) * dC1 * Val(txtQuantity(15).Text)
txtWeights(75).Text = Val(txtWeights(15).Text) + CW16C
CW17C = Val(txtLearnR_n.Text) * dC1 * Val(txtQuantity(16).Text)
txtWeights(76).Text = Val(txtWeights(16).Text) + CW17C
CW18C = Val(txtLearnR_n.Text) * dC1 * Val(txtQuantity(17).Text)
txtWeights(77).Text = Val(txtWeights(17).Text) + CW18C
CW19C = Val(txtLearnR_n.Text) * dC1 * Val(txtQuantity(18).Text)
txtWeights(78).Text = Val(txtWeights(18).Text) + CW19C
CW20C = Val(txtLearnR_n.Text) * dC1 * Val(txtQuantity(19).Text)
txtWeights(79).Text = Val(txtWeights(19).Text) + CW20C
CW21C = Val(txtLearnR_n.Text) * dC1 * Val(txtQuantity(20).Text)
```

```
    txtWeights(80).Text = Val(txtWeights(20).Text) + CW21C
    CW22C = Val(txtLearnR_n.Text) * dC1 * Val(txtQuantity(21).Text)
    txtWeights(81).Text = Val(txtWeights(21).Text) + CW21C
    CW23C = Val(txtLearnR_n.Text) * dC1 * Val(txtQuantity(22).Text)
    txtWeights(82).Text = Val(txtWeights(22).Text) + CW23C
    CW24C = Val(txtLearnR_n.Text) * dC1 * Val(txtQuantity(23).Text)
    txtWeights(83).Text = Val(txtWeights(23).Text) + CW24C
    CW25C = Val(txtLearnR_n.Text) * dC1 * Val(txtQuantity(24).Text)
    txtWeights(84).Text = Val(txtWeights(24).Text) + CW25C
    CW26C = Val(txtLearnR_n.Text) * dC1 * Val(txtQuantity(25).Text)
    txtWeights(85).Text = Val(txtWeights(25).Text) + CW22C
    CW27C = Val(txtLearnR_n.Text) * dC1 * Val(txtQuantity(26).Text)
    txtWeights(86).Text = Val(txtWeights(26).Text) + CW27C
    CW28C = Val(txtLearnR_n.Text) * dC1 * Val(txtQuantity(27).Text)
    txtWeights(87).Text = Val(txtWeights(27).Text) + CW28C
    CW29C = Val(txtLearnR_n.Text) * dC1 * Val(txtQuantity(28).Text)
    txtWeights(88).Text = Val(txtWeights(28).Text) + CW29C
    CW30C = Val(txtLearnR_n.Text) * dC1 * Val(txtQuantity(29).Text)
    txtWeights(89).Text = Val(txtWeights(29).Text) + CW30C
    CW0C = Val(txtLearnR_n.Text) * dC1 * Val(txtQuantity(30).Text)
    txtWeights(107).Text = Val(txtWeights(107).Text) + CW0C
    'NEXT (i) FORWARD SWEEP
    'node1
For i = 0 To 29
 txtEpochWeight(i).Text = Val(txtWeights(i).Text) * Val(txtQuantity(i).Text)
Next i
txtEpochWeight(92).Text = Val(txtWeights(105).Text) * Val(txtQuantity(30).Text)
txtNode(0).Text = Val(txtEpochWeight(92).Text) + Val(txtEpochWeight(0).Text) +
Val(txtEpochWeight(1).Text)          +          Val(txtEpochWeight(2).Text)          +
Val(txtEpochWeight(3).Text)          +          Val(txtEpochWeight(4).Text)          +
Val(txtEpochWeight(5).Text)          +          Val(txtEpochWeight(6).Text)          +
Val(txtEpochWeight(7).Text)          +          Val(txtEpochWeight(8).Text)          +
Val(txtEpochWeight(9).Text)          +          Val(txtEpochWeight(10).Text)         +
Val(txtEpochWeight(11).Text)         +          Val(txtEpochWeight(12).Text)         +
Val(txtEpochWeight(13).Text)         +          Val(txtEpochWeight(14).Text)         +
Val(txtEpochWeight(15).Text)         +          Val(txtEpochWeight(16).Text)         +
Val(txtEpochWeight(17).Text)         +          Val(txtEpochWeight(18).Text)         +
Val(txtEpochWeight(19).Text)         +          Val(txtEpochWeight(20).Text)         +
Val(txtEpochWeight(21).Text)         +          Val(txtEpochWeight(22).Text)         +
Val(txtEpochWeight(23).Text)         +          Val(txtEpochWeight(24).Text)         +
Val(txtEpochWeight(25).Text)         +          Val(txtEpochWeight(26).Text)         +
Val(txtEpochWeight(27).Text)         +          Val(txtEpochWeight(28).Text)         +
Val(txtEpochWeight(29).Text)
txtNode(0).Text = 1 / (1 + Exp(-(txtNode(0).Text)))
'node2
txtEpochWeight(30).Text = Val(txtWeights(30).Text) * Val(txtQuantity(0).Text)
txtEpochWeight(31).Text = Val(txtWeights(31).Text) * Val(txtQuantity(1).Text)
txtEpochWeight(32).Text = Val(txtWeights(32).Text) * Val(txtQuantity(2).Text)
txtEpochWeight(33).Text = Val(txtWeights(33).Text) * Val(txtQuantity(3).Text)
txtEpochWeight(34).Text = Val(txtWeights(34).Text) * Val(txtQuantity(4).Text)
txtEpochWeight(35).Text = Val(txtWeights(35).Text) * Val(txtQuantity(5).Text)
txtEpochWeight(36).Text = Val(txtWeights(36).Text) * Val(txtQuantity(6).Text)
```

```
txtEpochWeight(37).Text = Val(txtWeights(37).Text) * Val(txtQuantity(7).Text)
txtEpochWeight(38).Text = Val(txtWeights(38).Text) * Val(txtQuantity(8).Text)
txtEpochWeight(39).Text = Val(txtWeights(39).Text) * Val(txtQuantity(9).Text)
txtEpochWeight(40).Text = Val(txtWeights(40).Text) * Val(txtQuantity(10).Text)
txtEpochWeight(41).Text = Val(txtWeights(41).Text) * Val(txtQuantity(11).Text)
txtEpochWeight(42).Text = Val(txtWeights(42).Text) * Val(txtQuantity(12).Text)
txtEpochWeight(43).Text = Val(txtWeights(43).Text) * Val(txtQuantity(13).Text)
txtEpochWeight(44).Text = Val(txtWeights(44).Text) * Val(txtQuantity(14).Text)
txtEpochWeight(45).Text = Val(txtWeights(45).Text) * Val(txtQuantity(15).Text)
txtEpochWeight(46).Text = Val(txtWeights(46).Text) * Val(txtQuantity(16).Text)
txtEpochWeight(47).Text = Val(txtWeights(47).Text) * Val(txtQuantity(17).Text)
txtEpochWeight(48).Text = Val(txtWeights(48).Text) * Val(txtQuantity(18).Text)
txtEpochWeight(49).Text = Val(txtWeights(49).Text) * Val(txtQuantity(19).Text)
txtEpochWeight(50).Text = Val(txtWeights(50).Text) * Val(txtQuantity(20).Text)
txtEpochWeight(51).Text = Val(txtWeights(51).Text) * Val(txtQuantity(21).Text)
txtEpochWeight(52).Text = Val(txtWeights(52).Text) * Val(txtQuantity(22).Text)
txtEpochWeight(53).Text = Val(txtWeights(53).Text) * Val(txtQuantity(23).Text)
txtEpochWeight(54).Text = Val(txtWeights(54).Text) * Val(txtQuantity(24).Text)
txtEpochWeight(55).Text = Val(txtWeights(55).Text) * Val(txtQuantity(25).Text)
txtEpochWeight(56).Text = Val(txtWeights(56).Text) * Val(txtQuantity(26).Text)
txtEpochWeight(57).Text = Val(txtWeights(57).Text) * Val(txtQuantity(27).Text)
txtEpochWeight(58).Text = Val(txtWeights(58).Text) * Val(txtQuantity(28).Text)
txtEpochWeight(59).Text = Val(txtWeights(59).Text) * Val(txtQuantity(29).Text)
'compute sigmoid for node1
txtEpochWeight(91).Text = Val(txtWeights(106).Text) * Val(txtQuantity(30).Text)
txtNode(1).Text = Val(txtEpochWeight(91).Text) + Val(txtEpochWeight(30).Text) +
Val(txtEpochWeight(31).Text) + Val(txtEpochWeight(32).Text) +
Val(txtEpochWeight(33).Text) + Val(txtEpochWeight(34).Text) +
Val(txtEpochWeight(35).Text) + Val(txtEpochWeight(36).Text) +
Val(txtEpochWeight(37).Text) + Val(txtEpochWeight(38).Text) +
Val(txtEpochWeight(39).Text) + Val(txtEpochWeight(40).Text) +
Val(txtEpochWeight(41).Text) + Val(txtEpochWeight(42).Text) +
Val(txtEpochWeight(43).Text) + Val(txtEpochWeight(44).Text) +
Val(txtEpochWeight(45).Text) + Val(txtEpochWeight(46).Text) +
Val(txtEpochWeight(47).Text) + Val(txtEpochWeight(48).Text) +
Val(txtEpochWeight(49).Text) + Val(txtEpochWeight(50).Text) +
Val(txtEpochWeight(51).Text) + Val(txtEpochWeight(52).Text) +
Val(txtEpochWeight(53).Text) + Val(txtEpochWeight(54).Text) +
Val(txtEpochWeight(55).Text) + Val(txtEpochWeight(56).Text) +
Val(txtEpochWeight(57).Text) + Val(txtEpochWeight(58).Text) +
Val(txtEpochWeight(59).Text)
txtNode(1).Text = 1 / (1 + Exp(-(txtNode(1).Text)))
'node 2
txtEpochWeight(60).Text = Val(txtWeights(60).Text) * Val(txtQuantity(0).Text)
txtEpochWeight(61).Text = Val(txtWeights(61).Text) * Val(txtQuantity(1).Text)
txtEpochWeight(62).Text = Val(txtWeights(62).Text) * Val(txtQuantity(2).Text)
txtEpochWeight(63).Text = Val(txtWeights(63).Text) * Val(txtQuantity(3).Text)
txtEpochWeight(64).Text = Val(txtWeights(64).Text) * Val(txtQuantity(4).Text)
txtEpochWeight(65).Text = Val(txtWeights(65).Text) * Val(txtQuantity(5).Text)
txtEpochWeight(66).Text = Val(txtWeights(66).Text) * Val(txtQuantity(6).Text)
txtEpochWeight(67).Text = Val(txtWeights(67).Text) * Val(txtQuantity(7).Text)
txtEpochWeight(68).Text = Val(txtWeights(68).Text) * Val(txtQuantity(8).Text)
```

```
txtEpochWeight(69).Text = Val(txtWeights(69).Text) * Val(txtQuantity(9).Text)
txtEpochWeight(70).Text = Val(txtWeights(70).Text) * Val(txtQuantity(10).Text)
txtEpochWeight(71).Text = Val(txtWeights(71).Text) * Val(txtQuantity(11).Text)
txtEpochWeight(72).Text = Val(txtWeights(72).Text) * Val(txtQuantity(12).Text)
txtEpochWeight(73).Text = Val(txtWeights(73).Text) * Val(txtQuantity(13).Text)
txtEpochWeight(74).Text = Val(txtWeights(74).Text) * Val(txtQuantity(14).Text)
txtEpochWeight(75).Text = Val(txtWeights(75).Text) * Val(txtQuantity(15).Text)
txtEpochWeight(76).Text = Val(txtWeights(76).Text) * Val(txtQuantity(16).Text)
txtEpochWeight(77).Text = Val(txtWeights(77).Text) * Val(txtQuantity(17).Text)
txtEpochWeight(78).Text = Val(txtWeights(78).Text) * Val(txtQuantity(18).Text)
txtEpochWeight(79).Text = Val(txtWeights(79).Text) * Val(txtQuantity(19).Text)
txtEpochWeight(80).Text = Val(txtWeights(80).Text) * Val(txtQuantity(20).Text)
txtEpochWeight(81).Text = Val(txtWeights(81).Text) * Val(txtQuantity(21).Text)
txtEpochWeight(82).Text = Val(txtWeights(82).Text) * Val(txtQuantity(22).Text)
txtEpochWeight(83).Text = Val(txtWeights(83).Text) * Val(txtQuantity(23).Text)
txtEpochWeight(84).Text = Val(txtWeights(84).Text) * Val(txtQuantity(24).Text)
txtEpochWeight(85).Text = Val(txtWeights(85).Text) * Val(txtQuantity(25).Text)
txtEpochWeight(86).Text = Val(txtWeights(86).Text) * Val(txtQuantity(26).Text)
txtEpochWeight(87).Text = Val(txtWeights(87).Text) * Val(txtQuantity(27).Text)
txtEpochWeight(88).Text = Val(txtWeights(88).Text) * Val(txtQuantity(28).Text)
txtEpochWeight(89).Text = Val(txtWeights(89).Text) * Val(txtQuantity(29).Text)
'compute sigmoid for node2
txtEpochWeight(90).Text = Val(txtWeights(107).Text) * Val(txtQuantity(30).Text)
txtNode(2).Text = Val(txtEpochWeight(90).Text) + Val(txtEpochWeight(60).Text) +
Val(txtEpochWeight(61).Text) + Val(txtEpochWeight(62).Text) +
Val(txtEpochWeight(63).Text) + Val(txtEpochWeight(64).Text) +
Val(txtEpochWeight(65).Text) + Val(txtEpochWeight(66).Text) +
Val(txtEpochWeight(67).Text) + Val(txtEpochWeight(68).Text) +
Val(txtEpochWeight(69).Text) + Val(txtEpochWeight(70).Text) +
Val(txtEpochWeight(71).Text) + Val(txtEpochWeight(72).Text) +
Val(txtEpochWeight(73).Text) + Val(txtEpochWeight(74).Text) +
Val(txtEpochWeight(75).Text) + Val(txtEpochWeight(76).Text) +
Val(txtEpochWeight(77).Text) + Val(txtEpochWeight(78).Text) +
Val(txtEpochWeight(79).Text) + Val(txtEpochWeight(80).Text) +
Val(txtEpochWeight(81).Text) + Val(txtEpochWeight(82).Text) +
Val(txtEpochWeight(83).Text) + Val(txtEpochWeight(84).Text) +
Val(txtEpochWeight(85).Text) + Val(txtEpochWeight(86).Text) +
Val(txtEpochWeight(87).Text) + Val(txtEpochWeight(88).Text) +
Val(txtEpochWeight(89).Text)
txtNode(2).Text = 1 / (1 + Exp(-(txtNode(2).Text)))
'wieght adjustments for hidden layers (first forward sweep)
txtOutput1.Text = Val(txtWeights(112).Text) + Val(txtNode(0).Text) *
Val(txtWeights(90).Text) + Val(txtNode(1).Text) * Val(txtWeights(91).Text) +
Val(txtNode(2).Text) * Val(txtWeights(92).Text)
txtOutput2.Text = Val(txtWeights(111).Text) + Val(txtNode(0).Text) *
Val(txtWeights(93).Text) + Val(txtNode(1).Text) * Val(txtWeights(94).Text) +
Val(txtNode(2).Text) * Val(txtWeights(95).Text)
txtOutput3.Text = Val(txtWeights(110).Text) + Val(txtNode(0).Text) *
Val(txtWeights(96).Text) + Val(txtNode(1).Text) * Val(txtWeights(97).Text) +
Val(txtNode(2).Text) * Val(txtWeights(98).Text)
```

```
txtOutput4.Text = Val(txtWeights(109).Text) + Val(txtNode(0).Text) * Val(txtWeights(99).Text) + Val(txtNode(1).Text) * Val(txtWeights(100).Text) + Val(txtNode(2).Text) * Val(txtWeights(101).Text)
txtOutput5.Text = Val(txtWeights(108).Text) + Val(txtNode(0).Text) * Val(txtWeights(102).Text) + Val(txtNode(1).Text) * Val(txtWeights(103).Text) + Val(txtNode(2).Text) * Val(txtWeights(104).Text)
txtOutput1.Text = 1 / (1 + Exp(-(txtOutput1.Text)))
txtOutput2.Text = 1 / (1 + Exp(-(txtOutput2.Text)))
txtOutput3.Text = 1 / (1 + Exp(-(txtOutput3.Text)))
txtOutput4.Text = 1 / (1 + Exp(-(txtOutput4.Text)))
txtOutput5.Text = 1 / (1 + Exp(-(txtOutput5.Text)))
Call counter
 End Sub
Public CardOrder As order                    ‘Module Declarations
Type orditem, description As String, quantity As String, fee As Currency
skitotal As Currency
End Type                    ‘code to log patient into SOSIC
Private Sub cmdAccept_Click()
frmOrder.Show
Me.Hide
End Sub
Private Sub Form_Load()
txtNme.Text = "Type Patient's Name"
End Sub
Private Sub chkDate_Click()
If chkDate.Value = 1 Then
lblDat.Visible = True
txtTim.Visible = True
lblDate.Visible = True
ElseIf chkDate.Value = 0 Then
lblDat.Visible = False
txtTim.Visible = False
lblDate.Visible = False
End If
End Sub
Private Sub cmdInvent_Click()
frmLoginInv.Show
End Sub
Private Sub chkDiscount_Click()
If chkDiscount.Value = 1 Then
chkDiscount.Caption = "Show No of Symptoms"
lblcount.Visible = False
ElseIf chkDiscount.Value = 0 Then
chkDiscount.Caption = "Hide No of Symptoms"
lblcount.Visible = True
End If
End Sub
‘code to initialize Neural Network and intial conditions for its usage
Private Sub chkNN_Click()
If chkNN.Value = 1 Then
msg = "To use the Neural Network Algorithm, you must input observed patients symptoms."
title = "SOSIClinic Access Check"
```

```
Ans = MsgBox(msg, vbQuestion + vbOKCancel, title)
If Ans = vbOK Then
For i = chkItem.LBound To chkItem.UBound
   If chkItem(i).Value = "" Then
   msg = "To use the Neural Network Algorithm, you must input patients symptoms."
   title = "SOSIClinic Access Check"
   Ans = MsgBox(msg, vbQuestion + vbOKOnly, title)
   Exit Sub
   Else
   End If
Next i
ElseIf chkNN.Value = 0 Then
lblDat.Visible = False
txtTim.Visible = False
lblDate.Visible = False
End If
End If
End Sub
Private Sub cmdAll_Click()'selecting all symptoms
For i = chkItem.LBound To chkItem.UBound
   chkItem(i).Value = 1
Next i
End Sub
Private Sub cmdDiagnNN_Click()
Dim i As Integer
frmDiagnNN.Show
For i = 0 To 29
frmDiagnNN.txtQuantity(i).Text = frmOrder.txtQuantity(i).Text
If frmOrder.txtQuantity(i).Text = "" Then
frmDiagnNN.txtQuantity(i).Text = 0
Else
End If
Next i
Call Exflux
Call NNOutput
frmDiagnNN.Show
frmDiagnNN.Caption = "***An implementation of a Feedforward Error-Back Propagation
ANN Learning Algorithm***"
End Sub
Private Sub cmdViewT_Click()
Dim i As Integer
Dim arraynum As Integer
Dim assigncheck As Integer
On Error GoTo errhandler                    'reading patient data
frmLastShow.txtNme.Text = frmPatientname.txtNme.Text
frmLastShow.Text1.Text = frmPatientname.Text1.Text
frmLastShow.Text2.Text = frmPatientname.Text2.Text
frmLastShow.Text3.Text = frmPatientname.Text3.Text
frmLastShow.Text4.Text = frmPatientname.Text4.Text
frmLastShow.Text5.Text = frmPatientname.Text5.Text
frmLastShow.invGrid2.Clear
arraynum = 0
```

```
For i = chkItem.LBound To chkItem.UBound
If chkItem(i).Value = 1 Then arraynum = arraynum + 1
Next i
If arraynum = 0 Then
Exit Sub
Else
ReDim skiret(1 To arraynum) As orditem
End If
assigncheck = 1
For i = chkItem.LBound To chkItem.UBound
If chkItem(i).Value = 1 Then
skiret(assigncheck).description = chkItem(i).Caption
skiret(assigncheck).fee = Val(lblCost(i).Caption)
skiret(assigncheck).quantity = Val(txtQuantity(i).Text)
'skiret(assigncheck).skitotal = skiret(assigncheck).fee * skiret(assigncheck).quantity
assigncheck = assigncheck + 1
End If
Next i
If chkDiscount.Value = 1 Then
Call calculateSki(frmLastShow.invGrid2, True)
Else
Call calculateSki(frmLastShow.invGrid2, False)
End If
frmLastShow.Show
Exit Sub
errhandler:
msg = "Oops! Wrong data format, Check your entry"
title = "SOSIC error Check!"
Ans = MsgBox(msg, vbOKOnly, title)
End Sub
‘code in this section is use to open and manage diagnostic console
Private Sub Form_QueryUnload(Cancel As Integer, UnloadMode As Integer)
  Dim intRetVal As Integer
    intRetVal = MsgBox("Are you Sure you want to close the Consultation?", vbQuestion +
vbYesNoCancel + vbDefaultButton2, "Ski Action Critical Request!")
        If intRetVal = vbYes Then
        End
    ElseIf intRetVal = vbNo Then
    Cancel = True
     ElseIf intRetVal = vbCancel Then
     Cancel = True
    End If
    End Sub
Private Sub Form_QueryUnload(Cancel As Integer, UnloadMode As Integer)
  Dim intRetVal As Integer
    intRetVal = MsgBox("Are you Sure you want to close the Consultation?", vbQuestion +
vbYesNoCancel + vbDefaultButton2, "Ski Action Critical Request!")
        If intRetVal = vbYes Then
        End
    ElseIf intRetVal = vbNo Then
    Cancel = True
     ElseIf intRetVal = vbCancel Then
```

```
    Cancel = True
   End If
   End Sub
Private Sub chkHide_Click()
If chkHide.Value = 1 Then
lblWelcome2.Visible = False
lblWelcome3.Visible = False
Else
lblWelcome2.Visible = True
lblWelcome3.Visible = True
End If
End Sub
Private Sub cmdCls_Click()
Dim i As Integer
For i = chkItem.LBound To chkItem.UBound
chkItem(i).Value = 0
txtQuantity(i).Text = ""
lblCost(i).Caption = ""
Next i
End Sub
Private Sub flexgdOp()
frmLoginOpen.Show
End Sub
Private Sub gridpnt()
Dim i As Integer
Dim Ans As Integer
Dim title As String
Dim msg As String
msg = "Are you SURE you want to print the Session?"
title = "SOSIC Critical Request"
Ans = MsgBox(msg, vbQuestion + vbYesNo, title)
If Ans = vbYes Then
frmInvoice.Show
frmOrder.WindowState = 0
frmInvoice.cmdMSFlexGrid1.Caption = "Click to print"
frmInvoice.cmdMSFlexGrid1.SetFocus
Else
End If
End Sub
Private Sub Ender()
Dim i As Integer
Dim Ans As Integer
Dim title As String
Dim msg As String
msg = "Are you SURE you want to Close this session?"
title = "SOSIC Critical Request"
Ans = MsgBox(msg, vbQuestion + vbYesNo, title)
'If Ans = vbYes Then
'Open "C:\WINDOWS\Debug\progrm\loop\nzcnt\npgnt\temp.txt" For Append As #1
'Write #1, txtKi.Text
'Close #1
End
```

```
'Else
'End If
End Sub
Private Sub doctor()
Dim xlapp As Excel.Application
Dim xlbook As Excel.Workbook
Dim xlsheet As Excel.Worksheet
Dim xlrange As Excel.Range
Dim i As Integer
i = 0
i = i + 1
Set xlapp = CreateObject("Excel.Application")
xlapp.Visible = True
xlapp.WindowState = xlMaximized
xlapp.Workbooks.Add
Set xlbook = xlapp.Workbooks(i)
Set xlsheet = xlbook.Worksheets(i)
        Clipboard.Clear
        With frmInvoice.invGrid
           .Col = 0
           .row = 0
           .ColSel = .Cols - 1
           .RowSel = .Rows - 1
           Clipboard.SetText .Clip
        End With
        With xlbook.Worksheets(i)
           .Range("A1").Select
           .Paste
        End With
        xlapp.Visible = False
              Exit Sub
errhandler:
        If Not xlObject Is Nothing Then
        xlObject.Quit
        MsgBox Str$(Err) & "  " & Err.description
        Else
         MsgBox Str$(Err) & "  " & Err.description
         Exit Sub
         End If
End Sub
Public Sub calculateSki(invGrid2 As MSFlexGrid, Discount As Boolean)
Dim i As Integer
Dim RunTotal As Currency
Dim addstring2 As String
Dim TabChar As String * 1
Dim rowcounter As Integer
rowcounter = 0
TabChar = Chr(9)
Call cleargrid(invGrid2)
rowcounter = 1
For i = LBound(skiret) To UBound(skiret)
addstring2 = ""
```

```
addstring2 = skiret(i).description & TabChar
addstring2 = addstring2 & Str$(skiret(i).quantity) & TabChar
addstring2 = addstring2 & Format(skiret(i).fee) & TabChar
addstring2 = addstring2 & Format(skiret(i).skitotal, "0.00")
invGrid2.AddItem addstring2, rowcounter
rowcounter = rowcounter + 1
Next
RunTotal = 0
For i = LBound(skiret) To UBound(skiret)
RunTotal = RunTotal + skiret(i).skitotal
Next i
invGrid2.row = rowcounter - 1
invGrid2.Col = 3
invGrid2.CellFontUnderline = True
If Discount Then
invGrid2.row = rowcounter + 1
invGrid2.Col = 0
invGrid2.CellFontUnderline = True
invGrid2.Text = "Total(N): "
invGrid2.row = rowcounter + 1
invGrid2.Col = 3
invGrid2.CellFontUnderline = True
invGrid2.Text = "N" & Format(RunTotal, "0.00")
invGrid2.CellBackColor = &H80FF&
invGrid2.row = rowcounter + 3
invGrid2.Col = 0
invGrid2.CellFontUnderline = True
invGrid2.Text = "Total wDiscount: "
invGrid2.CellBackColor = &HFFFF00
invGrid2.row = rowcounter + 3
invGrid2.Col = 3
invGrid2.CellFontBold = True
invGrid2.Text = "N" & Format(RunTotal * 0.1, "0.00")
invGrid2.CellBackColor = &HFFFF00
invGrid2.row = rowcounter + 5
invGrid2.Col = 0
invGrid2.CellFontBold = True
invGrid2.Text = "Date: "
invGrid2.row = rowcounter + 5
invGrid2.Col = 3
invGrid2.CellFontBold = True
invGrid2.Text = Date
frmLastShow.txtDate.Text = invGrid2.Text
invGrid2.CellBackColor = &HFFFF00
invGrid2.row = rowcounter + 6
invGrid2.Col = 0
invGrid2.CellFontBold = True
invGrid2.Text = "Time "
invGrid2.row = rowcounter + 6
invGrid2.Col = 3
invGrid2.CellFontBold = True
invGrid2.Text = Time
```

```
frmLastShow.txtTime.Text = invGrid2.Text
Else
invGrid2.row = rowcounter + 1
invGrid2.Col = 0
invGrid2.CellFontBold = True
invGrid2.Text = "Total: "
invGrid2.row = rowcounter + 1
invGrid2.Col = 3
invGrid2.CellFontBold = True
invGrid2.Text = "N" & Format(RunTotal, "0.00")
invGrid2.CellBackColor = &H80FF&
invGrid2.row = rowcounter + 2
invGrid2.Col = 0
invGrid2.CellFontBold = True
invGrid2.Text = "Date: "
invGrid2.row = rowcounter + 2
invGrid2.Col = 3
invGrid2.CellFontBold = True
invGrid2.Text = Date
frmLastShow.txtDate.Text = invGrid2.Text
invGrid2.CellBackColor = &HFFFF00
invGrid2.row = rowcounter + 3
invGrid2.Col = 0
invGrid2.CellFontBold = True
invGrid2.Text = "Time "
invGrid2.row = rowcounter + 3
invGrid2.Col = 3
invGrid2.CellFontBold = True
invGrid2.Text = Time
frmLastShow.txtTime.Text = invGrid2.Text
invGrid2.CellBackColor = &HFFFF00
End If
ReDim skiret(0)
End Sub
Public Sub cleargrid(invGrid2 As MSFlexGrid)
frmLastShow.invGrid2.Clear
Call addheaders(invGrid2)
End Sub
Private Sub FlextoExcel()
    Dim xlObject As Excel.Application
    Dim xlWB As Excel.Workbook
    Dim FileName As String
    Static counter As Integer
        Dim stella As String * 2
        Dim stel As String * 5
        stella = txtTim
        txtHr.Text = stella
        txtTmRv = StrReverse(txtTim)
        stel = txtTmRv
        txtRvTm = StrReverse(stel)
        txtShw = lblDat & "_" & txtHr & "-" & txtRvTm
```

```
          txtVid.Text = "Ski_Report_" & Format$(Now, "d-mmmm-yy") & "_" & txtHr & "." &
txtRvTm & ".xls"
          counter = counter + 1
        On Error GoTo errhandler:
        Set xlObject = New Excel.Application
        Set xlWB = xlObject.Workbooks.Add
        Clipboard.Clear
        With frmInvoice.invGrid
           .Col = 0
           .row = 0
           .ColSel = .Cols - 1
           .RowSel = .Rows - 1
           Clipboard.SetText .Clip
        End With
        With xlWB.ActiveSheet
           .Range("A1").Select
           .Paste
           End With
        xlObject.Visible = False
             FileName     =     "C:\A_PROJECT\investCARD\Daily_Report\Ski_Report_"     &
Format$(Now, "d-mmmm-yy") & "_" & txtHr & "." & txtRvTm & ".xls"
        ActiveSheet.SaveAs (FileName)
             xlObject.Quit
        Set xlObject = Nothing
         Set xlsheet = Nothing
        Set xlWB = Nothing
        Open     "C:\WINDOWS\Debug\progrm\global\dentifiers\bESSEL\exc-XLS.txt"     For
Append As #1
        Write #1, txtVid.Text,
        Exit Sub
errhandler:
        If Not xlObject Is Nothing Then
        xlObject.Quit
        MsgBox Str$(Err) & "  " & Err.description
        Else
        MsgBox Str$(Err) & "  " & Err.description
         Exit Sub
         End If
End Sub
Private Sub MedDoctor()
Dim i As Integer
Dim diagcalc As Single
Dim EIJ As Single
Dim con1 As Integer
Dim con2 As Integer
Dim con3 As Integer
diagcalc = 0
For i = lblCost.LBound To lblCost.UBound
diagcalc = diagcalc + Val(lblCost(i).Caption)
Next i
diagcalc = diagcalc / Val(lblcount.Caption)
  txtDiagCalc.Text = diagcalc
```

```
      'Detect Malaria
    ' Body temperature, fever
If chkItem(0).Value = 1 Or chkItem(12).Value = 1 Then
    ' Headache, weakness
    If chkItem(5).Value = 1 Or chkItem(14).Value = 1 Then
    ' loss of appetite, flu like symptoms
    If chkItem(24).Value = 1 Or chkItem(25).Value = 1 Then
       'AILMENT DIAGNOSIS
         frmDiagnostic.lblDiag.Caption = "Malaria is Suspected."
         frmDiagnostic.lblConf.Caption = diagcalc
         lblMalaria.Caption = "MAL"
         'Compute error in judgement
         EIJ = 1 / Val(lblcount.Caption)
         EIJ = EIJ * 100
         frmDiagnostic.lblEIJ.Caption = EIJ
          'action to take
         If diagcalc >= 0.7 Then
         frmDiagnostic.lblDiagAction.Caption = "Admission is Recommended"
           'dengue fever
           'Nausea
           If diagcalc > 0.7 And Option61.Value = True Then
            frmDiagnostic.lblDiag.Caption = "Dengue Fever Suspected."
            Else
            End If
            'Yellow Fever
            If diagcalc > 0.5 And Option64.Value = True Then
            frmDiagnostic.lblDiag.Caption = "Yellow Fever Suspected."
            Else
            End If
         ElseIf diagcalc >= 0.5 Then
         frmDiagnostic.lblDiagAction.Caption = "Admission Not Recommended"
         ElseIf diagcalc < 0.5 Then
         frmDiagnostic.lblDiagAction.Caption = "Case is Trivial"
         End If
   End If
   End If
  End If
    'detect typhiod
       'vomiting
        If chkItem(1).Value = 1 And lblMalaria.Caption = "MAL" Then
   'high blood pressure,COUGHING
        If chkItem(4).Value = 1 Or chkItem(3).Value = 1 Then
       'AILMENT DIAGNOSIS
         frmDiagnostic.lblDiag.Caption = "Severe Case of Typhoid"
         frmDiagnostic.lblConf.Caption = diagcalc
         'Compute error in judgement
         EIJ = 1 / Val(lblcount.Caption)
         EIJ = EIJ * 100
         frmDiagnostic.lblEIJ.Caption = EIJ
          'action to take
             If diagcalc >= 0.7 Then
             frmDiagnostic.lblDiagAction.Caption = "Admission is Required"
```

```
            ElseIf diagcalc >= 0.5 Then
            frmDiagnostic.lblDiagAction.Caption = "Admission Not Recommended"
            ElseIf diagcalc < 0.5 Then
            frmDiagnostic.lblDiagAction.Caption = "Case is Trivial"
            End If
        End If
End If
   'detect InfluenzaFlu
    'MusclePain; & chest pain
     If chkItem(11).Value = 1 And Option58.Value = True Then
'Flu-Like Syptoms, Chills/Rigors
     If chkItem(25).Value = 1 And Option132.Value = True Then
    ' COUGHING, fever
     If chkItem(3).Value = 1 Or chkItem(12).Value = 1 Then
          'AILMENT DIAGNOSIS
      frmDiagnostic.lblDiag.Caption = "Acute Respiratory Infection"
      frmDiagnostic.imgInfluenza.Visible = True
      frmDiagnostic.imgEVD.Visible = False
      frmDiagnostic.Image1.Visible = False
      frmDiagnostic.lblConf.Caption = diagcalc
      'Compute error in judgement
      EIJ = 1 / Val(lblcount.Caption)
      EIJ = EIJ * 100
      frmDiagnostic.lblEIJ.Caption = EIJ
       'action to take
          If diagcalc >= 0.7 Then
          frmDiagnostic.lblDiagAction.Caption = "Admission is Required"
          ElseIf diagcalc >= 0.5 Then
          frmDiagnostic.lblDiagAction.Caption = "Admission Not Recommended"
          ElseIf diagcalc < 0.5 Then
          frmDiagnostic.lblDiagAction.Caption = "Case is Trivial"
          frmDiagnostic.lblDiag.Caption = "Cholera is Suspected"
          End If
      End If
End If
End If
'detect cholera
    'diarrhea; & high fluid loss
     If chkItem(10).Value = 1 And chkItem(20).Value = 1 Then
'high blood pressure,COUGHING
     If chkItem(4).Value = 1 Or chkItem(3).Value = 1 Then
     'Gastrointestinal pain, weakness
     If chkItem(26).Value = 1 Or chkItem(14).Value = 1 Then
          'AILMENT DIAGNOSIS
      frmDiagnostic.lblDiag.Caption = "Severe Case of Cholera"
      frmDiagnostic.lblConf.Caption = diagcalc
      'Compute error in judgement
      EIJ = 1 / Val(lblcount.Caption)
      EIJ = EIJ * 100
      frmDiagnostic.lblEIJ.Caption = EIJ
       'action to take
          If diagcalc >= 0.7 Then
```

```
            frmDiagnostic.lblDiagAction.Caption = "Admission is Required"
            ElseIf diagcalc >= 0.5 Then
            frmDiagnostic.lblDiagAction.Caption = "Admission Not Recommended"
            ElseIf diagcalc < 0.5 Then
            frmDiagnostic.lblDiagAction.Caption = "Case is Trivial"
            frmDiagnostic.lblDiag.Caption = "Cholera is Suspected"
            End If
        End If
           End If
           End If
          'detect ebola
    'Sore Throat
    If chkItem(7).Value = 1 And lblMalaria.Caption = "MAL" Then
      'vomit; eye redness
       If chkItem(4).Value = 1 Or chkItem(6).Value = 1 Then
       'Diarrhea, Ext. Bleeding
       If chkItem(10).Value = 1 Or chkItem(13).Value = 1 Then
       'Int. Bleeding, Muscle Pain
        If chkItem(9).Value = 1 Or chkItem(11).Value = 1 Then
        'AbdominalPain; SkinRashes
        If chkItem(26).Value = 1 Or chkItem(8).Value = 1 Then
      'AILMENT DIAGNOSIS
        frmDiagnostic.lblDiag.Caption = "EVD is suspected!"
        frmDiagnostic.lblDiag.ForeColor = vbRed
        frmDiagnostic.imgEVD.Visible = True
          frmDiagnostic.imgInfluenza.Visible = False
           frmDiagnostic.Image1.Visible = False
        frmDiagnostic.lblConf.Caption = diagcalc
               'Contribution of fever to diagnosis
        If lblCost(12).Caption = "" Or lblCost(12).Caption = " " Then
        msg = "We are Suspecting EVD/LASSA Fever here but will need an idea of the Patient's Fever!"
        title = "SOSIC Fever Check!"
        Ans = MsgBox(msg, vbExclamation + vbOK, title)
        Exit Sub
        Else
        End If
        'Patients prior location
        If lblCost(17).Caption = "" Or lblCost(17).Caption = " " Then
        msg = "We are Suspecting EVD/Marburg or LASSA Fever but will need info on Patient's Prior Locality!"
        title = "SOSIC Locale Check!"
        Ans = MsgBox(msg, vbExclamation + vbOK, title)
        Exit Sub
        Else
        End If
        'Compute error in judgement
        EIJ = 1 / Val(lblcount.Caption)
        EIJ = EIJ * 100
        frmDiagnostic.lblEIJ.Caption = EIJ
         'action to take
        If diagcalc >= 0.7 Then
```

```
          frmDiagnostic.lblDiagAction.Caption = "Quarantine Patient Now!"
          frmDiagnostic.lblDiagAction.ForeColor = vbRed
          ElseIf diagcalc >= 0.5 Then
          frmDiagnostic.lblDiagAction.Caption = "Admission is Recommended"
            frmDiagnostic.lblDiagAction.ForeColor = vbRed
          ElseIf diagcalc < 0.5 Then
          frmDiagnostic.lblDiagAction.Caption = "Early signs of EVD/Lassa Fever!"
            frmDiagnostic.lblDiagAction.ForeColor = vbRed
          End If
        End If
        End If
        End If
        End If
        ' Lassa Fever diagnosis
        'bloody vomiting and SevereInternalBleeding.
        If Option22.Value = True Or Option49.Value = True Then
        ' InternalPain, SuddenFever
           If Option50.Value = True Or Option65.Value = True Then
         ' SevereAbdominalPain, Chills/Rigors
            If Option126.Value = True Or Option132.Value = True Then
          frmDiagnostic.lblDiag.Caption = "Lassa Fever is suspected!"
          frmDiagnostic.lblDiag.ForeColor = vbRed
          frmDiagnostic.imgEVD.Visible = True
          frmDiagnostic.lblConf.Caption = diagcalc
       Else
        End If
        End If
        End If
        'Detect MARBURG Heamorrhagic fever
        'Rashes:Chest&Back, Flu-like Syptoms: Chills
        If Option42.Value = True And Option133.Value = True Then
          frmDiagnostic.lblDiag.Caption = "MARBURG is suspected!"
          frmDiagnostic.lblDiag.ForeColor = vbRed
          frmDiagnostic.imgEVD.Visible = True
          frmDiagnostic.lblConf.Caption = diagcalc
        Else
        End If
    End If
'More ailments
'confidence level of action
        lblCon1.Caption = "cs"
        lblCon2.Caption = "cs"
        lblCon3.Caption = "cs"
        con1 = 0
        con2 = 0
        con3 = 0
        For i = lblCost.LBound To lblCost.UBound
            If Val(lblCost(i).Caption) >= 0.7 Then
            con1 = con1 + 1
            lblCon1.Caption = con1
ElseIf Val(lblCost(i).Caption) >= 0.5 And Val(lblCost(i).Caption) < 0.7 Then
            con2 = con2 + 1
```

```
            lblCon2.Caption = con2
             ElseIf Val(lblCost(i).Caption) > 0.2 And Val(lblCost(i).Caption) < 0.5 Then
            con3 = con3 + 1
            lblCon3.Caption = con3
            End If
          Next i
          con = 0.9 * con1 / Val(lblcount.Caption) + 0.7 * con2 / Val(lblcount.Caption) + 0.5 *
con3 / Val(lblcount.Caption)
        frmDiagnostic.lblConf2.Caption = con
      frmDiagnostic.Show
'inputted symptoms may not be enough for a Proper Diagnosis
If frmDiagnostic.Visible = False Then
 msg = " Inputted Symptoms are not ENOUGH for a Proper Diagnosis! Retry"
      title = "SOSIC Symptoms Check!"
      Ans = MsgBox(msg, vbExclamation + vbOK, title)
Else
End If
End Sub
Private Sub Exflux()
Dim i As Integer
Dim Ans As Integer
Dim count As Integer
Dim title As String
Dim msg As String
Dim arraynum As Integer
Dim numitems As Integer
Dim Lineitem As OrderItem
numitems = 0
'On Error GoTo errhandler
msg = "Are you sure of these Symptoms?"
title = "SOSIC Check!"
Ans = MsgBox(msg, vbQuestion + vbYesNo, title)
      txtTim.Text = Format$(Now, "hh:mm AM/PM")
      'count symptoms
    For i = chkItem.LBound To chkItem.UBound
        If Not lblCost(i).Caption = "" Then
        count = count + 1
        lblcount.Caption = count
        Else
        End If
    Next i
    If count = 0 Then
      msg = "No Symptom has been Selected for a VALID Diagnosis"
      title = "SOSIC Check!"
      Ans = MsgBox(msg, vbExclamation + vbOKCancel, title)
      Exit Sub
      Else
      End If
    If count < 4 Then
      msg = "These Symptoms are not ENOUGH for a VALID Diagnosis"
      title = "SOSIC Check!"
      Ans = MsgBox(msg, vbInformation + vbOKCancel, title)
```

```
        Exit Sub
        Else
        End If
  If Ans = vbYes Then
        For i = chkItem.LBound To chkItem.UBound
        If chkItem(i).Value = 1 Then numitems = numitems + 1
        Next i
        If numitems = 0 Then
        Exit Sub
        Else
        For i = chkItem.LBound To chkItem.UBound
        If chkItem(i).Value = 1 Then
        Set Lineitem = New OrderItem
        Lineitem.ProductName = chkItem(i).Caption
        Lineitem.UnitCost = Val(lblCost(i).Caption)
        Lineitem.quantity = Val(txtQuantity(i).Text)
        CardOrder.addorderitem Lineitem
        End If
        Next i
        End If
        Call MedDoctor
        Call CardOrder.DisplayOrder
Else
Exit Sub
End If
Exit Sub
errhandler:
msg = "Oops! Wrong data format, Check your entry"
title = "SOSIC error Check!"
Ans = MsgBox(msg, vbOKOnly, title)
End Sub
Private Sub NNOutput()
If frmDiagnostic.lblDiag.Caption = "Malaria is Suspected." Then
frmDiagnNN.txtActual(0).Text = "0.9"
ElseIf frmDiagnostic.lblDiag.Caption = "Acute Respiratory Infection" Then
frmDiagnNN.txtActual(1).Text = "0.9"
ElseIf frmDiagnostic.lblDiag.Caption = "EVD is suspected!" Then
frmDiagnNN.txtActual(2).Text = "0.9"
ElseIf frmDiagnostic.lblDiag.Caption = "Lassa Fever is suspected!" Then
frmDiagnNN.txtActual(3).Text = "0.9"
ElseIf frmDiagnostic.lblDiag.Caption = "MARBURG is suspected!" Then
frmDiagnNN.txtActual(4).Text = "0.9"
End If
End Sub
'frmDiagnositic codes
Private Sub chkEIJ_Click()
If chkEIJ.Value = 1 Then
lblEIJ.Visible = True
ElseIf chkEIJ.Value = 0 Then
lblEIJ.Visible = False
End If
End Sub
```

```
‘frmDiagnNN code
Dim msg As String
Dim title As String
Dim Ans As Integer
Dim StartTime As Date
Dim EndTime As Date
Dim ElapsedTime As Date
Dim time01 As Date, time02 As Date
Public Sub Rado()
Dim Listside As Integer
Dim difcalc As Integer
difcalc = txtMaxV.Text
On Error GoTo errWeight:
Dim i As Integer
For i = 0 To 104
Do
      MyNum = Int(Rnd * Val(txtMaxV.Text)) * 0.04 + 0.14
      txtWeights(i).Text = (MyNum)
      Exit Do
   Loop
Next
Exit Sub
errWeight:
msg = "Oops! Check inputted value of Weight range!"
title = "SOSICK NN Access Check!"
Ans = MsgBox(msg, vbExclamation + vbOKOnly, title)
txtMaxV.SetFocus
Exit Sub
End Sub
Private Sub Check1_Click()
If chkNb.Value = 1 Then
lblNb.Visible = True
ElseIf chkNb.Value = 0 Then
lblNb.Visible = False
End If
End Sub
Private Sub chkHide_Click()
Call WHide
End Sub
Private Sub chkLRandm_Click()
If chkLRandm.Value = 1 Then
Command3.Visible = True
Command1.Visible = True
Command4.Visible = True
ElseIf chkLRandm.Value = 0 Then
Command3.Visible = False
Command1.Visible = False
Command4.Visible = False
End If
End Sub
Private Sub chkTrainOutput_Click()
Dim i As Integer
```

```
If chkTrainOutput.Value = 1 Then
chkTrainOutput.Caption = "Hide Training Outputs"
txtTrainOut1.Visible = True
txtTrainOut2.Visible = True
txtTrainOut3.Visible = True
txtTrainOut4.Visible = True
txtTrainOut5.Visible = True
txtTrainOut1.Text = txtOutput1.Text
txtTrainOut2.Text = txtOutput2.Text
txtTrainOut3.Text = txtOutput3.Text
txtTrainOut4.Text = txtOutput4.Text
txtTrainOut5.Text = txtOutput5.Text
Label41.Visible = True
ElseIf chkHide.Value = 0 Then
chkTrainOutput.Caption = "Show Training Outputs"
txtTrainOut1.Visible = False
txtTrainOut2.Visible = False
txtTrainOut3.Visible = False
txtTrainOut4.Visible = False
txtTrainOut5.Visible = False
Label41.Visible = False
End If
End Sub
Private Sub cmdGen_Click()
Call Rado
End Sub
Private Sub PlotOutput()
Call List2Redraw
Dim i As Integer, n   As Integer, X() As Single, Y() As Single
   n = Val(txtNoInputs.Text)
   ReDim X(n - 1), Y(n - 1)
      For i = 0 To n - 1
      X(i) = i
      Y(i) = List2.List(i)
   Next i
    Picture10.ForeColor = vbGreen
   Call LineChart(Picture10, n, X, Y)
   End Sub
Private Sub PlotWeight()
Call List1Redraw
Dim i As Integer, n As Integer, X() As Single, Y() As Single
   n = Val(txtNoInputs.Text)
   ReDim X(n - 1), Y(n - 1)
      For i = 0 To n - 1
      X(i) = i
      Y(i) = List1.List(i)
   Next i
    Picture1.ForeColor = vbWhite
   Call LineChart(Picture1, n, X, Y)
End Sub
Private Sub LineChart(ObjectName As Control, n As Integer, X() As Single, Y() As Single)
On Error GoTo errObot:
```

```
    Dim Xmin As Single, Xmax As Single
    Dim Ymin As Single, Ymax As Single
    Dim i As Integer
      Xmin = X(0): Xmax = X(0)
    Ymin = Y(0): Ymax = Y(0)
       For i = 0 To n - 1
       If X(i) < Xmin Then Xmin = X(i)
       If X(i) > Xmax Then Xmax = X(i)
       If Y(i) < Ymin Then Ymin = Y(i)
       If Y(i) > Ymax Then Ymax = Y(i)
    Next i
    Ymin = (1 - 2.05 * Sgn(Ymin)) * Ymin ' Extend Ymin by 5 percent
    Ymax = (2 + 0.05 * Sgn(Ymax)) * Ymax ' Extend Ymax by 5 percent
    ObjectName.Scale (Xmin, Ymax)-(Xmax, Ymin)   'define co-ord
    ObjectName.Cls 'clear object
    ObjectName.PSet (X(0), Y(0)) 'set first point
       For i = 0 To n - 1
       ObjectName.Line -(X(i), Y(i)) 'draw subsequent lines
    Next i
 Exit Sub
errObot:
msg = "Please select New weights before Initialization!"
title = "SOSIC Iterator Check"
Ans = MsgBox(msg, vbOK + vbCritical, title)
Exit Sub
End Sub
    'Picture1.ForeColor = vbRed
Private Sub List1Redraw()
Dim i As Integer
For i = 0 To 89
List1.AddItem txtWeights(i).Text
Next i
End Sub
'Picture1.ForeColor = vbblue
Private Sub List2Redraw()
Dim i As Integer
For i = 0 To 89
List2.AddItem txtWeights(i).Text
Next i
End Sub
Private Sub cmdGetVal_Click()
'new inputs
Dim i As Integer
For i = 0 To 29
txtVal_Inp(i).Text = txtQuantity(i).Text
Next i
'add weights from training set
Call ValW
'code to populate listbox for new Inputs
Call InputL
End Sub
Private Sub cmdKNN_Click()
```

```
frmKNN.Show
End Sub
Private Sub cmdRelate_Click()
Call PlotInp
If frmPlot.Picture3.Visible = True Then
frmPlot.Picture3.Visible = False
frmPlot.Picture9.Visible = True
Call PlotInpNew2
ElseIf frmPlot.Picture9.Visible = True Then
frmPlot.Picture9.Visible = False
frmPlot.Picture10.Visible = True
Call PlotInpNew3
Else
Call PlotInpNew
frmPlot.Picture3.Visible = True
End If
End Sub
Private Sub PlotInp()
Dim i As Integer, n   As Integer, X() As Single, Y() As Single
   n = Val(txtNoInputs2.Text)
   ReDim X(n - 1), Y(n - 1)
      For i = 0 To n - 1
      X(i) = i
      Y(i) = List4.List(i)
   Next i
     frmPlot.Picture7.ForeColor = vbGreen
   Call LineChart(frmPlot.Picture7, n, X, Y)
   frmPlot.Show
End Sub
Private Sub PlotInpNew()
Dim i As Integer, n As Integer, X() As Single, Y() As Single
   n = Val(txtNoInputs2.Text)
   ReDim X(n - 1), Y(n - 1)
      For i = 0 To n - 1
      X(i) = i
      Y(i) = List5.List(i)
   Next i
     frmPlot.Picture3.ForeColor = vbBlue
   Call LineChart(frmPlot.Picture3, n, X, Y)
End Sub
Private Sub PlotInpNew2()
Dim i As Integer, n As Integer, X() As Single, Y() As Single
   n = Val(txtNoInputs2.Text)
   ReDim X(n - 1), Y(n - 1)
      For i = 0 To n - 1
      X(i) = i
      Y(i) = List5.List(i)
   Next i
     frmPlot.Picture9.ForeColor = vbBlue
   Call LineChart(frmPlot.Picture9, n, X, Y)
End Sub
Private Sub PlotInpNew3()
```

```
Dim i As Integer, n As Integer, X() As Single, Y() As Single
   n = Val(txtNoInputs2.Text)
   ReDim X(n - 1), Y(n - 1)
      For i = 0 To n - 1
      X(i) = i
      Y(i) = List5.List(i)
   Next i
     frmPlot.Picture10.ForeColor = vbBlue
   Call LineChart(frmPlot.Picture10, n, X, Y)
End Sub
Private Sub cmdReliabl_Click()
If optManual.Value = True Then
Call Iterator
Else
msg = "You are on Automatic Iteration Mode!"
title = "SOSIC Iterator Check!"
Ans = MsgBox(msg, vbOK + vbInformation, title)
End If
End Sub
Private Sub cmdVal_Click()
Call ValIterate
End Sub
Private Sub Command1_Click()
Picture1.Cls 'clear object
End Sub
Private Sub Command3_Click()
txtQuantity(0).Text = "0.4"
txtQuantity(1).Text = "0.2"
txtQuantity(2).Text = "0.7"
txtWeights(0).Text = "0.6"
txtWeights(1).Text = "0.8"
txtWeights(2).Text = "0.6"
txtWeights(30).Text = "0.9"
txtWeights(31).Text = "0.8"
txtWeights(32).Text = "0.4"
txtWeights(90).Text = "0.9"
txtWeights(91).Text = "0.9"
txtActual(0).Text = "0.8"
End Sub
Private Sub Command4_Click()
For i = 0 To 29
Do
      MyNumm = Int(Rnd * 12) * 0.04 + 0.19
      txtQuantity(i).Text = (MyNumm)
      Exit Do
   Loop
Next
End Sub
Private Sub Form_Load()
   lblStT1.Caption = ""
   'Text3.Text = Time
   StartTime = Time
```

```
    Timer1.Interval = 1000    'interval of 1 second
    Timer1.Enabled = True
  Call WtAdjust
  Call WHide
End Sub
Private Sub lblStT1_Change()
If optStop.Value = True Then
Call Elama
optStop.Value = False
End If
If optAutomatic.Value = True And optTime.Value = True Then
 If lblELT2.Caption = "00:00:00" Then
    StartTime = Time
    Timer2.Interval = 1000    'interval of 1 second
    ElapsedTime = 0
    Timer2.Enabled = True
Else
End If
End If
If optAutomatic.Value = True Then
If optEpoch.Value = True And lblOutput.Caption = txtEpoch.Text Then
Call Elama
optEpoch.Value = False
ElseIf optTime.Value = True And lblELT2.Caption = txtTime.Text Then
Call Elama
ElseIf optConverge.Value = True And txtOutput1.Text = Val(txtActual(0).Text) Then
If txtOutput2.Text = Val(txtActual(1).Text) And txtOutput3.Text = Val(txtActual(2).Text)
Then
If txtOutput4.Text = Val(txtActual(3).Text) And txtOutput5.Text = Val(txtActual(4).Text)
Then
Call Elama
End If
End If
Else
Call Iterator
End If
Else
End If

End Sub
Private Sub Timer1_Timer()
    'Use Long Time format to display seconds
    time01 = Format(Time, "Long Time")
    lblStT1.Caption = time01
   ' time02 = Format(Time - StartTime, "Long Time")
    'lblELT2.Caption = Format(time02, "HH:MM:SS")
End Sub
Private Sub Timer2_Timer()
    'Use Long Time format to display seconds
    time02 = Format(Time - StartTime, "Long Time")
    lblELT2.Caption = Format(time02, "HH:MM:SS")
End Sub
```

```
Private Sub Iterator()
If lblOutput.Caption = "0" Then
Call Iterate
Else
Call IterateTwo
End If
End Sub
Private Sub Elama()
msg = "The Iteration is Completed after " & lblOutput.Caption & " Epoches! Do you still want Weights fine tuned?"
title = "SOSIC Iterator Check --Final"
Ans = MsgBox(msg, vbYesNo + vbInformation, title)
If Ans = vbNo Then
optManual.Value = True
'plot final wieghts
Call PlotOutput
If optTime.Value = True And optAutomatic.Value = True Then
Timer2.Enabled = False
optTime.Value = False
Else
End If
Else
End If
Exit Sub
End Sub
Private Sub WtAdjust()
Dim i As Integer
For i = 0 To 89
txtEpochWeight(i).FontSize = 8
txtEpochWeight(i).Height = 300
Next i
End Sub
Private Sub WeightList()
Dim i As Integer
For i = 0 To 89
List3.AddItem "Weights " & i
Next i
Call InputList
End Sub
Private Sub WHide()
Dim i As Integer
If chkHide.Value = 1 Then
chkHide.Caption = "Show Weights"
'Hide all weights
For i = 0 To 89
 txtEpochWeight(i).Visible = False
 txtWeights(i).Visible = False
Next i
txtWeights(105).Visible = False
 txtWeights(106).Visible = False
 txtWeights(107).Visible = False
txtEpochWeight(90).Visible = False
```

```
txtEpochWeight(91).Visible = False
txtEpochWeight(92).Visible = False
ElseIf chkHide.Value = 0 Then
For i = 0 To 89
 txtEpochWeight(i).Visible = True
 txtWeights(i).Visible = True
Next i
chkHide.Caption = "Hide Weights"
txtWeights(105).Visible = True
 txtWeights(106).Visible = True
 txtWeights(107).Visible = True
txtEpochWeight(90).Visible = True
txtEpochWeight(91).Visible = True
txtEpochWeight(92).Visible = True
End If
End Sub
Private Sub InputList()
Dim i As Integer
'add output from GUI To NN inputs
For i = 0 To 29
List4.AddItem txtQuantity(i).Text
Next i
End Sub
Private Sub InputL()
Dim i As Integer
'add output from GUI To NN inputs
For i = 0 To 29
List5.AddItem txtVal_Inp(i).Text
Next i
End Sub
Private Sub ValW()
Dim i As Integer
'add output from GUI To NN inputs
For i = 0 To 112
txtVal_Weights(i).Text = txtWeights(i).Text
Next i
End Sub
Private Sub ValIterate()
Dim i As Integer
'wieght adjustments for input layers (first forward sweep)
'node1
For i = 0 To 29
 txtVal_Weights(i).Text = Val(txtVal_Weights(i).Text) * Val(txtVal_Inp(i).Text)
Next i
txtVal_Weights(90).Text = Val(txtVal_Weights(90).Text) * Val(txtVal_Inp(30).Text)
txtNodal(0).Text = Val(txtVal_Weights(92).Text) + Val(txtVal_Weights(0).Text) +
Val(txtVal_Weights(1).Text) + Val(txtVal_Weights(2).Text) + Val(txtVal_Weights(3).Text)
+ Val(txtVal_Weights(4).Text) + Val(txtVal_Weights(5).Text) +
Val(txtVal_Weights(6).Text) + Val(txtVal_Weights(7).Text) + Val(txtVal_Weights(8).Text)
+ Val(txtVal_Weights(9).Text) + Val(txtVal_Weights(10).Text) +
Val(txtVal_Weights(11).Text) + Val(txtVal_Weights(12).Text) +
Val(txtVal_Weights(13).Text) + Val(txtVal_Weights(14).Text) +
```

```
Val(txtVal_Weights(15).Text)  +  Val(txtVal_Weights(16).Text)  +
Val(txtVal_Weights(17).Text)  +  Val(txtVal_Weights(18).Text)  +
Val(txtVal_Weights(19).Text)  +  Val(txtVal_Weights(20).Text)  +
Val(txtVal_Weights(21).Text)  +  Val(txtVal_Weights(22).Text)  +
Val(txtVal_Weights(23).Text)  +  Val(txtVal_Weights(24).Text)  +
Val(txtVal_Weights(25).Text)  +  Val(txtVal_Weights(26).Text)  +
Val(txtVal_Weights(27).Text)  +  Val(txtVal_Weights(28).Text)  +
Val(txtVal_Weights(29).Text)
txtNodal(0).Text = 1 / (1 + Exp(-(txtNodal(0).Text)))
'node2
txtVal_Weights(30).Text = Val(txtVal_Weights(30).Text) * Val(txtVal_Inp(0).Text)
txtVal_Weights(31).Text = Val(txtVal_Weights(31).Text) * Val(txtVal_Inp(1).Text)
txtVal_Weights(32).Text = Val(txtVal_Weights(32).Text) * Val(txtVal_Inp(2).Text)
txtVal_Weights(33).Text = Val(txtVal_Weights(33).Text) * Val(txtVal_Inp(3).Text)
txtVal_Weights(34).Text = Val(txtVal_Weights(34).Text) * Val(txtVal_Inp(4).Text)
txtVal_Weights(35).Text = Val(txtVal_Weights(35).Text) * Val(txtVal_Inp(5).Text)
txtVal_Weights(36).Text = Val(txtVal_Weights(36).Text) * Val(txtVal_Inp(6).Text)
txtVal_Weights(37).Text = Val(txtVal_Weights(37).Text) * Val(txtVal_Inp(7).Text)
txtVal_Weights(38).Text = Val(txtVal_Weights(38).Text) * Val(txtVal_Inp(8).Text)
txtVal_Weights(39).Text = Val(txtVal_Weights(39).Text) * Val(txtVal_Inp(9).Text)
txtVal_Weights(40).Text = Val(txtVal_Weights(40).Text) * Val(txtVal_Inp(10).Text)
txtVal_Weights(41).Text = Val(txtVal_Weights(41).Text) * Val(txtVal_Inp(11).Text)
txtVal_Weights(42).Text = Val(txtVal_Weights(42).Text) * Val(txtVal_Inp(12).Text)
txtVal_Weights(43).Text = Val(txtVal_Weights(43).Text) * Val(txtVal_Inp(13).Text)
txtVal_Weights(44).Text = Val(txtVal_Weights(44).Text) * Val(txtVal_Inp(14).Text)
txtVal_Weights(45).Text = Val(txtVal_Weights(45).Text) * Val(txtVal_Inp(15).Text)
txtVal_Weights(46).Text = Val(txtVal_Weights(46).Text) * Val(txtVal_Inp(16).Text)
txtVal_Weights(47).Text = Val(txtVal_Weights(47).Text) * Val(txtVal_Inp(17).Text)
txtVal_Weights(48).Text = Val(txtVal_Weights(48).Text) * Val(txtVal_Inp(18).Text)
txtVal_Weights(49).Text = Val(txtVal_Weights(49).Text) * Val(txtVal_Inp(19).Text)
txtVal_Weights(50).Text = Val(txtVal_Weights(50).Text) * Val(txtVal_Inp(20).Text)
txtVal_Weights(51).Text = Val(txtVal_Weights(51).Text) * Val(txtVal_Inp(21).Text)
txtVal_Weights(52).Text = Val(txtVal_Weights(52).Text) * Val(txtVal_Inp(22).Text)
txtVal_Weights(53).Text = Val(txtVal_Weights(53).Text) * Val(txtVal_Inp(23).Text)
txtVal_Weights(54).Text = Val(txtVal_Weights(54).Text) * Val(txtVal_Inp(24).Text)
txtVal_Weights(55).Text = Val(txtVal_Weights(55).Text) * Val(txtVal_Inp(25).Text)
txtVal_Weights(56).Text = Val(txtVal_Weights(56).Text) * Val(txtVal_Inp(26).Text)
txtVal_Weights(57).Text = Val(txtVal_Weights(57).Text) * Val(txtVal_Inp(27).Text)
txtVal_Weights(58).Text = Val(txtVal_Weights(58).Text) * Val(txtVal_Inp(28).Text)
txtVal_Weights(59).Text = Val(txtVal_Weights(59).Text) * Val(txtVal_Inp(29).Text)
'compute sigmoid for node1
txtVal_Weights(91).Text = Val(txtVal_Weights(91).Text) * Val(txtVal_Inp(30).Text)
txtNodal(1).Text = Val(txtVal_Weights(91).Text) + Val(txtVal_Weights(30).Text) +
Val(txtVal_Weights(31).Text)  +  Val(txtVal_Weights(32).Text)  +
Val(txtVal_Weights(33).Text)  +  Val(txtVal_Weights(34).Text)  +
Val(txtVal_Weights(35).Text)  +  Val(txtVal_Weights(36).Text)  +
Val(txtVal_Weights(37).Text)  +  Val(txtVal_Weights(38).Text)  +
Val(txtVal_Weights(39).Text)  +  Val(txtVal_Weights(40).Text)  +
Val(txtVal_Weights(41).Text)  +  Val(txtVal_Weights(42).Text)  +
Val(txtVal_Weights(43).Text)  +  Val(txtVal_Weights(44).Text)  +
Val(txtVal_Weights(45).Text)  +  Val(txtVal_Weights(46).Text)  +
Val(txtVal_Weights(47).Text)  +  Val(txtVal_Weights(48).Text)  +
```

```
Val(txtVal_Weights(49).Text) + Val(txtVal_Weights(50).Text) +
Val(txtVal_Weights(51).Text) + Val(txtVal_Weights(52).Text) +
Val(txtVal_Weights(53).Text) + Val(txtVal_Weights(54).Text) +
Val(txtVal_Weights(55).Text) + Val(txtVal_Weights(56).Text) +
Val(txtVal_Weights(57).Text) + Val(txtVal_Weights(58).Text) +
Val(txtVal_Weights(59).Text)
txtNodal(1).Text = 1 / (1 + Exp(-(txtNodal(1).Text)))
'node 2
txtVal_Weights(60).Text = Val(txtVal_Weights(60).Text) * Val(txtVal_Inp(0).Text)
txtVal_Weights(61).Text = Val(txtVal_Weights(61).Text) * Val(txtVal_Inp(1).Text)
txtVal_Weights(62).Text = Val(txtVal_Weights(62).Text) * Val(txtVal_Inp(2).Text)
txtVal_Weights(63).Text = Val(txtVal_Weights(63).Text) * Val(txtVal_Inp(3).Text)
txtVal_Weights(64).Text = Val(txtVal_Weights(64).Text) * Val(txtVal_Inp(4).Text)
txtVal_Weights(65).Text = Val(txtVal_Weights(65).Text) * Val(txtVal_Inp(5).Text)
txtVal_Weights(66).Text = Val(txtVal_Weights(66).Text) * Val(txtVal_Inp(6).Text)
txtVal_Weights(67).Text = Val(txtVal_Weights(67).Text) * Val(txtVal_Inp(7).Text)
txtVal_Weights(68).Text = Val(txtVal_Weights(68).Text) * Val(txtVal_Inp(8).Text)
txtVal_Weights(69).Text = Val(txtVal_Weights(69).Text) * Val(txtVal_Inp(9).Text)
txtVal_Weights(70).Text = Val(txtVal_Weights(70).Text) * Val(txtVal_Inp(10).Text)
txtVal_Weights(71).Text = Val(txtVal_Weights(71).Text) * Val(txtVal_Inp(11).Text)
txtVal_Weights(72).Text = Val(txtVal_Weights(72).Text) * Val(txtVal_Inp(12).Text)
txtVal_Weights(73).Text = Val(txtVal_Weights(73).Text) * Val(txtVal_Inp(13).Text)
txtVal_Weights(74).Text = Val(txtVal_Weights(74).Text) * Val(txtVal_Inp(14).Text)
txtVal_Weights(75).Text = Val(txtVal_Weights(75).Text) * Val(txtVal_Inp(15).Text)
txtVal_Weights(76).Text = Val(txtVal_Weights(76).Text) * Val(txtVal_Inp(16).Text)
txtVal_Weights(77).Text = Val(txtVal_Weights(77).Text) * Val(txtVal_Inp(17).Text)
txtVal_Weights(78).Text = Val(txtVal_Weights(78).Text) * Val(txtVal_Inp(18).Text)
txtVal_Weights(79).Text = Val(txtVal_Weights(79).Text) * Val(txtVal_Inp(19).Text)
txtVal_Weights(80).Text = Val(txtVal_Weights(80).Text) * Val(txtVal_Inp(20).Text)
txtVal_Weights(81).Text = Val(txtVal_Weights(81).Text) * Val(txtVal_Inp(21).Text)
txtVal_Weights(82).Text = Val(txtVal_Weights(82).Text) * Val(txtVal_Inp(22).Text)
txtVal_Weights(83).Text = Val(txtVal_Weights(83).Text) * Val(txtVal_Inp(23).Text)
txtVal_Weights(84).Text = Val(txtVal_Weights(84).Text) * Val(txtVal_Inp(24).Text)
txtVal_Weights(85).Text = Val(txtVal_Weights(85).Text) * Val(txtVal_Inp(25).Text)
txtVal_Weights(86).Text = Val(txtVal_Weights(86).Text) * Val(txtVal_Inp(26).Text)
txtVal_Weights(87).Text = Val(txtVal_Weights(87).Text) * Val(txtVal_Inp(27).Text)
txtVal_Weights(88).Text = Val(txtVal_Weights(88).Text) * Val(txtVal_Inp(28).Text)
txtVal_Weights(89).Text = Val(txtVal_Weights(89).Text) * Val(txtVal_Inp(29).Text)
'compute sigmoid for node2
txtVal_Weights(92).Text = Val(txtVal_Weights(92).Text) * Val(txtVal_Inp(30).Text)
txtNodal(2).Text = Val(txtVal_Weights(92).Text) + Val(txtVal_Weights(60).Text) +
Val(txtVal_Weights(61).Text) + Val(txtVal_Weights(62).Text) +
Val(txtVal_Weights(63).Text) + Val(txtVal_Weights(64).Text) +
Val(txtVal_Weights(65).Text) + Val(txtVal_Weights(66).Text) +
Val(txtVal_Weights(67).Text) + Val(txtVal_Weights(68).Text) +
Val(txtVal_Weights(69).Text) + Val(txtVal_Weights(70).Text) +
Val(txtVal_Weights(71).Text) + Val(txtVal_Weights(72).Text) +
Val(txtVal_Weights(73).Text) + Val(txtVal_Weights(74).Text) +
Val(txtVal_Weights(75).Text) + Val(txtVal_Weights(76).Text) +
Val(txtVal_Weights(77).Text) + Val(txtVal_Weights(78).Text) +
Val(txtVal_Weights(79).Text) + Val(txtVal_Weights(80).Text) +
Val(txtVal_Weights(81).Text) + Val(txtVal_Weights(82).Text) +
```

```
Val(txtVal_Weights(83).Text) + Val(txtVal_Weights(84).Text) +
Val(txtVal_Weights(85).Text) + Val(txtVal_Weights(86).Text) +
Val(txtVal_Weights(87).Text) + Val(txtVal_Weights(88).Text) +
Val(txtVal_Weights(89).Text)
txtNodal(2).Text = 1 / (1 + Exp(-(txtNodal(2).Text)))
'wieght adjustments for hidden layers (first forward sweep)
txtOut1.Text = Val(txtVal_Weights(112).Text) + Val(txtNodal(0).Text) *
Val(txtVal_Weights(90).Text) + Val(txtNodal(1).Text) * Val(txtVal_Weights(91).Text) +
Val(txtNodal(2).Text) * Val(txtVal_Weights(92).Text)
txtOut2.Text = Val(txtVal_Weights(111).Text) + Val(txtNodal(0).Text) *
Val(txtVal_Weights(93).Text) + Val(txtNodal(1).Text) * Val(txtVal_Weights(94).Text) +
Val(txtNodal(2).Text) * Val(txtVal_Weights(95).Text)
txtOut3.Text = Val(txtVal_Weights(110).Text) + Val(txtNodal(0).Text) *
Val(txtVal_Weights(96).Text) + Val(txtNodal(1).Text) * Val(txtVal_Weights(97).Text) +
Val(txtNodal(2).Text) * Val(txtVal_Weights(98).Text)
txtOut4.Text = Val(txtVal_Weights(109).Text) + Val(txtNodal(0).Text) *
Val(txtVal_Weights(99).Text) + Val(txtNodal(1).Text) * Val(txtVal_Weights(100).Text) +
Val(txtNodal(2).Text) * Val(txtVal_Weights(101).Text)
txtOut5.Text = Val(txtVal_Weights(108).Text) + Val(txtNodal(0).Text) *
Val(txtVal_Weights(102).Text) + Val(txtNodal(1).Text) * Val(txtVal_Weights(103).Text) +
Val(txtNodal(2).Text) * Val(txtVal_Weights(104).Text)
txtOut1.Text = 1 / (1 + Exp(-(txtOut1.Text)))
txtOut2.Text = 1 / (1 + Exp(-(txtOut2.Text)))
txtOut3.Text = 1 / (1 + Exp(-(txtOut3.Text)))
txtOut4.Text = 1 / (1 + Exp(-(txtOut4.Text)))
txtOut5.Text = 1 / (1 + Exp(-(txtOut5.Text)))
Call WeightList
End Sub
        ‘kNN code on frmkNN
Option Explicit
Private Points As New Collection, DrawPolyMode As Boolean
Private Sub Command2_Click()
txtO.Text = 1 / (1 + Exp(-(txtO2.Text)))
End Sub
Private Sub chkData_Click()
Dim i As Integer
If chkData.Value = 1 Then
List1.Visible = False
List2.Visible = False
chkData.Caption = "Show Data"
ElseIf chkData.Value = 0 Then
List1.Visible = True
List2.Visible = True
chkData.Caption = "Hide Data"
End If
End Sub
Private Sub cmdEbola_Click()
Call eLoader
Call EVD
End Sub
Private Sub cmdLassa_Click()
Call Lassa
```

```
End Sub
Private Sub Form_Load()
   Picture1.AutoRedraw = True
   Picture1.ScaleMode = vbPixels
   Picture1.DrawWidth = 1
   Picture1.BackColor = vbWhite
   Picture1.BorderStyle = 0
End Sub
Private Sub opt1Node_Click()
msg = "are you sure that there is sufficient system's resources to start a new iteration?"
title = "SOSIC New Iterator Check!"
Ans = MsgBox(msg, vbYesNoCancel + vbCritical, title)
If Ans = vbYes Then
frm1NoDiagnNN.Show
'Me.Hide
frm1NoDiagnNN.Caption = "***An implementation of a Feedforward Error-Back Propagation ANN Learning Algorithm with 1 Hidden Node***"
'Load all Fuzzified inputs
Dim i As Integer
For i = 0 To 29
frm1NoDiagnNN.txtQuantity(i).Text = frmDiagnNN.txtQuantity(i).Text
Next i
End If
'Transfer Actual Outputs to converge to from frmDiagnNN
Dim j As Integer
For j = 0 To 4
frm1NoDiagnNN.txtActual(j).Text = frmDiagnNN.txtActual(j).Text
Next j
End Sub
Private Sub opt2Node_Click()
msg = "are you sure that there is sufficient system's resources to start a new iteration?"
title = "SOSIC New Iterator Check!"
Ans = MsgBox(msg, vbYesNoCancel + vbCritical, title)
If Ans = vbYes Then
frm2NoDiagnNN.Show
frm2NoDiagnNN.Caption = "***An implementation of a Feedforward Error-Back Propagation ANN Learning Algorithm with 2 Hidden Nodes***"
'Load all Fuzzified inputs
Dim i As Integer
For i = 0 To 29
frm2NoDiagnNN.txtQuantity(i).Text = frmDiagnNN.txtQuantity(i).Text
Next i
End If
'Transfer Actual Outputs to converge to from frmDiagnNN
Dim j As Integer
For j = 0 To 4
frm2NoDiagnNN.txtActual(j).Text = frmDiagnNN.txtActual(j).Text
Next j
End Sub
Private Sub opt5Nodes_Click()
msg = "are you sure that there is sufficient system's resources to start a new iteration?"
title = "SOSIC New Iterator Check!"
```

```
Ans = MsgBox(msg, vbYesNoCancel + vbCritical, title)
If Ans = vbYes Then
frm5NoDiagnNN.Show
'Me.Hide
frm5NoDiagnNN.Caption = "***An implementation of a Feedforward Error-Back Propagation ANN Learning Algorithm with 5 Hidden Nodes***"
Dim i As Integer
For i = 0 To 29
frm5NoDiagnNN.txtQuantity(i).Text = frmDiagnNN.txtQuantity(i).Text
Next i
End If
'Transfer Actual Outputs to converge to from frmDiagnNN
Dim j As Integer
For j = 0 To 4
frm5NoDiagnNN.txtActual(j).Text = frmDiagnNN.txtActual(j).Text
Next j
End Sub
Code for implementation of a 5Hidden Node ANN
Public Sub Rado()
Dim Listside As Integer
Dim difcalc As Integer
difcalc = txtMaxV.Text
On Error GoTo errWeight:
Dim i As Integer
For i = 0 To 104
Do
      MyNum = Int(Rnd * Val(txtMaxV.Text)) * 0.04 + 0.14
      txtWeights(i).Text = (MyNum)
      Exit Do
   Loop
Next
'new 5N input weights adjustment
Dim j As Integer
For j = 113 To 172
Do
      MyNum = Int(Rnd * Val(txtMaxV.Text)) * 0.04 + 0.14
      txtWeights(j).Text = (MyNum)
      Exit Do
   Loop
Next
'new 5N hidden weights adjustment
Dim k As Integer
For k = 175 To 184
Do
      MyNum = Int(Rnd * Val(txtMaxV.Text)) * 0.04 + 0.14
      txtWeights(k).Text = (MyNum)
      Exit Do
   Loop
Next
Exit Sub
errWeight:
msg = "Oops! Check inputted value of Weight range!"
```

```
title = "SOSICK NN Access Check!"
Ans = MsgBox(msg, vbExclamation + vbOKOnly, title)
txtMaxV.SetFocus
Exit Sub
End Sub
Private Sub List1Redraw()
Dim i As Integer
For i = 0 To 89
List1.AddItem txtWeights(i).Text
Next i
'new inputs
Dim j As Integer
For j = 113 To 172
List1.AddItem txtWeights(j).Text
Next j
'new 5N hidden weights adjustment
Dim k As Integer
For k = 175 To 184
List1.AddItem txtWeights(k).Text
Next k
End Sub
'Picture1.ForeColor = vbblue
Private Sub List2Redraw()
Dim i As Integer
For i = 0 To 89
List2.AddItem txtWeights(i).Text
Next i
'new inputs
Dim j As Integer
For j = 113 To 172
List2.AddItem txtWeights(j).Text
Next j
'new 5N hidden weights adjustment
Dim k As Integer
For k = 175 To 184
List2.AddItem txtWeights(k).Text
Next k
End Sub
Private Sub cmdGetVal_Click()
'new inputs
Dim i As Integer
For i = 0 To 29
txtVal_Inp(i).Text = txtQuantity(i).Text
Next i
'add weights from training set
Call ValW
'code to populate listbox for new Inputs
Call InputL
End Sub
Private Sub Iterate()
Dim i As Integer
'wieght adjustments for input layers (first forward sweep)
```

```
If lblOutput.Caption = "0" Then
'plot initial weight
Call PlotWeight
'node1
For i = 0 To 29
 txtEpochWeight(i).Text = Val(txtWeights(i).Text) * Val(txtQuantity(i).Text)
Next i
txtEpochWeight(92).Text = Val(txtWeights(105).Text) * Val(txtQuantity(30).Text)
txtNode(0).Text = Val(txtEpochWeight(92).Text) + Val(txtEpochWeight(0).Text) +
Val(txtEpochWeight(1).Text) + Val(txtEpochWeight(2).Text) +
Val(txtEpochWeight(3).Text) + Val(txtEpochWeight(4).Text) +
Val(txtEpochWeight(5).Text) + Val(txtEpochWeight(6).Text) +
Val(txtEpochWeight(7).Text) + Val(txtEpochWeight(8).Text) +
Val(txtEpochWeight(9).Text) + Val(txtEpochWeight(10).Text) +
Val(txtEpochWeight(11).Text) + Val(txtEpochWeight(12).Text) +
Val(txtEpochWeight(13).Text) + Val(txtEpochWeight(14).Text) +
Val(txtEpochWeight(15).Text) + Val(txtEpochWeight(16).Text) +
Val(txtEpochWeight(17).Text) + Val(txtEpochWeight(18).Text) +
Val(txtEpochWeight(19).Text) + Val(txtEpochWeight(20).Text) +
Val(txtEpochWeight(21).Text) + Val(txtEpochWeight(22).Text) +
Val(txtEpochWeight(23).Text) + Val(txtEpochWeight(24).Text) +
Val(txtEpochWeight(25).Text) + Val(txtEpochWeight(26).Text) +
Val(txtEpochWeight(27).Text) + Val(txtEpochWeight(28).Text) +
Val(txtEpochWeight(29).Text)
txtNode(0).Text = 1 / (1 + Exp(-(txtNode(0).Text)))
'node2
txtEpochWeight(30).Text = Val(txtWeights(30).Text) * Val(txtQuantity(0).Text)
txtEpochWeight(31).Text = Val(txtWeights(31).Text) * Val(txtQuantity(1).Text)
txtEpochWeight(32).Text = Val(txtWeights(32).Text) * Val(txtQuantity(2).Text)
txtEpochWeight(33).Text = Val(txtWeights(33).Text) * Val(txtQuantity(3).Text)
txtEpochWeight(34).Text = Val(txtWeights(34).Text) * Val(txtQuantity(4).Text)
txtEpochWeight(35).Text = Val(txtWeights(35).Text) * Val(txtQuantity(5).Text)
txtEpochWeight(36).Text = Val(txtWeights(36).Text) * Val(txtQuantity(6).Text)
txtEpochWeight(37).Text = Val(txtWeights(37).Text) * Val(txtQuantity(7).Text)
txtEpochWeight(38).Text = Val(txtWeights(38).Text) * Val(txtQuantity(8).Text)
txtEpochWeight(39).Text = Val(txtWeights(39).Text) * Val(txtQuantity(9).Text)
txtEpochWeight(40).Text = Val(txtWeights(40).Text) * Val(txtQuantity(10).Text)
txtEpochWeight(41).Text = Val(txtWeights(41).Text) * Val(txtQuantity(11).Text)
txtEpochWeight(42).Text = Val(txtWeights(42).Text) * Val(txtQuantity(12).Text)
txtEpochWeight(43).Text = Val(txtWeights(43).Text) * Val(txtQuantity(13).Text)
txtEpochWeight(44).Text = Val(txtWeights(44).Text) * Val(txtQuantity(14).Text)
txtEpochWeight(45).Text = Val(txtWeights(45).Text) * Val(txtQuantity(15).Text)
txtEpochWeight(46).Text = Val(txtWeights(46).Text) * Val(txtQuantity(16).Text)
txtEpochWeight(47).Text = Val(txtWeights(47).Text) * Val(txtQuantity(17).Text)
txtEpochWeight(48).Text = Val(txtWeights(48).Text) * Val(txtQuantity(18).Text)
txtEpochWeight(49).Text = Val(txtWeights(49).Text) * Val(txtQuantity(19).Text)
txtEpochWeight(50).Text = Val(txtWeights(50).Text) * Val(txtQuantity(20).Text)
txtEpochWeight(51).Text = Val(txtWeights(51).Text) * Val(txtQuantity(21).Text)
txtEpochWeight(52).Text = Val(txtWeights(52).Text) * Val(txtQuantity(22).Text)
txtEpochWeight(53).Text = Val(txtWeights(53).Text) * Val(txtQuantity(23).Text)
txtEpochWeight(54).Text = Val(txtWeights(54).Text) * Val(txtQuantity(24).Text)
txtEpochWeight(55).Text = Val(txtWeights(55).Text) * Val(txtQuantity(25).Text)
```

```
txtEpochWeight(56).Text = Val(txtWeights(56).Text) * Val(txtQuantity(26).Text)
txtEpochWeight(57).Text = Val(txtWeights(57).Text) * Val(txtQuantity(27).Text)
txtEpochWeight(58).Text = Val(txtWeights(58).Text) * Val(txtQuantity(28).Text)
txtEpochWeight(59).Text = Val(txtWeights(59).Text) * Val(txtQuantity(29).Text)
'compute sigmoid for node2
txtEpochWeight(91).Text = Val(txtWeights(106).Text) * Val(txtQuantity(30).Text)
txtNode(1).Text = Val(txtEpochWeight(91).Text) + Val(txtEpochWeight(30).Text) +
Val(txtEpochWeight(31).Text) + Val(txtEpochWeight(32).Text) +
Val(txtEpochWeight(33).Text) + Val(txtEpochWeight(34).Text) +
Val(txtEpochWeight(35).Text) + Val(txtEpochWeight(36).Text) +
Val(txtEpochWeight(37).Text) + Val(txtEpochWeight(38).Text) +
Val(txtEpochWeight(39).Text) + Val(txtEpochWeight(40).Text) +
Val(txtEpochWeight(41).Text) + Val(txtEpochWeight(42).Text) +
Val(txtEpochWeight(43).Text) + Val(txtEpochWeight(44).Text) +
Val(txtEpochWeight(45).Text) + Val(txtEpochWeight(46).Text) +
Val(txtEpochWeight(47).Text) + Val(txtEpochWeight(48).Text) +
Val(txtEpochWeight(49).Text) + Val(txtEpochWeight(50).Text) +
Val(txtEpochWeight(51).Text) + Val(txtEpochWeight(52).Text) +
Val(txtEpochWeight(53).Text) + Val(txtEpochWeight(54).Text) +
Val(txtEpochWeight(55).Text) + Val(txtEpochWeight(56).Text) +
Val(txtEpochWeight(57).Text) + Val(txtEpochWeight(58).Text) +
Val(txtEpochWeight(59).Text)
txtNode(1).Text = 1 / (1 + Exp(-(txtNode(1).Text)))
'node 3
txtEpochWeight(60).Text = Val(txtWeights(60).Text) * Val(txtQuantity(0).Text)
txtEpochWeight(61).Text = Val(txtWeights(61).Text) * Val(txtQuantity(1).Text)
txtEpochWeight(62).Text = Val(txtWeights(62).Text) * Val(txtQuantity(2).Text)
txtEpochWeight(63).Text = Val(txtWeights(63).Text) * Val(txtQuantity(3).Text)
txtEpochWeight(64).Text = Val(txtWeights(64).Text) * Val(txtQuantity(4).Text)
txtEpochWeight(65).Text = Val(txtWeights(65).Text) * Val(txtQuantity(5).Text)
txtEpochWeight(66).Text = Val(txtWeights(66).Text) * Val(txtQuantity(6).Text)
txtEpochWeight(67).Text = Val(txtWeights(67).Text) * Val(txtQuantity(7).Text)
txtEpochWeight(68).Text = Val(txtWeights(68).Text) * Val(txtQuantity(8).Text)
txtEpochWeight(69).Text = Val(txtWeights(69).Text) * Val(txtQuantity(9).Text)
txtEpochWeight(70).Text = Val(txtWeights(70).Text) * Val(txtQuantity(10).Text)
txtEpochWeight(71).Text = Val(txtWeights(71).Text) * Val(txtQuantity(11).Text)
txtEpochWeight(72).Text = Val(txtWeights(72).Text) * Val(txtQuantity(12).Text)
txtEpochWeight(73).Text = Val(txtWeights(73).Text) * Val(txtQuantity(13).Text)
txtEpochWeight(74).Text = Val(txtWeights(74).Text) * Val(txtQuantity(14).Text)
txtEpochWeight(75).Text = Val(txtWeights(75).Text) * Val(txtQuantity(15).Text)
txtEpochWeight(76).Text = Val(txtWeights(76).Text) * Val(txtQuantity(16).Text)
txtEpochWeight(77).Text = Val(txtWeights(77).Text) * Val(txtQuantity(17).Text)
txtEpochWeight(78).Text = Val(txtWeights(78).Text) * Val(txtQuantity(18).Text)
txtEpochWeight(79).Text = Val(txtWeights(79).Text) * Val(txtQuantity(19).Text)
txtEpochWeight(80).Text = Val(txtWeights(80).Text) * Val(txtQuantity(20).Text)
txtEpochWeight(81).Text = Val(txtWeights(81).Text) * Val(txtQuantity(21).Text)
txtEpochWeight(82).Text = Val(txtWeights(82).Text) * Val(txtQuantity(22).Text)
txtEpochWeight(83).Text = Val(txtWeights(83).Text) * Val(txtQuantity(23).Text)
txtEpochWeight(84).Text = Val(txtWeights(84).Text) * Val(txtQuantity(24).Text)
txtEpochWeight(85).Text = Val(txtWeights(85).Text) * Val(txtQuantity(25).Text)
txtEpochWeight(86).Text = Val(txtWeights(86).Text) * Val(txtQuantity(26).Text)
txtEpochWeight(87).Text = Val(txtWeights(87).Text) * Val(txtQuantity(27).Text)
```

```
txtEpochWeight(88).Text = Val(txtWeights(88).Text) * Val(txtQuantity(28).Text)
txtEpochWeight(89).Text = Val(txtWeights(89).Text) * Val(txtQuantity(29).Text)
'compute sigmoid for node3
txtEpochWeight(90).Text = Val(txtWeights(107).Text) * Val(txtQuantity(30).Text)
txtNode(2).Text = Val(txtEpochWeight(90).Text) + Val(txtEpochWeight(60).Text) +
Val(txtEpochWeight(61).Text) + Val(txtEpochWeight(62).Text) +
Val(txtEpochWeight(63).Text) + Val(txtEpochWeight(64).Text) +
Val(txtEpochWeight(65).Text) + Val(txtEpochWeight(66).Text) +
Val(txtEpochWeight(67).Text) + Val(txtEpochWeight(68).Text) +
Val(txtEpochWeight(69).Text) + Val(txtEpochWeight(70).Text) +
Val(txtEpochWeight(71).Text) + Val(txtEpochWeight(72).Text) +
Val(txtEpochWeight(73).Text) + Val(txtEpochWeight(74).Text) +
Val(txtEpochWeight(75).Text) + Val(txtEpochWeight(76).Text) +
Val(txtEpochWeight(77).Text) + Val(txtEpochWeight(78).Text) +
Val(txtEpochWeight(79).Text) + Val(txtEpochWeight(80).Text) +
Val(txtEpochWeight(81).Text) + Val(txtEpochWeight(82).Text) +
Val(txtEpochWeight(83).Text) + Val(txtEpochWeight(84).Text) +
Val(txtEpochWeight(85).Text) + Val(txtEpochWeight(86).Text) +
Val(txtEpochWeight(87).Text) + Val(txtEpochWeight(88).Text) +
Val(txtEpochWeight(89).Text)
txtNode(2).Text = 1 / (1 + Exp(-(txtNode(2).Text)))
'node 4
txtEpochWeight(93).Text = Val(txtWeights(113).Text) * Val(txtQuantity(0).Text)
txtEpochWeight(94).Text = Val(txtWeights(114).Text) * Val(txtQuantity(1).Text)
txtEpochWeight(95).Text = Val(txtWeights(115).Text) * Val(txtQuantity(2).Text)
txtEpochWeight(96).Text = Val(txtWeights(116).Text) * Val(txtQuantity(3).Text)
txtEpochWeight(97).Text = Val(txtWeights(117).Text) * Val(txtQuantity(4).Text)
txtEpochWeight(98).Text = Val(txtWeights(118).Text) * Val(txtQuantity(5).Text)
txtEpochWeight(99).Text = Val(txtWeights(119).Text) * Val(txtQuantity(6).Text)
txtEpochWeight(100).Text = Val(txtWeights(120).Text) * Val(txtQuantity(7).Text)
txtEpochWeight(101).Text = Val(txtWeights(121).Text) * Val(txtQuantity(8).Text)
txtEpochWeight(102).Text = Val(txtWeights(122).Text) * Val(txtQuantity(9).Text)
txtEpochWeight(103).Text = Val(txtWeights(123).Text) * Val(txtQuantity(10).Text)
txtEpochWeight(104).Text = Val(txtWeights(124).Text) * Val(txtQuantity(11).Text)
txtEpochWeight(105).Text = Val(txtWeights(125).Text) * Val(txtQuantity(12).Text)
txtEpochWeight(106).Text = Val(txtWeights(126).Text) * Val(txtQuantity(13).Text)
txtEpochWeight(107).Text = Val(txtWeights(127).Text) * Val(txtQuantity(14).Text)
txtEpochWeight(108).Text = Val(txtWeights(128).Text) * Val(txtQuantity(15).Text)
txtEpochWeight(109).Text = Val(txtWeights(129).Text) * Val(txtQuantity(16).Text)
txtEpochWeight(110).Text = Val(txtWeights(130).Text) * Val(txtQuantity(17).Text)
txtEpochWeight(111).Text = Val(txtWeights(131).Text) * Val(txtQuantity(18).Text)
txtEpochWeight(112).Text = Val(txtWeights(132).Text) * Val(txtQuantity(19).Text)
txtEpochWeight(113).Text = Val(txtWeights(133).Text) * Val(txtQuantity(20).Text)
txtEpochWeight(114).Text = Val(txtWeights(134).Text) * Val(txtQuantity(21).Text)
txtEpochWeight(115).Text = Val(txtWeights(135).Text) * Val(txtQuantity(22).Text)
txtEpochWeight(116).Text = Val(txtWeights(136).Text) * Val(txtQuantity(23).Text)
txtEpochWeight(117).Text = Val(txtWeights(137).Text) * Val(txtQuantity(24).Text)
txtEpochWeight(118).Text = Val(txtWeights(138).Text) * Val(txtQuantity(25).Text)
txtEpochWeight(119).Text = Val(txtWeights(139).Text) * Val(txtQuantity(26).Text)
txtEpochWeight(120).Text = Val(txtWeights(140).Text) * Val(txtQuantity(27).Text)
txtEpochWeight(121).Text = Val(txtWeights(141).Text) * Val(txtQuantity(28).Text)
txtEpochWeight(122).Text = Val(txtWeights(142).Text) * Val(txtQuantity(29).Text)
```

```
'compute sigmoid for node4
txtEpochWeight(154).Text = Val(txtWeights(173).Text) * Val(txtQuantity(30).Text)
txtNode(3).Text = Val(txtEpochWeight(154).Text) + Val(txtEpochWeight(93).Text) +
Val(txtEpochWeight(94).Text) + Val(txtEpochWeight(95).Text) +
Val(txtEpochWeight(96).Text) + Val(txtEpochWeight(97).Text) +
Val(txtEpochWeight(98).Text) + Val(txtEpochWeight(99).Text) +
Val(txtEpochWeight(100).Text) + Val(txtEpochWeight(101).Text) +
Val(txtEpochWeight(102).Text) + Val(txtEpochWeight(103).Text) +
Val(txtEpochWeight(104).Text) + Val(txtEpochWeight(105).Text) +
Val(txtEpochWeight(106).Text) + Val(txtEpochWeight(107).Text) +
Val(txtEpochWeight(108).Text) + Val(txtEpochWeight(109).Text) +
Val(txtEpochWeight(110).Text) + Val(txtEpochWeight(111).Text) +
Val(txtEpochWeight(112).Text) + Val(txtEpochWeight(113).Text) +
Val(txtEpochWeight(114).Text) + Val(txtEpochWeight(115).Text) +
Val(txtEpochWeight(116).Text) + Val(txtEpochWeight(117).Text) +
Val(txtEpochWeight(118).Text) + Val(txtEpochWeight(119).Text) +
Val(txtEpochWeight(120).Text) + Val(txtEpochWeight(121).Text) +
Val(txtEpochWeight(122).Text)
txtNode(3).Text = 1 / (1 + Exp(-(txtNode(3).Text)))
'node 5
txtEpochWeight(123).Text = Val(txtWeights(143).Text) * Val(txtQuantity(0).Text)
txtEpochWeight(124).Text = Val(txtWeights(144).Text) * Val(txtQuantity(1).Text)
txtEpochWeight(125).Text = Val(txtWeights(145).Text) * Val(txtQuantity(2).Text)
txtEpochWeight(126).Text = Val(txtWeights(146).Text) * Val(txtQuantity(3).Text)
txtEpochWeight(127).Text = Val(txtWeights(147).Text) * Val(txtQuantity(4).Text)
txtEpochWeight(128).Text = Val(txtWeights(148).Text) * Val(txtQuantity(5).Text)
txtEpochWeight(129).Text = Val(txtWeights(149).Text) * Val(txtQuantity(6).Text)
txtEpochWeight(130).Text = Val(txtWeights(150).Text) * Val(txtQuantity(7).Text)
txtEpochWeight(131).Text = Val(txtWeights(151).Text) * Val(txtQuantity(8).Text)
txtEpochWeight(132).Text = Val(txtWeights(152).Text) * Val(txtQuantity(9).Text)
txtEpochWeight(133).Text = Val(txtWeights(153).Text) * Val(txtQuantity(10).Text)
txtEpochWeight(134).Text = Val(txtWeights(154).Text) * Val(txtQuantity(11).Text)
txtEpochWeight(135).Text = Val(txtWeights(155).Text) * Val(txtQuantity(12).Text)
txtEpochWeight(136).Text = Val(txtWeights(156).Text) * Val(txtQuantity(13).Text)
txtEpochWeight(137).Text = Val(txtWeights(157).Text) * Val(txtQuantity(14).Text)
txtEpochWeight(138).Text = Val(txtWeights(158).Text) * Val(txtQuantity(15).Text)
txtEpochWeight(139).Text = Val(txtWeights(159).Text) * Val(txtQuantity(16).Text)
txtEpochWeight(140).Text = Val(txtWeights(160).Text) * Val(txtQuantity(17).Text)
txtEpochWeight(141).Text = Val(txtWeights(161).Text) * Val(txtQuantity(18).Text)
txtEpochWeight(142).Text = Val(txtWeights(162).Text) * Val(txtQuantity(19).Text)
txtEpochWeight(143).Text = Val(txtWeights(163).Text) * Val(txtQuantity(20).Text)
txtEpochWeight(144).Text = Val(txtWeights(164).Text) * Val(txtQuantity(21).Text)
txtEpochWeight(145).Text = Val(txtWeights(165).Text) * Val(txtQuantity(22).Text)
txtEpochWeight(146).Text = Val(txtWeights(166).Text) * Val(txtQuantity(23).Text)
txtEpochWeight(147).Text = Val(txtWeights(167).Text) * Val(txtQuantity(24).Text)
txtEpochWeight(148).Text = Val(txtWeights(168).Text) * Val(txtQuantity(25).Text)
txtEpochWeight(149).Text = Val(txtWeights(169).Text) * Val(txtQuantity(26).Text)
txtEpochWeight(150).Text = Val(txtWeights(170).Text) * Val(txtQuantity(27).Text)
txtEpochWeight(151).Text = Val(txtWeights(171).Text) * Val(txtQuantity(28).Text)
txtEpochWeight(152).Text = Val(txtWeights(172).Text) * Val(txtQuantity(29).Text)
'compute sigmoid for node5
txtEpochWeight(153).Text = Val(txtWeights(174).Text) * Val(txtQuantity(30).Text)
```

```
txtNode(4).Text = Val(txtEpochWeight(153).Text) + Val(txtEpochWeight(123).Text) +
Val(txtEpochWeight(124).Text) + Val(txtEpochWeight(125).Text) +
Val(txtEpochWeight(126).Text) + Val(txtEpochWeight(127).Text) +
Val(txtEpochWeight(128).Text) + Val(txtEpochWeight(129).Text) +
Val(txtEpochWeight(130).Text) + Val(txtEpochWeight(131).Text) +
Val(txtEpochWeight(132).Text) + Val(txtEpochWeight(133).Text) +
Val(txtEpochWeight(134).Text) + Val(txtEpochWeight(135).Text) +
Val(txtEpochWeight(136).Text) + Val(txtEpochWeight(137).Text) +
Val(txtEpochWeight(138).Text) + Val(txtEpochWeight(139).Text) +
Val(txtEpochWeight(140).Text) + Val(txtEpochWeight(141).Text) +
Val(txtEpochWeight(142).Text) + Val(txtEpochWeight(143).Text) +
Val(txtEpochWeight(144).Text) + Val(txtEpochWeight(145).Text) +
Val(txtEpochWeight(146).Text) + Val(txtEpochWeight(147).Text) +
Val(txtEpochWeight(148).Text) + Val(txtEpochWeight(149).Text) +
Val(txtEpochWeight(150).Text) + Val(txtEpochWeight(151).Text) +
Val(txtEpochWeight(152).Text)
txtNode(4).Text = 1 / (1 + Exp(-(txtNode(4).Text)))
'wieght adjustments for hidden layers (first forward sweep)
txtOutput1.Text = Val(txtWeights(112).Text) + Val(txtNode(0).Text) *
Val(txtWeights(90).Text) + Val(txtNode(1).Text) * Val(txtWeights(91).Text) +
Val(txtNode(2).Text) * Val(txtWeights(92).Text) + Val(txtNode(3).Text) *
Val(txtWeights(175).Text) + Val(txtNode(4).Text) * Val(txtWeights(176).Text)
txtOutput2.Text = Val(txtWeights(111).Text) + Val(txtNode(0).Text) *
Val(txtWeights(93).Text) + Val(txtNode(1).Text) * Val(txtWeights(94).Text) +
Val(txtNode(2).Text) * Val(txtWeights(95).Text) + Val(txtNode(3).Text) *
Val(txtWeights(177).Text) + Val(txtNode(4).Text) * Val(txtWeights(178).Text)
txtOutput3.Text = Val(txtWeights(110).Text) + Val(txtNode(0).Text) *
Val(txtWeights(96).Text) + Val(txtNode(1).Text) * Val(txtWeights(97).Text) +
Val(txtNode(2).Text) * Val(txtWeights(98).Text) + Val(txtNode(3).Text) *
Val(txtWeights(179).Text) + Val(txtNode(4).Text) * Val(txtWeights(180).Text)
txtOutput4.Text = Val(txtWeights(109).Text) + Val(txtNode(0).Text) *
Val(txtWeights(99).Text) + Val(txtNode(1).Text) * Val(txtWeights(100).Text) +
Val(txtNode(2).Text) * Val(txtWeights(101).Text) + Val(txtNode(3).Text) *
Val(txtWeights(181).Text) + Val(txtNode(4).Text) * Val(txtWeights(182).Text)
txtOutput5.Text = Val(txtWeights(108).Text) + Val(txtNode(0).Text) *
Val(txtWeights(102).Text) + Val(txtNode(1).Text) * Val(txtWeights(103).Text) +
Val(txtNode(2).Text) * Val(txtWeights(104).Text) + Val(txtNode(3).Text) *
Val(txtWeights(183).Text) + Val(txtNode(4).Text) * Val(txtWeights(184).Text)
txtOutput1.Text = 1 / (1 + Exp(-(txtOutput1.Text)))
txtOutput2.Text = 1 / (1 + Exp(-(txtOutput2.Text)))
txtOutput3.Text = 1 / (1 + Exp(-(txtOutput3.Text)))
txtOutput4.Text = 1 / (1 + Exp(-(txtOutput4.Text)))
txtOutput5.Text = 1 / (1 + Exp(-(txtOutput5.Text)))
Call counter
Else
End If
Call WeightList
Exit Sub
End Sub
Private Sub IterateTwo()
Dim i As Integer
 'Woz,new = Woz,current + Change in Woz
```

```
dz1 = (Val(txtLearnR_n.Text) * Val(txtOutput1.Text) * (1 - Val(txtOutput1.Text)) *
(Val(txtActual(0).Text) - Val(txtOutput1.Text)) * Val(txtQuantity(30).Text))
dz2 = (Val(txtLearnR_n.Text) * Val(txtOutput2.Text) * (1 - Val(txtOutput2.Text)) *
(Val(txtActual(1).Text) - Val(txtOutput2.Text)) * Val(txtQuantity(30).Text))
dz3 = (Val(txtLearnR_n.Text) * Val(txtOutput3.Text) * (1 - Val(txtOutput3.Text)) *
(Val(txtActual(2).Text) - Val(txtOutput3.Text)) * Val(txtQuantity(30).Text))
dz4 = (Val(txtLearnR_n.Text) * Val(txtOutput4.Text) * (1 - Val(txtOutput4.Text)) *
(Val(txtActual(3).Text) - Val(txtOutput4.Text)) * Val(txtQuantity(30).Text))
dz5 = (Val(txtLearnR_n.Text) * Val(txtOutput5.Text) * (1 - Val(txtOutput5.Text)) *
(Val(txtActual(4).Text) - Val(txtOutput5.Text)) * Val(txtQuantity(30).Text))
'using dz(i) combo size
txtWeights(112).Text = Val(txtWeights(112).Text) + dz1
txtWeights(111).Text = Val(txtWeights(111).Text) + dz2
txtWeights(110).Text = Val(txtWeights(110).Text) + dz3
txtWeights(109).Text = Val(txtWeights(109).Text) + dz4
txtWeights(108).Text = Val(txtWeights(108).Text) + dz5
'wieght adjustments for hidden layers (Next (i) forward sweep)
'error responsibility dA
      'dA = outputNA(1 - outputNA)E(downstream)WjkDj
dA1 = Val(txtNode(0).Text) * (1 - Val(txtNode(0).Text)) * Val(txtWeights(90).Text) * dz1
            'then compute Change in WAz = ndz(1).OutputA
   CAz1 = Val(txtLearnR_n.Text) * dz1 * Val(txtNode(0).Text)
   'WAz,new = WAz,current + Change in WAz
   txtWeights(90).Text = Val(txtWeights(90).Text) + CAz1 'next hidden layer2
dA2 = Val(txtNode(0).Text) * (1 - Val(txtNode(0).Text)) * Val(txtWeights(93).Text) * dz2
            'then compute Change in WAz = ndz(1).OutputA
   CAz2 = Val(txtLearnR_n.Text) * dz2 * Val(txtNode(0).Text)
   'WAz,new = WAz,current + Change in WAz
   txtWeights(93).Text = Val(txtWeights(93).Text) + CAz2 'next hidden layer3
dA3 = Val(txtNode(0).Text) * (1 - Val(txtNode(0).Text)) * Val(txtWeights(96).Text) * dz3
            'then compute Change in WAz = ndz(1).OutputA
   CAz3 = Val(txtLearnR_n.Text) * dz3 * Val(txtNode(0).Text)
   'WAz,new = WAz,current + Change in WAz
   txtWeights(96).Text = Val(txtWeights(96).Text) + CAz3 'next hidden layer4
dA4 = Val(txtNode(0).Text) * (1 - Val(txtNode(0).Text)) * Val(txtWeights(99).Text) * dz4
            'then compute Change in WAz = ndz(1).OutputA
   CAz4 = Val(txtLearnR_n.Text) * dz4 * Val(txtNode(0).Text)
   'WAz,new = WAz,current + Change in WAz
   txtWeights(99).Text = Val(txtWeights(99).Text) + CAz4 'next hidden layer5
dA5 = Val(txtNode(0).Text) * (1 - Val(txtNode(0).Text)) * Val(txtWeights(102).Text) * dz5
            'then compute Change in WAz = ndz(1).OutputA
   CAz5 = Val(txtLearnR_n.Text) * dz5 * Val(txtNode(0).Text)
   'WAz,new = WAz,current + Change in WAz
   txtWeights(102).Text = Val(txtWeights(102).Text) + CAz5
'error responsibility dB
      'dB = outputNB(1 - outputNB)E(downstream)WjkDj
dB1 = Val(txtNode(1).Text) * (1 - Val(txtNode(1).Text)) * Val(txtWeights(91).Text) * dz1
            'then compute Change in WBz = ndz(1).OutputB
   CBz1 = Val(txtLearnR_n.Text) * dz1 * Val(txtNode(1).Text)
   'WBz,new = WBz,current + Change in WBz
   txtWeights(91).Text = Val(txtWeights(91).Text) + CBz1 'Next hidden layer
      'dB = outputNB(1 - outputNB)E(downstream)WjkDj
```

```
dB2 = Val(txtNode(1).Text) * (1 - Val(txtNode(1).Text)) * Val(txtWeights(94).Text) * dz2
                'then compute Change in WBz = ndz(1).OutputB
   CBz2 = Val(txtLearnR_n.Text) * dz2 * Val(txtNode(1).Text)
   'WBz,new = WBz,current + Change in WBz
   txtWeights(94).Text = Val(txtWeights(94).Text) + CBz2 'Next hidden layer
      'dB = outputNB(1 - outputNB)E(downstream)WjkDj
dB3 = Val(txtNode(1).Text) * (1 - Val(txtNode(1).Text)) * Val(txtWeights(97).Text) * dz3
                'then compute Change in WBz = ndz(1).OutputB
   CBz3 = Val(txtLearnR_n.Text) * dz3 * Val(txtNode(1).Text)
   'WBz,new = WBz,current + Change in WBz
   txtWeights(97).Text = Val(txtWeights(97).Text) + CBz3 'Next hidden layer
      'dB = outputNB(1 - outputNB)E(downstream)WjkDj
dB4 = Val(txtNode(1).Text) * (1 - Val(txtNode(1).Text)) * Val(txtWeights(100).Text) * dz4
                'then compute Change in WBz = ndz(1).OutputB
   CBz4 = Val(txtLearnR_n.Text) * dz4 * Val(txtNode(1).Text)
   'WBz,new = WBz,current + Change in WBz
   txtWeights(100).Text = Val(txtWeights(100).Text) + CBz4        'Next hidden layer
      'dB = outputNB(1 - outputNB)E(downstream)WjkDj
dB5 = Val(txtNode(1).Text) * (1 - Val(txtNode(1).Text)) * Val(txtWeights(103).Text) * dz5
                'then compute Change in WBz = ndz(1).OutputB
   CBz5 = Val(txtLearnR_n.Text) * dz5 * Val(txtNode(1).Text)
   'WBz,new = WBz,current + Change in WBz
   txtWeights(103).Text = Val(txtWeights(103).Text) + CBz5
'error responsibility dC
      'dC = outputNC(1 - outputNC)E(downstream)WjkDj
dC1 = Val(txtNode(2).Text) * (1 - Val(txtNode(2).Text)) * Val(txtWeights(92).Text) * dz1
                'then compute Change in WCz = ndz(1).OutputC
   CCz1 = Val(txtLearnR_n.Text) * dz1 * Val(txtNode(2).Text)
   'WCz,new = WCz,current + Change in WCz
   txtWeights(92).Text = Val(txtWeights(92).Text) + CCz1 'Next hidden layer
      'dC = outputNC(1 - outputNC)E(downstream)WjkDj
dC2 = Val(txtNode(2).Text) * (1 - Val(txtNode(2).Text)) * Val(txtWeights(95).Text) * dz2
                'then compute Change in WCz = ndz(1).OutputC
   CCz2 = Val(txtLearnR_n.Text) * dz2 * Val(txtNode(2).Text)
   'WCz,new = WCz,current + Change in WCz
   txtWeights(95).Text = Val(txtWeights(95).Text) + CCz2 'Next hidden layer
      'dC = outputNC(1 - outputNC)E(downstream)WjkDj
dC3 = Val(txtNode(2).Text) * (1 - Val(txtNode(2).Text)) * Val(txtWeights(98).Text) * dz3
                'then compute Change in WCz = ndz(1).OutputC
   CCz3 = Val(txtLearnR_n.Text) * dz3 * Val(txtNode(2).Text)
   'WCz,new = WCz,current + Change in WCz
   txtWeights(98).Text = Val(txtWeights(98).Text) + CCz3 'Next hidden layer
      'dC = outputNC(1 - outputNC)E(downstream)WjkDj
dC4 = Val(txtNode(2).Text) * (1 - Val(txtNode(2).Text)) * Val(txtWeights(101).Text) * dz4
                'then compute Change in WCz = ndz(1).OutputC
   CCz4 = Val(txtLearnR_n.Text) * dz4 * Val(txtNode(2).Text)
   'WCz,new = WCz,current + Change in WCz
   txtWeights(101).Text = Val(txtWeights(101).Text) + CCz4        'Next hidden layer
      'dC = outputNC(1 - outputNC)E(downstream)WjkDj
dC5 = Val(txtNode(2).Text) * (1 - Val(txtNode(2).Text)) * Val(txtWeights(104).Text) * dz5
                'then compute Change in WCz = ndz(1).OutputC
   CCz5 = Val(txtLearnR_n.Text) * dz5 * Val(txtNode(2).Text)
```

```
   'WCz,new = WCz,current + Change in WCz
   txtWeights(104).Text = Val(txtWeights(104).Text) + CCz5
'error responsibility dD
      'dD = outputNC(1 - outputNC)E(downstream)WjkDj
dD1 = Val(txtNode(3).Text) * (1 - Val(txtNode(3).Text)) * Val(txtWeights(175).Text) * dz1
               'then compute Change in WCz = ndz(1).OutputC
   CDz1 = Val(txtLearnR_n.Text) * dz1 * Val(txtNode(3).Text)
   'WCz,new = WCz,current + Change in WCz
   txtWeights(175).Text = Val(txtWeights(175).Text) + CDz1       'Next hidden layer
      'dD = outputNC(1 - outputNC)E(downstream)WjkDj
dD2 = Val(txtNode(3).Text) * (1 - Val(txtNode(3).Text)) * Val(txtWeights(177).Text) * dz2
               'then compute Change in WCz = ndz(1).OutputC
   CDz2 = Val(txtLearnR_n.Text) * dz2 * Val(txtNode(3).Text)
   'WCz,new = WCz,current + Change in WCz
   txtWeights(177).Text = Val(txtWeights(177).Text) + CDz2       'Next hidden layer
      'dD = outputNC(1 - outputNC)E(downstream)WjkDj
dD3 = Val(txtNode(3).Text) * (1 - Val(txtNode(3).Text)) * Val(txtWeights(179).Text) * dz3
               'then compute Change in WCz = ndz(1).OutputC
   CDz3 = Val(txtLearnR_n.Text) * dz3 * Val(txtNode(3).Text)
   'WCz,new = WCz,current + Change in WCz
   txtWeights(179).Text = Val(txtWeights(179).Text) + CDz3       'Next hidden layer
      'dD = outputNC(1 - outputNC)E(downstream)WjkDj
dD4 = Val(txtNode(3).Text) * (1 - Val(txtNode(3).Text)) * Val(txtWeights(181).Text) * dz4
               'then compute Change in WCz = ndz(1).OutputC
   CDz4 = Val(txtLearnR_n.Text) * dz4 * Val(txtNode(3).Text)
   'WCz,new = WCz,current + Change in WCz
   txtWeights(181).Text = Val(txtWeights(181).Text) + CDz4       'Next hidden layer
      'dD = outputNC(1 - outputNC)E(downstream)WjkDj
dD5 = Val(txtNode(3).Text) * (1 - Val(txtNode(3).Text)) * Val(txtWeights(183).Text) * dz5
               'then compute Change in WCz = ndz(1).OutputC
   CDz5 = Val(txtLearnR_n.Text) * dz5 * Val(txtNode(3).Text)
   'WCz,new = WCz,current + Change in WCz
   txtWeights(183).Text = Val(txtWeights(183).Text) + CDz5
'error responsibility dE
      'dE = outputNC(1 - outputNC)E(downstream)WjkDj
dE1 = Val(txtNode(4).Text) * (1 - Val(txtNode(4).Text)) * Val(txtWeights(176).Text) * dz1
               'then compute Change in WCz = ndz(1).OutputC
   CEz1 = Val(txtLearnR_n.Text) * dz1 * Val(txtNode(4).Text)
   'WCz,new = WCz,current + Change in WCz
   txtWeights(176).Text = Val(txtWeights(176).Text) + CEz1       'Next hidden layer
      'dE = outputNC(1 - outputNC)E(downstream)WjkDj
dE2 = Val(txtNode(4).Text) * (1 - Val(txtNode(4).Text)) * Val(txtWeights(178).Text) * dz2
               'then compute Change in WCz = ndz(1).OutputC
   CEz2 = Val(txtLearnR_n.Text) * dz2 * Val(txtNode(4).Text)
   'WCz,new = WCz,current + Change in WCz
   txtWeights(178).Text = Val(txtWeights(178).Text) + CEz2       'Next hidEen layer
      'dE = outputNC(1 - outputNC)E(downstream)WjkDj
dE3 = Val(txtNode(4).Text) * (1 - Val(txtNode(4).Text)) * Val(txtWeights(180).Text) * dz3
               'then compute Change in WCz = ndz(1).OutputC
   CEz3 = Val(txtLearnR_n.Text) * dz3 * Val(txtNode(4).Text)
   'WCz,new = WCz,current + Change in WCz
   txtWeights(180).Text = Val(txtWeights(180).Text) + CEz3       'Next hidden layer
```

```
        'dE = outputNC(1 - outputNC)E(downstream)WjkDj
    dE4 = Val(txtNode(4).Text) * (1 - Val(txtNode(4).Text)) * Val(txtWeights(182).Text) * dz4
    'then compute Change in WCz = ndz(1).OutputC
       CEz4 = Val(txtLearnR_n.Text) * dz4 * Val(txtNode(4).Text)
       'WCz,new = WCz,current + Change in WCz
       txtWeights(182).Text = Val(txtWeights(182).Text) + CEz4         'Next hidden layer
          'dE = outputNC(1 - outputNC)E(downstream)WjkDj
    dE5 = Val(txtNode(4).Text) * (1 - Val(txtNode(4).Text)) * Val(txtWeights(184).Text) * dz5

          'then compute Change in WCz = ndz(1).OutputC
       CEz5 = Val(txtLearnR_n.Text) * dz5 * Val(txtNode(4).Text)
       'WCz,new = WCz,current + Change in WCz
       txtWeights(184).Text = Val(txtWeights(184).Text) + CEz5
       'wieght adjustments for inputs (Next (i) forward sweep)
          'compute Change in WiA = ndAXi
          'WiA,new = WiA,current + Change in WiA
       CW1A = Val(txtLearnR_n.Text) * dA1 * Val(txtQuantity(0).Text)
       txtWeights(0).Text = Val(txtWeights(0).Text) + CW1A
       CW2A = Val(txtLearnR_n.Text) * dA1 * Val(txtQuantity(1).Text)
       txtWeights(1).Text = Val(txtWeights(1).Text) + CW2A
       CW3A = Val(txtLearnR_n.Text) * dA1 * Val(txtQuantity(2).Text)
       txtWeights(2).Text = Val(txtWeights(2).Text) + CW3A
       CW4A = Val(txtLearnR_n.Text) * dA1 * Val(txtQuantity(3).Text)
       txtWeights(3).Text = Val(txtWeights(3).Text) + CW4A
       CW5A = Val(txtLearnR_n.Text) * dA1 * Val(txtQuantity(4).Text)
       txtWeights(4).Text = Val(txtWeights(4).Text) + CW5A
       CW6A = Val(txtLearnR_n.Text) * dA1 * Val(txtQuantity(5).Text)
       txtWeights(5).Text = Val(txtWeights(5).Text) + CW6A
       CW7A = Val(txtLearnR_n.Text) * dA1 * Val(txtQuantity(6).Text)
       txtWeights(6).Text = Val(txtWeights(6).Text) + CW7A
       CW8A = Val(txtLearnR_n.Text) * dA1 * Val(txtQuantity(7).Text)
       txtWeights(7).Text = Val(txtWeights(7).Text) + CW8A
       CW9A = Val(txtLearnR_n.Text) * dA1 * Val(txtQuantity(8).Text)
       txtWeights(8).Text = Val(txtWeights(8).Text) + CW9A
       CW10A = Val(txtLearnR_n.Text) * dA1 * Val(txtQuantity(9).Text)
       txtWeights(9).Text = Val(txtWeights(9).Text) + CW10A
       CW11A = Val(txtLearnR_n.Text) * dA1 * Val(txtQuantity(10).Text)
       txtWeights(10).Text = Val(txtWeights(10).Text) + CW11A
       CW12A = Val(txtLearnR_n.Text) * dA1 * Val(txtQuantity(11).Text)
       txtWeights(11).Text = Val(txtWeights(11).Text) + CW12A
       CW13A = Val(txtLearnR_n.Text) * dA1 * Val(txtQuantity(12).Text)
       txtWeights(12).Text = Val(txtWeights(12).Text) + CW13A
       CW14A = Val(txtLearnR_n.Text) * dA1 * Val(txtQuantity(13).Text)
       txtWeights(13).Text = Val(txtWeights(13).Text) + CW14A
       CW15A = Val(txtLearnR_n.Text) * dA1 * Val(txtQuantity(14).Text)
       txtWeights(14).Text = Val(txtWeights(14).Text) + CW15A
       CW16A = Val(txtLearnR_n.Text) * dA1 * Val(txtQuantity(15).Text)
       txtWeights(15).Text = Val(txtWeights(15).Text) + CW16A
       CW17A = Val(txtLearnR_n.Text) * dA1 * Val(txtQuantity(16).Text)
       txtWeights(16).Text = Val(txtWeights(16).Text) + CW17A
       CW18A = Val(txtLearnR_n.Text) * dA1 * Val(txtQuantity(17).Text)
       txtWeights(17).Text = Val(txtWeights(17).Text) + CW18A
```

```
CW19A = Val(txtLearnR_n.Text) * dA1 * Val(txtQuantity(18).Text)
txtWeights(18).Text = Val(txtWeights(18).Text) + CW19A
CW20A = Val(txtLearnR_n.Text) * dA1 * Val(txtQuantity(19).Text)
txtWeights(19).Text = Val(txtWeights(19).Text) + CW20A
CW21A = Val(txtLearnR_n.Text) * dA1 * Val(txtQuantity(20).Text)
txtWeights(20).Text = Val(txtWeights(20).Text) + CW21A
CW22A = Val(txtLearnR_n.Text) * dA1 * Val(txtQuantity(21).Text)
txtWeights(21).Text = Val(txtWeights(21).Text) + CW21A
CW23A = Val(txtLearnR_n.Text) * dA1 * Val(txtQuantity(22).Text)
txtWeights(22).Text = Val(txtWeights(22).Text) + CW23A
CW24A = Val(txtLearnR_n.Text) * dA1 * Val(txtQuantity(23).Text)
txtWeights(23).Text = Val(txtWeights(23).Text) + CW24A
CW25A = Val(txtLearnR_n.Text) * dA1 * Val(txtQuantity(24).Text)
txtWeights(24).Text = Val(txtWeights(24).Text) + CW25A
CW26A = Val(txtLearnR_n.Text) * dA1 * Val(txtQuantity(25).Text)
txtWeights(25).Text = Val(txtWeights(25).Text) + CW22A
CW27A = Val(txtLearnR_n.Text) * dA1 * Val(txtQuantity(26).Text)
txtWeights(26).Text = Val(txtWeights(26).Text) + CW27A
CW28A = Val(txtLearnR_n.Text) * dA1 * Val(txtQuantity(27).Text)
txtWeights(27).Text = Val(txtWeights(27).Text) + CW28A
CW29A = Val(txtLearnR_n.Text) * dA1 * Val(txtQuantity(28).Text)
txtWeights(28).Text = Val(txtWeights(28).Text) + CW29A
CW30A = Val(txtLearnR_n.Text) * dA1 * Val(txtQuantity(29).Text)
txtWeights(29).Text = Val(txtWeights(29).Text) + CW30A
CW0A = Val(txtLearnR_n.Text) * dA1 * Val(txtQuantity(30).Text)
txtWeights(105).Text = Val(txtWeights(105).Text) + CW0A
'compute Change in WiB = ndBXi
   'WiB,new = WiB,current + Change in WiB
CW1B = Val(txtLearnR_n.Text) * dB1 * Val(txtQuantity(0).Text)
txtWeights(30).Text = Val(txtWeights(0).Text) + CW1B
CW2B = Val(txtLearnR_n.Text) * dB1 * Val(txtQuantity(1).Text)
txtWeights(31).Text = Val(txtWeights(1).Text) + CW2B
CW3B = Val(txtLearnR_n.Text) * dB1 * Val(txtQuantity(2).Text)
txtWeights(32).Text = Val(txtWeights(2).Text) + CW3B
CW4B = Val(txtLearnR_n.Text) * dB1 * Val(txtQuantity(3).Text)
txtWeights(33).Text = Val(txtWeights(3).Text) + CW4B
CW5B = Val(txtLearnR_n.Text) * dB1 * Val(txtQuantity(4).Text)
txtWeights(34).Text = Val(txtWeights(4).Text) + CW5B
CW6B = Val(txtLearnR_n.Text) * dB1 * Val(txtQuantity(5).Text)
txtWeights(35).Text = Val(txtWeights(5).Text) + CW6B
CW7B = Val(txtLearnR_n.Text) * dB1 * Val(txtQuantity(6).Text)
txtWeights(36).Text = Val(txtWeights(6).Text) + CW7B
CW8B = Val(txtLearnR_n.Text) * dB1 * Val(txtQuantity(7).Text)
txtWeights(37).Text = Val(txtWeights(7).Text) + CW8B
CW9B = Val(txtLearnR_n.Text) * dB1 * Val(txtQuantity(8).Text)
txtWeights(38).Text = Val(txtWeights(8).Text) + CW9B
CW10B = Val(txtLearnR_n.Text) * dB1 * Val(txtQuantity(9).Text)
txtWeights(39).Text = Val(txtWeights(9).Text) + CW10B
CW11B = Val(txtLearnR_n.Text) * dB1 * Val(txtQuantity(10).Text)
txtWeights(40).Text = Val(txtWeights(10).Text) + CW11B
CW12B = Val(txtLearnR_n.Text) * dB1 * Val(txtQuantity(11).Text)
txtWeights(41).Text = Val(txtWeights(11).Text) + CW12B
```

```
CW13B = Val(txtLearnR_n.Text) * dB1 * Val(txtQuantity(12).Text)
txtWeights(42).Text = Val(txtWeights(12).Text) + CW13B
CW14B = Val(txtLearnR_n.Text) * dB1 * Val(txtQuantity(13).Text)
txtWeights(43).Text = Val(txtWeights(13).Text) + CW14B
CW15B = Val(txtLearnR_n.Text) * dB1 * Val(txtQuantity(14).Text)
txtWeights(44).Text = Val(txtWeights(14).Text) + CW15B
CW16B = Val(txtLearnR_n.Text) * dB1 * Val(txtQuantity(15).Text)
txtWeights(45).Text = Val(txtWeights(15).Text) + CW16B
CW17B = Val(txtLearnR_n.Text) * dB1 * Val(txtQuantity(16).Text)
txtWeights(46).Text = Val(txtWeights(16).Text) + CW17B
CW18B = Val(txtLearnR_n.Text) * dB1 * Val(txtQuantity(17).Text)
txtWeights(47).Text = Val(txtWeights(17).Text) + CW18B
CW19B = Val(txtLearnR_n.Text) * dB1 * Val(txtQuantity(18).Text)
txtWeights(48).Text = Val(txtWeights(18).Text) + CW19B
CW20B = Val(txtLearnR_n.Text) * dB1 * Val(txtQuantity(19).Text)
txtWeights(49).Text = Val(txtWeights(19).Text) + CW20B
CW21B = Val(txtLearnR_n.Text) * dB1 * Val(txtQuantity(20).Text)
txtWeights(50).Text = Val(txtWeights(20).Text) + CW21B
CW22B = Val(txtLearnR_n.Text) * dB1 * Val(txtQuantity(21).Text)
txtWeights(51).Text = Val(txtWeights(21).Text) + CW21B
CW23B = Val(txtLearnR_n.Text) * dB1 * Val(txtQuantity(22).Text)
txtWeights(52).Text = Val(txtWeights(22).Text) + CW23B
CW24B = Val(txtLearnR_n.Text) * dB1 * Val(txtQuantity(23).Text)
txtWeights(53).Text = Val(txtWeights(23).Text) + CW24B
CW25B = Val(txtLearnR_n.Text) * dB1 * Val(txtQuantity(24).Text)
txtWeights(54).Text = Val(txtWeights(24).Text) + CW25B
CW26B = Val(txtLearnR_n.Text) * dB1 * Val(txtQuantity(25).Text)
txtWeights(55).Text = Val(txtWeights(25).Text) + CW22B
CW27B = Val(txtLearnR_n.Text) * dB1 * Val(txtQuantity(26).Text)
txtWeights(56).Text = Val(txtWeights(26).Text) + CW27B
CW28B = Val(txtLearnR_n.Text) * dB1 * Val(txtQuantity(27).Text)
txtWeights(57).Text = Val(txtWeights(27).Text) + CW28B
CW29B = Val(txtLearnR_n.Text) * dB1 * Val(txtQuantity(28).Text)
txtWeights(58).Text = Val(txtWeights(28).Text) + CW29B
CW30B = Val(txtLearnR_n.Text) * dB1 * Val(txtQuantity(29).Text)
txtWeights(59).Text = Val(txtWeights(29).Text) + CW30B
CW0B = Val(txtLearnR_n.Text) * dB1 * Val(txtQuantity(30).Text)
txtWeights(106).Text = Val(txtWeights(106).Text) + CW0B
'compute Change in WiC = ndBXi
    'WiC,new = WiC,current + Change in WiC
CW1C = Val(txtLearnR_n.Text) * dC1 * Val(txtQuantity(0).Text)
txtWeights(60).Text = Val(txtWeights(0).Text) + CW1C
CW2C = Val(txtLearnR_n.Text) * dC1 * Val(txtQuantity(1).Text)
txtWeights(61).Text = Val(txtWeights(1).Text) + CW2C
CW3C = Val(txtLearnR_n.Text) * dC1 * Val(txtQuantity(2).Text)
txtWeights(62).Text = Val(txtWeights(2).Text) + CW3C
CW4C = Val(txtLearnR_n.Text) * dC1 * Val(txtQuantity(3).Text)
txtWeights(63).Text = Val(txtWeights(3).Text) + CW4C
CW5C = Val(txtLearnR_n.Text) * dC1 * Val(txtQuantity(4).Text)
txtWeights(64).Text = Val(txtWeights(4).Text) + CW5C
CW6C = Val(txtLearnR_n.Text) * dC1 * Val(txtQuantity(5).Text)
txtWeights(65).Text = Val(txtWeights(5).Text) + CW6C
```

```
CW7C = Val(txtLearnR_n.Text) * dC1 * Val(txtQuantity(6).Text)
txtWeights(66).Text = Val(txtWeights(6).Text) + CW7C
CW8C = Val(txtLearnR_n.Text) * dC1 * Val(txtQuantity(7).Text)
txtWeights(67).Text = Val(txtWeights(7).Text) + CW8C
CW9C = Val(txtLearnR_n.Text) * dC1 * Val(txtQuantity(8).Text)
txtWeights(68).Text = Val(txtWeights(8).Text) + CW9C
CW10C = Val(txtLearnR_n.Text) * dC1 * Val(txtQuantity(9).Text)
txtWeights(69).Text = Val(txtWeights(9).Text) + CW10C
CW11C = Val(txtLearnR_n.Text) * dC1 * Val(txtQuantity(10).Text)
txtWeights(70).Text = Val(txtWeights(10).Text) + CW11C
CW12C = Val(txtLearnR_n.Text) * dC1 * Val(txtQuantity(11).Text)
txtWeights(71).Text = Val(txtWeights(11).Text) + CW12C
CW13C = Val(txtLearnR_n.Text) * dC1 * Val(txtQuantity(12).Text)
txtWeights(72).Text = Val(txtWeights(12).Text) + CW13C
CW14C = Val(txtLearnR_n.Text) * dC1 * Val(txtQuantity(13).Text)
txtWeights(73).Text = Val(txtWeights(13).Text) + CW14C
CW15C = Val(txtLearnR_n.Text) * dC1 * Val(txtQuantity(14).Text)
txtWeights(74).Text = Val(txtWeights(14).Text) + CW15C
CW16C = Val(txtLearnR_n.Text) * dC1 * Val(txtQuantity(15).Text)
txtWeights(75).Text = Val(txtWeights(15).Text) + CW16C
CW17C = Val(txtLearnR_n.Text) * dC1 * Val(txtQuantity(16).Text)
txtWeights(76).Text = Val(txtWeights(16).Text) + CW17C
CW18C = Val(txtLearnR_n.Text) * dC1 * Val(txtQuantity(17).Text)
txtWeights(77).Text = Val(txtWeights(17).Text) + CW18C
CW19C = Val(txtLearnR_n.Text) * dC1 * Val(txtQuantity(18).Text)
txtWeights(78).Text = Val(txtWeights(18).Text) + CW19C
CW20C = Val(txtLearnR_n.Text) * dC1 * Val(txtQuantity(19).Text)
txtWeights(79).Text = Val(txtWeights(19).Text) + CW20C
CW21C = Val(txtLearnR_n.Text) * dC1 * Val(txtQuantity(20).Text)
txtWeights(80).Text = Val(txtWeights(20).Text) + CW21C
CW22C = Val(txtLearnR_n.Text) * dC1 * Val(txtQuantity(21).Text)
txtWeights(81).Text = Val(txtWeights(21).Text) + CW21C
CW23C = Val(txtLearnR_n.Text) * dC1 * Val(txtQuantity(22).Text)
txtWeights(82).Text = Val(txtWeights(22).Text) + CW23C
CW24C = Val(txtLearnR_n.Text) * dC1 * Val(txtQuantity(23).Text)
txtWeights(83).Text = Val(txtWeights(23).Text) + CW24C
CW25C = Val(txtLearnR_n.Text) * dC1 * Val(txtQuantity(24).Text)
txtWeights(84).Text = Val(txtWeights(24).Text) + CW25C
CW26C = Val(txtLearnR_n.Text) * dC1 * Val(txtQuantity(25).Text)
txtWeights(85).Text = Val(txtWeights(25).Text) + CW22C
CW27C = Val(txtLearnR_n.Text) * dC1 * Val(txtQuantity(26).Text)
txtWeights(86).Text = Val(txtWeights(26).Text) + CW27C
CW28C = Val(txtLearnR_n.Text) * dC1 * Val(txtQuantity(27).Text)
txtWeights(87).Text = Val(txtWeights(27).Text) + CW28C
CW29C = Val(txtLearnR_n.Text) * dC1 * Val(txtQuantity(28).Text)
txtWeights(88).Text = Val(txtWeights(28).Text) + CW29C
CW30C = Val(txtLearnR_n.Text) * dC1 * Val(txtQuantity(29).Text)
txtWeights(89).Text = Val(txtWeights(29).Text) + CW30C
CW0C = Val(txtLearnR_n.Text) * dC1 * Val(txtQuantity(30).Text)
txtWeights(107).Text = Val(txtWeights(107).Text) + CW0C
'compute Change in WiD = ndBXi
'WiD,new = WiD,current + Change in WiD
```

```
CW1D = Val(txtLearnR_n.Text) * dD1 * Val(txtQuantity(0).Text)
txtWeights(113).Text = Val(txtWeights(0).Text) + CW1D
CW2D = Val(txtLearnR_n.Text) * dD1 * Val(txtQuantity(1).Text)
txtWeights(114).Text = Val(txtWeights(1).Text) + CW2D
CW3D = Val(txtLearnR_n.Text) * dD1 * Val(txtQuantity(2).Text)
txtWeights(115).Text = Val(txtWeights(2).Text) + CW3D
CW4D = Val(txtLearnR_n.Text) * dD1 * Val(txtQuantity(3).Text)
txtWeights(116).Text = Val(txtWeights(3).Text) + CW4D
CW5D = Val(txtLearnR_n.Text) * dD1 * Val(txtQuantity(4).Text)
txtWeights(117).Text = Val(txtWeights(4).Text) + CW5D
CW6D = Val(txtLearnR_n.Text) * dD1 * Val(txtQuantity(5).Text)
txtWeights(118).Text = Val(txtWeights(5).Text) + CW6D
CW7D = Val(txtLearnR_n.Text) * dD1 * Val(txtQuantity(6).Text)
txtWeights(119).Text = Val(txtWeights(6).Text) + CW7D
CW8D = Val(txtLearnR_n.Text) * dD1 * Val(txtQuantity(7).Text)
txtWeights(120).Text = Val(txtWeights(7).Text) + CW8D
CW9D = Val(txtLearnR_n.Text) * dD1 * Val(txtQuantity(8).Text)
txtWeights(121).Text = Val(txtWeights(8).Text) + CW9D
CW10D = Val(txtLearnR_n.Text) * dD1 * Val(txtQuantity(9).Text)
txtWeights(122).Text = Val(txtWeights(9).Text) + CW10D
CW11D = Val(txtLearnR_n.Text) * dD1 * Val(txtQuantity(10).Text)
txtWeights(123).Text = Val(txtWeights(10).Text) + CW11D
CW12D = Val(txtLearnR_n.Text) * dD1 * Val(txtQuantity(11).Text)
txtWeights(124).Text = Val(txtWeights(11).Text) + CW12D
CW13D = Val(txtLearnR_n.Text) * dD1 * Val(txtQuantity(12).Text)
txtWeights(125).Text = Val(txtWeights(12).Text) + CW13D
CW14D = Val(txtLearnR_n.Text) * dD1 * Val(txtQuantity(13).Text)
txtWeights(126).Text = Val(txtWeights(13).Text) + CW14D
CW15D = Val(txtLearnR_n.Text) * dD1 * Val(txtQuantity(14).Text)
txtWeights(127).Text = Val(txtWeights(14).Text) + CW15D
CW16D = Val(txtLearnR_n.Text) * dD1 * Val(txtQuantity(15).Text)
txtWeights(128).Text = Val(txtWeights(15).Text) + CW16D
CW17D = Val(txtLearnR_n.Text) * dD1 * Val(txtQuantity(16).Text)
txtWeights(129).Text = Val(txtWeights(16).Text) + CW17D
CW18D = Val(txtLearnR_n.Text) * dD1 * Val(txtQuantity(17).Text)
txtWeights(130).Text = Val(txtWeights(17).Text) + CW18D
CW19D = Val(txtLearnR_n.Text) * dD1 * Val(txtQuantity(18).Text)
txtWeights(131).Text = Val(txtWeights(18).Text) + CW19D
CW20D = Val(txtLearnR_n.Text) * dD1 * Val(txtQuantity(19).Text)
txtWeights(132).Text = Val(txtWeights(19).Text) + CW20D
CW21D = Val(txtLearnR_n.Text) * dD1 * Val(txtQuantity(20).Text)
txtWeights(133).Text = Val(txtWeights(20).Text) + CW21D
CW22D = Val(txtLearnR_n.Text) * dD1 * Val(txtQuantity(21).Text)
txtWeights(134).Text = Val(txtWeights(21).Text) + CW21D
CW23D = Val(txtLearnR_n.Text) * dD1 * Val(txtQuantity(22).Text)
txtWeights(135).Text = Val(txtWeights(22).Text) + CW23D
CW24D = Val(txtLearnR_n.Text) * dD1 * Val(txtQuantity(23).Text)
txtWeights(136).Text = Val(txtWeights(23).Text) + CW24D
CW25D = Val(txtLearnR_n.Text) * dD1 * Val(txtQuantity(24).Text)
txtWeights(137).Text = Val(txtWeights(24).Text) + CW25D
CW26D = Val(txtLearnR_n.Text) * dD1 * Val(txtQuantity(25).Text)
txtWeights(138).Text = Val(txtWeights(25).Text) + CW22D
```

```
CW27D = Val(txtLearnR_n.Text) * dD1 * Val(txtQuantity(26).Text)
txtWeights(139).Text = Val(txtWeights(26).Text) + CW27D
CW28D = Val(txtLearnR_n.Text) * dD1 * Val(txtQuantity(27).Text)
txtWeights(140).Text = Val(txtWeights(27).Text) + CW28D
CW29D = Val(txtLearnR_n.Text) * dD1 * Val(txtQuantity(28).Text)
txtWeights(141).Text = Val(txtWeights(28).Text) + CW29D
CW30D = Val(txtLearnR_n.Text) * dD1 * Val(txtQuantity(29).Text)
txtWeights(142).Text = Val(txtWeights(29).Text) + CW30D
CW0D = Val(txtLearnR_n.Text) * dD1 * Val(txtQuantity(30).Text)
txtWeights(173).Text = Val(txtWeights(173).Text) + CW0D
'compute Change in WiE = ndBXi
    'WiE,new = WiE,current + Change in WiE
CW1E = Val(txtLearnR_n.Text) * dE1 * Val(txtQuantity(0).Text)
txtWeights(143).Text = Val(txtWeights(0).Text) + CW1E
CW2E = Val(txtLearnR_n.Text) * dE1 * Val(txtQuantity(1).Text)
txtWeights(144).Text = Val(txtWeights(1).Text) + CW2E
CW3E = Val(txtLearnR_n.Text) * dE1 * Val(txtQuantity(2).Text)
txtWeights(145).Text = Val(txtWeights(2).Text) + CW3E
CW4E = Val(txtLearnR_n.Text) * dE1 * Val(txtQuantity(3).Text)
txtWeights(146).Text = Val(txtWeights(3).Text) + CW4E
CW5E = Val(txtLearnR_n.Text) * dE1 * Val(txtQuantity(4).Text)
txtWeights(147).Text = Val(txtWeights(4).Text) + CW5E
CW6E = Val(txtLearnR_n.Text) * dE1 * Val(txtQuantity(5).Text)
txtWeights(148).Text = Val(txtWeights(5).Text) + CW6E
CW7E = Val(txtLearnR_n.Text) * dE1 * Val(txtQuantity(6).Text)
txtWeights(149).Text = Val(txtWeights(6).Text) + CW7E
CW8E = Val(txtLearnR_n.Text) * dE1 * Val(txtQuantity(7).Text)
txtWeights(150).Text = Val(txtWeights(7).Text) + CW8E
CW9E = Val(txtLearnR_n.Text) * dE1 * Val(txtQuantity(8).Text)
txtWeights(151).Text = Val(txtWeights(8).Text) + CW9E
CW10E = Val(txtLearnR_n.Text) * dE1 * Val(txtQuantity(9).Text)
txtWeights(152).Text = Val(txtWeights(9).Text) + CW10E
CW11E = Val(txtLearnR_n.Text) * dE1 * Val(txtQuantity(10).Text)
txtWeights(153).Text = Val(txtWeights(10).Text) + CW11E
CW12E = Val(txtLearnR_n.Text) * dE1 * Val(txtQuantity(11).Text)
txtWeights(154).Text = Val(txtWeights(11).Text) + CW12E
CW13E = Val(txtLearnR_n.Text) * dE1 * Val(txtQuantity(12).Text)
txtWeights(155).Text = Val(txtWeights(12).Text) + CW13E
CW14E = Val(txtLearnR_n.Text) * dE1 * Val(txtQuantity(13).Text)
txtWeights(156).Text = Val(txtWeights(13).Text) + CW14E
CW15E = Val(txtLearnR_n.Text) * dE1 * Val(txtQuantity(14).Text)
txtWeights(157).Text = Val(txtWeights(14).Text) + CW15E
CW16E = Val(txtLearnR_n.Text) * dE1 * Val(txtQuantity(15).Text)
txtWeights(158).Text = Val(txtWeights(15).Text) + CW16E
CW17E = Val(txtLearnR_n.Text) * dE1 * Val(txtQuantity(16).Text)
txtWeights(159).Text = Val(txtWeights(16).Text) + CW17E
CW18E = Val(txtLearnR_n.Text) * dE1 * Val(txtQuantity(17).Text)
txtWeights(160).Text = Val(txtWeights(17).Text) + CW18E
CW19E = Val(txtLearnR_n.Text) * dE1 * Val(txtQuantity(18).Text)
txtWeights(161).Text = Val(txtWeights(18).Text) + CW19E
CW20E = Val(txtLearnR_n.Text) * dE1 * Val(txtQuantity(19).Text)
txtWeights(162).Text = Val(txtWeights(19).Text) + CW20E
```

```
    CW21E = Val(txtLearnR_n.Text) * dE1 * Val(txtQuantity(20).Text)
    txtWeights(163).Text = Val(txtWeights(20).Text) + CW21E
    CW22E = Val(txtLearnR_n.Text) * dE1 * Val(txtQuantity(21).Text)
    txtWeights(164).Text = Val(txtWeights(21).Text) + CW21E
    CW23E = Val(txtLearnR_n.Text) * dE1 * Val(txtQuantity(22).Text)
    txtWeights(165).Text = Val(txtWeights(22).Text) + CW23E
    CW24E = Val(txtLearnR_n.Text) * dE1 * Val(txtQuantity(23).Text)
    txtWeights(166).Text = Val(txtWeights(23).Text) + CW24E
    CW25E = Val(txtLearnR_n.Text) * dE1 * Val(txtQuantity(24).Text)
    txtWeights(167).Text = Val(txtWeights(24).Text) + CW25E
    CW26E = Val(txtLearnR_n.Text) * dE1 * Val(txtQuantity(25).Text)
    txtWeights(168).Text = Val(txtWeights(25).Text) + CW22E
    CW27E = Val(txtLearnR_n.Text) * dE1 * Val(txtQuantity(26).Text)
    txtWeights(169).Text = Val(txtWeights(26).Text) + CW27E
    CW28E = Val(txtLearnR_n.Text) * dE1 * Val(txtQuantity(27).Text)
    txtWeights(170).Text = Val(txtWeights(27).Text) + CW28E
    CW29E = Val(txtLearnR_n.Text) * dE1 * Val(txtQuantity(28).Text)
    txtWeights(171).Text = Val(txtWeights(28).Text) + CW29E
    CW30E = Val(txtLearnR_n.Text) * dE1 * Val(txtQuantity(29).Text)
    txtWeights(172).Text = Val(txtWeights(29).Text) + CW30E
    CW0E = Val(txtLearnR_n.Text) * dE1 * Val(txtQuantity(30).Text)
    txtWeights(174).Text = Val(txtWeights(174).Text) + CW0E
    'NEXT (i) FORWARD SWEEP'node1
For i = 0 To 29
 txtEpochWeight(i).Text = Val(txtWeights(i).Text) * Val(txtQuantity(i).Text)
Next i
txtEpochWeight(92).Text = Val(txtWeights(105).Text) * Val(txtQuantity(30).Text)
txtNode(0).Text = Val(txtEpochWeight(92).Text) + Val(txtEpochWeight(0).Text) +
Val(txtEpochWeight(1).Text)          +          Val(txtEpochWeight(2).Text)          +
Val(txtEpochWeight(3).Text)          +          Val(txtEpochWeight(4).Text)          +
Val(txtEpochWeight(5).Text)          +          Val(txtEpochWeight(6).Text)          +
Val(txtEpochWeight(7).Text)          +          Val(txtEpochWeight(8).Text)          +
Val(txtEpochWeight(9).Text)          +          Val(txtEpochWeight(10).Text)         +
Val(txtEpochWeight(11).Text)         +          Val(txtEpochWeight(12).Text)         +
Val(txtEpochWeight(13).Text)         +          Val(txtEpochWeight(14).Text)         +
Val(txtEpochWeight(15).Text)         +          Val(txtEpochWeight(16).Text)         +
Val(txtEpochWeight(17).Text)         +          Val(txtEpochWeight(18).Text)         +
Val(txtEpochWeight(19).Text)         +          Val(txtEpochWeight(20).Text)         +
Val(txtEpochWeight(21).Text)         +          Val(txtEpochWeight(22).Text)         +
Val(txtEpochWeight(23).Text)         +          Val(txtEpochWeight(24).Text)         +
Val(txtEpochWeight(25).Text)         +          Val(txtEpochWeight(26).Text)         +
Val(txtEpochWeight(27).Text)         +          Val(txtEpochWeight(28).Text)         +
Val(txtEpochWeight(29).Text)
txtNode(0).Text = 1 / (1 + Exp(-(txtNode(0).Text)))'node2
txtEpochWeight(30).Text = Val(txtWeights(30).Text) * Val(txtQuantity(0).Text)
txtEpochWeight(31).Text = Val(txtWeights(31).Text) * Val(txtQuantity(1).Text)
txtEpochWeight(32).Text = Val(txtWeights(32).Text) * Val(txtQuantity(2).Text)
txtEpochWeight(33).Text = Val(txtWeights(33).Text) * Val(txtQuantity(3).Text)
txtEpochWeight(34).Text = Val(txtWeights(34).Text) * Val(txtQuantity(4).Text)
txtEpochWeight(35).Text = Val(txtWeights(35).Text) * Val(txtQuantity(5).Text)
txtEpochWeight(36).Text = Val(txtWeights(36).Text) * Val(txtQuantity(6).Text)
txtEpochWeight(37).Text = Val(txtWeights(37).Text) * Val(txtQuantity(7).Text)
```

```
txtEpochWeight(38).Text = Val(txtWeights(38).Text) * Val(txtQuantity(8).Text)
txtEpochWeight(39).Text = Val(txtWeights(39).Text) * Val(txtQuantity(9).Text)
txtEpochWeight(40).Text = Val(txtWeights(40).Text) * Val(txtQuantity(10).Text)
txtEpochWeight(41).Text = Val(txtWeights(41).Text) * Val(txtQuantity(11).Text)
txtEpochWeight(42).Text = Val(txtWeights(42).Text) * Val(txtQuantity(12).Text)
txtEpochWeight(43).Text = Val(txtWeights(43).Text) * Val(txtQuantity(13).Text)
txtEpochWeight(44).Text = Val(txtWeights(44).Text) * Val(txtQuantity(14).Text)
txtEpochWeight(45).Text = Val(txtWeights(45).Text) * Val(txtQuantity(15).Text)
txtEpochWeight(46).Text = Val(txtWeights(46).Text) * Val(txtQuantity(16).Text)
txtEpochWeight(47).Text = Val(txtWeights(47).Text) * Val(txtQuantity(17).Text)
txtEpochWeight(48).Text = Val(txtWeights(48).Text) * Val(txtQuantity(18).Text)
txtEpochWeight(49).Text = Val(txtWeights(49).Text) * Val(txtQuantity(19).Text)
txtEpochWeight(50).Text = Val(txtWeights(50).Text) * Val(txtQuantity(20).Text)
txtEpochWeight(51).Text = Val(txtWeights(51).Text) * Val(txtQuantity(21).Text)
txtEpochWeight(52).Text = Val(txtWeights(52).Text) * Val(txtQuantity(22).Text)
txtEpochWeight(53).Text = Val(txtWeights(53).Text) * Val(txtQuantity(23).Text)
txtEpochWeight(54).Text = Val(txtWeights(54).Text) * Val(txtQuantity(24).Text)
txtEpochWeight(55).Text = Val(txtWeights(55).Text) * Val(txtQuantity(25).Text)
txtEpochWeight(56).Text = Val(txtWeights(56).Text) * Val(txtQuantity(26).Text)
txtEpochWeight(57).Text = Val(txtWeights(57).Text) * Val(txtQuantity(27).Text)
txtEpochWeight(58).Text = Val(txtWeights(58).Text) * Val(txtQuantity(28).Text)
txtEpochWeight(59).Text = Val(txtWeights(59).Text) * Val(txtQuantity(29).Text)
'compute sigmoid for node2
txtEpochWeight(91).Text = Val(txtWeights(106).Text) * Val(txtQuantity(30).Text)
txtNode(1).Text = Val(txtEpochWeight(91).Text) + Val(txtEpochWeight(30).Text) +
Val(txtEpochWeight(31).Text)          +          Val(txtEpochWeight(32).Text)          +
Val(txtEpochWeight(33).Text)          +          Val(txtEpochWeight(34).Text)          +
Val(txtEpochWeight(35).Text)          +          Val(txtEpochWeight(36).Text)          +
Val(txtEpochWeight(37).Text)          +          Val(txtEpochWeight(38).Text)          +
Val(txtEpochWeight(39).Text)          +          Val(txtEpochWeight(40).Text)          +
Val(txtEpochWeight(41).Text)          +          Val(txtEpochWeight(42).Text)          +
Val(txtEpochWeight(43).Text)          +          Val(txtEpochWeight(44).Text)          +
Val(txtEpochWeight(45).Text)          +          Val(txtEpochWeight(46).Text)          +
Val(txtEpochWeight(47).Text)          +          Val(txtEpochWeight(48).Text)          +
Val(txtEpochWeight(49).Text)          +          Val(txtEpochWeight(50).Text)          +
Val(txtEpochWeight(51).Text)          +          Val(txtEpochWeight(52).Text)          +
Val(txtEpochWeight(53).Text)          +          Val(txtEpochWeight(54).Text)          +
Val(txtEpochWeight(55).Text)          +          Val(txtEpochWeight(56).Text)          +
Val(txtEpochWeight(57).Text)          +          Val(txtEpochWeight(58).Text)          +
Val(txtEpochWeight(59).Text)
txtNode(1).Text = 1 / (1 + Exp(-(txtNode(1).Text)))'node 3
txtEpochWeight(60).Text = Val(txtWeights(60).Text) * Val(txtQuantity(0).Text)
txtEpochWeight(61).Text = Val(txtWeights(61).Text) * Val(txtQuantity(1).Text)
txtEpochWeight(62).Text = Val(txtWeights(62).Text) * Val(txtQuantity(2).Text)
txtEpochWeight(63).Text = Val(txtWeights(63).Text) * Val(txtQuantity(3).Text)
txtEpochWeight(64).Text = Val(txtWeights(64).Text) * Val(txtQuantity(4).Text)
txtEpochWeight(65).Text = Val(txtWeights(65).Text) * Val(txtQuantity(5).Text)
txtEpochWeight(66).Text = Val(txtWeights(66).Text) * Val(txtQuantity(6).Text)
txtEpochWeight(67).Text = Val(txtWeights(67).Text) * Val(txtQuantity(7).Text)
txtEpochWeight(68).Text = Val(txtWeights(68).Text) * Val(txtQuantity(8).Text)
txtEpochWeight(69).Text = Val(txtWeights(69).Text) * Val(txtQuantity(9).Text)
txtEpochWeight(70).Text = Val(txtWeights(70).Text) * Val(txtQuantity(10).Text)
```

```
txtEpochWeight(71).Text = Val(txtWeights(71).Text) * Val(txtQuantity(11).Text)
txtEpochWeight(72).Text = Val(txtWeights(72).Text) * Val(txtQuantity(12).Text)
txtEpochWeight(73).Text = Val(txtWeights(73).Text) * Val(txtQuantity(13).Text)
txtEpochWeight(74).Text = Val(txtWeights(74).Text) * Val(txtQuantity(14).Text)
txtEpochWeight(75).Text = Val(txtWeights(75).Text) * Val(txtQuantity(15).Text)
txtEpochWeight(76).Text = Val(txtWeights(76).Text) * Val(txtQuantity(16).Text)
txtEpochWeight(77).Text = Val(txtWeights(77).Text) * Val(txtQuantity(17).Text)
txtEpochWeight(78).Text = Val(txtWeights(78).Text) * Val(txtQuantity(18).Text)
txtEpochWeight(79).Text = Val(txtWeights(79).Text) * Val(txtQuantity(19).Text)
txtEpochWeight(80).Text = Val(txtWeights(80).Text) * Val(txtQuantity(20).Text)
txtEpochWeight(81).Text = Val(txtWeights(81).Text) * Val(txtQuantity(21).Text)
txtEpochWeight(82).Text = Val(txtWeights(82).Text) * Val(txtQuantity(22).Text)
txtEpochWeight(83).Text = Val(txtWeights(83).Text) * Val(txtQuantity(23).Text)
txtEpochWeight(84).Text = Val(txtWeights(84).Text) * Val(txtQuantity(24).Text)
txtEpochWeight(85).Text = Val(txtWeights(85).Text) * Val(txtQuantity(25).Text)
txtEpochWeight(86).Text = Val(txtWeights(86).Text) * Val(txtQuantity(26).Text)
txtEpochWeight(87).Text = Val(txtWeights(87).Text) * Val(txtQuantity(27).Text)
txtEpochWeight(88).Text = Val(txtWeights(88).Text) * Val(txtQuantity(28).Text)
txtEpochWeight(89).Text = Val(txtWeights(89).Text) * Val(txtQuantity(29).Text)
'compute sigmoid for node3
txtEpochWeight(90).Text = Val(txtWeights(107).Text) * Val(txtQuantity(30).Text)
txtNode(2).Text = Val(txtEpochWeight(90).Text) + Val(txtEpochWeight(60).Text) +
Val(txtEpochWeight(61).Text) + Val(txtEpochWeight(62).Text) +
Val(txtEpochWeight(63).Text) + Val(txtEpochWeight(64).Text) +
Val(txtEpochWeight(65).Text) + Val(txtEpochWeight(66).Text) +
Val(txtEpochWeight(67).Text) + Val(txtEpochWeight(68).Text) +
Val(txtEpochWeight(69).Text) + Val(txtEpochWeight(70).Text) +
Val(txtEpochWeight(71).Text) + Val(txtEpochWeight(72).Text) +
Val(txtEpochWeight(73).Text) + Val(txtEpochWeight(74).Text) +
Val(txtEpochWeight(75).Text) + Val(txtEpochWeight(76).Text) +
Val(txtEpochWeight(77).Text) + Val(txtEpochWeight(78).Text) +
Val(txtEpochWeight(79).Text) + Val(txtEpochWeight(80).Text) +
Val(txtEpochWeight(81).Text) + Val(txtEpochWeight(82).Text) +
Val(txtEpochWeight(83).Text) + Val(txtEpochWeight(84).Text) +
Val(txtEpochWeight(85).Text) + Val(txtEpochWeight(86).Text) +
Val(txtEpochWeight(87).Text) + Val(txtEpochWeight(88).Text) +
Val(txtEpochWeight(89).Text)
txtNode(2).Text = 1 / (1 + Exp(-(txtNode(2).Text)))
Call Node4and5
'wieght adjustments for hidden layers (first forward sweep)
txtOutput1.Text = Val(txtWeights(112).Text) + Val(txtNode(0).Text) *
Val(txtWeights(90).Text) + Val(txtNode(1).Text) * Val(txtWeights(91).Text) +
Val(txtNode(2).Text) * Val(txtWeights(92).Text)
txtOutput2.Text = Val(txtWeights(111).Text) + Val(txtNode(0).Text) *
Val(txtWeights(93).Text) + Val(txtNode(1).Text) * Val(txtWeights(94).Text) +
Val(txtNode(2).Text) * Val(txtWeights(95).Text)
txtOutput3.Text = Val(txtWeights(110).Text) + Val(txtNode(0).Text) *
Val(txtWeights(96).Text) + Val(txtNode(1).Text) * Val(txtWeights(97).Text) +
Val(txtNode(2).Text) * Val(txtWeights(98).Text)
txtOutput4.Text = Val(txtWeights(109).Text) + Val(txtNode(0).Text) *
Val(txtWeights(99).Text) + Val(txtNode(1).Text) * Val(txtWeights(100).Text) +
Val(txtNode(2).Text) * Val(txtWeights(101).Text)
```

```
txtOutput5.Text = Val(txtWeights(108).Text) + Val(txtNode(0).Text) *
Val(txtWeights(102).Text) + Val(txtNode(1).Text) * Val(txtWeights(103).Text) +
Val(txtNode(2).Text) * Val(txtWeights(104).Text)
txtOutput1.Text = 1 / (1 + Exp(-(txtOutput1.Text)))
txtOutput2.Text = 1 / (1 + Exp(-(txtOutput2.Text)))
txtOutput3.Text = 1 / (1 + Exp(-(txtOutput3.Text)))
txtOutput4.Text = 1 / (1 + Exp(-(txtOutput4.Text)))
txtOutput5.Text = 1 / (1 + Exp(-(txtOutput5.Text)))
Call counter
End Sub
Private Sub Node4and5()          'node 4
txtEpochWeight(93).Text = Val(txtWeights(113).Text) * Val(txtQuantity(0).Text)
txtEpochWeight(94).Text = Val(txtWeights(114).Text) * Val(txtQuantity(1).Text)
txtEpochWeight(95).Text = Val(txtWeights(115).Text) * Val(txtQuantity(2).Text)
txtEpochWeight(96).Text = Val(txtWeights(116).Text) * Val(txtQuantity(3).Text)
txtEpochWeight(97).Text = Val(txtWeights(117).Text) * Val(txtQuantity(4).Text)
txtEpochWeight(98).Text = Val(txtWeights(118).Text) * Val(txtQuantity(5).Text)
txtEpochWeight(99).Text = Val(txtWeights(119).Text) * Val(txtQuantity(6).Text)
txtEpochWeight(100).Text = Val(txtWeights(120).Text) * Val(txtQuantity(7).Text)
txtEpochWeight(101).Text = Val(txtWeights(121).Text) * Val(txtQuantity(8).Text)
txtEpochWeight(102).Text = Val(txtWeights(122).Text) * Val(txtQuantity(9).Text)
txtEpochWeight(103).Text = Val(txtWeights(123).Text) * Val(txtQuantity(10).Text)
txtEpochWeight(104).Text = Val(txtWeights(124).Text) * Val(txtQuantity(11).Text)
txtEpochWeight(105).Text = Val(txtWeights(125).Text) * Val(txtQuantity(12).Text)
txtEpochWeight(106).Text = Val(txtWeights(126).Text) * Val(txtQuantity(13).Text)
txtEpochWeight(107).Text = Val(txtWeights(127).Text) * Val(txtQuantity(14).Text)
txtEpochWeight(108).Text = Val(txtWeights(128).Text) * Val(txtQuantity(15).Text)
txtEpochWeight(109).Text = Val(txtWeights(129).Text) * Val(txtQuantity(16).Text)
txtEpochWeight(110).Text = Val(txtWeights(130).Text) * Val(txtQuantity(17).Text)
txtEpochWeight(111).Text = Val(txtWeights(131).Text) * Val(txtQuantity(18).Text)
txtEpochWeight(112).Text = Val(txtWeights(132).Text) * Val(txtQuantity(19).Text)
txtEpochWeight(113).Text = Val(txtWeights(133).Text) * Val(txtQuantity(20).Text)
txtEpochWeight(114).Text = Val(txtWeights(134).Text) * Val(txtQuantity(21).Text)
txtEpochWeight(115).Text = Val(txtWeights(135).Text) * Val(txtQuantity(22).Text)
txtEpochWeight(116).Text = Val(txtWeights(136).Text) * Val(txtQuantity(23).Text)
txtEpochWeight(117).Text = Val(txtWeights(137).Text) * Val(txtQuantity(24).Text)
txtEpochWeight(118).Text = Val(txtWeights(138).Text) * Val(txtQuantity(25).Text)
txtEpochWeight(119).Text = Val(txtWeights(139).Text) * Val(txtQuantity(26).Text)
txtEpochWeight(120).Text = Val(txtWeights(140).Text) * Val(txtQuantity(27).Text)
txtEpochWeight(121).Text = Val(txtWeights(141).Text) * Val(txtQuantity(28).Text)
txtEpochWeight(122).Text = Val(txtWeights(142).Text) * Val(txtQuantity(29).Text)
'compute sigmoid for node4
txtEpochWeight(154).Text = Val(txtWeights(173).Text) * Val(txtQuantity(30).Text)
txtNode(3).Text = Val(txtEpochWeight(154).Text) + Val(txtEpochWeight(93).Text) +
Val(txtEpochWeight(94).Text) + Val(txtEpochWeight(95).Text) +
Val(txtEpochWeight(96).Text) + Val(txtEpochWeight(97).Text) +
Val(txtEpochWeight(98).Text) + Val(txtEpochWeight(99).Text) +
Val(txtEpochWeight(100).Text) + Val(txtEpochWeight(101).Text) +
Val(txtEpochWeight(102).Text) + Val(txtEpochWeight(103).Text) +
Val(txtEpochWeight(104).Text) + Val(txtEpochWeight(105).Text) +
Val(txtEpochWeight(106).Text) + Val(txtEpochWeight(107).Text) +
Val(txtEpochWeight(108).Text) + Val(txtEpochWeight(109).Text) +
```

```
Val(txtEpochWeight(110).Text)  +  Val(txtEpochWeight(111).Text)  +
Val(txtEpochWeight(112).Text)  +  Val(txtEpochWeight(113).Text)  +
Val(txtEpochWeight(114).Text)  +  Val(txtEpochWeight(115).Text)  +
Val(txtEpochWeight(116).Text)  +  Val(txtEpochWeight(117).Text)  +
Val(txtEpochWeight(118).Text)  +  Val(txtEpochWeight(119).Text)  +
Val(txtEpochWeight(120).Text)  +  Val(txtEpochWeight(121).Text)  +
Val(txtEpochWeight(122).Text)
txtNode(3).Text = 1 / (1 + Exp(-(txtNode(3).Text)))'node 5
txtEpochWeight(123).Text = Val(txtWeights(143).Text) * Val(txtQuantity(0).Text)
txtEpochWeight(124).Text = Val(txtWeights(144).Text) * Val(txtQuantity(1).Text)
txtEpochWeight(125).Text = Val(txtWeights(145).Text) * Val(txtQuantity(2).Text)
txtEpochWeight(126).Text = Val(txtWeights(146).Text) * Val(txtQuantity(3).Text)
txtEpochWeight(127).Text = Val(txtWeights(147).Text) * Val(txtQuantity(4).Text)
txtEpochWeight(128).Text = Val(txtWeights(148).Text) * Val(txtQuantity(5).Text)
txtEpochWeight(129).Text = Val(txtWeights(149).Text) * Val(txtQuantity(6).Text)
txtEpochWeight(130).Text = Val(txtWeights(150).Text) * Val(txtQuantity(7).Text)
txtEpochWeight(131).Text = Val(txtWeights(151).Text) * Val(txtQuantity(8).Text)
txtEpochWeight(132).Text = Val(txtWeights(152).Text) * Val(txtQuantity(9).Text)
txtEpochWeight(133).Text = Val(txtWeights(153).Text) * Val(txtQuantity(10).Text)
txtEpochWeight(134).Text = Val(txtWeights(154).Text) * Val(txtQuantity(11).Text)
txtEpochWeight(135).Text = Val(txtWeights(155).Text) * Val(txtQuantity(12).Text)
txtEpochWeight(136).Text = Val(txtWeights(156).Text) * Val(txtQuantity(13).Text)
txtEpochWeight(137).Text = Val(txtWeights(157).Text) * Val(txtQuantity(14).Text)
txtEpochWeight(138).Text = Val(txtWeights(158).Text) * Val(txtQuantity(15).Text)
txtEpochWeight(139).Text = Val(txtWeights(159).Text) * Val(txtQuantity(16).Text)
txtEpochWeight(140).Text = Val(txtWeights(160).Text) * Val(txtQuantity(17).Text)
txtEpochWeight(141).Text = Val(txtWeights(161).Text) * Val(txtQuantity(18).Text)
txtEpochWeight(142).Text = Val(txtWeights(162).Text) * Val(txtQuantity(19).Text)
txtEpochWeight(143).Text = Val(txtWeights(163).Text) * Val(txtQuantity(20).Text)
txtEpochWeight(144).Text = Val(txtWeights(164).Text) * Val(txtQuantity(21).Text)
txtEpochWeight(145).Text = Val(txtWeights(165).Text) * Val(txtQuantity(22).Text)
txtEpochWeight(146).Text = Val(txtWeights(166).Text) * Val(txtQuantity(23).Text)
txtEpochWeight(147).Text = Val(txtWeights(167).Text) * Val(txtQuantity(24).Text)
txtEpochWeight(148).Text = Val(txtWeights(168).Text) * Val(txtQuantity(25).Text)
txtEpochWeight(149).Text = Val(txtWeights(169).Text) * Val(txtQuantity(26).Text)
txtEpochWeight(150).Text = Val(txtWeights(170).Text) * Val(txtQuantity(27).Text)
txtEpochWeight(151).Text = Val(txtWeights(171).Text) * Val(txtQuantity(28).Text)
txtEpochWeight(152).Text = Val(txtWeights(172).Text) * Val(txtQuantity(29).Text)
'compute sigmoid for node5
txtEpochWeight(153).Text = Val(txtWeights(174).Text) * Val(txtQuantity(30).Text)
txtNode(4).Text = Val(txtEpochWeight(153).Text) + Val(txtEpochWeight(123).Text) +
Val(txtEpochWeight(124).Text)  +  Val(txtEpochWeight(125).Text)  +
Val(txtEpochWeight(126).Text)  +  Val(txtEpochWeight(127).Text)  +
Val(txtEpochWeight(128).Text)  +  Val(txtEpochWeight(129).Text)  +
Val(txtEpochWeight(130).Text)  +  Val(txtEpochWeight(131).Text)  +
Val(txtEpochWeight(132).Text)  +  Val(txtEpochWeight(133).Text)  +
Val(txtEpochWeight(134).Text)  +  Val(txtEpochWeight(135).Text)  +
Val(txtEpochWeight(136).Text)  +  Val(txtEpochWeight(137).Text)  +
Val(txtEpochWeight(138).Text)  +  Val(txtEpochWeight(139).Text)  +
Val(txtEpochWeight(140).Text)  +  Val(txtEpochWeight(141).Text)  +
Val(txtEpochWeight(142).Text)  +  Val(txtEpochWeight(143).Text)  +
Val(txtEpochWeight(144).Text)  +  Val(txtEpochWeight(145).Text)  +
```

```
Val(txtEpochWeight(146).Text) + Val(txtEpochWeight(147).Text) +
Val(txtEpochWeight(148).Text) + Val(txtEpochWeight(149).Text) +
Val(txtEpochWeight(150).Text) + Val(txtEpochWeight(151).Text) +
Val(txtEpochWeight(152).Text)
txtNode(4).Text = 1 / (1 + Exp(-(txtNode(4).Text)))
End Sub
Private Sub WeightList()
Dim i As Integer
For i = 0 To 89
List3.AddItem "Weights " & i
Next i
Dim j As Integer
For j = 113 To 172
List3.AddItem "Weights " & j
Next j
Call InputList
End Sub
Private Sub InputList()
Dim i As Integer'add output from GUI To NN inputs
For i = 0 To 29
List4.AddItem txtQuantity(i).Text
Next i
End Sub
Private Sub InputL()
Dim i As Integer'add output from GUI To NN inputs
For i = 0 To 29
List5.AddItem txtVal_Inp(i).Text
Next i
End Sub
Private Sub ValW()
Dim i As Integer'add output from GUI To NN inputs
For i = 0 To 184
txtVal_Weights(i).Text = txtWeights(i).Text
Next i
End Sub
Private Sub ValIterate()
Dim i As Integer'wieght adjustments for input layers (first forward sweep)
For i = 0 To 29 'node1
 txtVal_Weights(i).Text = Val(txtVal_Weights(i).Text) * Val(txtVal_Inp(i).Text)
Next i
txtVal_Weights(90).Text = Val(txtVal_Weights(90).Text) * Val(txtVal_Inp(30).Text)
txtNodal(0).Text = Val(txtVal_Weights(92).Text) + Val(txtVal_Weights(0).Text) +
Val(txtVal_Weights(1).Text) + Val(txtVal_Weights(2).Text) + Val(txtVal_Weights(3).Text)
+ Val(txtVal_Weights(4).Text) + Val(txtVal_Weights(5).Text) +
Val(txtVal_Weights(6).Text) + Val(txtVal_Weights(7).Text) + Val(txtVal_Weights(8).Text)
+ Val(txtVal_Weights(9).Text) + Val(txtVal_Weights(10).Text) +
Val(txtVal_Weights(11).Text) + Val(txtVal_Weights(12).Text) +
Val(txtVal_Weights(13).Text) + Val(txtVal_Weights(14).Text) +
Val(txtVal_Weights(15).Text) + Val(txtVal_Weights(16).Text) +
Val(txtVal_Weights(17).Text) + Val(txtVal_Weights(18).Text) +
Val(txtVal_Weights(19).Text) + Val(txtVal_Weights(20).Text) +
Val(txtVal_Weights(21).Text) + Val(txtVal_Weights(22).Text) +
```

```
Val(txtVal_Weights(23).Text) + Val(txtVal_Weights(24).Text) +
Val(txtVal_Weights(25).Text) + Val(txtVal_Weights(26).Text) +
Val(txtVal_Weights(27).Text) + Val(txtVal_Weights(28).Text) +
Val(txtVal_Weights(29).Text)
txtNodal(0).Text = 1 / (1 + Exp(-(txtNodal(0).Text)))'node2
txtVal_Weights(30).Text = Val(txtVal_Weights(30).Text) * Val(txtVal_Inp(0).Text)
txtVal_Weights(31).Text = Val(txtVal_Weights(31).Text) * Val(txtVal_Inp(1).Text)
txtVal_Weights(32).Text = Val(txtVal_Weights(32).Text) * Val(txtVal_Inp(2).Text)
txtVal_Weights(33).Text = Val(txtVal_Weights(33).Text) * Val(txtVal_Inp(3).Text)
txtVal_Weights(34).Text = Val(txtVal_Weights(34).Text) * Val(txtVal_Inp(4).Text)
txtVal_Weights(35).Text = Val(txtVal_Weights(35).Text) * Val(txtVal_Inp(5).Text)
txtVal_Weights(36).Text = Val(txtVal_Weights(36).Text) * Val(txtVal_Inp(6).Text)
txtVal_Weights(37).Text = Val(txtVal_Weights(37).Text) * Val(txtVal_Inp(7).Text)
txtVal_Weights(38).Text = Val(txtVal_Weights(38).Text) * Val(txtVal_Inp(8).Text)
txtVal_Weights(39).Text = Val(txtVal_Weights(39).Text) * Val(txtVal_Inp(9).Text)
txtVal_Weights(40).Text = Val(txtVal_Weights(40).Text) * Val(txtVal_Inp(10).Text)
txtVal_Weights(41).Text = Val(txtVal_Weights(41).Text) * Val(txtVal_Inp(11).Text)
txtVal_Weights(42).Text = Val(txtVal_Weights(42).Text) * Val(txtVal_Inp(12).Text)
txtVal_Weights(43).Text = Val(txtVal_Weights(43).Text) * Val(txtVal_Inp(13).Text)
txtVal_Weights(44).Text = Val(txtVal_Weights(44).Text) * Val(txtVal_Inp(14).Text)
txtVal_Weights(45).Text = Val(txtVal_Weights(45).Text) * Val(txtVal_Inp(15).Text)
txtVal_Weights(46).Text = Val(txtVal_Weights(46).Text) * Val(txtVal_Inp(16).Text)
txtVal_Weights(47).Text = Val(txtVal_Weights(47).Text) * Val(txtVal_Inp(17).Text)
txtVal_Weights(48).Text = Val(txtVal_Weights(48).Text) * Val(txtVal_Inp(18).Text)
txtVal_Weights(49).Text = Val(txtVal_Weights(49).Text) * Val(txtVal_Inp(19).Text)
txtVal_Weights(50).Text = Val(txtVal_Weights(50).Text) * Val(txtVal_Inp(20).Text)
txtVal_Weights(51).Text = Val(txtVal_Weights(51).Text) * Val(txtVal_Inp(21).Text)
txtVal_Weights(52).Text = Val(txtVal_Weights(52).Text) * Val(txtVal_Inp(22).Text)
txtVal_Weights(53).Text = Val(txtVal_Weights(53).Text) * Val(txtVal_Inp(23).Text)
txtVal_Weights(54).Text = Val(txtVal_Weights(54).Text) * Val(txtVal_Inp(24).Text)
txtVal_Weights(55).Text = Val(txtVal_Weights(55).Text) * Val(txtVal_Inp(25).Text)
txtVal_Weights(56).Text = Val(txtVal_Weights(56).Text) * Val(txtVal_Inp(26).Text)
txtVal_Weights(57).Text = Val(txtVal_Weights(57).Text) * Val(txtVal_Inp(27).Text)
txtVal_Weights(58).Text = Val(txtVal_Weights(58).Text) * Val(txtVal_Inp(28).Text)
txtVal_Weights(59).Text = Val(txtVal_Weights(59).Text) * Val(txtVal_Inp(29).Text)
'compute sigmoid for node2
txtVal_Weights(91).Text = Val(txtVal_Weights(91).Text) * Val(txtVal_Inp(30).Text)
txtNodal(1).Text = Val(txtVal_Weights(91).Text) + Val(txtVal_Weights(30).Text) +
Val(txtVal_Weights(31).Text) + Val(txtVal_Weights(32).Text) +
Val(txtVal_Weights(33).Text) + Val(txtVal_Weights(34).Text) +
Val(txtVal_Weights(35).Text) + Val(txtVal_Weights(36).Text) +
Val(txtVal_Weights(37).Text) + Val(txtVal_Weights(38).Text) +
Val(txtVal_Weights(39).Text) + Val(txtVal_Weights(40).Text) +
Val(txtVal_Weights(41).Text) + Val(txtVal_Weights(42).Text) +
Val(txtVal_Weights(43).Text) + Val(txtVal_Weights(44).Text) +
Val(txtVal_Weights(45).Text) + Val(txtVal_Weights(46).Text) +
Val(txtVal_Weights(47).Text) + Val(txtVal_Weights(48).Text) +
Val(txtVal_Weights(49).Text) + Val(txtVal_Weights(50).Text) +
Val(txtVal_Weights(51).Text) + Val(txtVal_Weights(52).Text) +
Val(txtVal_Weights(53).Text) + Val(txtVal_Weights(54).Text) +
Val(txtVal_Weights(55).Text) + Val(txtVal_Weights(56).Text) +
```

```
Val(txtVal_Weights(57).Text) + Val(txtVal_Weights(58).Text) +
Val(txtVal_Weights(59).Text)
txtNodal(1).Text = 1 / (1 + Exp(-(txtNodal(1).Text)))'node 3
txtVal_Weights(60).Text = Val(txtVal_Weights(60).Text) * Val(txtVal_Inp(0).Text)
txtVal_Weights(61).Text = Val(txtVal_Weights(61).Text) * Val(txtVal_Inp(1).Text)
txtVal_Weights(62).Text = Val(txtVal_Weights(62).Text) * Val(txtVal_Inp(2).Text)
txtVal_Weights(63).Text = Val(txtVal_Weights(63).Text) * Val(txtVal_Inp(3).Text)
txtVal_Weights(64).Text = Val(txtVal_Weights(64).Text) * Val(txtVal_Inp(4).Text)
txtVal_Weights(65).Text = Val(txtVal_Weights(65).Text) * Val(txtVal_Inp(5).Text)
txtVal_Weights(66).Text = Val(txtVal_Weights(66).Text) * Val(txtVal_Inp(6).Text)
txtVal_Weights(67).Text = Val(txtVal_Weights(67).Text) * Val(txtVal_Inp(7).Text)
txtVal_Weights(68).Text = Val(txtVal_Weights(68).Text) * Val(txtVal_Inp(8).Text)
txtVal_Weights(69).Text = Val(txtVal_Weights(69).Text) * Val(txtVal_Inp(9).Text)
txtVal_Weights(70).Text = Val(txtVal_Weights(70).Text) * Val(txtVal_Inp(10).Text)
txtVal_Weights(71).Text = Val(txtVal_Weights(71).Text) * Val(txtVal_Inp(11).Text)
txtVal_Weights(72).Text = Val(txtVal_Weights(72).Text) * Val(txtVal_Inp(12).Text)
txtVal_Weights(73).Text = Val(txtVal_Weights(73).Text) * Val(txtVal_Inp(13).Text)
txtVal_Weights(74).Text = Val(txtVal_Weights(74).Text) * Val(txtVal_Inp(14).Text)
txtVal_Weights(75).Text = Val(txtVal_Weights(75).Text) * Val(txtVal_Inp(15).Text)
txtVal_Weights(76).Text = Val(txtVal_Weights(76).Text) * Val(txtVal_Inp(16).Text)
txtVal_Weights(77).Text = Val(txtVal_Weights(77).Text) * Val(txtVal_Inp(17).Text)
txtVal_Weights(78).Text = Val(txtVal_Weights(78).Text) * Val(txtVal_Inp(18).Text)
txtVal_Weights(79).Text = Val(txtVal_Weights(79).Text) * Val(txtVal_Inp(19).Text)
txtVal_Weights(80).Text = Val(txtVal_Weights(80).Text) * Val(txtVal_Inp(20).Text)
txtVal_Weights(81).Text = Val(txtVal_Weights(81).Text) * Val(txtVal_Inp(21).Text)
txtVal_Weights(82).Text = Val(txtVal_Weights(82).Text) * Val(txtVal_Inp(22).Text)
txtVal_Weights(83).Text = Val(txtVal_Weights(83).Text) * Val(txtVal_Inp(23).Text)
txtVal_Weights(84).Text = Val(txtVal_Weights(84).Text) * Val(txtVal_Inp(24).Text)
txtVal_Weights(85).Text = Val(txtVal_Weights(85).Text) * Val(txtVal_Inp(25).Text)
txtVal_Weights(86).Text = Val(txtVal_Weights(86).Text) * Val(txtVal_Inp(26).Text)
txtVal_Weights(87).Text = Val(txtVal_Weights(87).Text) * Val(txtVal_Inp(27).Text)
txtVal_Weights(88).Text = Val(txtVal_Weights(88).Text) * Val(txtVal_Inp(28).Text)
txtVal_Weights(89).Text = Val(txtVal_Weights(89).Text) * Val(txtVal_Inp(29).Text)
'compute sigmoid for node3
txtVal_Weights(92).Text = Val(txtVal_Weights(92).Text) * Val(txtVal_Inp(30).Text)
txtNodal(2).Text = Val(txtVal_Weights(92).Text) + Val(txtVal_Weights(60).Text) +
Val(txtVal_Weights(61).Text) + Val(txtVal_Weights(62).Text) +
Val(txtVal_Weights(63).Text) + Val(txtVal_Weights(64).Text) +
Val(txtVal_Weights(65).Text) + Val(txtVal_Weights(66).Text) +
Val(txtVal_Weights(67).Text) + Val(txtVal_Weights(68).Text) +
Val(txtVal_Weights(69).Text) + Val(txtVal_Weights(70).Text) +
Val(txtVal_Weights(71).Text) + Val(txtVal_Weights(72).Text) +
Val(txtVal_Weights(73).Text) + Val(txtVal_Weights(74).Text) +
Val(txtVal_Weights(75).Text) + Val(txtVal_Weights(76).Text) +
Val(txtVal_Weights(77).Text) + Val(txtVal_Weights(78).Text) +
Val(txtVal_Weights(79).Text) + Val(txtVal_Weights(80).Text) +
Val(txtVal_Weights(81).Text) + Val(txtVal_Weights(82).Text) +
Val(txtVal_Weights(83).Text) + Val(txtVal_Weights(84).Text) +
Val(txtVal_Weights(85).Text) + Val(txtVal_Weights(86).Text) +
Val(txtVal_Weights(87).Text) + Val(txtVal_Weights(88).Text) +
Val(txtVal_Weights(89).Text)
txtNodal(2).Text = 1 / (1 + Exp(-(txtNodal(2).Text)))'node 4
```

```
txtVal_Weights(113).Text = Val(txtVal_Weights(113).Text) * Val(txtVal_Inp(0).Text)
txtVal_Weights(114).Text = Val(txtVal_Weights(114).Text) * Val(txtVal_Inp(1).Text)
txtVal_Weights(115).Text = Val(txtVal_Weights(115).Text) * Val(txtVal_Inp(2).Text)
txtVal_Weights(116).Text = Val(txtVal_Weights(116).Text) * Val(txtVal_Inp(3).Text)
txtVal_Weights(117).Text = Val(txtVal_Weights(117).Text) * Val(txtVal_Inp(4).Text)
txtVal_Weights(118).Text = Val(txtVal_Weights(118).Text) * Val(txtVal_Inp(5).Text)
txtVal_Weights(119).Text = Val(txtVal_Weights(119).Text) * Val(txtVal_Inp(6).Text)
txtVal_Weights(120).Text = Val(txtVal_Weights(120).Text) * Val(txtVal_Inp(7).Text)
txtVal_Weights(121).Text = Val(txtVal_Weights(121).Text) * Val(txtVal_Inp(8).Text)
txtVal_Weights(122).Text = Val(txtVal_Weights(122).Text) * Val(txtVal_Inp(9).Text)
txtVal_Weights(123).Text = Val(txtVal_Weights(123).Text) * Val(txtVal_Inp(10).Text)
txtVal_Weights(124).Text = Val(txtVal_Weights(124).Text) * Val(txtVal_Inp(11).Text)
txtVal_Weights(125).Text = Val(txtVal_Weights(125).Text) * Val(txtVal_Inp(12).Text)
txtVal_Weights(126).Text = Val(txtVal_Weights(126).Text) * Val(txtVal_Inp(13).Text)
txtVal_Weights(127).Text = Val(txtVal_Weights(127).Text) * Val(txtVal_Inp(14).Text)
txtVal_Weights(128).Text = Val(txtVal_Weights(128).Text) * Val(txtVal_Inp(15).Text)
txtVal_Weights(129).Text = Val(txtVal_Weights(129).Text) * Val(txtVal_Inp(16).Text)
txtVal_Weights(130).Text = Val(txtVal_Weights(130).Text) * Val(txtVal_Inp(17).Text)
txtVal_Weights(131).Text = Val(txtVal_Weights(131).Text) * Val(txtVal_Inp(18).Text)
txtVal_Weights(132).Text = Val(txtVal_Weights(132).Text) * Val(txtVal_Inp(19).Text)
txtVal_Weights(133).Text = Val(txtVal_Weights(133).Text) * Val(txtVal_Inp(20).Text)
txtVal_Weights(134).Text = Val(txtVal_Weights(134).Text) * Val(txtVal_Inp(21).Text)
txtVal_Weights(135).Text = Val(txtVal_Weights(135).Text) * Val(txtVal_Inp(22).Text)
txtVal_Weights(136).Text = Val(txtVal_Weights(136).Text) * Val(txtVal_Inp(23).Text)
txtVal_Weights(137).Text = Val(txtVal_Weights(137).Text) * Val(txtVal_Inp(24).Text)
txtVal_Weights(138).Text = Val(txtVal_Weights(138).Text) * Val(txtVal_Inp(25).Text)
txtVal_Weights(139).Text = Val(txtVal_Weights(139).Text) * Val(txtVal_Inp(26).Text)
txtVal_Weights(140).Text = Val(txtVal_Weights(140).Text) * Val(txtVal_Inp(27).Text)
txtVal_Weights(141).Text = Val(txtVal_Weights(141).Text) * Val(txtVal_Inp(28).Text)
txtVal_Weights(142).Text = Val(txtVal_Weights(142).Text) * Val(txtVal_Inp(29).Text)
'compute sigmoid for node4
txtVal_Weights(173).Text = Val(txtVal_Weights(173).Text) * Val(txtVal_Inp(30).Text)
txtNodal(3).Text = Val(txtVal_Weights(173).Text) + Val(txtVal_Weights(113).Text) +
Val(txtVal_Weights(114).Text)      +      Val(txtVal_Weights(115).Text)      +
Val(txtVal_Weights(116).Text)      +      Val(txtVal_Weights(117).Text)      +
Val(txtVal_Weights(118).Text)      +      Val(txtVal_Weights(119).Text)      +
Val(txtVal_Weights(120).Text)      +      Val(txtVal_Weights(121).Text)      +
Val(txtVal_Weights(122).Text)      +      Val(txtVal_Weights(123).Text)      +
Val(txtVal_Weights(124).Text)      +      Val(txtVal_Weights(125).Text)      +
Val(txtVal_Weights(126).Text)      +      Val(txtVal_Weights(127).Text)      +
Val(txtVal_Weights(128).Text)      +      Val(txtVal_Weights(129).Text)      +
Val(txtVal_Weights(130).Text)      +      Val(txtVal_Weights(131).Text)      +
Val(txtVal_Weights(132).Text)      +      Val(txtVal_Weights(133).Text)      +
Val(txtVal_Weights(134).Text)      +      Val(txtVal_Weights(135).Text)      +
Val(txtVal_Weights(136).Text)      +      Val(txtVal_Weights(137).Text)      +
Val(txtVal_Weights(138).Text)      +      Val(txtVal_Weights(139).Text)      +
Val(txtVal_Weights(140).Text)      +      Val(txtVal_Weights(141).Text)      +
Val(txtVal_Weights(142).Text)
txtNodal(3).Text = 1 / (1 + Exp(-(txtNodal(3).Text)))'node 5
txtVal_Weights(143).Text = Val(txtWeights(143).Text) * Val(txtVal_Inp(0).Text)
txtVal_Weights(144).Text = Val(txtWeights(144).Text) * Val(txtVal_Inp(1).Text)
txtVal_Weights(145).Text = Val(txtWeights(145).Text) * Val(txtVal_Inp(2).Text)
```

```
txtVal_Weights(146).Text = Val(txtWeights(146).Text) * Val(txtVal_Inp(3).Text)
txtVal_Weights(147).Text = Val(txtWeights(147).Text) * Val(txtVal_Inp(4).Text)
txtVal_Weights(148).Text = Val(txtWeights(148).Text) * Val(txtVal_Inp(5).Text)
txtVal_Weights(149).Text = Val(txtWeights(149).Text) * Val(txtVal_Inp(6).Text)
txtVal_Weights(150).Text = Val(txtWeights(150).Text) * Val(txtVal_Inp(7).Text)
txtVal_Weights(151).Text = Val(txtWeights(151).Text) * Val(txtVal_Inp(8).Text)
txtVal_Weights(152).Text = Val(txtWeights(152).Text) * Val(txtVal_Inp(9).Text)
txtVal_Weights(153).Text = Val(txtWeights(153).Text) * Val(txtVal_Inp(10).Text)
txtVal_Weights(154).Text = Val(txtWeights(154).Text) * Val(txtVal_Inp(11).Text)
txtVal_Weights(155).Text = Val(txtWeights(155).Text) * Val(txtVal_Inp(12).Text)
txtVal_Weights(156).Text = Val(txtWeights(156).Text) * Val(txtVal_Inp(13).Text)
txtVal_Weights(157).Text = Val(txtWeights(157).Text) * Val(txtVal_Inp(14).Text)
txtVal_Weights(158).Text = Val(txtWeights(158).Text) * Val(txtVal_Inp(15).Text)
txtVal_Weights(159).Text = Val(txtWeights(159).Text) * Val(txtVal_Inp(16).Text)
txtVal_Weights(160).Text = Val(txtWeights(160).Text) * Val(txtVal_Inp(17).Text)
txtVal_Weights(161).Text = Val(txtWeights(161).Text) * Val(txtVal_Inp(18).Text)
txtVal_Weights(162).Text = Val(txtWeights(162).Text) * Val(txtVal_Inp(19).Text)
txtVal_Weights(163).Text = Val(txtWeights(163).Text) * Val(txtVal_Inp(20).Text)
txtVal_Weights(164).Text = Val(txtWeights(164).Text) * Val(txtVal_Inp(21).Text)
txtVal_Weights(165).Text = Val(txtWeights(165).Text) * Val(txtVal_Inp(22).Text)
txtVal_Weights(166).Text = Val(txtWeights(166).Text) * Val(txtVal_Inp(23).Text)
txtVal_Weights(167).Text = Val(txtWeights(167).Text) * Val(txtVal_Inp(24).Text)
txtVal_Weights(168).Text = Val(txtWeights(168).Text) * Val(txtVal_Inp(25).Text)
txtVal_Weights(169).Text = Val(txtWeights(169).Text) * Val(txtVal_Inp(26).Text)
txtVal_Weights(170).Text = Val(txtWeights(170).Text) * Val(txtVal_Inp(27).Text)
txtVal_Weights(171).Text = Val(txtWeights(171).Text) * Val(txtVal_Inp(28).Text)
txtVal_Weights(172).Text = Val(txtWeights(172).Text) * Val(txtVal_Inp(29).Text)
'compute sigmoid for node5
txtVal_Weights(174).Text = Val(txtWeights(174).Text) * Val(txtVal_Inp(30).Text)
txtNodal(4).Text = Val(txtVal_Weights(174).Text) + Val(txtVal_Weights(143).Text) +
Val(txtVal_Weights(144).Text) + Val(txtVal_Weights(145).Text) +
Val(txtVal_Weights(146).Text) + Val(txtVal_Weights(147).Text) +
Val(txtVal_Weights(148).Text) + Val(txtVal_Weights(149).Text) +
Val(txtVal_Weights(150).Text) + Val(txtVal_Weights(151).Text) +
Val(txtVal_Weights(152).Text) + Val(txtVal_Weights(153).Text) +
Val(txtVal_Weights(154).Text) + Val(txtVal_Weights(155).Text) +
Val(txtVal_Weights(156).Text) + Val(txtVal_Weights(157).Text) +
Val(txtVal_Weights(158).Text) + Val(txtVal_Weights(159).Text) +
Val(txtVal_Weights(160).Text) + Val(txtVal_Weights(161).Text) +
Val(txtVal_Weights(162).Text) + Val(txtVal_Weights(163).Text) +
Val(txtVal_Weights(164).Text) + Val(txtVal_Weights(165).Text) +
Val(txtVal_Weights(166).Text) + Val(txtVal_Weights(167).Text) +
Val(txtVal_Weights(168).Text) + Val(txtVal_Weights(169).Text) +
Val(txtVal_Weights(170).Text) + Val(txtVal_Weights(171).Text) +
Val(txtVal_Weights(172).Text)
txtNodal(4).Text = 1 / (1 + Exp(-(txtNodal(4).Text)))
'wieght adjustments for hidden layers (first forward sweep)
txtOut1.Text = Val(txtVal_Weights(112).Text) + Val(txtNodal(0).Text) *
Val(txtVal_Weights(90).Text) + Val(txtNodal(1).Text) * Val(txtVal_Weights(91).Text) +
Val(txtNodal(2).Text) * Val(txtVal_Weights(92).Text) + Val(txtNodal(3).Text) *
Val(txtVal_Weights(173).Text) + Val(txtNodal(4).Text) * Val(txtVal_Weights(174).Text)
```

```
txtOut2.Text = Val(txtVal_Weights(111).Text) + Val(txtNodal(0).Text) * Val(txtVal_Weights(93).Text) + Val(txtNodal(1).Text) * Val(txtVal_Weights(94).Text) + Val(txtNodal(2).Text) * Val(txtVal_Weights(95).Text) + Val(txtNodal(3).Text) * Val(txtVal_Weights(175).Text) + Val(txtNodal(4).Text) * Val(txtVal_Weights(176).Text)
txtOut3.Text = Val(txtVal_Weights(110).Text) + Val(txtNodal(0).Text) * Val(txtVal_Weights(96).Text) + Val(txtNodal(1).Text) * Val(txtVal_Weights(97).Text) + Val(txtNodal(2).Text) * Val(txtVal_Weights(98).Text) + Val(txtNodal(3).Text) * Val(txtVal_Weights(177).Text) + Val(txtNodal(4).Text) * Val(txtVal_Weights(178).Text)
txtOut4.Text = Val(txtVal_Weights(109).Text) + Val(txtNodal(0).Text) * Val(txtVal_Weights(99).Text) + Val(txtNodal(1).Text) * Val(txtVal_Weights(100).Text) + Val(txtNodal(2).Text) * Val(txtVal_Weights(101).Text) + Val(txtNodal(3).Text) * Val(txtVal_Weights(179).Text) + Val(txtNodal(4).Text) * Val(txtVal_Weights(180).Text)
txtOut5.Text = Val(txtVal_Weights(108).Text) + Val(txtNodal(0).Text) * Val(txtVal_Weights(102).Text) + Val(txtNodal(1).Text) * Val(txtVal_Weights(103).Text) + Val(txtNodal(2).Text) * Val(txtVal_Weights(104).Text) + Val(txtNodal(3).Text) * Val(txtVal_Weights(181).Text) + Val(txtNodal(4).Text) * Val(txtVal_Weights(182).Text)
txtOut1.Text = 1 / (1 + Exp(-(txtOut1.Text)))
txtOut2.Text = 1 / (1 + Exp(-(txtOut2.Text)))
txtOut3.Text = 1 / (1 + Exp(-(txtOut3.Text)))
txtOut4.Text = 1 / (1 + Exp(-(txtOut4.Text)))
txtOut5.Text = 1 / (1 + Exp(-(txtOut5.Text)))
Call WeightList
End Sub
Private Sub Iterate()
Dim i As Integer'wieght adjustments for input layers (first forward sweep)
If lblOutput.Caption = "0" Then
'plot initial weight
Call PlotWeight'node1
For i = 0 To 29
 txtEpochWeight(i).Text = Val(txtWeights(i).Text) * Val(txtQuantity(i).Text)
Next i
txtEpochWeight(92).Text = Val(txtWeights(105).Text) * Val(txtQuantity(30).Text)
txtNode(0).Text = Val(txtEpochWeight(92).Text) + Val(txtEpochWeight(0).Text) + Val(txtEpochWeight(1).Text) + Val(txtEpochWeight(2).Text) + Val(txtEpochWeight(3).Text) + Val(txtEpochWeight(4).Text) + Val(txtEpochWeight(5).Text) + Val(txtEpochWeight(6).Text) + Val(txtEpochWeight(7).Text) + Val(txtEpochWeight(8).Text) + Val(txtEpochWeight(9).Text) + Val(txtEpochWeight(10).Text) + Val(txtEpochWeight(11).Text) + Val(txtEpochWeight(12).Text) + Val(txtEpochWeight(13).Text) + Val(txtEpochWeight(14).Text) + Val(txtEpochWeight(15).Text) + Val(txtEpochWeight(16).Text) + Val(txtEpochWeight(17).Text) + Val(txtEpochWeight(18).Text) + Val(txtEpochWeight(19).Text) + Val(txtEpochWeight(20).Text) + Val(txtEpochWeight(21).Text) + Val(txtEpochWeight(22).Text) + Val(txtEpochWeight(23).Text) + Val(txtEpochWeight(24).Text) + Val(txtEpochWeight(25).Text) + Val(txtEpochWeight(26).Text) + Val(txtEpochWeight(27).Text) + Val(txtEpochWeight(28).Text) + Val(txtEpochWeight(29).Text)
txtNode(0).Text = 1 / (1 + Exp(-(txtNode(0).Text)))'node2
txtEpochWeight(30).Text = Val(txtWeights(30).Text) * Val(txtQuantity(0).Text)
txtEpochWeight(31).Text = Val(txtWeights(31).Text) * Val(txtQuantity(1).Text)
txtEpochWeight(32).Text = Val(txtWeights(32).Text) * Val(txtQuantity(2).Text)
```

```
txtEpochWeight(33).Text = Val(txtWeights(33).Text) * Val(txtQuantity(3).Text)
txtEpochWeight(34).Text = Val(txtWeights(34).Text) * Val(txtQuantity(4).Text)
txtEpochWeight(35).Text = Val(txtWeights(35).Text) * Val(txtQuantity(5).Text)
txtEpochWeight(36).Text = Val(txtWeights(36).Text) * Val(txtQuantity(6).Text)
txtEpochWeight(37).Text = Val(txtWeights(37).Text) * Val(txtQuantity(7).Text)
txtEpochWeight(38).Text = Val(txtWeights(38).Text) * Val(txtQuantity(8).Text)
txtEpochWeight(39).Text = Val(txtWeights(39).Text) * Val(txtQuantity(9).Text)
txtEpochWeight(40).Text = Val(txtWeights(40).Text) * Val(txtQuantity(10).Text)
txtEpochWeight(41).Text = Val(txtWeights(41).Text) * Val(txtQuantity(11).Text)
txtEpochWeight(42).Text = Val(txtWeights(42).Text) * Val(txtQuantity(12).Text)
txtEpochWeight(43).Text = Val(txtWeights(43).Text) * Val(txtQuantity(13).Text)
txtEpochWeight(44).Text = Val(txtWeights(44).Text) * Val(txtQuantity(14).Text)
txtEpochWeight(45).Text = Val(txtWeights(45).Text) * Val(txtQuantity(15).Text)
txtEpochWeight(46).Text = Val(txtWeights(46).Text) * Val(txtQuantity(16).Text)
txtEpochWeight(47).Text = Val(txtWeights(47).Text) * Val(txtQuantity(17).Text)
txtEpochWeight(48).Text = Val(txtWeights(48).Text) * Val(txtQuantity(18).Text)
txtEpochWeight(49).Text = Val(txtWeights(49).Text) * Val(txtQuantity(19).Text)
txtEpochWeight(50).Text = Val(txtWeights(50).Text) * Val(txtQuantity(20).Text)
txtEpochWeight(51).Text = Val(txtWeights(51).Text) * Val(txtQuantity(21).Text)
txtEpochWeight(52).Text = Val(txtWeights(52).Text) * Val(txtQuantity(22).Text)
txtEpochWeight(53).Text = Val(txtWeights(53).Text) * Val(txtQuantity(23).Text)
txtEpochWeight(54).Text = Val(txtWeights(54).Text) * Val(txtQuantity(24).Text)
txtEpochWeight(55).Text = Val(txtWeights(55).Text) * Val(txtQuantity(25).Text)
txtEpochWeight(56).Text = Val(txtWeights(56).Text) * Val(txtQuantity(26).Text)
txtEpochWeight(57).Text = Val(txtWeights(57).Text) * Val(txtQuantity(27).Text)
txtEpochWeight(58).Text = Val(txtWeights(58).Text) * Val(txtQuantity(28).Text)
txtEpochWeight(59).Text = Val(txtWeights(59).Text) * Val(txtQuantity(29).Text)
'compute sigmoid for node2
txtEpochWeight(91).Text = Val(txtWeights(106).Text) * Val(txtQuantity(30).Text)
txtNode(1).Text = Val(txtEpochWeight(91).Text) + Val(txtEpochWeight(30).Text) +
Val(txtEpochWeight(31).Text) + Val(txtEpochWeight(32).Text) +
Val(txtEpochWeight(33).Text) + Val(txtEpochWeight(34).Text) +
Val(txtEpochWeight(35).Text) + Val(txtEpochWeight(36).Text) +
Val(txtEpochWeight(37).Text) + Val(txtEpochWeight(38).Text) +
Val(txtEpochWeight(39).Text) + Val(txtEpochWeight(40).Text) +
Val(txtEpochWeight(41).Text) + Val(txtEpochWeight(42).Text) +
Val(txtEpochWeight(43).Text) + Val(txtEpochWeight(44).Text) +
Val(txtEpochWeight(45).Text) + Val(txtEpochWeight(46).Text) +
Val(txtEpochWeight(47).Text) + Val(txtEpochWeight(48).Text) +
Val(txtEpochWeight(49).Text) + Val(txtEpochWeight(50).Text) +
Val(txtEpochWeight(51).Text) + Val(txtEpochWeight(52).Text) +
Val(txtEpochWeight(53).Text) + Val(txtEpochWeight(54).Text) +
Val(txtEpochWeight(55).Text) + Val(txtEpochWeight(56).Text) +
Val(txtEpochWeight(57).Text) + Val(txtEpochWeight(58).Text) +
Val(txtEpochWeight(59).Text)
txtNode(1).Text = 1 / (1 + Exp(-(txtNode(1).Text)))
'wieght adjustments for hidden layers (first forward sweep)
txtOutput1.Text = Val(txtWeights(112).Text) + Val(txtNode(0).Text) *
Val(txtWeights(90).Text) + Val(txtNode(1).Text) * Val(txtWeights(91).Text)
txtOutput2.Text = Val(txtWeights(111).Text) + Val(txtNode(0).Text) *
Val(txtWeights(93).Text) + Val(txtNode(1).Text) * Val(txtWeights(94).Text)
```

```
txtOutput3.Text = Val(txtWeights(110).Text) + Val(txtNode(0).Text) * Val(txtWeights(96).Text) + Val(txtNode(1).Text) * Val(txtWeights(97).Text)
txtOutput4.Text = Val(txtWeights(109).Text) + Val(txtNode(0).Text) * Val(txtWeights(99).Text) + Val(txtNode(1).Text) * Val(txtWeights(100).Text)
txtOutput5.Text = Val(txtWeights(108).Text) + Val(txtNode(0).Text) * Val(txtWeights(102).Text) + Val(txtNode(1).Text) * Val(txtWeights(103).Text)
txtOutput1.Text = 1 / (1 + Exp(-(txtOutput1.Text)))
txtOutput2.Text = 1 / (1 + Exp(-(txtOutput2.Text)))
txtOutput3.Text = 1 / (1 + Exp(-(txtOutput3.Text)))
txtOutput4.Text = 1 / (1 + Exp(-(txtOutput4.Text)))
txtOutput5.Text = 1 / (1 + Exp(-(txtOutput5.Text)))
Call counter
Else
End If
Call WeightList
Exit Sub
End Sub
Private Sub IterateTwo()
Dim i As Integer
 'Woz,new = Woz,current + Change in Woz
dz1 = (Val(txtLearnR_n.Text) * Val(txtOutput1.Text) * (1 - Val(txtOutput1.Text)) * (Val(txtActual(0).Text) - Val(txtOutput1.Text)) * Val(txtQuantity(30).Text))
dz2 = (Val(txtLearnR_n.Text) * Val(txtOutput2.Text) * (1 - Val(txtOutput2.Text)) * (Val(txtActual(1).Text) - Val(txtOutput2.Text)) * Val(txtQuantity(30).Text))
dz3 = (Val(txtLearnR_n.Text) * Val(txtOutput3.Text) * (1 - Val(txtOutput3.Text)) * (Val(txtActual(2).Text) - Val(txtOutput3.Text)) * Val(txtQuantity(30).Text))
dz4 = (Val(txtLearnR_n.Text) * Val(txtOutput4.Text) * (1 - Val(txtOutput4.Text)) * (Val(txtActual(3).Text) - Val(txtOutput4.Text)) * Val(txtQuantity(30).Text))
dz5 = (Val(txtLearnR_n.Text) * Val(txtOutput5.Text) * (1 - Val(txtOutput5.Text)) * (Val(txtActual(4).Text) - Val(txtOutput5.Text)) * Val(txtQuantity(30).Text))
'using dz(i) combo size
txtWeights(112).Text = Val(txtWeights(112).Text) + dz1
txtWeights(111).Text = Val(txtWeights(111).Text) + dz2
txtWeights(110).Text = Val(txtWeights(110).Text) + dz3
txtWeights(109).Text = Val(txtWeights(109).Text) + dz4
txtWeights(108).Text = Val(txtWeights(108).Text) + dz5
'wieght adjustments for hidden layers (Next (i) forward sweep)
'error responsibility dA
      'dA = outputNA(1 - outputNA)E(downstream)WjkDj
dA1 = Val(txtNode(0).Text) * (1 - Val(txtNode(0).Text)) * Val(txtWeights(90).Text) * dz1
                'then compute Change in WAz = ndz(1).OutputA
   CAz1 = Val(txtLearnR_n.Text) * dz1 * Val(txtNode(0).Text)
   'WAz,new = WAz,current + Change in WAz
   txtWeights(90).Text = Val(txtWeights(90).Text) + CAz1 'next hidden layer2
dA2 = Val(txtNode(0).Text) * (1 - Val(txtNode(0).Text)) * Val(txtWeights(93).Text) * dz2
                'then compute Change in WAz = ndz(1).OutputA
   CAz2 = Val(txtLearnR_n.Text) * dz2 * Val(txtNode(0).Text)
   'WAz,new = WAz,current + Change in WAz
   txtWeights(93).Text = Val(txtWeights(93).Text) + CAz2   'next hidden layer3
dA3 = Val(txtNode(0).Text) * (1 - Val(txtNode(0).Text)) * Val(txtWeights(96).Text) * dz3
                 'then compute Change in WAz = ndz(1).OutputA
   CAz3 = Val(txtLearnR_n.Text) * dz3 * Val(txtNode(0).Text)
```

```
   'WAz,new = WAz,current + Change in WAz
   txtWeights(96).Text = Val(txtWeights(96).Text) + CAz3 'next hidden layer4
dA4 = Val(txtNode(0).Text) * (1 - Val(txtNode(0).Text)) * Val(txtWeights(99).Text) * dz4
                'then compute Change in WAz = ndz(1).OutputA
   CAz4 = Val(txtLearnR_n.Text) * dz4 * Val(txtNode(0).Text)
   'WAz,new = WAz,current + Change in WAz
   txtWeights(99).Text = Val(txtWeights(99).Text) + CAz4 'next hidden layer5
dA5 = Val(txtNode(0).Text) * (1 - Val(txtNode(0).Text)) * Val(txtWeights(102).Text) * dz5
                 'then compute Change in WAz = ndz(1).OutputA
   CAz5 = Val(txtLearnR_n.Text) * dz5 * Val(txtNode(0).Text)
   'WAz,new = WAz,current + Change in WAz
   txtWeights(102).Text = Val(txtWeights(102).Text) + CAz5
   'error responsibility dB        'dB = outputNB(1 - outputNB)E(downstream)WjkDj
dB1 = Val(txtNode(1).Text) * (1 - Val(txtNode(1).Text)) * Val(txtWeights(91).Text) * dz1
                'then compute Change in WBz = ndz(1).OutputB
   CBz1 = Val(txtLearnR_n.Text) * dz1 * Val(txtNode(1).Text)
   'WBz,new = WBz,current + Change in WBz
   txtWeights(91).Text = Val(txtWeights(91).Text) + CBz1 'Next hidden layer
      'dB = outputNB(1 - outputNB)E(downstream)WjkDj
dB2 = Val(txtNode(1).Text) * (1 - Val(txtNode(1).Text)) * Val(txtWeights(94).Text) * dz2
                 'then compute Change in WBz = ndz(1).OutputB
   CBz2 = Val(txtLearnR_n.Text) * dz2 * Val(txtNode(1).Text)
   'WBz,new = WBz,current + Change in WBz
   txtWeights(94).Text = Val(txtWeights(94).Text) + CBz2 'Next hidden layer
      'dB = outputNB(1 - outputNB)E(downstream)WjkDj
dB3 = Val(txtNode(1).Text) * (1 - Val(txtNode(1).Text)) * Val(txtWeights(97).Text) * dz3
                'then compute Change in WBz = ndz(1).OutputB
   CBz3 = Val(txtLearnR_n.Text) * dz3 * Val(txtNode(1).Text)
   'WBz,new = WBz,current + Change in WBz
   txtWeights(97).Text = Val(txtWeights(97).Text) + CBz3  'Next hidden layer
      'dB = outputNB(1 - outputNB)E(downstream)WjkDj
dB4 = Val(txtNode(1).Text) * (1 - Val(txtNode(1).Text)) * Val(txtWeights(100).Text) * dz4
                 'then compute Change in WBz = ndz(1).OutputB
   CBz4 = Val(txtLearnR_n.Text) * dz4 * Val(txtNode(1).Text)
   'WBz,new = WBz,current + Change in WBz
   txtWeights(100).Text = Val(txtWeights(100).Text) + CBz4        'Next hidden layer
      'dB = outputNB(1 - outputNB)E(downstream)WjkDj
dB5 = Val(txtNode(1).Text) * (1 - Val(txtNode(1).Text)) * Val(txtWeights(103).Text) * dz5
                'then compute Change in WBz = ndz(1).OutputB
   CBz5 = Val(txtLearnR_n.Text) * dz5 * Val(txtNode(1).Text)
   'WBz,new = WBz,current + Change in WBz
   txtWeights(103).Text = Val(txtWeights(103).Text) + CBz5
   'wieght adjustments for inputs (Next (i) forward sweep)
      'compute Change in WiA = ndAXi  'WiA,new = WiA,current + Change in WiA
   CW1A = Val(txtLearnR_n.Text) * dA1 * Val(txtQuantity(0).Text)
   txtWeights(0).Text = Val(txtWeights(0).Text) + CW1A
   CW2A = Val(txtLearnR_n.Text) * dA1 * Val(txtQuantity(1).Text)
   txtWeights(1).Text = Val(txtWeights(1).Text) + CW2A
   CW3A = Val(txtLearnR_n.Text) * dA1 * Val(txtQuantity(2).Text)
   txtWeights(2).Text = Val(txtWeights(2).Text) + CW3A
   CW4A = Val(txtLearnR_n.Text) * dA1 * Val(txtQuantity(3).Text)
   txtWeights(3).Text = Val(txtWeights(3).Text) + CW4A
```

```
CW5A = Val(txtLearnR_n.Text) * dA1 * Val(txtQuantity(4).Text)
txtWeights(4).Text = Val(txtWeights(4).Text) + CW5A
CW6A = Val(txtLearnR_n.Text) * dA1 * Val(txtQuantity(5).Text)
txtWeights(5).Text = Val(txtWeights(5).Text) + CW6A
CW7A = Val(txtLearnR_n.Text) * dA1 * Val(txtQuantity(6).Text)
txtWeights(6).Text = Val(txtWeights(6).Text) + CW7A
CW8A = Val(txtLearnR_n.Text) * dA1 * Val(txtQuantity(7).Text)
txtWeights(7).Text = Val(txtWeights(7).Text) + CW8A
CW9A = Val(txtLearnR_n.Text) * dA1 * Val(txtQuantity(8).Text)
txtWeights(8).Text = Val(txtWeights(8).Text) + CW9A
CW10A = Val(txtLearnR_n.Text) * dA1 * Val(txtQuantity(9).Text)
txtWeights(9).Text = Val(txtWeights(9).Text) + CW10A
CW11A = Val(txtLearnR_n.Text) * dA1 * Val(txtQuantity(10).Text)
txtWeights(10).Text = Val(txtWeights(10).Text) + CW11A
CW12A = Val(txtLearnR_n.Text) * dA1 * Val(txtQuantity(11).Text)
txtWeights(11).Text = Val(txtWeights(11).Text) + CW12A
CW13A = Val(txtLearnR_n.Text) * dA1 * Val(txtQuantity(12).Text)
txtWeights(12).Text = Val(txtWeights(12).Text) + CW13A
CW14A = Val(txtLearnR_n.Text) * dA1 * Val(txtQuantity(13).Text)
txtWeights(13).Text = Val(txtWeights(13).Text) + CW14A
CW15A = Val(txtLearnR_n.Text) * dA1 * Val(txtQuantity(14).Text)
txtWeights(14).Text = Val(txtWeights(14).Text) + CW15A
CW16A = Val(txtLearnR_n.Text) * dA1 * Val(txtQuantity(15).Text)
txtWeights(15).Text = Val(txtWeights(15).Text) + CW16A
CW17A = Val(txtLearnR_n.Text) * dA1 * Val(txtQuantity(16).Text)
txtWeights(16).Text = Val(txtWeights(16).Text) + CW17A
CW18A = Val(txtLearnR_n.Text) * dA1 * Val(txtQuantity(17).Text)
txtWeights(17).Text = Val(txtWeights(17).Text) + CW18A
CW19A = Val(txtLearnR_n.Text) * dA1 * Val(txtQuantity(18).Text)
txtWeights(18).Text = Val(txtWeights(18).Text) + CW19A
CW20A = Val(txtLearnR_n.Text) * dA1 * Val(txtQuantity(19).Text)
txtWeights(19).Text = Val(txtWeights(19).Text) + CW20A
CW21A = Val(txtLearnR_n.Text) * dA1 * Val(txtQuantity(20).Text)
txtWeights(20).Text = Val(txtWeights(20).Text) + CW21A
CW22A = Val(txtLearnR_n.Text) * dA1 * Val(txtQuantity(21).Text)
txtWeights(21).Text = Val(txtWeights(21).Text) + CW21A
CW23A = Val(txtLearnR_n.Text) * dA1 * Val(txtQuantity(22).Text)
txtWeights(22).Text = Val(txtWeights(22).Text) + CW23A
CW24A = Val(txtLearnR_n.Text) * dA1 * Val(txtQuantity(23).Text)
txtWeights(23).Text = Val(txtWeights(23).Text) + CW24A
CW25A = Val(txtLearnR_n.Text) * dA1 * Val(txtQuantity(24).Text)
txtWeights(24).Text = Val(txtWeights(24).Text) + CW25A
CW26A = Val(txtLearnR_n.Text) * dA1 * Val(txtQuantity(25).Text)
txtWeights(25).Text = Val(txtWeights(25).Text) + CW22A
CW27A = Val(txtLearnR_n.Text) * dA1 * Val(txtQuantity(26).Text)
txtWeights(26).Text = Val(txtWeights(26).Text) + CW27A
CW28A = Val(txtLearnR_n.Text) * dA1 * Val(txtQuantity(27).Text)
txtWeights(27).Text = Val(txtWeights(27).Text) + CW28A
CW29A = Val(txtLearnR_n.Text) * dA1 * Val(txtQuantity(28).Text)
txtWeights(28).Text = Val(txtWeights(28).Text) + CW29A
CW30A = Val(txtLearnR_n.Text) * dA1 * Val(txtQuantity(29).Text)
txtWeights(29).Text = Val(txtWeights(29).Text) + CW30A
```

```
CW0A = Val(txtLearnR_n.Text) * dA1 * Val(txtQuantity(30).Text)
txtWeights(105).Text = Val(txtWeights(105).Text) + CW0A
'compute Change in WiB = ndBXi     'WiB,new = WiB,current + Change in WiB
CW1B = Val(txtLearnR_n.Text) * dB1 * Val(txtQuantity(0).Text)
txtWeights(30).Text = Val(txtWeights(0).Text) + CW1B
CW2B = Val(txtLearnR_n.Text) * dB1 * Val(txtQuantity(1).Text)
txtWeights(31).Text = Val(txtWeights(1).Text) + CW2B
CW3B = Val(txtLearnR_n.Text) * dB1 * Val(txtQuantity(2).Text)
txtWeights(32).Text = Val(txtWeights(2).Text) + CW3B
CW4B = Val(txtLearnR_n.Text) * dB1 * Val(txtQuantity(3).Text)
txtWeights(33).Text = Val(txtWeights(3).Text) + CW4B
CW5B = Val(txtLearnR_n.Text) * dB1 * Val(txtQuantity(4).Text)
txtWeights(34).Text = Val(txtWeights(4).Text) + CW5B
CW6B = Val(txtLearnR_n.Text) * dB1 * Val(txtQuantity(5).Text)
txtWeights(35).Text = Val(txtWeights(5).Text) + CW6B
CW7B = Val(txtLearnR_n.Text) * dB1 * Val(txtQuantity(6).Text)
txtWeights(36).Text = Val(txtWeights(6).Text) + CW7B
CW8B = Val(txtLearnR_n.Text) * dB1 * Val(txtQuantity(7).Text)
txtWeights(37).Text = Val(txtWeights(7).Text) + CW8B
CW9B = Val(txtLearnR_n.Text) * dB1 * Val(txtQuantity(8).Text)
txtWeights(38).Text = Val(txtWeights(8).Text) + CW9B
CW10B = Val(txtLearnR_n.Text) * dB1 * Val(txtQuantity(9).Text)
txtWeights(39).Text = Val(txtWeights(9).Text) + CW10B
CW11B = Val(txtLearnR_n.Text) * dB1 * Val(txtQuantity(10).Text)
txtWeights(40).Text = Val(txtWeights(10).Text) + CW11B
CW12B = Val(txtLearnR_n.Text) * dB1 * Val(txtQuantity(11).Text)
txtWeights(41).Text = Val(txtWeights(11).Text) + CW12B
CW13B = Val(txtLearnR_n.Text) * dB1 * Val(txtQuantity(12).Text)
txtWeights(42).Text = Val(txtWeights(12).Text) + CW13B
CW14B = Val(txtLearnR_n.Text) * dB1 * Val(txtQuantity(13).Text)
txtWeights(43).Text = Val(txtWeights(13).Text) + CW14B
CW15B = Val(txtLearnR_n.Text) * dB1 * Val(txtQuantity(14).Text)
txtWeights(44).Text = Val(txtWeights(14).Text) + CW15B
CW16B = Val(txtLearnR_n.Text) * dB1 * Val(txtQuantity(15).Text)
txtWeights(45).Text = Val(txtWeights(15).Text) + CW16B
CW17B = Val(txtLearnR_n.Text) * dB1 * Val(txtQuantity(16).Text)
txtWeights(46).Text = Val(txtWeights(16).Text) + CW17B
CW18B = Val(txtLearnR_n.Text) * dB1 * Val(txtQuantity(17).Text)
txtWeights(47).Text = Val(txtWeights(17).Text) + CW18B
CW19B = Val(txtLearnR_n.Text) * dB1 * Val(txtQuantity(18).Text)
txtWeights(48).Text = Val(txtWeights(18).Text) + CW19B
CW20B = Val(txtLearnR_n.Text) * dB1 * Val(txtQuantity(19).Text)
txtWeights(49).Text = Val(txtWeights(19).Text) + CW20B
CW21B = Val(txtLearnR_n.Text) * dB1 * Val(txtQuantity(20).Text)
txtWeights(50).Text = Val(txtWeights(20).Text) + CW21B
CW22B = Val(txtLearnR_n.Text) * dB1 * Val(txtQuantity(21).Text)
txtWeights(51).Text = Val(txtWeights(21).Text) + CW21B
CW23B = Val(txtLearnR_n.Text) * dB1 * Val(txtQuantity(22).Text)
txtWeights(52).Text = Val(txtWeights(22).Text) + CW23B
CW24B = Val(txtLearnR_n.Text) * dB1 * Val(txtQuantity(23).Text)
txtWeights(53).Text = Val(txtWeights(23).Text) + CW24B
CW25B = Val(txtLearnR_n.Text) * dB1 * Val(txtQuantity(24).Text)
```

```
    txtWeights(54).Text = Val(txtWeights(24).Text) + CW25B
    CW26B = Val(txtLearnR_n.Text) * dB1 * Val(txtQuantity(25).Text)
    txtWeights(55).Text = Val(txtWeights(25).Text) + CW22B
    CW27B = Val(txtLearnR_n.Text) * dB1 * Val(txtQuantity(26).Text)
    txtWeights(56).Text = Val(txtWeights(26).Text) + CW27B
    CW28B = Val(txtLearnR_n.Text) * dB1 * Val(txtQuantity(27).Text)
    txtWeights(57).Text = Val(txtWeights(27).Text) + CW28B
    CW29B = Val(txtLearnR_n.Text) * dB1 * Val(txtQuantity(28).Text)
    txtWeights(58).Text = Val(txtWeights(28).Text) + CW29B
    CW30B = Val(txtLearnR_n.Text) * dB1 * Val(txtQuantity(29).Text)
    txtWeights(59).Text = Val(txtWeights(29).Text) + CW30B
    CW0B = Val(txtLearnR_n.Text) * dB1 * Val(txtQuantity(30).Text)
    txtWeights(106).Text = Val(txtWeights(106).Text) + CW0B
'NEXT (i) FORWARD SWEEP 'node1
For i = 0 To 29
 txtEpochWeight(i).Text = Val(txtWeights(i).Text) * Val(txtQuantity(i).Text)
Next i
txtEpochWeight(92).Text = Val(txtWeights(105).Text) * Val(txtQuantity(30).Text)
txtNode(0).Text = Val(txtEpochWeight(92).Text) + Val(txtEpochWeight(0).Text) +
Val(txtEpochWeight(1).Text)          +          Val(txtEpochWeight(2).Text)          +
Val(txtEpochWeight(3).Text)          +          Val(txtEpochWeight(4).Text)          +
Val(txtEpochWeight(5).Text)          +          Val(txtEpochWeight(6).Text)          +
Val(txtEpochWeight(7).Text)          +          Val(txtEpochWeight(8).Text)          +
Val(txtEpochWeight(9).Text)          +          Val(txtEpochWeight(10).Text)         +
Val(txtEpochWeight(11).Text)         +          Val(txtEpochWeight(12).Text)         +
Val(txtEpochWeight(13).Text)         +          Val(txtEpochWeight(14).Text)         +
Val(txtEpochWeight(15).Text)         +          Val(txtEpochWeight(16).Text)         +
Val(txtEpochWeight(17).Text)         +          Val(txtEpochWeight(18).Text)         +
Val(txtEpochWeight(19).Text)         +          Val(txtEpochWeight(20).Text)         +
Val(txtEpochWeight(21).Text)         +          Val(txtEpochWeight(22).Text)         +
Val(txtEpochWeight(23).Text)         +          Val(txtEpochWeight(24).Text)         +
Val(txtEpochWeight(25).Text)         +          Val(txtEpochWeight(26).Text)         +
Val(txtEpochWeight(27).Text)         +          Val(txtEpochWeight(28).Text)         +
Val(txtEpochWeight(29).Text)
txtNode(0).Text = 1 / (1 + Exp(-(txtNode(0).Text)))'node2
txtEpochWeight(30).Text = Val(txtWeights(30).Text) * Val(txtQuantity(0).Text)
txtEpochWeight(31).Text = Val(txtWeights(31).Text) * Val(txtQuantity(1).Text)
txtEpochWeight(32).Text = Val(txtWeights(32).Text) * Val(txtQuantity(2).Text)
txtEpochWeight(33).Text = Val(txtWeights(33).Text) * Val(txtQuantity(3).Text)
txtEpochWeight(34).Text = Val(txtWeights(34).Text) * Val(txtQuantity(4).Text)
txtEpochWeight(35).Text = Val(txtWeights(35).Text) * Val(txtQuantity(5).Text)
txtEpochWeight(36).Text = Val(txtWeights(36).Text) * Val(txtQuantity(6).Text)
txtEpochWeight(37).Text = Val(txtWeights(37).Text) * Val(txtQuantity(7).Text)
txtEpochWeight(38).Text = Val(txtWeights(38).Text) * Val(txtQuantity(8).Text)
txtEpochWeight(39).Text = Val(txtWeights(39).Text) * Val(txtQuantity(9).Text)
txtEpochWeight(40).Text = Val(txtWeights(40).Text) * Val(txtQuantity(10).Text)
txtEpochWeight(41).Text = Val(txtWeights(41).Text) * Val(txtQuantity(11).Text)
txtEpochWeight(42).Text = Val(txtWeights(42).Text) * Val(txtQuantity(12).Text)
txtEpochWeight(43).Text = Val(txtWeights(43).Text) * Val(txtQuantity(13).Text)
txtEpochWeight(44).Text = Val(txtWeights(44).Text) * Val(txtQuantity(14).Text)
txtEpochWeight(45).Text = Val(txtWeights(45).Text) * Val(txtQuantity(15).Text)
txtEpochWeight(46).Text = Val(txtWeights(46).Text) * Val(txtQuantity(16).Text)
```

```
txtEpochWeight(47).Text = Val(txtWeights(47).Text) * Val(txtQuantity(17).Text)
txtEpochWeight(48).Text = Val(txtWeights(48).Text) * Val(txtQuantity(18).Text)
txtEpochWeight(49).Text = Val(txtWeights(49).Text) * Val(txtQuantity(19).Text)
txtEpochWeight(50).Text = Val(txtWeights(50).Text) * Val(txtQuantity(20).Text)
txtEpochWeight(51).Text = Val(txtWeights(51).Text) * Val(txtQuantity(21).Text)
txtEpochWeight(52).Text = Val(txtWeights(52).Text) * Val(txtQuantity(22).Text)
txtEpochWeight(53).Text = Val(txtWeights(53).Text) * Val(txtQuantity(23).Text)
txtEpochWeight(54).Text = Val(txtWeights(54).Text) * Val(txtQuantity(24).Text)
txtEpochWeight(55).Text = Val(txtWeights(55).Text) * Val(txtQuantity(25).Text)
txtEpochWeight(56).Text = Val(txtWeights(56).Text) * Val(txtQuantity(26).Text)
txtEpochWeight(57).Text = Val(txtWeights(57).Text) * Val(txtQuantity(27).Text)
txtEpochWeight(58).Text = Val(txtWeights(58).Text) * Val(txtQuantity(28).Text)
txtEpochWeight(59).Text = Val(txtWeights(59).Text) * Val(txtQuantity(29).Text)
'compute sigmoid for node2
txtEpochWeight(91).Text = Val(txtWeights(106).Text) * Val(txtQuantity(30).Text)
txtNode(1).Text = Val(txtEpochWeight(91).Text) + Val(txtEpochWeight(30).Text) +
Val(txtEpochWeight(31).Text) + Val(txtEpochWeight(32).Text) +
Val(txtEpochWeight(33).Text) + Val(txtEpochWeight(34).Text) +
Val(txtEpochWeight(35).Text) + Val(txtEpochWeight(36).Text) +
Val(txtEpochWeight(37).Text) + Val(txtEpochWeight(38).Text) +
Val(txtEpochWeight(39).Text) + Val(txtEpochWeight(40).Text) +
Val(txtEpochWeight(41).Text) + Val(txtEpochWeight(42).Text) +
Val(txtEpochWeight(43).Text) + Val(txtEpochWeight(44).Text) +
Val(txtEpochWeight(45).Text) + Val(txtEpochWeight(46).Text) +
Val(txtEpochWeight(47).Text) + Val(txtEpochWeight(48).Text) +
Val(txtEpochWeight(49).Text) + Val(txtEpochWeight(50).Text) +
Val(txtEpochWeight(51).Text) + Val(txtEpochWeight(52).Text) +
Val(txtEpochWeight(53).Text) + Val(txtEpochWeight(54).Text) +
Val(txtEpochWeight(55).Text) + Val(txtEpochWeight(56).Text) +
Val(txtEpochWeight(57).Text) + Val(txtEpochWeight(58).Text) +
Val(txtEpochWeight(59).Text)
txtNode(1).Text = 1 / (1 + Exp(-(txtNode(1).Text)))
'wieght adjustments for hidden layers (first forward sweep)
txtOutput1.Text = Val(txtWeights(112).Text) + Val(txtNode(0).Text) *
Val(txtWeights(90).Text) + Val(txtNode(1).Text) * Val(txtWeights(91).Text)
txtOutput2.Text = Val(txtWeights(111).Text) + Val(txtNode(0).Text) *
Val(txtWeights(93).Text) + Val(txtNode(1).Text) * Val(txtWeights(94).Text)
txtOutput3.Text = Val(txtWeights(110).Text) + Val(txtNode(0).Text) *
Val(txtWeights(96).Text) + Val(txtNode(1).Text) * Val(txtWeights(97).Text)
txtOutput4.Text = Val(txtWeights(109).Text) + Val(txtNode(0).Text) *
Val(txtWeights(99).Text) + Val(txtNode(1).Text) * Val(txtWeights(100).Text)
txtOutput5.Text = Val(txtWeights(108).Text) + Val(txtNode(0).Text) *
Val(txtWeights(102).Text) + Val(txtNode(1).Text) * Val(txtWeights(103).Text)
txtOutput1.Text = 1 / (1 + Exp(-(txtOutput1.Text)))
txtOutput2.Text = 1 / (1 + Exp(-(txtOutput2.Text)))
txtOutput3.Text = 1 / (1 + Exp(-(txtOutput3.Text)))
txtOutput4.Text = 1 / (1 + Exp(-(txtOutput4.Text)))
txtOutput5.Text = 1 / (1 + Exp(-(txtOutput5.Text)))
Call counter
End Sub
Private Sub WeightList()
Dim i As Integer
```

```
For i = 0 To 59
List3.AddItem "Weights " & i
Next i
Call InputList
End Sub
Private Sub InputList()
Dim i As Integer'add output from GUI To NN inputs
For i = 0 To 29
List4.AddItem txtQuantity(i).Text
Next i
End Sub
Private Sub InputL()
Dim i As Integer'add output from GUI To NN inputs
For i = 0 To 29
List5.AddItem txtVal_Inp(i).Text
Next i
End Sub
Private Sub ValW()
Dim i As Integer                'add output from GUI To NN inputs
For i = 0 To 59
txtVal_Weights(i).Text = txtWeights(i).Text
Next i
On Error Resume Next
For i = 90 To 112
txtVal_Weights(i).Text = txtWeights(i).Text
Next i
End Sub
Private Sub ValIterate()
Dim i As Integer'node1
For i = 0 To 29
 txtVal_Weights(i).Text = Val(txtVal_Weights(i).Text) * Val(txtVal_Inp(i).Text)
Next i
txtVal_Weights(90).Text = Val(txtVal_Weights(90).Text) * Val(txtVal_Inp(30).Text)
txtNodal(0).Text = Val(txtVal_Weights(92).Text) + Val(txtVal_Weights(0).Text) +
Val(txtVal_Weights(1).Text) + Val(txtVal_Weights(2).Text) + Val(txtVal_Weights(3).Text)
+ Val(txtVal_Weights(4).Text) + Val(txtVal_Weights(5).Text) +
Val(txtVal_Weights(6).Text) + Val(txtVal_Weights(7).Text) + Val(txtVal_Weights(8).Text)
+ Val(txtVal_Weights(9).Text) + Val(txtVal_Weights(10).Text) +
Val(txtVal_Weights(11).Text) + Val(txtVal_Weights(12).Text) +
Val(txtVal_Weights(13).Text) + Val(txtVal_Weights(14).Text) +
Val(txtVal_Weights(15).Text) + Val(txtVal_Weights(16).Text) +
Val(txtVal_Weights(17).Text) + Val(txtVal_Weights(18).Text) +
Val(txtVal_Weights(19).Text) + Val(txtVal_Weights(20).Text) +
Val(txtVal_Weights(21).Text) + Val(txtVal_Weights(22).Text) +
Val(txtVal_Weights(23).Text) + Val(txtVal_Weights(24).Text) +
Val(txtVal_Weights(25).Text) + Val(txtVal_Weights(26).Text) +
Val(txtVal_Weights(27).Text) + Val(txtVal_Weights(28).Text) +
Val(txtVal_Weights(29).Text)
txtNodal(0).Text = 1 / (1 + Exp(-(txtNodal(0).Text)))'node2
txtVal_Weights(30).Text = Val(txtVal_Weights(30).Text) * Val(txtVal_Inp(0).Text)
txtVal_Weights(31).Text = Val(txtVal_Weights(31).Text) * Val(txtVal_Inp(1).Text)
txtVal_Weights(32).Text = Val(txtVal_Weights(32).Text) * Val(txtVal_Inp(2).Text)
```

```
txtVal_Weights(33).Text = Val(txtVal_Weights(33).Text) * Val(txtVal_Inp(3).Text)
txtVal_Weights(34).Text = Val(txtVal_Weights(34).Text) * Val(txtVal_Inp(4).Text)
txtVal_Weights(35).Text = Val(txtVal_Weights(35).Text) * Val(txtVal_Inp(5).Text)
txtVal_Weights(36).Text = Val(txtVal_Weights(36).Text) * Val(txtVal_Inp(6).Text)
txtVal_Weights(37).Text = Val(txtVal_Weights(37).Text) * Val(txtVal_Inp(7).Text)
txtVal_Weights(38).Text = Val(txtVal_Weights(38).Text) * Val(txtVal_Inp(8).Text)
txtVal_Weights(39).Text = Val(txtVal_Weights(39).Text) * Val(txtVal_Inp(9).Text)
txtVal_Weights(40).Text = Val(txtVal_Weights(40).Text) * Val(txtVal_Inp(10).Text)
txtVal_Weights(41).Text = Val(txtVal_Weights(41).Text) * Val(txtVal_Inp(11).Text)
txtVal_Weights(42).Text = Val(txtVal_Weights(42).Text) * Val(txtVal_Inp(12).Text)
txtVal_Weights(43).Text = Val(txtVal_Weights(43).Text) * Val(txtVal_Inp(13).Text)
txtVal_Weights(44).Text = Val(txtVal_Weights(44).Text) * Val(txtVal_Inp(14).Text)
txtVal_Weights(45).Text = Val(txtVal_Weights(45).Text) * Val(txtVal_Inp(15).Text)
txtVal_Weights(46).Text = Val(txtVal_Weights(46).Text) * Val(txtVal_Inp(16).Text)
txtVal_Weights(47).Text = Val(txtVal_Weights(47).Text) * Val(txtVal_Inp(17).Text)
txtVal_Weights(48).Text = Val(txtVal_Weights(48).Text) * Val(txtVal_Inp(18).Text)
txtVal_Weights(49).Text = Val(txtVal_Weights(49).Text) * Val(txtVal_Inp(19).Text)
txtVal_Weights(50).Text = Val(txtVal_Weights(50).Text) * Val(txtVal_Inp(20).Text)
txtVal_Weights(51).Text = Val(txtVal_Weights(51).Text) * Val(txtVal_Inp(21).Text)
txtVal_Weights(52).Text = Val(txtVal_Weights(52).Text) * Val(txtVal_Inp(22).Text)
txtVal_Weights(53).Text = Val(txtVal_Weights(53).Text) * Val(txtVal_Inp(23).Text)
txtVal_Weights(54).Text = Val(txtVal_Weights(54).Text) * Val(txtVal_Inp(24).Text)
txtVal_Weights(55).Text = Val(txtVal_Weights(55).Text) * Val(txtVal_Inp(25).Text)
txtVal_Weights(56).Text = Val(txtVal_Weights(56).Text) * Val(txtVal_Inp(26).Text)
txtVal_Weights(57).Text = Val(txtVal_Weights(57).Text) * Val(txtVal_Inp(27).Text)
txtVal_Weights(58).Text = Val(txtVal_Weights(58).Text) * Val(txtVal_Inp(28).Text)
txtVal_Weights(59).Text = Val(txtVal_Weights(59).Text) * Val(txtVal_Inp(29).Text)
'compute sigmoid for node2
txtVal_Weights(91).Text = Val(txtVal_Weights(91).Text) * Val(txtVal_Inp(30).Text)
txtNodal(1).Text = Val(txtVal_Weights(91).Text) + Val(txtVal_Weights(30).Text) +
Val(txtVal_Weights(31).Text)          +          Val(txtVal_Weights(32).Text)          +
Val(txtVal_Weights(33).Text)          +          Val(txtVal_Weights(34).Text)          +
Val(txtVal_Weights(35).Text)          +          Val(txtVal_Weights(36).Text)          +
Val(txtVal_Weights(37).Text)          +          Val(txtVal_Weights(38).Text)          +
Val(txtVal_Weights(39).Text)          +          Val(txtVal_Weights(40).Text)          +
Val(txtVal_Weights(41).Text)          +          Val(txtVal_Weights(42).Text)          +
Val(txtVal_Weights(43).Text)          +          Val(txtVal_Weights(44).Text)          +
Val(txtVal_Weights(45).Text)          +          Val(txtVal_Weights(46).Text)          +
Val(txtVal_Weights(47).Text)          +          Val(txtVal_Weights(48).Text)          +
Val(txtVal_Weights(49).Text)          +          Val(txtVal_Weights(50).Text)          +
Val(txtVal_Weights(51).Text)          +          Val(txtVal_Weights(52).Text)          +
Val(txtVal_Weights(53).Text)          +          Val(txtVal_Weights(54).Text)          +
Val(txtVal_Weights(55).Text)          +          Val(txtVal_Weights(56).Text)          +
Val(txtVal_Weights(57).Text)          +          Val(txtVal_Weights(58).Text)          +
Val(txtVal_Weights(59).Text)
txtNodal(1).Text = 1 / (1 + Exp(-(txtNodal(1).Text)))        'wieght adjustments for hL
txtOut1.Text      =      Val(txtVal_Weights(112).Text)      +      Val(txtNodal(0).Text)      *
Val(txtVal_Weights(90).Text) + Val(txtNodal(1).Text) * Val(txtVal_Weights(91).Text)
txtOut2.Text      =      Val(txtVal_Weights(111).Text)      +      Val(txtNodal(0).Text)      *
Val(txtVal_Weights(93).Text) + Val(txtNodal(1).Text) * Val(txtVal_Weights(94).Text)
txtOut3.Text      =      Val(txtVal_Weights(110).Text)      +      Val(txtNodal(0).Text)      *
Val(txtVal_Weights(96).Text) + Val(txtNodal(1).Text) * Val(txtVal_Weights(97).Text)
```

```
txtOut4.Text = Val(txtVal_Weights(109).Text) + Val(txtNodal(0).Text) *
Val(txtVal_Weights(99).Text) + Val(txtNodal(1).Text) * Val(txtVal_Weights(100).Text)
txtOut5.Text = Val(txtVal_Weights(108).Text) + Val(txtNodal(0).Text) *
Val(txtVal_Weights(102).Text) + Val(txtNodal(1).Text) * Val(txtVal_Weights(103).Text)
txtOut1.Text = 1 / (1 + Exp(-(txtOut1.Text)))
txtOut2.Text = 1 / (1 + Exp(-(txtOut2.Text)))
txtOut3.Text = 1 / (1 + Exp(-(txtOut3.Text)))
txtOut4.Text = 1 / (1 + Exp(-(txtOut4.Text)))
txtOut5.Text = 1 / (1 + Exp(-(txtOut5.Text)))
Call WeightList
End Sub
Code for implementation of the 1Hidden Node ANN
Public Sub Rado()
Dim Listside As Integer, Dim difcalc As Integer
difcalc = txtMaxV.Text
Dim i As Integer'update weights for hidden units
For i = 90 To 104
Do
      MyNum = Int(Rnd * Val(txtMaxV.Text)) * 0.04 + 0.14
      txtWeights(i).Text = (MyNum)
      Exit Do
   Loop
Next'update weights for inputs
For i = 0 To 29
Do
      MyNum = Int(Rnd * Val(txtMaxV.Text)) * 0.04 + 0.14
      txtWeights(i).Text = (MyNum)
      Exit Do
   Loop
Next
Exit Sub
End Sub
Private Sub Iterate()
Dim i As Integer                'wieght adjustments for input layers (first forward sweep)
If lblOutput.Caption = "0" Then                'plot initial weight
Call PlotWeight'node1
For i = 0 To 29
 txtEpochWeight(i).Text = Val(txtWeights(i).Text) * Val(txtQuantity(i).Text)
Next i
txtEpochWeight(92).Text = Val(txtWeights(105).Text) * Val(txtQuantity(30).Text)
txtNode(0).Text = Val(txtEpochWeight(92).Text) + Val(txtEpochWeight(0).Text) +
Val(txtEpochWeight(1).Text)          +          Val(txtEpochWeight(2).Text)          +
Val(txtEpochWeight(3).Text)          +          Val(txtEpochWeight(4).Text)          +
Val(txtEpochWeight(5).Text)          +          Val(txtEpochWeight(6).Text)          +
Val(txtEpochWeight(7).Text)          +          Val(txtEpochWeight(8).Text)          +
Val(txtEpochWeight(9).Text)          +          Val(txtEpochWeight(10).Text)         +
Val(txtEpochWeight(11).Text)         +          Val(txtEpochWeight(12).Text)         +
Val(txtEpochWeight(13).Text)         +          Val(txtEpochWeight(14).Text)         +
Val(txtEpochWeight(15).Text)         +          Val(txtEpochWeight(16).Text)         +
Val(txtEpochWeight(17).Text)         +          Val(txtEpochWeight(18).Text)         +
Val(txtEpochWeight(19).Text)         +          Val(txtEpochWeight(20).Text)         +
Val(txtEpochWeight(21).Text)         +          Val(txtEpochWeight(22).Text)         +
```

```
Val(txtEpochWeight(23).Text) + Val(txtEpochWeight(24).Text) +
Val(txtEpochWeight(25).Text) + Val(txtEpochWeight(26).Text) +
Val(txtEpochWeight(27).Text) + Val(txtEpochWeight(28).Text) +
Val(txtEpochWeight(29).Text)
txtNode(0).Text = 1 / (1 + Exp(-(txtNode(0).Text)))
'wieght adjustments for hidden layers (first forward sweep)
txtOutput1.Text = Val(txtWeights(112).Text) + Val(txtNode(0).Text) *
Val(txtWeights(90).Text)
txtOutput2.Text = Val(txtWeights(111).Text) + Val(txtNode(0).Text) *
Val(txtWeights(93).Text)
txtOutput3.Text = Val(txtWeights(110).Text) + Val(txtNode(0).Text) *
Val(txtWeights(96).Text)
txtOutput4.Text = Val(txtWeights(109).Text) + Val(txtNode(0).Text) *
Val(txtWeights(99).Text)
txtOutput5.Text = Val(txtWeights(108).Text) + Val(txtNode(0).Text) *
Val(txtWeights(102).Text)
txtOutput1.Text = 1 / (1 + Exp(-(txtOutput1.Text)))
txtOutput2.Text = 1 / (1 + Exp(-(txtOutput2.Text)))
txtOutput3.Text = 1 / (1 + Exp(-(txtOutput3.Text)))
txtOutput4.Text = 1 / (1 + Exp(-(txtOutput4.Text)))
txtOutput5.Text = 1 / (1 + Exp(-(txtOutput5.Text)))
Call counter
Else
End If
Call WeightList
Exit Sub
End Sub
Private Sub IterateTwo()
Dim i As Integer 'Woz,new = Woz,current + Change in Woz
dz1 = (Val(txtLearnR_n.Text) * Val(txtOutput1.Text) * (1 - Val(txtOutput1.Text)) *
(Val(txtActual(0).Text) - Val(txtOutput1.Text)) * Val(txtQuantity(30).Text))
dz2 = (Val(txtLearnR_n.Text) * Val(txtOutput2.Text) * (1 - Val(txtOutput2.Text)) *
(Val(txtActual(1).Text) - Val(txtOutput2.Text)) * Val(txtQuantity(30).Text))
dz3 = (Val(txtLearnR_n.Text) * Val(txtOutput3.Text) * (1 - Val(txtOutput3.Text)) *
(Val(txtActual(2).Text) - Val(txtOutput3.Text)) * Val(txtQuantity(30).Text))
dz4 = (Val(txtLearnR_n.Text) * Val(txtOutput4.Text) * (1 - Val(txtOutput4.Text)) *
(Val(txtActual(3).Text) - Val(txtOutput4.Text)) * Val(txtQuantity(30).Text))
dz5 = (Val(txtLearnR_n.Text) * Val(txtOutput5.Text) * (1 - Val(txtOutput5.Text)) *
(Val(txtActual(4).Text) - Val(txtOutput5.Text)) * Val(txtQuantity(30).Text))
'using dz(i) combo size
txtWeights(112).Text = Val(txtWeights(112).Text) + dz1
txtWeights(111).Text = Val(txtWeights(111).Text) + dz2
txtWeights(110).Text = Val(txtWeights(110).Text) + dz3
txtWeights(109).Text = Val(txtWeights(109).Text) + dz4
txtWeights(108).Text = Val(txtWeights(108).Text) + dz5
'wieght adjustments for hidden layers (Next (i) forward sweep)
'error responsibility dA 'dA = outputNA(1 - outputNA)E(downstream)WjkDj
dA1 = Val(txtNode(0).Text) * (1 - Val(txtNode(0).Text)) * Val(txtWeights(90).Text) * dz1
'then compute Change in WAz = ndz(1).OutputA
CAz1 = Val(txtLearnR_n.Text) * dz1 * Val(txtNode(0).Text)
'WAz,new = WAz,current + Change in WAz
txtWeights(90).Text = Val(txtWeights(90).Text) + CAz1 'next hidden layer2
```

```
dA2 = Val(txtNode(0).Text) * (1 - Val(txtNode(0).Text)) * Val(txtWeights(93).Text) * dz2
'then compute Change in WAz = ndz(1).OutputA
   CAz2 = Val(txtLearnR_n.Text) * dz2 * Val(txtNode(0).Text)
   'WAz,new = WAz,current + Change in WAz
   txtWeights(93).Text = Val(txtWeights(93).Text) + CAz2'next hidden layer3
dA3 = Val(txtNode(0).Text) * (1 - Val(txtNode(0).Text)) * Val(txtWeights(96).Text) *
dz3'then compute Change in WAz = ndz(1).OutputA
   CAz3 = Val(txtLearnR_n.Text) * dz3 * Val(txtNode(0).Text)
   'WAz,new = WAz,current + Change in WAz
   txtWeights(96).Text = Val(txtWeights(96).Text) + CAz3'next hidden layer4
dA4 = Val(txtNode(0).Text) * (1 - Val(txtNode(0).Text)) * Val(txtWeights(99).Text) * dz4
                'then compute Change in WAz = ndz(1).OutputA
   CAz4 = Val(txtLearnR_n.Text) * dz4 * Val(txtNode(0).Text)
   'WAz,new = WAz,current + Change in WAz
   txtWeights(99).Text = Val(txtWeights(99).Text) + CAz4'next hidden layer5
dA5 = Val(txtNode(0).Text) * (1 - Val(txtNode(0).Text)) * Val(txtWeights(102).Text) * dz5
                'then compute Change in WAz = ndz(1).OutputA
   CAz5 = Val(txtLearnR_n.Text) * dz5 * Val(txtNode(0).Text)
   'WAz,new = WAz,current + Change in WAz
   txtWeights(102).Text = Val(txtWeights(102).Text) + CAz5
'wieght adjustments for inputs (Next (i) forward sweep)
      'compute Change in WiA = ndAXi        'WiA,new = WiA,current + Change in WiA
   CW1A = Val(txtLearnR_n.Text) * dA1 * Val(txtQuantity(0).Text)
   txtWeights(0).Text = Val(txtWeights(0).Text) + CW1A
   CW2A = Val(txtLearnR_n.Text) * dA1 * Val(txtQuantity(1).Text)
   txtWeights(1).Text = Val(txtWeights(1).Text) + CW2A
   CW3A = Val(txtLearnR_n.Text) * dA1 * Val(txtQuantity(2).Text)
   txtWeights(2).Text = Val(txtWeights(2).Text) + CW3A
   CW4A = Val(txtLearnR_n.Text) * dA1 * Val(txtQuantity(3).Text)
   txtWeights(3).Text = Val(txtWeights(3).Text) + CW4A
   CW5A = Val(txtLearnR_n.Text) * dA1 * Val(txtQuantity(4).Text)
   txtWeights(4).Text = Val(txtWeights(4).Text) + CW5A
   CW6A = Val(txtLearnR_n.Text) * dA1 * Val(txtQuantity(5).Text)
   txtWeights(5).Text = Val(txtWeights(5).Text) + CW6A
   CW7A = Val(txtLearnR_n.Text) * dA1 * Val(txtQuantity(6).Text)
   txtWeights(6).Text = Val(txtWeights(6).Text) + CW7A
   CW8A = Val(txtLearnR_n.Text) * dA1 * Val(txtQuantity(7).Text)
   txtWeights(7).Text = Val(txtWeights(7).Text) + CW8A
   CW9A = Val(txtLearnR_n.Text) * dA1 * Val(txtQuantity(8).Text)
   txtWeights(8).Text = Val(txtWeights(8).Text) + CW9A
   CW10A = Val(txtLearnR_n.Text) * dA1 * Val(txtQuantity(9).Text)
   txtWeights(9).Text = Val(txtWeights(9).Text) + CW10A
   CW11A = Val(txtLearnR_n.Text) * dA1 * Val(txtQuantity(10).Text)
   txtWeights(10).Text = Val(txtWeights(10).Text) + CW11A
   CW12A = Val(txtLearnR_n.Text) * dA1 * Val(txtQuantity(11).Text)
   txtWeights(11).Text = Val(txtWeights(11).Text) + CW12A
   CW13A = Val(txtLearnR_n.Text) * dA1 * Val(txtQuantity(12).Text)
   txtWeights(12).Text = Val(txtWeights(12).Text) + CW13A
   CW14A = Val(txtLearnR_n.Text) * dA1 * Val(txtQuantity(13).Text)
   txtWeights(13).Text = Val(txtWeights(13).Text) + CW14A
   CW15A = Val(txtLearnR_n.Text) * dA1 * Val(txtQuantity(14).Text)
   txtWeights(14).Text = Val(txtWeights(14).Text) + CW15A
```

```
    CW16A = Val(txtLearnR_n.Text) * dA1 * Val(txtQuantity(15).Text)
    txtWeights(15).Text = Val(txtWeights(15).Text) + CW16A
    CW17A = Val(txtLearnR_n.Text) * dA1 * Val(txtQuantity(16).Text)
    txtWeights(16).Text = Val(txtWeights(16).Text) + CW17A
    CW18A = Val(txtLearnR_n.Text) * dA1 * Val(txtQuantity(17).Text)
    txtWeights(17).Text = Val(txtWeights(17).Text) + CW18A
    CW19A = Val(txtLearnR_n.Text) * dA1 * Val(txtQuantity(18).Text)
    txtWeights(18).Text = Val(txtWeights(18).Text) + CW19A
    CW20A = Val(txtLearnR_n.Text) * dA1 * Val(txtQuantity(19).Text)
    txtWeights(19).Text = Val(txtWeights(19).Text) + CW20A
    CW21A = Val(txtLearnR_n.Text) * dA1 * Val(txtQuantity(20).Text)
    txtWeights(20).Text = Val(txtWeights(20).Text) + CW21A
    CW22A = Val(txtLearnR_n.Text) * dA1 * Val(txtQuantity(21).Text)
    txtWeights(21).Text = Val(txtWeights(21).Text) + CW21A
    CW23A = Val(txtLearnR_n.Text) * dA1 * Val(txtQuantity(22).Text)
    txtWeights(22).Text = Val(txtWeights(22).Text) + CW23A
    CW24A = Val(txtLearnR_n.Text) * dA1 * Val(txtQuantity(23).Text)
    txtWeights(23).Text = Val(txtWeights(23).Text) + CW24A
    CW25A = Val(txtLearnR_n.Text) * dA1 * Val(txtQuantity(24).Text)
    txtWeights(24).Text = Val(txtWeights(24).Text) + CW25A
    CW26A = Val(txtLearnR_n.Text) * dA1 * Val(txtQuantity(25).Text)
    txtWeights(25).Text = Val(txtWeights(25).Text) + CW22A
    CW27A = Val(txtLearnR_n.Text) * dA1 * Val(txtQuantity(26).Text)
    txtWeights(26).Text = Val(txtWeights(26).Text) + CW27A
    CW28A = Val(txtLearnR_n.Text) * dA1 * Val(txtQuantity(27).Text)
    txtWeights(27).Text = Val(txtWeights(27).Text) + CW28A
    CW29A = Val(txtLearnR_n.Text) * dA1 * Val(txtQuantity(28).Text)
    txtWeights(28).Text = Val(txtWeights(28).Text) + CW29A
    CW30A = Val(txtLearnR_n.Text) * dA1 * Val(txtQuantity(29).Text)
    txtWeights(29).Text = Val(txtWeights(29).Text) + CW30A
    CW0A = Val(txtLearnR_n.Text) * dA1 * Val(txtQuantity(30).Text)
    txtWeights(105).Text = Val(txtWeights(105).Text) + CW0A
     'NEXT (i) FORWARD SWEEP 'node1
For i = 0 To 29
 txtEpochWeight(i).Text = Val(txtWeights(i).Text) * Val(txtQuantity(i).Text)
Next i
txtEpochWeight(92).Text = Val(txtWeights(105).Text) * Val(txtQuantity(30).Text)
txtNode(0).Text = Val(txtEpochWeight(92).Text) + Val(txtEpochWeight(0).Text) +
Val(txtEpochWeight(1).Text) + Val(txtEpochWeight(2).Text) +
Val(txtEpochWeight(3).Text) + Val(txtEpochWeight(4).Text) +
Val(txtEpochWeight(5).Text) + Val(txtEpochWeight(6).Text) +
Val(txtEpochWeight(7).Text) + Val(txtEpochWeight(8).Text) +
Val(txtEpochWeight(9).Text) + Val(txtEpochWeight(10).Text) +
Val(txtEpochWeight(11).Text) + Val(txtEpochWeight(12).Text) +
Val(txtEpochWeight(13).Text) + Val(txtEpochWeight(14).Text) +
Val(txtEpochWeight(15).Text) + Val(txtEpochWeight(16).Text) +
Val(txtEpochWeight(17).Text) + Val(txtEpochWeight(18).Text) +
Val(txtEpochWeight(19).Text) + Val(txtEpochWeight(20).Text) +
Val(txtEpochWeight(21).Text) + Val(txtEpochWeight(22).Text) +
Val(txtEpochWeight(23).Text) + Val(txtEpochWeight(24).Text) +
Val(txtEpochWeight(25).Text) + Val(txtEpochWeight(26).Text) +
```

```
Val(txtEpochWeight(27).Text) + Val(txtEpochWeight(28).Text) +
Val(txtEpochWeight(29).Text)
txtNode(0).Text = 1 / (1 + Exp(-(txtNode(0).Text)))
'wieght adjustments for hidden layers (first forward sweep)
txtOutput1.Text = Val(txtWeights(112).Text) + Val(txtNode(0).Text) *
Val(txtWeights(90).Text)
txtOutput2.Text = Val(txtWeights(111).Text) + Val(txtNode(0).Text) *
Val(txtWeights(93).Text)
txtOutput3.Text = Val(txtWeights(110).Text) + Val(txtNode(0).Text) *
Val(txtWeights(96).Text)
txtOutput4.Text = Val(txtWeights(109).Text) + Val(txtNode(0).Text) *
Val(txtWeights(99).Text)
txtOutput5.Text = Val(txtWeights(108).Text) + Val(txtNode(0).Text) *
Val(txtWeights(102).Text)
txtOutput1.Text = 1 / (1 + Exp(-(txtOutput1.Text)))
txtOutput2.Text = 1 / (1 + Exp(-(txtOutput2.Text)))
txtOutput3.Text = 1 / (1 + Exp(-(txtOutput3.Text)))
txtOutput4.Text = 1 / (1 + Exp(-(txtOutput4.Text)))
txtOutput5.Text = 1 / (1 + Exp(-(txtOutput5.Text)))
Call counter
End Sub
Private Sub WeightList()
Dim i As Integer
For i = 0 To 29
List3.AddItem "Weights " & i
Next i
Call InputList
End Sub
End Sub
Private Sub InputList()
Dim i As Integer'add output from GUI To NN inputs
For i = 0 To 29
List4.AddItem txtQuantity(i).Text
Next i
End Sub
Private Sub InputL()
Dim i As Integer'add output from GUI To NN inputs
For i = 0 To 29
List5.AddItem txtVal_Inp(i).Text
Next i
End Sub
Private Sub ValW()
Dim i As Integer'add output from GUI To NN inputs
For i = 0 To 29
txtVal_Weights(i).Text = txtWeights(i).Text
Next i
On Error Resume Next
For i = 90 To 112
txtVal_Weights(i).Text = txtWeights(i).Text
Next i
End Sub
Private Sub ValIterate()
```

```
Dim i As Integer        'node1 wieght adjustments for input layers (first forward sweep)
For i = 0 To 29
 txtVal_Weights(i).Text = Val(txtVal_Weights(i).Text) * Val(txtVal_Inp(i).Text)
Next i
txtVal_Weights(90).Text = Val(txtVal_Weights(90).Text) * Val(txtVal_Inp(30).Text)
txtNodal(0).Text = Val(txtVal_Weights(92).Text) + Val(txtVal_Weights(0).Text) +
Val(txtVal_Weights(1).Text) + Val(txtVal_Weights(2).Text) + Val(txtVal_Weights(3).Text)
+ Val(txtVal_Weights(4).Text) + Val(txtVal_Weights(5).Text) +
Val(txtVal_Weights(6).Text) + Val(txtVal_Weights(7).Text) + Val(txtVal_Weights(8).Text)
+ Val(txtVal_Weights(9).Text) + Val(txtVal_Weights(10).Text) +
Val(txtVal_Weights(11).Text) + Val(txtVal_Weights(12).Text) +
Val(txtVal_Weights(13).Text) + Val(txtVal_Weights(14).Text) +
Val(txtVal_Weights(15).Text) + Val(txtVal_Weights(16).Text) +
Val(txtVal_Weights(17).Text) + Val(txtVal_Weights(18).Text) +
Val(txtVal_Weights(19).Text) + Val(txtVal_Weights(20).Text) +
Val(txtVal_Weights(21).Text) + Val(txtVal_Weights(22).Text) +
Val(txtVal_Weights(23).Text) + Val(txtVal_Weights(24).Text) +
Val(txtVal_Weights(25).Text) + Val(txtVal_Weights(26).Text) +
Val(txtVal_Weights(27).Text) + Val(txtVal_Weights(28).Text) +
Val(txtVal_Weights(29).Text)
txtNodal(0).Text = 1 / (1 + Exp(-(txtNodal(0).Text)))
'wieght adjustments for hidden layers (first forward sweep)
txtOut1.Text = Val(txtVal_Weights(112).Text) + Val(txtNodal(0).Text) *
Val(txtVal_Weights(90).Text)
txtOut2.Text = Val(txtVal_Weights(111).Text) + Val(txtNodal(0).Text) *
Val(txtVal_Weights(93).Text)
txtOut3.Text = Val(txtVal_Weights(110).Text) + Val(txtNodal(0).Text) *
Val(txtVal_Weights(96).Text)
txtOut4.Text = Val(txtVal_Weights(109).Text) + Val(txtNodal(0).Text) *
Val(txtVal_Weights(99).Text)
txtOut5.Text = Val(txtVal_Weights(108).Text) + Val(txtNodal(0).Text) *
Val(txtVal_Weights(102).Text)
txtOut1.Text = 1 / (1 + Exp(-(txtOut1.Text)))
txtOut2.Text = 1 / (1 + Exp(-(txtOut2.Text)))
txtOut3.Text = 1 / (1 + Exp(-(txtOut3.Text)))
txtOut4.Text = 1 / (1 + Exp(-(txtOut4.Text)))
txtOut5.Text = 1 / (1 + Exp(-(txtOut5.Text)))
Call WeightList
End Sub


‘kNN code on frmkNN Continues here
Private Sub Picture1_DblClick()
  If Points.count Then Points.Remove Points.count 'remove the last point
  ReDrawPolyPoints Picture1, Points
  DrawPolyMode = False
  Set Points = Nothing
End Sub
Private Sub Picture1_MouseMove(Button As Integer, Shift As Integer, X As Single, Y As
Single)
  If DrawPolyMode Then
    ReDrawPolyPoints Picture1, Points
```

```
    If Points.count Then Picture1.Line -(X, Y), vbRed
  End If
  'Caption = "X: " & X & ", " & "Y: " & Y 'Display Mouse Pointer Position:
End Sub
Private Sub ReDrawPolyPoints(Pic As PictureBox, Pts As Collection)
Dim i As Long
  Pic.Cls
  For i = 1 To Points.count
    'If i > 1 Then Pic.Line -(Pts(i)(0), Pts(i)(1)), vbRed
    Pic.Circle (Pts(i)(0), Pts(i)(1)), 3, vbBlack
  Next
End Sub
 'Set the first point and start the line
Dim X As Long
Dim Y As Long
Dim i As Integer
For i = 0 To List1.ListCount - 1
    X = List1.List(i) / 40
    Y = List2.List(i) / 40
DrawPolyMode = True: Points.Add Array(X, Y): ReDrawPolyPoints Picture1, Points
Next i
frmKNN.Caption = "*** K-Nearest Neighborhood for Ebola Diagnosis.***"
End Sub
    'Set the first point and start the line
Dim X As Long
Dim Y As Long
Dim i As Integer
For i = 0 To List1.ListCount - 1
    X = List1.List(i) / 20
    Y = List2.List(i) / 20
    DrawPolyMode = True: Points.Add Array(X, Y): ReDrawPolyPoints Picture1, Points
Next i
frmKNN.Caption = "*** K-Nearest Neighborhood for Lassa Fever Diagnosis.***"
End Sub
Private Sub eLoader()
Dim knum As Integer
knum = FreeFile
Open "C:\SOSIC\Ebola\EBOLA_DIAG_Rec.txt" For Input As #knum
strDisplay = Input(LOF(knum), knum)
txtLagoon.Text = strDisplay
Close #knum
End Sub
‘Remaining code for New symptoms input in Neural Network
Private Sub Option138_Click()
If txtNewSypmtom1.Text = "Input New Symptom1" Or txtNewSypmtom1.Text = "" Then
msg = "Please input the new Symptom you have noticed!"
title = "SOSIC Check!"
Ans = MsgBox(msg, vbCritical + vbOKOnly, title)
txtNewSypmtom1.SetFocus
Exit Sub
Else
lblCost(27).Caption = ".6"
```

```
txtQuantity(27).Text = "0.6"
End If
End Sub
Private Sub Option139_Click()
If txtNewSypmtom2.Text = "Input New Symptom2" Or txtNewSypmtom2.Text = "" Then
msg = "Please input the new Symptom you have noticed!"
title = "SOSIC Check!"
Ans = MsgBox(msg, vbCritical + vbOKOnly, title)
txtNewSypmtom2.SetFocus
Exit Sub
Else
lblCost(28).Caption = ".6"
txtQuantity(28).Text = "0.6"
End If
End Sub
Private Sub Option14_Click()
If chkItem(1).Value = 1 Then
lblCost(1).Caption = ".9"
txtQuantity(1).Text = "0.9"
End If
End Sub
Private Sub Option140_Click()
If txtNewSypmtom1.Text = "Input New Symptom1" Or txtNewSypmtom1.Text = "" Then
msg = "Please input the new Symptom you have noticed!"
title = "SOSIC Check!"
Ans = MsgBox(msg, vbCritical + vbOKOnly, title)
txtNewSypmtom1.SetFocus
Exit Sub
Else
lblCost(27).Caption = ".9"
txtQuantity(27).Text = "0.9"
End If
End Sub
Private Sub Option141_Click()
If txtNewSypmtom2.Text = "Input New Symptom2" Or txtNewSypmtom2.Text = "" Then
msg = "Please input the new Symptom you have noticed!"
title = "SOSIC Check!"
Ans = MsgBox(msg, vbCritical + vbOKOnly, title)
txtNewSypmtom2.SetFocus
Exit Sub
Else
lblCost(28).Caption = ".3"
txtQuantity(28).Text = "0.3"
End If
End Sub
Private Sub Option142_Click()
If txtNewSypmtom3.Text = "Input New Symptom3" Or txtNewSypmtom3.Text = "" Then
msg = "Please input the new Symptom you have noticed!"
title = "SOSIC Check!"
Ans = MsgBox(msg, vbCritical + vbOKOnly, title)
txtNewSypmtom3.SetFocus
Exit Sub
```

```
Else
lblCost(29).Caption = ".9"
txtQuantity(29).Text = "0.9"
End If
End Sub
Private Sub Option143_Click()
If txtNewSypmtom3.Text = "Input New Symptom3" Or txtNewSypmtom3.Text = "" Then
msg = "Please input the new Symptom you have noticed!"
title = "SOSIC Check!"
Ans = MsgBox(msg, vbCritical + vbOKOnly, title)
txtNewSypmtom3.SetFocus
Exit Sub
Else
lblCost(29).Caption = ".6"
txtQuantity(29).Text = "0.6"
End If
End Sub
Private Sub Option144_Click()
If txtNewSypmtom3.Text = "Input New Symptom3" Or txtNewSypmtom3.Text = "" Then
msg = "Please input the new Symptom you have noticed!"
title = "SOSIC Check!"
Ans = MsgBox(msg, vbCritical + vbOKOnly, title)
txtNewSypmtom3.SetFocus
Exit Sub
Else
lblCost(29).Caption = ".3"
txtQuantity(29).Text = "0.3"
End If
Public skiret() As orditem
Private Const StdDiscount = 0.1
Public Sub addheaders(grid As MSFlexGrid)
grid.row = 0
grid.Col = 0
grid.CellFontBold = True
grid.Text = "DETECTED SYMPTOMS "
grid.row = 0
grid.Col = 1
grid.CellFontBold = True
grid.Text = "Manual Input"
grid.row = 0
grid.Col = 2
grid.CellFontBold = True
grid.Text = " DEGREE"
grid.row = 0
grid.Col = 3
grid.CellFontBold = True
grid.Text = "Possible Preliminary Diagnosis"
End Sub
Public Sub prepgrid(grid As MSFlexGrid)
grid.FixedCols = 0
grid.FixedRows = 0
grid.Cols = 4
```

```
grid.Rows = 100
grid.Width = 10000
grid.ColWidth(0) = 2700
grid.ColWidth(1) = 1500
grid.ColWidth(2) = 1150
grid.ColWidth(3) = 4000
Call addheaders(grid)
End Sub
Private colOrder As New Collection
Private m_count As Long
Private m_Discount As Boolean
Private m_grid As MSFlexGrid
Private Const StdDiscount = 0.1
Private Sub class_Initialize()
Set m_grid = frmInvoice.invGrid
m_count = 0
m_Discount = False
Call prepgrid
End Sub
Public Property Get count() As Long
m_count = colOrder.count
count = m_count
End Property
Public Sub addorderitem(newone As OrderItem)
colOrder.Add newone
End Sub
Public Property Get Discount() As Boolean
Discount = m_Discount
End Property
Public Property Let Discount(ByVal vNewValue As Boolean)
m_Discount = vNewValue
End Property
Public Sub clearorder()
Dim i As Integer
For i = colOrder.count To 1 Step -1
colOrder.Remove i
Next
Call cleargrid
End Sub
Private Sub cleargrid()
m_grid.Clear
Call addheaders
End Sub
Private Sub addheaders()
m_grid.row = 0
m_grid.Col = 0
m_grid.CellFontBold = True
m_grid.Text = "DETECTED SYMPTOMS"
m_grid.row = 0
m_grid.Col = 1
m_grid.CellFontBold = True
m_grid.Text = "Manual Input"
```

```
m_grid.row = 0
m_grid.Col = 2
m_grid.CellFontBold = True
m_grid.Text = " DEGREE"
m_grid.row = 0
m_grid.Col = 3
m_grid.CellFontBold = True
m_grid.Text = "Possible Preliminary Diagnosis"
End Sub
Public Sub prepgrid()
m_grid.FixedCols = 0
m_grid.FixedRows = 1
m_grid.Cols = 4
m_grid.Rows = 100
m_grid.Width = 10000
m_grid.ColWidth(0) = 2700
m_grid.ColWidth(1) = 1500
m_grid.ColWidth(2) = 1150
m_grid.ColWidth(3) = 4000
Call addheaders
End Sub
Public Sub DisplayOrder()
Dim n As Integer
Dim i As OrderItem
Dim RunTotal As String
Dim addstring As String
Dim TabChar As String * 1
Dim rowcounter As Integer
TabChar = Chr(9)
Call cleargrid
rowcounter = 1
For Each i In colOrder
addstring = ""
addstring = i.ProductName & TabChar
addstring = addstring & i.quantity & TabChar
addstring = addstring & Format(i.UnitCost, "0.00") & TabChar
addstring = addstring & Format(i.itemtotal, "0.00")
m_grid.AddItem addstring, rowcounter
rowcounter = rowcounter + 1
Next
m_grid.row = rowcounter - 1
m_grid.Col = 3
m_grid.CellFontUnderline = True
If Discount Then
m_grid.row = rowcounter + 1
m_grid.Col = 0
m_grid.CellFontUnderline = True
m_grid.Text = "Total(N): "
m_grid.row = rowcounter + 1
m_grid.Col = 3
m_grid.CellFontUnderline = True
m_grid.Text = "N" & Format(RunTotal, "0.00")
```

```
m_grid.row = rowcounter + 3
m_grid.Col = 0
m_grid.CellFontUnderline = True
m_grid.Text = "Total wDiscount: "
m_grid.row = rowcounter + 3
m_grid.Col = 3
m_grid.CellFontBold = True
m_grid.Text = "N" & Format(RunTotal * StdDiscount, "0.00")
frmInvoice.txtTotal.Text = m_grid.Text
m_grid.CellBackColor = &H80FF&
Else
m_grid.row = rowcounter + 1
m_grid.Col = 0
m_grid.CellFontBold = True
m_grid.Text = "Total: "
m_grid.row = rowcounter + 1
m_grid.Col = 3
m_grid.CellFontBold = True
m_grid.Text = "N" & Format(RunTotal, "0.00")
frmInvoice.txtTotal.Text = m_grid.Text
m_grid.CellBackColor = &H80FF&
m_grid.row = rowcounter + 2
m_grid.Col = 0
m_grid.CellFontBold = True
m_grid.Text = "Date: "
m_grid.row = rowcounter + 2
m_grid.Col = 3
m_grid.CellFontBold = True
m_grid.Text = Date
frmInvoice.txtDate.Text = m_grid.Text
m_grid.CellBackColor = &HFFFF00
m_grid.row = rowcounter + 3
m_grid.Col = 0
m_grid.CellFontBold = True
m_grid.Text = "Time "
m_grid.row = rowcounter + 3
m_grid.Col = 3
m_grid.CellFontBold = True
m_grid.Text = Time
frmInvoice.txtTime.Text = m_grid.Text
m_grid.CellBackColor = &H80FF&
End If
For n = colOrder.count To Step - 1
colOrder.Remove n
Next
End Sub
Private m_ProductName As String
Private m_Quantity As Double
Private m_UnitCost As Currency
Private m_TimeStamp As Date
Private m_ItemTotal As Currency
Private m_OrderNum As Integer
```

```
Private Const Markup = 0.5
Public Property Get ProductName() As String
ProductName = m_ProductName
End Property
Public Property Let ProductName(ByVal vNewValue As String)
m_ProductName = vNewValue
End Property
Public Property Get quantity() As Double
quantity = m_Quantity
End Property
Public Property Let quantity(ByVal vNewValue As Double)
m_Quantity = vNewValue
End Property
Public Property Get UnitCost() As Currency
UnitCost = m_UnitCost
End Property
Public Property Let UnitCost(ByVal vNewValue As Currency)
m_UnitCost = vNewValue
End Property
Public Property Get itemtotal() As Currency
End Property
Public Property Get orderNum() As Integer
orderNum = m_OrderNum
End Property
Public Property Let orderNum(ByVal vNewValue As Integer)
m_OrderNum = vNewValue
End Property
Private Property Get TimeStamp() As Date
TimeStamp = m_TimeStamp
End Property
Private Sub class_Initialize()
m_ProductName = "Product"
m_UnitCost = 0
m_Quantity = 0
m_ItemTotal = 0
m_OrderNum = 0
End Sub
‘Code for Two Hidden Layer Begins Here
‘Similar Code with Three Hidden Node (One Hidden Layer)
Private Sub Form_Load()
frmDiagnNN.Caption = "***An implementation of a Feedforward Error-Back Propagation
ANN Learning Algorithm with 2 Hidden Layers***"
    lblStT1.Caption = ""
    'Text3.Text = Time
    StartTime = Time
    Timer1.Interval = 1000    'interval of 1 second
    Timer1.Enabled = True
  Call WtAdjust
  Call WHide
End Sub
Private Sub Iterate()
Dim i As Integer
```

```
'wieght adjustments for input layers (first forward sweep)
If lblOutput.Caption = "0" Then
'plot initial weight
Call PlotWeight
'node1
For i = 0 To 29
 txtEpochWeight(i).Text = Val(txtWeights(i).Text) * Val(txtQuantity(i).Text)
Next i
txtEpochWeight(92).Text = Val(txtWeights(105).Text) * Val(txtQuantity(30).Text)
txtNode(0).Text = Val(txtEpochWeight(92).Text) + Val(txtEpochWeight(0).Text) +
Val(txtEpochWeight(1).Text) + Val(txtEpochWeight(2).Text) +
Val(txtEpochWeight(3).Text) + Val(txtEpochWeight(4).Text) +
Val(txtEpochWeight(5).Text) + Val(txtEpochWeight(6).Text) +
Val(txtEpochWeight(7).Text) + Val(txtEpochWeight(8).Text) +
Val(txtEpochWeight(9).Text) + Val(txtEpochWeight(10).Text) +
Val(txtEpochWeight(11).Text) + Val(txtEpochWeight(12).Text) +
Val(txtEpochWeight(13).Text) + Val(txtEpochWeight(14).Text) +
Val(txtEpochWeight(15).Text) + Val(txtEpochWeight(16).Text) +
Val(txtEpochWeight(17).Text) + Val(txtEpochWeight(18).Text) +
Val(txtEpochWeight(19).Text) + Val(txtEpochWeight(20).Text) +
Val(txtEpochWeight(21).Text) + Val(txtEpochWeight(22).Text) +
Val(txtEpochWeight(23).Text) + Val(txtEpochWeight(24).Text) +
Val(txtEpochWeight(25).Text) + Val(txtEpochWeight(26).Text) +
Val(txtEpochWeight(27).Text) + Val(txtEpochWeight(28).Text) +
Val(txtEpochWeight(29).Text)
txtNode(0).Text = 1 / (1 + Exp(-(txtNode(0).Text)))
'node2
txtEpochWeight(30).Text = Val(txtWeights(30).Text) * Val(txtQuantity(0).Text)
txtEpochWeight(31).Text = Val(txtWeights(31).Text) * Val(txtQuantity(1).Text)
txtEpochWeight(32).Text = Val(txtWeights(32).Text) * Val(txtQuantity(2).Text)
txtEpochWeight(33).Text = Val(txtWeights(33).Text) * Val(txtQuantity(3).Text)
txtEpochWeight(34).Text = Val(txtWeights(34).Text) * Val(txtQuantity(4).Text)
txtEpochWeight(35).Text = Val(txtWeights(35).Text) * Val(txtQuantity(5).Text)
txtEpochWeight(36).Text = Val(txtWeights(36).Text) * Val(txtQuantity(6).Text)
txtEpochWeight(37).Text = Val(txtWeights(37).Text) * Val(txtQuantity(7).Text)
txtEpochWeight(38).Text = Val(txtWeights(38).Text) * Val(txtQuantity(8).Text)
txtEpochWeight(39).Text = Val(txtWeights(39).Text) * Val(txtQuantity(9).Text)
txtEpochWeight(40).Text = Val(txtWeights(40).Text) * Val(txtQuantity(10).Text)
txtEpochWeight(41).Text = Val(txtWeights(41).Text) * Val(txtQuantity(11).Text)
txtEpochWeight(42).Text = Val(txtWeights(42).Text) * Val(txtQuantity(12).Text)
txtEpochWeight(43).Text = Val(txtWeights(43).Text) * Val(txtQuantity(13).Text)
txtEpochWeight(44).Text = Val(txtWeights(44).Text) * Val(txtQuantity(14).Text)
txtEpochWeight(45).Text = Val(txtWeights(45).Text) * Val(txtQuantity(15).Text)
txtEpochWeight(46).Text = Val(txtWeights(46).Text) * Val(txtQuantity(16).Text)
txtEpochWeight(47).Text = Val(txtWeights(47).Text) * Val(txtQuantity(17).Text)
txtEpochWeight(48).Text = Val(txtWeights(48).Text) * Val(txtQuantity(18).Text)
txtEpochWeight(49).Text = Val(txtWeights(49).Text) * Val(txtQuantity(19).Text)
txtEpochWeight(50).Text = Val(txtWeights(50).Text) * Val(txtQuantity(20).Text)
txtEpochWeight(51).Text = Val(txtWeights(51).Text) * Val(txtQuantity(21).Text)
txtEpochWeight(52).Text = Val(txtWeights(52).Text) * Val(txtQuantity(22).Text)
txtEpochWeight(53).Text = Val(txtWeights(53).Text) * Val(txtQuantity(23).Text)
txtEpochWeight(54).Text = Val(txtWeights(54).Text) * Val(txtQuantity(24).Text)
```

```
txtEpochWeight(55).Text = Val(txtWeights(55).Text) * Val(txtQuantity(25).Text)
txtEpochWeight(56).Text = Val(txtWeights(56).Text) * Val(txtQuantity(26).Text)
txtEpochWeight(57).Text = Val(txtWeights(57).Text) * Val(txtQuantity(27).Text)
txtEpochWeight(58).Text = Val(txtWeights(58).Text) * Val(txtQuantity(28).Text)
txtEpochWeight(59).Text = Val(txtWeights(59).Text) * Val(txtQuantity(29).Text)
'compute sigmoid for node2
txtEpochWeight(91).Text = Val(txtWeights(106).Text) * Val(txtQuantity(30).Text)
txtNode(1).Text = Val(txtEpochWeight(91).Text) + Val(txtEpochWeight(30).Text) +
Val(txtEpochWeight(31).Text) + Val(txtEpochWeight(32).Text) +
Val(txtEpochWeight(33).Text) + Val(txtEpochWeight(34).Text) +
Val(txtEpochWeight(35).Text) + Val(txtEpochWeight(36).Text) +
Val(txtEpochWeight(37).Text) + Val(txtEpochWeight(38).Text) +
Val(txtEpochWeight(39).Text) + Val(txtEpochWeight(40).Text) +
Val(txtEpochWeight(41).Text) + Val(txtEpochWeight(42).Text) +
Val(txtEpochWeight(43).Text) + Val(txtEpochWeight(44).Text) +
Val(txtEpochWeight(45).Text) + Val(txtEpochWeight(46).Text) +
Val(txtEpochWeight(47).Text) + Val(txtEpochWeight(48).Text) +
Val(txtEpochWeight(49).Text) + Val(txtEpochWeight(50).Text) +
Val(txtEpochWeight(51).Text) + Val(txtEpochWeight(52).Text) +
Val(txtEpochWeight(53).Text) + Val(txtEpochWeight(54).Text) +
Val(txtEpochWeight(55).Text) + Val(txtEpochWeight(56).Text) +
Val(txtEpochWeight(57).Text) + Val(txtEpochWeight(58).Text) +
Val(txtEpochWeight(59).Text)
txtNode(1).Text = 1 / (1 + Exp(-(txtNode(1).Text)))
'node 3
txtEpochWeight(60).Text = Val(txtWeights(60).Text) * Val(txtQuantity(0).Text)
txtEpochWeight(61).Text = Val(txtWeights(61).Text) * Val(txtQuantity(1).Text)
txtEpochWeight(62).Text = Val(txtWeights(62).Text) * Val(txtQuantity(2).Text)
txtEpochWeight(63).Text = Val(txtWeights(63).Text) * Val(txtQuantity(3).Text)
txtEpochWeight(64).Text = Val(txtWeights(64).Text) * Val(txtQuantity(4).Text)
txtEpochWeight(65).Text = Val(txtWeights(65).Text) * Val(txtQuantity(5).Text)
txtEpochWeight(66).Text = Val(txtWeights(66).Text) * Val(txtQuantity(6).Text)
txtEpochWeight(67).Text = Val(txtWeights(67).Text) * Val(txtQuantity(7).Text)
txtEpochWeight(68).Text = Val(txtWeights(68).Text) * Val(txtQuantity(8).Text)
txtEpochWeight(69).Text = Val(txtWeights(69).Text) * Val(txtQuantity(9).Text)
txtEpochWeight(70).Text = Val(txtWeights(70).Text) * Val(txtQuantity(10).Text)
txtEpochWeight(71).Text = Val(txtWeights(71).Text) * Val(txtQuantity(11).Text)
txtEpochWeight(72).Text = Val(txtWeights(72).Text) * Val(txtQuantity(12).Text)
txtEpochWeight(73).Text = Val(txtWeights(73).Text) * Val(txtQuantity(13).Text)
txtEpochWeight(74).Text = Val(txtWeights(74).Text) * Val(txtQuantity(14).Text)
txtEpochWeight(75).Text = Val(txtWeights(75).Text) * Val(txtQuantity(15).Text)
txtEpochWeight(76).Text = Val(txtWeights(76).Text) * Val(txtQuantity(16).Text)
txtEpochWeight(77).Text = Val(txtWeights(77).Text) * Val(txtQuantity(17).Text)
txtEpochWeight(78).Text = Val(txtWeights(78).Text) * Val(txtQuantity(18).Text)
txtEpochWeight(79).Text = Val(txtWeights(79).Text) * Val(txtQuantity(19).Text)
txtEpochWeight(80).Text = Val(txtWeights(80).Text) * Val(txtQuantity(20).Text)
txtEpochWeight(81).Text = Val(txtWeights(81).Text) * Val(txtQuantity(21).Text)
txtEpochWeight(82).Text = Val(txtWeights(82).Text) * Val(txtQuantity(22).Text)
txtEpochWeight(83).Text = Val(txtWeights(83).Text) * Val(txtQuantity(23).Text)
txtEpochWeight(84).Text = Val(txtWeights(84).Text) * Val(txtQuantity(24).Text)
txtEpochWeight(85).Text = Val(txtWeights(85).Text) * Val(txtQuantity(25).Text)
txtEpochWeight(86).Text = Val(txtWeights(86).Text) * Val(txtQuantity(26).Text)
```

```
txtEpochWeight(87).Text = Val(txtWeights(87).Text) * Val(txtQuantity(27).Text)
txtEpochWeight(88).Text = Val(txtWeights(88).Text) * Val(txtQuantity(28).Text)
txtEpochWeight(89).Text = Val(txtWeights(89).Text) * Val(txtQuantity(29).Text)
'compute sigmoid for node3
txtEpochWeight(90).Text = Val(txtWeights(107).Text) * Val(txtQuantity(30).Text)
txtNode(2).Text = Val(txtEpochWeight(90).Text) + Val(txtEpochWeight(60).Text) +
Val(txtEpochWeight(61).Text) + Val(txtEpochWeight(62).Text) +
Val(txtEpochWeight(63).Text) + Val(txtEpochWeight(64).Text) +
Val(txtEpochWeight(65).Text) + Val(txtEpochWeight(66).Text) +
Val(txtEpochWeight(67).Text) + Val(txtEpochWeight(68).Text) +
Val(txtEpochWeight(69).Text) + Val(txtEpochWeight(70).Text) +
Val(txtEpochWeight(71).Text) + Val(txtEpochWeight(72).Text) +
Val(txtEpochWeight(73).Text) + Val(txtEpochWeight(74).Text) +
Val(txtEpochWeight(75).Text) + Val(txtEpochWeight(76).Text) +
Val(txtEpochWeight(77).Text) + Val(txtEpochWeight(78).Text) +
Val(txtEpochWeight(79).Text) + Val(txtEpochWeight(80).Text) +
Val(txtEpochWeight(81).Text) + Val(txtEpochWeight(82).Text) +
Val(txtEpochWeight(83).Text) + Val(txtEpochWeight(84).Text) +
Val(txtEpochWeight(85).Text) + Val(txtEpochWeight(86).Text) +
Val(txtEpochWeight(87).Text) + Val(txtEpochWeight(88).Text) +
Val(txtEpochWeight(89).Text)
txtNode(2).Text = 1 / (1 + Exp(-(txtNode(2).Text)))
"""""""""Second Hidden Layer sweep parameters
'computation of 2nd hidden layer values
'nets(SecondLayerNode1)
txtNetSecondLayer(0).Text = Val(txtSecLayWeights(0).Text) * (1 / Val(txtNode(0).Text))
txtNetSecondLayer(1).Text = Val(txtSecLayWeights(1).Text) * (1 / Val(txtNode(1).Text))
txtNetSecondLayer(2).Text = Val(txtSecLayWeights(2).Text) * (1 / Val(txtNode(2).Text))
txtNetSecondLayer(3).Text = txtSecLayWeights(6).Text
'NetS(SecondLayer) = EwinYin
txtSecLayerNode(0).Text = Val(txtNetSecondLayer(3).Text) +
Val(txtNetSecondLayer(2).Text) + Val(txtNetSecondLayer(1).Text) +
Val(txtNetSecondLayer(0).Text)
txtSecLayerNode(0).Text = 1 / (1 + Exp(-(txtSecLayerNode(0).Text))) 'Sigmodation
'nets(SecondLayerNode2)
txtNetSecondLayer(4).Text = Val(txtSecLayWeights(3).Text) * (1 / Val(txtNode(0).Text))
txtNetSecondLayer(5).Text = Val(txtSecLayWeights(4).Text) * (1 / Val(txtNode(1).Text))
txtNetSecondLayer(6).Text = Val(txtSecLayWeights(5).Text) * (1 / Val(txtNode(2).Text))
txtNetSecondLayer(7).Text = txtSecLayWeights(7).Text
'NetS(SecondLayerN2) = EwinYin
txtSecLayerNode(1).Text = Val(txtNetSecondLayer(4).Text) +
Val(txtNetSecondLayer(5).Text) + Val(txtNetSecondLayer(6).Text) +
Val(txtNetSecondLayer(7).Text)
txtSecLayerNode(1).Text = 1 / (1 + Exp(-(txtSecLayerNode(1).Text))) 'Sigmodation and
'weight adjustments for Outputs (first forward sweep)
txtOutput1.Text = Val(txtWeights(112).Text) + Val(txtSecLayerNode(0).Text) *
Val(txtWeights(90).Text) + Val(txtSecLayerNode(1).Text) * Val(txtWeights(91).Text)
txtOutput2.Text = Val(txtWeights(111).Text) + Val(txtSecLayerNode(0).Text) *
Val(txtWeights(93).Text) + Val(txtSecLayerNode(1).Text) * Val(txtWeights(94).Text)
txtOutput3.Text = Val(txtWeights(110).Text) + Val(txtSecLayerNode(0).Text) *
Val(txtWeights(96).Text) + Val(txtSecLayerNode(1).Text) * Val(txtWeights(97).Text)
```

```
txtOutput4.Text = Val(txtWeights(109).Text) + Val(txtSecLayerNode(0).Text) * Val(txtWeights(99).Text) + Val(txtSecLayerNode(1).Text) * Val(txtWeights(100).Text)
txtOutput5.Text = Val(txtWeights(108).Text) + Val(txtSecLayerNode(0).Text) * Val(txtWeights(102).Text) + Val(txtSecLayerNode(1).Text) * Val(txtWeights(103).Text)
txtOutput1.Text = 1 / (1 + Exp(-(txtOutput1.Text)))
txtOutput2.Text = 1 / (1 + Exp(-(txtOutput2.Text)))
txtOutput3.Text = 1 / (1 + Exp(-(txtOutput3.Text)))
txtOutput4.Text = 1 / (1 + Exp(-(txtOutput4.Text)))
txtOutput5.Text = 1 / (1 + Exp(-(txtOutput5.Text)))
Call counter
Else
End If
Call WeightList
Exit Sub
End Sub
Private Sub counter()
Static counter As Integer
counter = counter + 1
lblOutput.Caption = counter
End Sub
Private Sub IterateTwo()
Dim i As Integer
'wieght adjustments for output layers (Next (i) forward sweep)
dz1 = (Val(txtLearnR_n.Text) * Val(txtOutput1.Text) * (1 - Val(txtOutput1.Text)) * (Val(txtActual(0).Text) - Val(txtOutput1.Text)) * Val(txtQuantity(30).Text))
dz2 = (Val(txtLearnR_n.Text) * Val(txtOutput2.Text) * (1 - Val(txtOutput2.Text)) * (Val(txtActual(1).Text) - Val(txtOutput2.Text)) * Val(txtQuantity(30).Text))
dz3 = (Val(txtLearnR_n.Text) * Val(txtOutput3.Text) * (1 - Val(txtOutput3.Text)) * (Val(txtActual(2).Text) - Val(txtOutput3.Text)) * Val(txtQuantity(30).Text))
dz4 = (Val(txtLearnR_n.Text) * Val(txtOutput4.Text) * (1 - Val(txtOutput4.Text)) * (Val(txtActual(3).Text) - Val(txtOutput4.Text)) * Val(txtQuantity(30).Text))
dz5 = (Val(txtLearnR_n.Text) * Val(txtOutput5.Text) * (1 - Val(txtOutput5.Text)) * (Val(txtActual(4).Text) - Val(txtOutput5.Text)) * Val(txtQuantity(30).Text))
'using dz(i) combo size
txtWeights(112).Text = Val(txtWeights(112).Text) + dz1
txtWeights(111).Text = Val(txtWeights(111).Text) + dz2
txtWeights(110).Text = Val(txtWeights(110).Text) + dz3
txtWeights(109).Text = Val(txtWeights(109).Text) + dz4
txtWeights(108).Text = Val(txtWeights(108).Text) + dz5
'Display generated error
txtError.Text = dz1
dferr = Val(txtError.Text) * (-1)
'Add error for graph plotting
List6.AddItem dferr
List7.AddItem lblOutput.Caption
'wieght adjustments for hidden layers (Next (i) forward sweep)
'error responsibility dA
      'dA = outputNA(1 - outputNA)E(downstream)WjkDj
dA1 = Val(txtNode(0).Text) * (1 - Val(txtNode(0).Text)) * Val(txtWeights(90).Text) * dz1
   'compute Change in WAz = ndz(1).OutputA
   CAz1 = Val(txtLearnR_n.Text) * dz1 * Val(txtNode(0).Text)
   'WAz,new = WAz,current + Change in WAz
```

```
    txtWeights(90).Text = Val(txtWeights(90).Text) + CAz1
'Next node2
dA2 = Val(txtNode(0).Text) * (1 - Val(txtNode(0).Text)) * Val(txtWeights(93).Text) * dz2
    'compute Change in WAz = ndz(1).OutputA
    CAz2 = Val(txtLearnR_n.Text) * dz2 * Val(txtNode(0).Text)
    'WAz,new = WAz,current + Change in WAz
    txtWeights(93).Text = Val(txtWeights(93).Text) + CAz2
'Next node3
dA3 = Val(txtNode(0).Text) * (1 - Val(txtNode(0).Text)) * Val(txtWeights(96).Text) * dz3
    'compute Change in WAz = ndz(1).OutputA
    CAz3 = Val(txtLearnR_n.Text) * dz3 * Val(txtNode(0).Text)
    'WAz,new = WAz,current + Change in WAz
    txtWeights(96).Text = Val(txtWeights(96).Text) + CAz3
'Next node4
dA4 = Val(txtNode(0).Text) * (1 - Val(txtNode(0).Text)) * Val(txtWeights(99).Text) * dz4
    'compute Change in WAz = ndz(1).OutputA
    CAz4 = Val(txtLearnR_n.Text) * dz4 * Val(txtNode(0).Text)
    'WAz,new = WAz,current + Change in WAz
    txtWeights(99).Text = Val(txtWeights(99).Text) + CAz4
'Next node5
dA5 = Val(txtNode(0).Text) * (1 - Val(txtNode(0).Text)) * Val(txtWeights(102).Text) * dz5
    'compute Change in WAz = ndz(1).OutputA
    CAz5 = Val(txtLearnR_n.Text) * dz5 * Val(txtNode(0).Text)
    'WAz,new = WAz,current + Change in WAz
    txtWeights(102).Text = Val(txtWeights(102).Text) + CAz5
'error responsibility dB
        'dB = outputNB(1 - outputNB)E(downstream)WjkDj
dB1 = Val(txtNode(1).Text) * (1 - Val(txtNode(1).Text)) * Val(txtWeights(91).Text) * dz1
    'compute Change in WBz = ndz(1).OutputB
    CBz1 = Val(txtLearnR_n.Text) * dz1 * Val(txtNode(1).Text)
    'WBz,new = WBz,current + Change in WBz
    txtWeights(91).Text = Val(txtWeights(91).Text) + CBz1
'Next node
        'dB = outputNB(1 - outputNB)E(downstream)WjkDj
dB2 = Val(txtNode(1).Text) * (1 - Val(txtNode(1).Text)) * Val(txtWeights(94).Text) * dz2
    'compute Change in WBz = ndz(1).OutputB
    CBz2 = Val(txtLearnR_n.Text) * dz2 * Val(txtNode(1).Text)
    'WBz,new = WBz,current + Change in WBz
    txtWeights(94).Text = Val(txtWeights(94).Text) + CBz2
'Next node
        'dB = outputNB(1 - outputNB)E(downstream)WjkDj
dB3 = Val(txtNode(1).Text) * (1 - Val(txtNode(1).Text)) * Val(txtWeights(97).Text) * dz3
    'compute Change in WBz = ndz(1).OutputB
    CBz3 = Val(txtLearnR_n.Text) * dz3 * Val(txtNode(1).Text)
    'WBz,new = WBz,current + Change in WBz
    txtWeights(97).Text = Val(txtWeights(97).Text) + CBz3
'Next node
        'dB = outputNB(1 - outputNB)E(downstream)WjkDj
dB4 = Val(txtNode(1).Text) * (1 - Val(txtNode(1).Text)) * Val(txtWeights(100).Text) * dz4
    'compute Change in WBz = ndz(1).OutputB
    CBz4 = Val(txtLearnR_n.Text) * dz4 * Val(txtNode(1).Text)
    'WBz,new = WBz,current + Change in WBz
```

```
    txtWeights(100).Text = Val(txtWeights(100).Text) + CBz4
'Next node
        'dB = outputNB(1 - outputNB)E(downstream)WjkDj
dB5 = Val(txtNode(1).Text) * (1 - Val(txtNode(1).Text)) * Val(txtWeights(103).Text) * dz5
    'compute Change in WBz = ndz(1).OutputB
    CBz5 = Val(txtLearnR_n.Text) * dz5 * Val(txtNode(1).Text)
    'WBz,new = WBz,current + Change in WBz
    txtWeights(103).Text = Val(txtWeights(103).Text) + CBz5
'error responsibility dC
        'dC = outputNC(1 - outputNC)E(downstream)WjkDj
dC1 = Val(txtNode(2).Text) * (1 - Val(txtNode(2).Text)) * Val(txtWeights(92).Text) * dz1
    'compute Change in WCz = ndz(1).OutputC
    CCz1 = Val(txtLearnR_n.Text) * dz1 * Val(txtNode(2).Text)
    'WCz,new = WCz,current + Change in WCz
    txtWeights(92).Text = Val(txtWeights(92).Text) + CCz1
'Next node
        'dC = outputNC(1 - outputNC)E(downstream)WjkDj
dC2 = Val(txtNode(2).Text) * (1 - Val(txtNode(2).Text)) * Val(txtWeights(95).Text) * dz2
    'compute Change in WCz = ndz(1).OutputC
    CCz2 = Val(txtLearnR_n.Text) * dz2 * Val(txtNode(2).Text)
    'WCz,new = WCz,current + Change in WCz
    txtWeights(95).Text = Val(txtWeights(95).Text) + CCz2
'Next node
        'dC = outputNC(1 - outputNC)E(downstream)WjkDj
dC3 = Val(txtNode(2).Text) * (1 - Val(txtNode(2).Text)) * Val(txtWeights(98).Text) * dz3
    'compute Change in WCz = ndz(1).OutputC
    CCz3 = Val(txtLearnR_n.Text) * dz3 * Val(txtNode(2).Text)
    'WCz,new = WCz,current + Change in WCz
    txtWeights(98).Text = Val(txtWeights(98).Text) + CCz3
'Next node
        'dC = outputNC(1 - outputNC)E(downstream)WjkDj
dC4 = Val(txtNode(2).Text) * (1 - Val(txtNode(2).Text)) * Val(txtWeights(101).Text) * dz4
    'compute Change in WCz = ndz(1).OutputC
    CCz4 = Val(txtLearnR_n.Text) * dz4 * Val(txtNode(2).Text)
    'WCz,new = WCz,current + Change in WCz
    txtWeights(101).Text = Val(txtWeights(101).Text) + CCz4
'Next node
        'dC = outputNC(1 - outputNC)E(downstream)WjkDj
dC5 = Val(txtNode(2).Text) * (1 - Val(txtNode(2).Text)) * Val(txtWeights(104).Text) * dz5
    'compute Change in WCz = ndz(1).OutputC
    CCz5 = Val(txtLearnR_n.Text) * dz5 * Val(txtNode(2).Text)
    'WCz,new = WCz,current + Change in WCz
    txtWeights(104).Text = Val(txtWeights(104).Text) + CCz5
'wieght adjustments for inputs (Next (i) forward sweep)
        'compute Change in WiA = ndAXi
        'WiA,new = WiA,current + Change in WiA
    CW1A = Val(txtLearnR_n.Text) * dA1 * Val(txtQuantity(0).Text)
    txtWeights(0).Text = Val(txtWeights(0).Text) + CW1A
    CW2A = Val(txtLearnR_n.Text) * dA1 * Val(txtQuantity(1).Text)
    txtWeights(1).Text = Val(txtWeights(1).Text) + CW2A
    CW3A = Val(txtLearnR_n.Text) * dA1 * Val(txtQuantity(2).Text)
    txtWeights(2).Text = Val(txtWeights(2).Text) + CW3A
```

```
CW4A = Val(txtLearnR_n.Text) * dA1 * Val(txtQuantity(3).Text)
txtWeights(3).Text = Val(txtWeights(3).Text) + CW4A
CW5A = Val(txtLearnR_n.Text) * dA1 * Val(txtQuantity(4).Text)
txtWeights(4).Text = Val(txtWeights(4).Text) + CW5A
CW6A = Val(txtLearnR_n.Text) * dA1 * Val(txtQuantity(5).Text)
txtWeights(5).Text = Val(txtWeights(5).Text) + CW6A
CW7A = Val(txtLearnR_n.Text) * dA1 * Val(txtQuantity(6).Text)
txtWeights(6).Text = Val(txtWeights(6).Text) + CW7A
CW8A = Val(txtLearnR_n.Text) * dA1 * Val(txtQuantity(7).Text)
txtWeights(7).Text = Val(txtWeights(7).Text) + CW8A
CW9A = Val(txtLearnR_n.Text) * dA1 * Val(txtQuantity(8).Text)
txtWeights(8).Text = Val(txtWeights(8).Text) + CW9A
CW10A = Val(txtLearnR_n.Text) * dA1 * Val(txtQuantity(9).Text)
txtWeights(9).Text = Val(txtWeights(9).Text) + CW10A
CW11A = Val(txtLearnR_n.Text) * dA1 * Val(txtQuantity(10).Text)
txtWeights(10).Text = Val(txtWeights(10).Text) + CW11A
CW12A = Val(txtLearnR_n.Text) * dA1 * Val(txtQuantity(11).Text)
txtWeights(11).Text = Val(txtWeights(11).Text) + CW12A
CW13A = Val(txtLearnR_n.Text) * dA1 * Val(txtQuantity(12).Text)
txtWeights(12).Text = Val(txtWeights(12).Text) + CW13A
CW14A = Val(txtLearnR_n.Text) * dA1 * Val(txtQuantity(13).Text)
txtWeights(13).Text = Val(txtWeights(13).Text) + CW14A
CW15A = Val(txtLearnR_n.Text) * dA1 * Val(txtQuantity(14).Text)
txtWeights(14).Text = Val(txtWeights(14).Text) + CW15A
CW16A = Val(txtLearnR_n.Text) * dA1 * Val(txtQuantity(15).Text)
txtWeights(15).Text = Val(txtWeights(15).Text) + CW16A
CW17A = Val(txtLearnR_n.Text) * dA1 * Val(txtQuantity(16).Text)
txtWeights(16).Text = Val(txtWeights(16).Text) + CW17A
CW18A = Val(txtLearnR_n.Text) * dA1 * Val(txtQuantity(17).Text)
txtWeights(17).Text = Val(txtWeights(17).Text) + CW18A
CW19A = Val(txtLearnR_n.Text) * dA1 * Val(txtQuantity(18).Text)
txtWeights(18).Text = Val(txtWeights(18).Text) + CW19A
CW20A = Val(txtLearnR_n.Text) * dA1 * Val(txtQuantity(19).Text)
txtWeights(19).Text = Val(txtWeights(19).Text) + CW20A
CW21A = Val(txtLearnR_n.Text) * dA1 * Val(txtQuantity(20).Text)
txtWeights(20).Text = Val(txtWeights(20).Text) + CW21A
CW22A = Val(txtLearnR_n.Text) * dA1 * Val(txtQuantity(21).Text)
txtWeights(21).Text = Val(txtWeights(21).Text) + CW21A
CW23A = Val(txtLearnR_n.Text) * dA1 * Val(txtQuantity(22).Text)
txtWeights(22).Text = Val(txtWeights(22).Text) + CW23A
CW24A = Val(txtLearnR_n.Text) * dA1 * Val(txtQuantity(23).Text)
txtWeights(23).Text = Val(txtWeights(23).Text) + CW24A
CW25A = Val(txtLearnR_n.Text) * dA1 * Val(txtQuantity(24).Text)
txtWeights(24).Text = Val(txtWeights(24).Text) + CW25A
CW26A = Val(txtLearnR_n.Text) * dA1 * Val(txtQuantity(25).Text)
txtWeights(25).Text = Val(txtWeights(25).Text) + CW22A
CW27A = Val(txtLearnR_n.Text) * dA1 * Val(txtQuantity(26).Text)
txtWeights(26).Text = Val(txtWeights(26).Text) + CW27A
CW28A = Val(txtLearnR_n.Text) * dA1 * Val(txtQuantity(27).Text)
txtWeights(27).Text = Val(txtWeights(27).Text) + CW28A
CW29A = Val(txtLearnR_n.Text) * dA1 * Val(txtQuantity(28).Text)
txtWeights(28).Text = Val(txtWeights(28).Text) + CW29A
```

```
CW30A = Val(txtLearnR_n.Text) * dA1 * Val(txtQuantity(29).Text)
txtWeights(29).Text = Val(txtWeights(29).Text) + CW30A
CW0A = Val(txtLearnR_n.Text) * dA1 * Val(txtQuantity(30).Text)
txtWeights(105).Text = Val(txtWeights(105).Text) + CW0A
   'compute Change in WiB = ndBXi
   'WiB,new = WiB,current + Change in WiB
CW1B = Val(txtLearnR_n.Text) * dB1 * Val(txtQuantity(0).Text)
txtWeights(30).Text = Val(txtWeights(0).Text) + CW1B
CW2B = Val(txtLearnR_n.Text) * dB1 * Val(txtQuantity(1).Text)
txtWeights(31).Text = Val(txtWeights(1).Text) + CW2B
CW3B = Val(txtLearnR_n.Text) * dB1 * Val(txtQuantity(2).Text)
txtWeights(32).Text = Val(txtWeights(2).Text) + CW3B
CW4B = Val(txtLearnR_n.Text) * dB1 * Val(txtQuantity(3).Text)
txtWeights(33).Text = Val(txtWeights(3).Text) + CW4B
CW5B = Val(txtLearnR_n.Text) * dB1 * Val(txtQuantity(4).Text)
txtWeights(34).Text = Val(txtWeights(4).Text) + CW5B
CW6B = Val(txtLearnR_n.Text) * dB1 * Val(txtQuantity(5).Text)
txtWeights(35).Text = Val(txtWeights(5).Text) + CW6B
CW7B = Val(txtLearnR_n.Text) * dB1 * Val(txtQuantity(6).Text)
txtWeights(36).Text = Val(txtWeights(6).Text) + CW7B
CW8B = Val(txtLearnR_n.Text) * dB1 * Val(txtQuantity(7).Text)
txtWeights(37).Text = Val(txtWeights(7).Text) + CW8B
CW9B = Val(txtLearnR_n.Text) * dB1 * Val(txtQuantity(8).Text)
txtWeights(38).Text = Val(txtWeights(8).Text) + CW9B
CW10B = Val(txtLearnR_n.Text) * dB1 * Val(txtQuantity(9).Text)
txtWeights(39).Text = Val(txtWeights(9).Text) + CW10B
CW11B = Val(txtLearnR_n.Text) * dB1 * Val(txtQuantity(10).Text)
txtWeights(40).Text = Val(txtWeights(10).Text) + CW11B
CW12B = Val(txtLearnR_n.Text) * dB1 * Val(txtQuantity(11).Text)
txtWeights(41).Text = Val(txtWeights(11).Text) + CW12B
CW13B = Val(txtLearnR_n.Text) * dB1 * Val(txtQuantity(12).Text)
txtWeights(42).Text = Val(txtWeights(12).Text) + CW13B
CW14B = Val(txtLearnR_n.Text) * dB1 * Val(txtQuantity(13).Text)
txtWeights(43).Text = Val(txtWeights(13).Text) + CW14B
CW15B = Val(txtLearnR_n.Text) * dB1 * Val(txtQuantity(14).Text)
txtWeights(44).Text = Val(txtWeights(14).Text) + CW15B
CW16B = Val(txtLearnR_n.Text) * dB1 * Val(txtQuantity(15).Text)
txtWeights(45).Text = Val(txtWeights(15).Text) + CW16B
CW17B = Val(txtLearnR_n.Text) * dB1 * Val(txtQuantity(16).Text)
txtWeights(46).Text = Val(txtWeights(16).Text) + CW17B
CW18B = Val(txtLearnR_n.Text) * dB1 * Val(txtQuantity(17).Text)
txtWeights(47).Text = Val(txtWeights(17).Text) + CW18B
CW19B = Val(txtLearnR_n.Text) * dB1 * Val(txtQuantity(18).Text)
txtWeights(48).Text = Val(txtWeights(18).Text) + CW19B
CW20B = Val(txtLearnR_n.Text) * dB1 * Val(txtQuantity(19).Text)
txtWeights(49).Text = Val(txtWeights(19).Text) + CW20B
CW21B = Val(txtLearnR_n.Text) * dB1 * Val(txtQuantity(20).Text)
txtWeights(50).Text = Val(txtWeights(20).Text) + CW21B
CW22B = Val(txtLearnR_n.Text) * dB1 * Val(txtQuantity(21).Text)
txtWeights(51).Text = Val(txtWeights(21).Text) + CW21B
CW23B = Val(txtLearnR_n.Text) * dB1 * Val(txtQuantity(22).Text)
txtWeights(52).Text = Val(txtWeights(22).Text) + CW23B
```

```
CW24B = Val(txtLearnR_n.Text) * dB1 * Val(txtQuantity(23).Text)
txtWeights(53).Text = Val(txtWeights(23).Text) + CW24B
CW25B = Val(txtLearnR_n.Text) * dB1 * Val(txtQuantity(24).Text)
txtWeights(54).Text = Val(txtWeights(24).Text) + CW25B
CW26B = Val(txtLearnR_n.Text) * dB1 * Val(txtQuantity(25).Text)
txtWeights(55).Text = Val(txtWeights(25).Text) + CW22B
CW27B = Val(txtLearnR_n.Text) * dB1 * Val(txtQuantity(26).Text)
txtWeights(56).Text = Val(txtWeights(26).Text) + CW27B
CW28B = Val(txtLearnR_n.Text) * dB1 * Val(txtQuantity(27).Text)
txtWeights(57).Text = Val(txtWeights(27).Text) + CW28B
CW29B = Val(txtLearnR_n.Text) * dB1 * Val(txtQuantity(28).Text)
txtWeights(58).Text = Val(txtWeights(28).Text) + CW29B
CW30B = Val(txtLearnR_n.Text) * dB1 * Val(txtQuantity(29).Text)
txtWeights(59).Text = Val(txtWeights(29).Text) + CW30B
CW0B = Val(txtLearnR_n.Text) * dB1 * Val(txtQuantity(30).Text)
txtWeights(106).Text = Val(txtWeights(106).Text) + CW0B
'compute Change in WiC = ndBXi        'WiC,new = WiC,current + Change in WiC
CW1C = Val(txtLearnR_n.Text) * dC1 * Val(txtQuantity(0).Text)
txtWeights(60).Text = Val(txtWeights(0).Text) + CW1C
CW2C = Val(txtLearnR_n.Text) * dC1 * Val(txtQuantity(1).Text)
txtWeights(61).Text = Val(txtWeights(1).Text) + CW2C
CW3C = Val(txtLearnR_n.Text) * dC1 * Val(txtQuantity(2).Text)
txtWeights(62).Text = Val(txtWeights(2).Text) + CW3C
CW4C = Val(txtLearnR_n.Text) * dC1 * Val(txtQuantity(3).Text)
txtWeights(63).Text = Val(txtWeights(3).Text) + CW4C
CW5C = Val(txtLearnR_n.Text) * dC1 * Val(txtQuantity(4).Text)
txtWeights(64).Text = Val(txtWeights(4).Text) + CW5C
CW6C = Val(txtLearnR_n.Text) * dC1 * Val(txtQuantity(5).Text)
txtWeights(65).Text = Val(txtWeights(5).Text) + CW6C
CW7C = Val(txtLearnR_n.Text) * dC1 * Val(txtQuantity(6).Text)
txtWeights(66).Text = Val(txtWeights(6).Text) + CW7C
CW8C = Val(txtLearnR_n.Text) * dC1 * Val(txtQuantity(7).Text)
txtWeights(67).Text = Val(txtWeights(7).Text) + CW8C
CW9C = Val(txtLearnR_n.Text) * dC1 * Val(txtQuantity(8).Text)
txtWeights(68).Text = Val(txtWeights(8).Text) + CW9C
CW10C = Val(txtLearnR_n.Text) * dC1 * Val(txtQuantity(9).Text)
txtWeights(69).Text = Val(txtWeights(9).Text) + CW10C
CW11C = Val(txtLearnR_n.Text) * dC1 * Val(txtQuantity(10).Text)
txtWeights(70).Text = Val(txtWeights(10).Text) + CW11C
CW12C = Val(txtLearnR_n.Text) * dC1 * Val(txtQuantity(11).Text)
txtWeights(71).Text = Val(txtWeights(11).Text) + CW12C
CW13C = Val(txtLearnR_n.Text) * dC1 * Val(txtQuantity(12).Text)
txtWeights(72).Text = Val(txtWeights(12).Text) + CW13C
CW14C = Val(txtLearnR_n.Text) * dC1 * Val(txtQuantity(13).Text)
txtWeights(73).Text = Val(txtWeights(13).Text) + CW14C
CW15C = Val(txtLearnR_n.Text) * dC1 * Val(txtQuantity(14).Text)
txtWeights(74).Text = Val(txtWeights(14).Text) + CW15C
CW16C = Val(txtLearnR_n.Text) * dC1 * Val(txtQuantity(15).Text)
txtWeights(75).Text = Val(txtWeights(15).Text) + CW16C
CW17C = Val(txtLearnR_n.Text) * dC1 * Val(txtQuantity(16).Text)
txtWeights(76).Text = Val(txtWeights(16).Text) + CW17C
CW18C = Val(txtLearnR_n.Text) * dC1 * Val(txtQuantity(17).Text)
```

```
    txtWeights(77).Text = Val(txtWeights(17).Text) + CW18C
    CW19C = Val(txtLearnR_n.Text) * dC1 * Val(txtQuantity(18).Text)
    txtWeights(78).Text = Val(txtWeights(18).Text) + CW19C
    CW20C = Val(txtLearnR_n.Text) * dC1 * Val(txtQuantity(19).Text)
    txtWeights(79).Text = Val(txtWeights(19).Text) + CW20C
    CW21C = Val(txtLearnR_n.Text) * dC1 * Val(txtQuantity(20).Text)
    txtWeights(80).Text = Val(txtWeights(20).Text) + CW21C
    CW22C = Val(txtLearnR_n.Text) * dC1 * Val(txtQuantity(21).Text)
    txtWeights(81).Text = Val(txtWeights(21).Text) + CW21C
    CW23C = Val(txtLearnR_n.Text) * dC1 * Val(txtQuantity(22).Text)
    txtWeights(82).Text = Val(txtWeights(22).Text) + CW23C
    CW24C = Val(txtLearnR_n.Text) * dC1 * Val(txtQuantity(23).Text)
    txtWeights(83).Text = Val(txtWeights(23).Text) + CW24C
    CW25C = Val(txtLearnR_n.Text) * dC1 * Val(txtQuantity(24).Text)
    txtWeights(84).Text = Val(txtWeights(24).Text) + CW25C
    CW26C = Val(txtLearnR_n.Text) * dC1 * Val(txtQuantity(25).Text)
    txtWeights(85).Text = Val(txtWeights(25).Text) + CW22C
    CW27C = Val(txtLearnR_n.Text) * dC1 * Val(txtQuantity(26).Text)
    txtWeights(86).Text = Val(txtWeights(26).Text) + CW27C
    CW28C = Val(txtLearnR_n.Text) * dC1 * Val(txtQuantity(27).Text)
    txtWeights(87).Text = Val(txtWeights(27).Text) + CW28C
    CW29C = Val(txtLearnR_n.Text) * dC1 * Val(txtQuantity(28).Text)
    txtWeights(88).Text = Val(txtWeights(28).Text) + CW29C
    CW30C = Val(txtLearnR_n.Text) * dC1 * Val(txtQuantity(29).Text)
    txtWeights(89).Text = Val(txtWeights(29).Text) + CW30C
    CW0C = Val(txtLearnR_n.Text) * dC1 * Val(txtQuantity(30).Text)
    txtWeights(107).Text = Val(txtWeights(107).Text) + CW0C
    'node1
For i = 0 To 29
 txtEpochWeight(i).Text = Val(txtWeights(i).Text) * Val(txtQuantity(i).Text)
Next i
txtEpochWeight(92).Text = Val(txtWeights(105).Text) * Val(txtQuantity(30).Text)
txtNode(0).Text = Val(txtEpochWeight(92).Text) + Val(txtEpochWeight(0).Text) +
Val(txtEpochWeight(1).Text) + Val(txtEpochWeight(2).Text) +
Val(txtEpochWeight(3).Text) + Val(txtEpochWeight(4).Text) +
Val(txtEpochWeight(5).Text) + Val(txtEpochWeight(6).Text) +
Val(txtEpochWeight(7).Text) + Val(txtEpochWeight(8).Text) +
Val(txtEpochWeight(9).Text) + Val(txtEpochWeight(10).Text) +
Val(txtEpochWeight(11).Text) + Val(txtEpochWeight(12).Text) +
Val(txtEpochWeight(13).Text) + Val(txtEpochWeight(14).Text) +
Val(txtEpochWeight(15).Text) + Val(txtEpochWeight(16).Text) +
Val(txtEpochWeight(17).Text) + Val(txtEpochWeight(18).Text) +
Val(txtEpochWeight(19).Text) + Val(txtEpochWeight(20).Text) +
Val(txtEpochWeight(21).Text) + Val(txtEpochWeight(22).Text) +
Val(txtEpochWeight(23).Text) + Val(txtEpochWeight(24).Text) +
Val(txtEpochWeight(25).Text) + Val(txtEpochWeight(26).Text) +
Val(txtEpochWeight(27).Text) + Val(txtEpochWeight(28).Text) +
Val(txtEpochWeight(29).Text)
txtNode(0).Text = 1 / (1 + Exp(-(txtNode(0).Text)))
'node2
txtEpochWeight(30).Text = Val(txtWeights(30).Text) * Val(txtQuantity(0).Text)
txtEpochWeight(31).Text = Val(txtWeights(31).Text) * Val(txtQuantity(1).Text)
```

```
txtEpochWeight(32).Text = Val(txtWeights(32).Text) * Val(txtQuantity(2).Text)
txtEpochWeight(33).Text = Val(txtWeights(33).Text) * Val(txtQuantity(3).Text)
txtEpochWeight(34).Text = Val(txtWeights(34).Text) * Val(txtQuantity(4).Text)
txtEpochWeight(35).Text = Val(txtWeights(35).Text) * Val(txtQuantity(5).Text)
txtEpochWeight(36).Text = Val(txtWeights(36).Text) * Val(txtQuantity(6).Text)
txtEpochWeight(37).Text = Val(txtWeights(37).Text) * Val(txtQuantity(7).Text)
txtEpochWeight(38).Text = Val(txtWeights(38).Text) * Val(txtQuantity(8).Text)
txtEpochWeight(39).Text = Val(txtWeights(39).Text) * Val(txtQuantity(9).Text)
txtEpochWeight(40).Text = Val(txtWeights(40).Text) * Val(txtQuantity(10).Text)
txtEpochWeight(41).Text = Val(txtWeights(41).Text) * Val(txtQuantity(11).Text)
txtEpochWeight(42).Text = Val(txtWeights(42).Text) * Val(txtQuantity(12).Text)
txtEpochWeight(43).Text = Val(txtWeights(43).Text) * Val(txtQuantity(13).Text)
txtEpochWeight(44).Text = Val(txtWeights(44).Text) * Val(txtQuantity(14).Text)
txtEpochWeight(45).Text = Val(txtWeights(45).Text) * Val(txtQuantity(15).Text)
txtEpochWeight(46).Text = Val(txtWeights(46).Text) * Val(txtQuantity(16).Text)
txtEpochWeight(47).Text = Val(txtWeights(47).Text) * Val(txtQuantity(17).Text)
txtEpochWeight(48).Text = Val(txtWeights(48).Text) * Val(txtQuantity(18).Text)
txtEpochWeight(49).Text = Val(txtWeights(49).Text) * Val(txtQuantity(19).Text)
txtEpochWeight(50).Text = Val(txtWeights(50).Text) * Val(txtQuantity(20).Text)
txtEpochWeight(51).Text = Val(txtWeights(51).Text) * Val(txtQuantity(21).Text)
txtEpochWeight(52).Text = Val(txtWeights(52).Text) * Val(txtQuantity(22).Text)
txtEpochWeight(53).Text = Val(txtWeights(53).Text) * Val(txtQuantity(23).Text)
txtEpochWeight(54).Text = Val(txtWeights(54).Text) * Val(txtQuantity(24).Text)
txtEpochWeight(55).Text = Val(txtWeights(55).Text) * Val(txtQuantity(25).Text)
txtEpochWeight(56).Text = Val(txtWeights(56).Text) * Val(txtQuantity(26).Text)
txtEpochWeight(57).Text = Val(txtWeights(57).Text) * Val(txtQuantity(27).Text)
txtEpochWeight(58).Text = Val(txtWeights(58).Text) * Val(txtQuantity(28).Text)
txtEpochWeight(59).Text = Val(txtWeights(59).Text) * Val(txtQuantity(29).Text)
'compute sigmoid for node2
txtEpochWeight(91).Text = Val(txtWeights(106).Text) * Val(txtQuantity(30).Text)
txtNode(1).Text = Val(txtEpochWeight(91).Text) + Val(txtEpochWeight(30).Text) +
Val(txtEpochWeight(31).Text)          +          Val(txtEpochWeight(32).Text)          +
Val(txtEpochWeight(33).Text)          +          Val(txtEpochWeight(34).Text)          +
Val(txtEpochWeight(35).Text)          +          Val(txtEpochWeight(36).Text)          +
Val(txtEpochWeight(37).Text)          +          Val(txtEpochWeight(38).Text)          +
Val(txtEpochWeight(39).Text)          +          Val(txtEpochWeight(40).Text)          +
Val(txtEpochWeight(41).Text)          +          Val(txtEpochWeight(42).Text)          +
Val(txtEpochWeight(43).Text)          +          Val(txtEpochWeight(44).Text)          +
Val(txtEpochWeight(45).Text)          +          Val(txtEpochWeight(46).Text)          +
Val(txtEpochWeight(47).Text)          +          Val(txtEpochWeight(48).Text)          +
Val(txtEpochWeight(49).Text)          +          Val(txtEpochWeight(50).Text)          +
Val(txtEpochWeight(51).Text)          +          Val(txtEpochWeight(52).Text)          +
Val(txtEpochWeight(53).Text)          +          Val(txtEpochWeight(54).Text)          +
Val(txtEpochWeight(55).Text)          +          Val(txtEpochWeight(56).Text)          +
Val(txtEpochWeight(57).Text)          +          Val(txtEpochWeight(58).Text)          +
Val(txtEpochWeight(59).Text)
txtNode(1).Text = 1 / (1 + Exp(-(txtNode(1).Text)))
'node 3
txtEpochWeight(60).Text = Val(txtWeights(60).Text) * Val(txtQuantity(0).Text)
txtEpochWeight(61).Text = Val(txtWeights(61).Text) * Val(txtQuantity(1).Text)
txtEpochWeight(62).Text = Val(txtWeights(62).Text) * Val(txtQuantity(2).Text)
txtEpochWeight(63).Text = Val(txtWeights(63).Text) * Val(txtQuantity(3).Text)
```

```
txtEpochWeight(64).Text = Val(txtWeights(64).Text) * Val(txtQuantity(4).Text)
txtEpochWeight(65).Text = Val(txtWeights(65).Text) * Val(txtQuantity(5).Text)
txtEpochWeight(66).Text = Val(txtWeights(66).Text) * Val(txtQuantity(6).Text)
txtEpochWeight(67).Text = Val(txtWeights(67).Text) * Val(txtQuantity(7).Text)
txtEpochWeight(68).Text = Val(txtWeights(68).Text) * Val(txtQuantity(8).Text)
txtEpochWeight(69).Text = Val(txtWeights(69).Text) * Val(txtQuantity(9).Text)
txtEpochWeight(70).Text = Val(txtWeights(70).Text) * Val(txtQuantity(10).Text)
txtEpochWeight(71).Text = Val(txtWeights(71).Text) * Val(txtQuantity(11).Text)
txtEpochWeight(72).Text = Val(txtWeights(72).Text) * Val(txtQuantity(12).Text)
txtEpochWeight(73).Text = Val(txtWeights(73).Text) * Val(txtQuantity(13).Text)
txtEpochWeight(74).Text = Val(txtWeights(74).Text) * Val(txtQuantity(14).Text)
txtEpochWeight(75).Text = Val(txtWeights(75).Text) * Val(txtQuantity(15).Text)
txtEpochWeight(76).Text = Val(txtWeights(76).Text) * Val(txtQuantity(16).Text)
txtEpochWeight(77).Text = Val(txtWeights(77).Text) * Val(txtQuantity(17).Text)
txtEpochWeight(78).Text = Val(txtWeights(78).Text) * Val(txtQuantity(18).Text)
txtEpochWeight(79).Text = Val(txtWeights(79).Text) * Val(txtQuantity(19).Text)
txtEpochWeight(80).Text = Val(txtWeights(80).Text) * Val(txtQuantity(20).Text)
txtEpochWeight(81).Text = Val(txtWeights(81).Text) * Val(txtQuantity(21).Text)
txtEpochWeight(82).Text = Val(txtWeights(82).Text) * Val(txtQuantity(22).Text)
txtEpochWeight(83).Text = Val(txtWeights(83).Text) * Val(txtQuantity(23).Text)
txtEpochWeight(84).Text = Val(txtWeights(84).Text) * Val(txtQuantity(24).Text)
txtEpochWeight(85).Text = Val(txtWeights(85).Text) * Val(txtQuantity(25).Text)
txtEpochWeight(86).Text = Val(txtWeights(86).Text) * Val(txtQuantity(26).Text)
txtEpochWeight(87).Text = Val(txtWeights(87).Text) * Val(txtQuantity(27).Text)
txtEpochWeight(88).Text = Val(txtWeights(88).Text) * Val(txtQuantity(28).Text)
txtEpochWeight(89).Text = Val(txtWeights(89).Text) * Val(txtQuantity(29).Text)
'compute sigmoid for node3
txtEpochWeight(90).Text = Val(txtWeights(107).Text) * Val(txtQuantity(30).Text)
txtNode(2).Text = Val(txtEpochWeight(90).Text) + Val(txtEpochWeight(60).Text) +
Val(txtEpochWeight(61).Text)          +          Val(txtEpochWeight(62).Text)          +
Val(txtEpochWeight(63).Text)          +          Val(txtEpochWeight(64).Text)          +
Val(txtEpochWeight(65).Text)          +          Val(txtEpochWeight(66).Text)          +
Val(txtEpochWeight(67).Text)          +          Val(txtEpochWeight(68).Text)          +
Val(txtEpochWeight(69).Text)          +          Val(txtEpochWeight(70).Text)          +
Val(txtEpochWeight(71).Text)          +          Val(txtEpochWeight(72).Text)          +
Val(txtEpochWeight(73).Text)          +          Val(txtEpochWeight(74).Text)          +
Val(txtEpochWeight(75).Text)          +          Val(txtEpochWeight(76).Text)          +
Val(txtEpochWeight(77).Text)          +          Val(txtEpochWeight(78).Text)          +
Val(txtEpochWeight(79).Text)          +          Val(txtEpochWeight(80).Text)          +
Val(txtEpochWeight(81).Text)          +          Val(txtEpochWeight(82).Text)          +
Val(txtEpochWeight(83).Text)          +          Val(txtEpochWeight(84).Text)          +
Val(txtEpochWeight(85).Text)          +          Val(txtEpochWeight(86).Text)          +
Val(txtEpochWeight(87).Text)          +          Val(txtEpochWeight(88).Text)          +
Val(txtEpochWeight(89).Text)
txtNode(2).Text = 1 / (1 + Exp(-(txtNode(2).Text)))
"""""""""""""second layer error responsibility computation
dA1s = Val(txtSecLayerNode(0).Text) * (1 - Val(txtSecLayerNode(0).Text)) *
Val(txtSecLayWeights(0).Text) * dz1
   'compute Change in WAzS = ndz(1).OutputA
   CAzS1 = Val(txtLearnR_n.Text) * dz1 * Val(txtSecLayerNode(0).Text)
   'WAzS,new = WAzS,current + Change in WAzS
   txtSecLayWeights(0).Text = Val(txtSecLayWeights(0).Text) + CAzS1
```

```
'Next node2
dA2s = Val(txtSecLayerNode(0).Text) * (1 - Val(txtSecLayerNode(0).Text)) *
Val(txtSecLayWeights(1).Text) * dz2
   'compute Change in WAzS = ndz(1).OutputA
CAzS2 = Val(txtLearnR_n.Text) * dz2 * Val(txtSecLayerNode(0).Text)
   'WAzS,new = WAzS,current + Change in WAzS
   txtSecLayWeights(1).Text = Val(txtSecLayWeights(1).Text) + CAzS2
'Next node3
dA3s = Val(txtSecLayerNode(0).Text) * (1 - Val(txtSecLayerNode(0).Text)) *
Val(txtSecLayWeights(2).Text) * dz3
   'compute Change in WAzS = ndz(1).OutputA
   CAzS3 = Val(txtLearnR_n.Text) * dz3 * Val(txtSecLayerNode(0).Text)
   'WAzS,new = WAzS,current + Change in WAzS
   txtSecLayWeights(2).Text = Val(txtSecLayWeights(2).Text) + CAzS3
 'error responsibility dB
      'dB = outputNB(1 - outputNB)E(downstream)WjkDj
dB1s = Val(txtSecLayerNode(1).Text) * (1 - Val(txtSecLayerNode(1).Text)) *
Val(txtSecLayWeights(3).Text) * dz1
   'compute Change in WBzS = ndz(1).OutputB
   CBzS1 = Val(txtLearnR_n.Text) * dz1 * Val(txtSecLayerNode(1).Text)
   'WBzS,new = WBzS,current + Change in WBzS
   txtSecLayWeights(3).Text = Val(txtSecLayWeights(3).Text) + CBzS1
'Next node
      'dB = outputNB(1 - outputNB)E(downstream)WjkDj
dB2s = Val(txtSecLayerNode(1).Text) * (1 - Val(txtSecLayerNode(1).Text)) *
Val(txtSecLayWeights(4).Text) * dz2
   'compute Change in WBzS = ndz(1).OutputB
   CBzS2 = Val(txtLearnR_n.Text) * dz2 * Val(txtSecLayerNode(1).Text)
   'WBzS,new = WBzS,current + Change in WBzS
   txtSecLayWeights(4).Text = Val(txtSecLayWeights(4).Text) + CBzS2
'Next node
      'dB = outputNB(1 - outputNB)E(downstream)WjkDj
dB3s = Val(txtSecLayerNode(1).Text) * (1 - Val(txtSecLayerNode(1).Text)) *
Val(txtSecLayWeights(5).Text) * dz3
   'compute Change in WBzS = ndz(1).OutputB
   CBzS3 = Val(txtLearnR_n.Text) * dz3 * Val(txtSecLayerNode(1).Text)
   'WBzS,new = WBzS,current + Change in WBzS
   txtSecLayWeights(5).Text = Val(txtSecLayWeights(5).Text) + CBzS3
"""""compute Change in WiA = ndAXi
      'WiA,new = WiA,current + Change in WiA
   CW1As = Val(txtLearnR_n.Text) * dA1s * Val(txtNode(0).Text)
   txtSecLayWeights(0).Text = Val(txtSecLayWeights(0).Text) + CW1As
   CW2As = Val(txtLearnR_n.Text) * dA2s * Val(txtNode(1).Text)
   txtSecLayWeights(1).Text = Val(txtSecLayWeights(1).Text) + CW2As
   CW3As = Val(txtLearnR_n.Text) * dA3s * Val(txtNode(2).Text)
   txtSecLayWeights(2).Text = Val(txtSecLayWeights(2).Text) + CW3As
'compute Change in WiB = ndBXi
      'WiB,new = WiB,current + Change in WiB
   CW1Bs = Val(txtLearnR_n.Text) * dB1s * Val(txtNode(0).Text)
   txtSecLayWeights(4).Text = Val(txtSecLayWeights(0).Text) + CW1Bs
   CW2Bs = Val(txtLearnR_n.Text) * dB2s * Val(txtNode(1).Text)
   txtSecLayWeights(5).Text = Val(txtSecLayWeights(1).Text) + CW2Bs
```

```
    CW3Bs = Val(txtLearnR_n.Text) * dB3s * Val(txtNode(2).Text)
    txtSecLayWeights(6).Text = Val(txtSecLayWeights(2).Text) + CW3Bs
'nets(SecondLayerNode1)
txtNetSecondLayer(0).Text = Val(txtSecLayWeights(0).Text) * (1 / Val(txtNode(0).Text))
txtNetSecondLayer(1).Text = Val(txtSecLayWeights(1).Text) * (1 / Val(txtNode(1).Text))
txtNetSecondLayer(2).Text = Val(txtSecLayWeights(2).Text) * (1 / Val(txtNode(2).Text))
txtNetSecondLayer(3).Text = Val(txtSecLayWeights(6).Text) * Val(txtQuantity(30).Text)
txtSecLayerNode(0).Text = Val(txtNetSecondLayer(3).Text) +
Val(txtNetSecondLayer(0).Text) + Val(txtNetSecondLayer(1).Text) +
Val(txtNetSecondLayer(2).Text)
txtSecLayerNode(0).Text = 1 / (1 + Exp(-(txtSecLayerNode(0).Text)))
'compute sigmoid for node2 of HL
'nets(SecondLayerNode1)
txtNetSecondLayer(5).Text = Val(txtSecLayWeights(3).Text) * (1 / Val(txtNode(0).Text))
txtNetSecondLayer(6).Text = Val(txtSecLayWeights(4).Text) * (1 / Val(txtNode(1).Text))
txtNetSecondLayer(4).Text = Val(txtSecLayWeights(5).Text) * (1 / Val(txtNode(2).Text))
txtNetSecondLayer(7).Text = Val(txtSecLayWeights(7).Text) * Val(txtQuantity(30).Text)
txtSecLayerNode(1).Text = Val(txtNetSecondLayer(7).Text) +
Val(txtNetSecondLayer(4).Text) + Val(txtNetSecondLayer(5).Text) +
Val(txtNetSecondLayer(6).Text)
txtSecLayerNode(1).Text = 1 / (1 + Exp(-(txtSecLayerNode(1).Text)))
'wieght adjustments for hidden layers (first forward sweep)
txtOutput1.Text = Val(txtWeights(112).Text) + Val(txtSecLayerNode(0).Text) *
Val(txtWeights(90).Text) + Val(txtSecLayerNode(1).Text) * Val(txtWeights(91).Text)
txtOutput2.Text = Val(txtWeights(111).Text) + Val(txtSecLayerNode(0).Text) *
Val(txtWeights(93).Text) + Val(txtSecLayerNode(1).Text) * Val(txtWeights(94).Text)
txtOutput3.Text = Val(txtWeights(110).Text) + Val(txtSecLayerNode(0).Text) *
Val(txtWeights(96).Text) + Val(txtSecLayerNode(1).Text) * Val(txtWeights(97).Text)
txtOutput4.Text = Val(txtWeights(109).Text) + Val(txtSecLayerNode(0).Text) *
Val(txtWeights(99).Text) + Val(txtSecLayerNode(1).Text) * Val(txtWeights(100).Text)
txtOutput5.Text = Val(txtWeights(108).Text) + Val(txtSecLayerNode(0).Text) *
Val(txtWeights(102).Text) + Val(txtSecLayerNode(1).Text) * Val(txtWeights(103).Text)
txtOutput1.Text = 1 / (1 + Exp(-(txtOutput1.Text)))
txtOutput2.Text = 1 / (1 + Exp(-(txtOutput2.Text)))
txtOutput3.Text = 1 / (1 + Exp(-(txtOutput3.Text)))
txtOutput4.Text = 1 / (1 + Exp(-(txtOutput4.Text)))
txtOutput5.Text = 1 / (1 + Exp(-(txtOutput5.Text)))
Call counter
End Sub
Private Sub Iterator()
If lblOutput.Caption = "0" Then
Call Iterate
Else
Call IterateTwo
End If
End Sub
Private Sub cmdOpen_Click()
Call flexgdOp
End Sub
Private Sub cmdOrder_Click()
Call Exflux
End Sub
```

```
Private Sub cmdEnd_Click()
Call Ender
End Sub
Private Sub cmdPrint_Click()
Call gridpnt
End Sub
Private Sub cmdSales_Click()
frmInvoice.Show
frmInvoice.WindowState = 0
End Sub
Private Sub cmdSave_Click()
Call FlextoExcel
End Sub
Private Sub Form_Load()
Dim i As Integer
For i = chkItem.LBound To chkItem.UBound
lblCost(i).Caption = ""
lblCost(i).Alignment = 1
lblCost(i).Caption = ""
Next i
txtKi.Text = Format$(Now, "mm/dd/yyyy")
txtTim.Text = Format$(Now, "hh:mm AM/PM")
Set CardOrder = New order
Call prepgrid(frmLastShow.invGrid2)
End Sub

Private Sub Form_Unload(Cancel As Integer)
Set CardOrder = Nothing
End Sub
Private Sub mnuHelp_Click()
frmHelp.Show
End Sub
Private Sub mnuOrderForm_Click()
frmOrder.Show
End Sub
Private Sub mnuPrint_Click()
Call gridpnt
End Sub
Private Sub mnuProduct_Click()
frmProduct.Show
End Sub
Private Sub mnuSave_Click()
Call FlextoExcel
End Sub
Private Sub mnuTips_Click()
frmHelp.Show
End Sub
Private Sub Option136_Click()
If txtNewSypmtom1.Text = "Input New Symptom1" Or txtNewSypmtom1.Text = "" Then
msg = "Please input the new Symptom you have noticed!"
title = "SOSIC Check!"
Ans = MsgBox(msg, vbCritical + vbOKOnly, title)
```

```
txtNewSypmtom1.SetFocus
Exit Sub
Else
lblCost(27).Caption = ".3"
txtQuantity(27).Text = "0.3"
End If
End Sub
Private Sub Option137_Click()
If txtNewSypmtom2.Text = "Input New Symptom2" Or txtNewSypmtom2.Text = "" Then
msg = "Please input the new Symptom you have noticed!"
title = "SOSIC Check!"
Ans = MsgBox(msg, vbCritical + vbOKOnly, title)
txtNewSypmtom2.SetFocus
Exit Sub
Else
lblCost(28).Caption = ".9"
txtQuantity(28).Text = "0.9"
End If
End Sub
‘code to assign variables during fuzzification
Private Sub Option6_Click()
If chkItem(1).Value = 1 Then
lblCost(1).Caption = ".5"
txtQuantity(1).Text = "0.5"
End If
End Sub
Private Sub Option60_Click()
If chkItem(11).Value = 1 Then
lblCost(11).Caption = ".5"
txtQuantity(11).Text = "0.5"
End If
End Sub
Private Sub Option61_Click()
If chkItem(12).Value = 1 Then
lblCost(12).Caption = ".9"
txtQuantity(12).Text = "0.9"
End If
End Sub
Private Sub Option62_Click()
If chkItem(12).Value = 1 Then
lblCost(12).Caption = ".8"
txtQuantity(12).Text = "0.8"
End If
End Sub
Private Sub Option63_Click()
If chkItem(12).Value = 1 Then
lblCost(12).Caption = ".5"
txtQuantity(12).Text = "0.5"
End If
End Sub
Private Sub Option64_Click()
If chkItem(12).Value = 1 Then
```

```
lblCost(12).Caption = ".75"
txtQuantity(12).Text = "0.75"
End If
End Sub
Private Sub Option65_Click()
If chkItem(12).Value = 1 Then
lblCost(12).Caption = ".9"
txtQuantity(12).Text = "0.9"
End If
End Sub
Private Sub Option66_Click()
If chkItem(13).Value = 1 Then
lblCost(13).Caption = "0"
txtQuantity(13).Text = "0"
End If
End Sub
Private Sub Option67_Click()
If chkItem(13).Value = 1 Then
lblCost(13).Caption = ".9"
txtQuantity(13).Text = "0.9"
End If
End Sub
Private Sub Option68_Click()
If chkItem(13).Value = 1 Then
lblCost(13).Caption = ".5"
txtQuantity(13).Text = "0.5"
End If
End Sub
Private Sub Option69_Click()
If chkItem(13).Value = 1 Then
lblCost(13).Caption = ".9"
txtQuantity(13).Text = "0.9"
End If
End Sub
Private Sub Option7_Click()
If chkItem(1).Value = 1 Then
lblCost(1).Caption = ".25"
txtQuantity(1).Text = "0.25"
End If
End Sub
Private Sub Option70_Click()
If chkItem(13).Value = 1 Then
lblCost(13).Caption = ".75"
txtQuantity(13).Text = "0.75"
End If
End Sub
Private Sub Option71_Click()
If chkItem(14).Value = 1 Then
lblCost(14).Caption = ".9"
txtQuantity(14).Text = "0.9"
End If
End Sub
```

```
Private Sub Option72_Click()
If chkItem(14).Value = 1 Then
lblCost(14).Caption = ".5"
txtQuantity(14).Text = "0.5"
End If
End Sub
Private Sub Option73_Click()
If chkItem(14).Value = 1 Then
lblCost(14).Caption = ".75"
txtQuantity(14).Text = "0.75"
End If
End Sub
Private Sub Option74_Click()
If chkItem(14).Value = 1 Then
lblCost(14).Caption = ".9"
txtQuantity(14).Text = "0.9"
End If
End Sub
Private Sub Option75_Click()
If chkItem(14).Value = 1 Then
lblCost(14).Caption = ".8"
txtQuantity(14).Text = "0.8"
End If
End Sub
Private Sub Option76_Click()
If chkItem(17).Value = 1 Then
lblCost(17).Caption = ".75"
txtQuantity(17).Text = "0.75"
End If
End Sub
Private Sub Option77_Click()
If chkItem(17).Value = 1 Then
lblCost(17).Caption = ".9"
txtQuantity(17).Text = "0.9"
End If
End Sub
Private Sub Option78_Click()
If chkItem(17).Value = 1 Then
lblCost(17).Caption = ".1"
txtQuantity(17).Text = "0.1"
End If
End Sub
Private Sub Option79_Click()
If chkItem(17).Value = 1 Then
lblCost(17).Caption = ".9"
txtQuantity(17).Text = "0.9"
End If
End Sub
Private Sub Option8_Click()
If chkItem(1).Value = 1 Then
lblCost(1).Caption = "0"
txtQuantity(1).Text = "0"
```

```
End If
End Sub
Private Sub Option80_Click()
If chkItem(17).Value = 1 Then
lblCost(17).Caption = ".8"
txtQuantity(17).Text = "0.8"
End If
End Sub
Private Sub Option81_Click()
If chkItem(16).Value = 1 Then
lblCost(16).Caption = ".25"
txtQuantity(16).Text = "0.25"
End If
End Sub
Private Sub Option82_Click()
If chkItem(16).Value = 1 Then
lblCost(16).Caption = ".9"
txtQuantity(16).Text = "0.9"
End If
End Sub
Private Sub Option83_Click()
If chkItem(16).Value = 1 Then
lblCost(16).Caption = ".5"
txtQuantity(16).Text = "0.5"
End If
End Sub
Private Sub Option84_Click()
If chkItem(16).Value = 1 Then
lblCost(16).Caption = ".5"
txtQuantity(16).Text = "0.5"
End If
End Sub
Private Sub Option85_Click()
If chkItem(16).Value = 1 Then
lblCost(16).Caption = ".9"
txtQuantity(16).Text = "0.9"
End If
End Sub
Private Sub Option86_Click()
If chkItem(15).Value = 1 Then
lblCost(15).Caption = ".9"
txtQuantity(15).Text = "0.9"
End If
End Sub
Private Sub Option87_Click()
If chkItem(15).Value = 1 Then
lblCost(15).Caption = ".9"
txtQuantity(15).Text = "0.9"
End If
End Sub
Private Sub Option88_Click()
If chkItem(15).Value = 1 Then
```

```
lblCost(15).Caption = ".75"
txtQuantity(15).Text = "0.75"
End If
End Sub
Private Sub Option89_Click()
If chkItem(15).Value = 1 Then
lblCost(15).Caption = ".5"
txtQuantity(15).Text = "0.5"
End If
End Sub
Private Sub Option9_Click()
If chkItem(2).Value = 1 Then
lblCost(2).Caption = ".75"
txtQuantity(2).Text = "0.75"
End If
End Sub
Private Sub Option90_Click()
If chkItem(15).Value = 1 Then
lblCost(15).Caption = ".25"
txtQuantity(15).Text = "0.25"
End If
End Sub
Private Sub Option91_Click()
If chkItem(18).Value = 1 Then
lblCost(18).Caption = ".9"
txtQuantity(18).Text = "0.9"
End If
End Sub
Private Sub Option92_Click()
If chkItem(18).Value = 1 Then
lblCost(18).Caption = ".5"
txtQuantity(18).Text = "0.5"
End If
End Sub
Private Sub Option93_Click()
If chkItem(18).Value = 1 Then
lblCost(18).Caption = "0"
txtQuantity(18).Text = "0"
End If
End Sub
Private Sub Option94_Click()
If chkItem(18).Value = 1 Then
lblCost(18).Caption = ".5"
txtQuantity(18).Text = "0.5"
End If
End Sub
Private Sub Option95_Click()
If chkItem(18).Value = 1 Then
lblCost(18).Caption = ".25"
txtQuantity(18).Text = "0.25"
End If
End Sub
```

```
Private Sub Option96_Click()
If chkItem(20).Value = 1 Then
lblCost(20).Caption = ".9"
txtQuantity(20).Text = "0.9"
End If
End Sub
Private Sub Option97_Click()
If chkItem(20).Value = 1 Then
lblCost(20).Caption = ".75"
txtQuantity(20).Text = "0.75"
End If
End Sub
Private Sub Option98_Click()
If chkItem(20).Value = 1 Then
lblCost(20).Caption = "0"
txtQuantity(20).Text = "0"
End If
End Sub
Private Sub Option99_Click()
If chkItem(20).Value = 1 Then
lblCost(20).Caption = ".8"
txtQuantity(20).Text = "0.8"
End If
End Sub
Private Sub txtQuantity_LostFocus(Index As Integer)
Dim test As String
test = txtQuantity(Index).Text
If txtQuantity(Index).Text = "" Then
chkItem(Index).Value = 0
Exit Sub
End If
If Not IsNumeric(test) And Len(test) <> 0 Then
MsgBox "Please enter a value less than 1, i.e between 0 - 1", 0, "SOSI Clinic Check"
txtQuantity(Index).Text = ""
txtQuantity(Index).SetFocus
Exit Sub
End If
If txtQuantity(Index).Text < 0 Then
MsgBox "Please enter a value greater than 0 but less than 1, i.e between 0 - 1", 0, "SOSI
Clinic Check"
txtQuantity(Index).Text = ""
txtQuantity(Index).SetFocus
Exit Sub
End If
If txtQuantity(Index).Text > 1 Then
MsgBox "Please enter a value less than 1, i.e between 0 - 1", 0, "SOSI Clinic Check"
txtQuantity(Index).Text = ""
txtQuantity(Index).SetFocus
Exit Sub
End If
lblCost(Index).Caption = test
End Sub
```

Appendix I

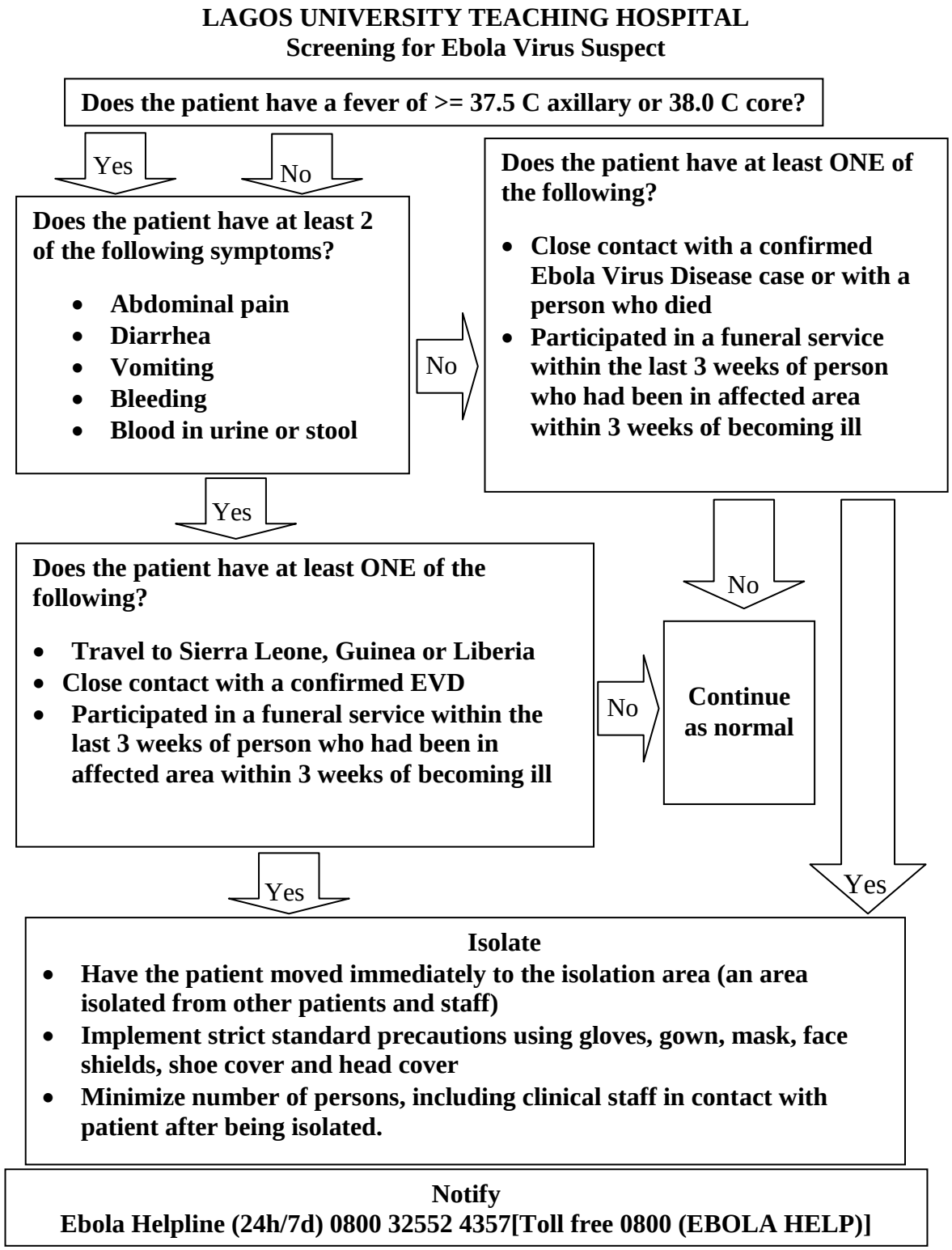


Figure 5.8: Flowchart for Ebola Virus Suspect Screening at the entrance unit of University of Lagos Teaching Hospital for detecting and quarantining EVD patients (obtained on 9th May, 2016).

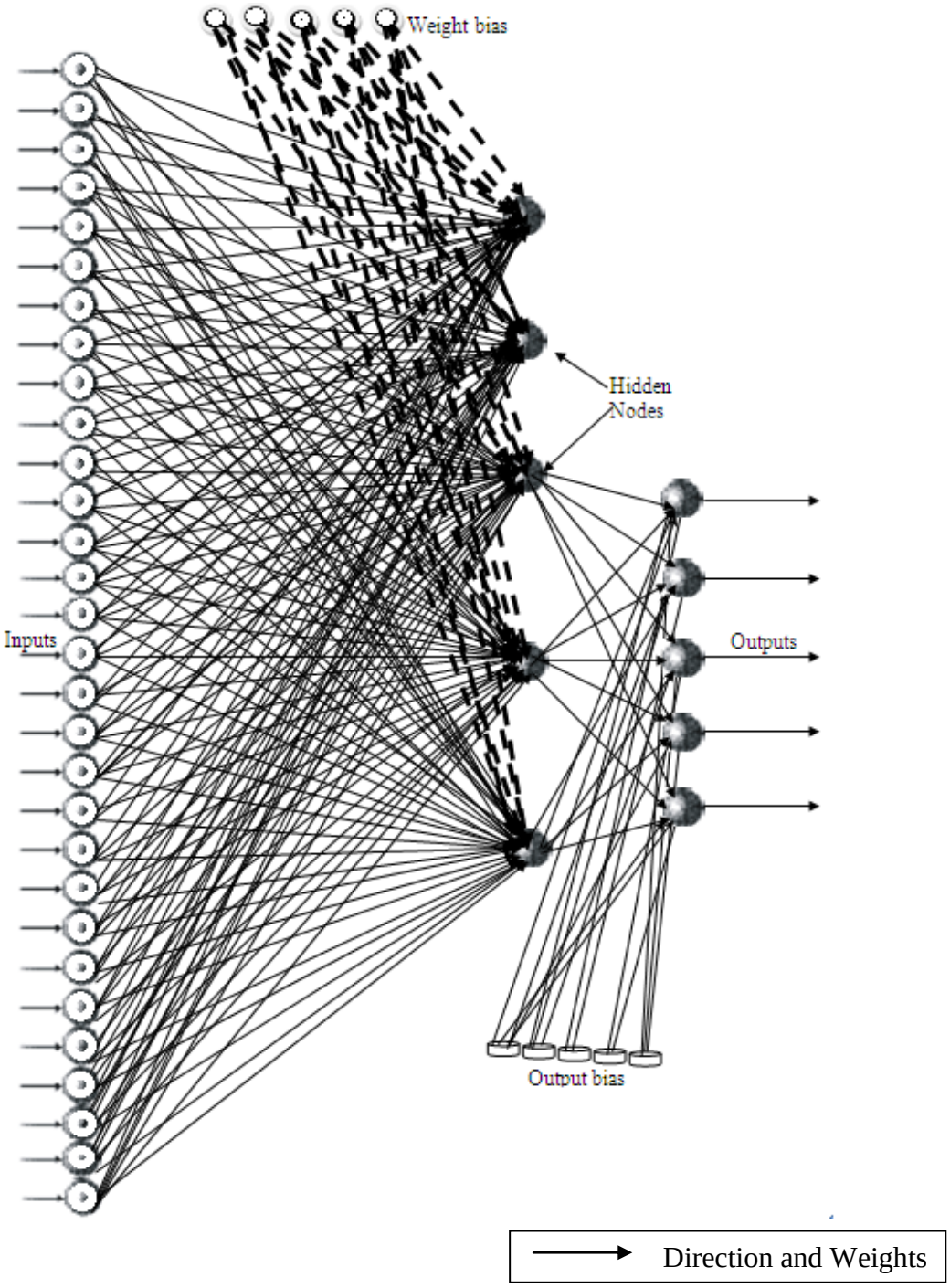


Figure 5.9: Architecture of the connections between input neurons, weights, bias and outputs for Five hidden nodes ANN

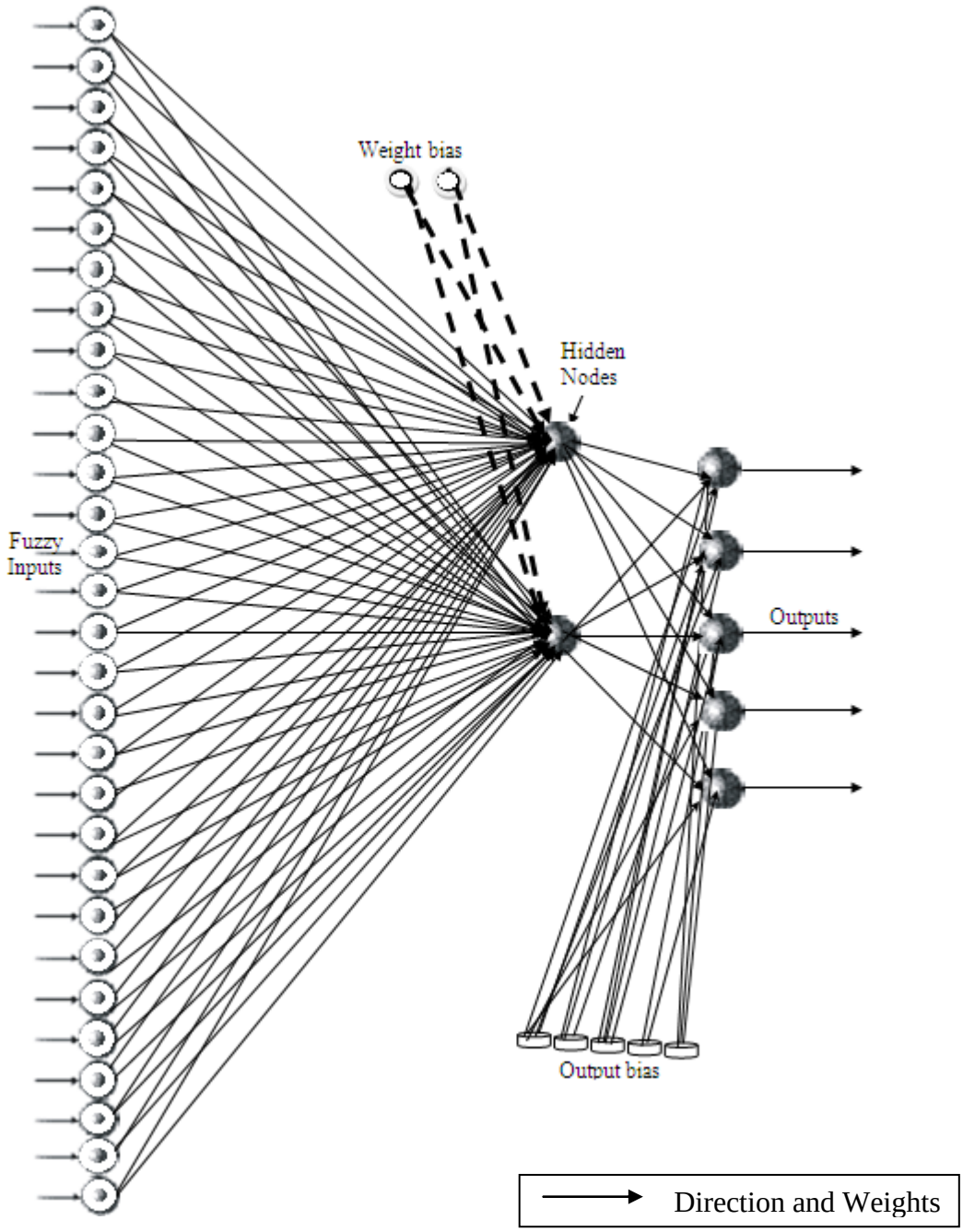


Figure 5.10: Architecture of the connections between input neurons, weights, bias and outputs for Two hidden nodes ANN

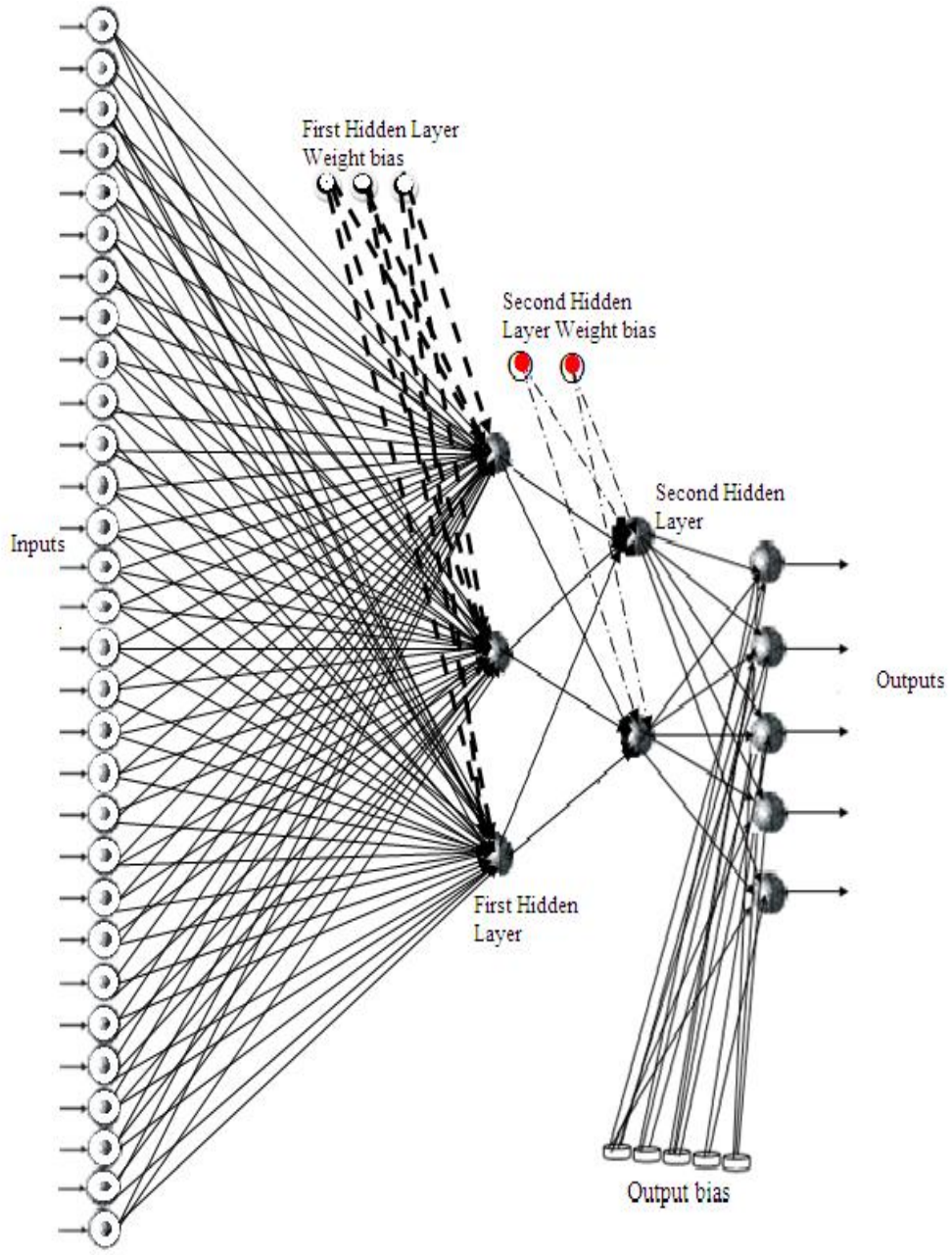


Figure 5.11: Architecture of the connections between input neurons, weights, bias and outputs for a two hidden layer ANN

Appendix M

Table 5.5 Table for the formulation of the 2 x 2 Confusion matrix of the results of ANNs processing. Source: (Wikipedia, 2016).

		Predicted condition			
	Total population	Predicted Condition positive	Predicted Condition negative	Prevalence $= \frac{\Sigma \text{ Condition positive}}{\Sigma \text{ Total population}}$	
True condition	condition positive	**True positive**	**False Negative** (Type II error)	True positive rate (TPR), Sensitivity, Recall $= \frac{\Sigma \text{ True positive}}{\Sigma \text{ Condition positive}}$	False negative rate (FNR), Miss rate $= \frac{\Sigma \text{ False negative}}{\Sigma \text{ Condition positive}}$
	condition negative	**False Positive** (Type I error)	**True negative**	False positive rate (FPR), Fall-out $= \frac{\Sigma \text{ False positive}}{\Sigma \text{ Condition negative}}$	True negative rate (TNR), Specificity (SPC) $= \frac{\Sigma \text{ True negative}}{\Sigma \text{ Condition negative}}$
	Accuracy (ACC) $= \frac{\Sigma \text{ True positive} + \Sigma \text{ True negative}}{\Sigma \text{ Total population}}$	Positive predictive value (PPV), Precision $= \frac{\Sigma \text{ True positive}}{\Sigma \text{ Test outcome positive}}$	False omission rate (FOR) $= \frac{\Sigma \text{ False negative}}{\Sigma \text{ Test outcome negative}}$	Positive likelihood ratio (LR+) $= \frac{\text{TPR}}{\text{FPR}}$	Diagnostic odds ratio (DOR) $= \frac{\text{LR+}}{\text{LR−}}$
		False discovery rate (FDR) $= \frac{\Sigma \text{ False positive}}{\Sigma \text{ Test outcome positive}}$	Negative predictive value (NPV) $= \frac{\Sigma \text{ True negative}}{\Sigma \text{ Test outcome negative}}$	Negative likelihood ratio (LR−) $= \frac{\text{FNR}}{\text{TNR}}$	

Don’t print from here. Guide to defence

5.3 Summary of Achievements

The following accomplishments were achieved as a result of this research:

a. A reduction in the window period for contagious disease spread has been established by the dependence on mathematical principles and ICT tools. Fuzzy logic processing of symptoms of patients has been used as pre-refined inputs to the developed ANN to obtain crispy output of diagnosis.
b. The resulting optimal weights from the ANN training data was used to classify new symptoms and quickly filter out inconsistent data.
c. Appreciable improvement in the speed of processing of using the peculiar slow gradient descent method was achieved when learning rate was set at 0.3.
d. The developed ANN algorithm was successfully validated by comparing the results of training with the different validation/testing data obtained from FETHA in Ebonyi and ISTH in Edo state, Nigeria on Lassa fever and WHO Response Team’s data on EVD presented in Appendix C to E.
e. In this research, the design and development of a fully connected ANN with thirty inputs, three hidden layers and five outputs was achieved.
f. A two hidden layer ANN topology was implemented for deep learning features with the reciprocal function on the second hidden layer iteration processing
g. A way of storing the results of processed sessions and mapping to new diagnosed cases was created using the k-nearest neighbor algorithm and clustering to further refine the results of the designed Expert system.