ADDIS ABEBA UNIVERSITY

FACULTY OF INFORMATICS

DEPARTMENT OF INFORMATION SCIENCE

# APPLICATION OF MULTILAYER FEED FORWARD ARTIFICIAL NEURAL NETWORK PERCEPTRON IN PREDICTION OF COURT CASE'S TIME SPAN: THE CASE OF FEDERAL SUPREME COURTS'

By

ESKINDER MESFIN CHERINET

A THESIS SUBMITTED TO THE SCHOOL OF GRADUATE STUDIES OF ADDIS ABABA UNIVERSITY IN PARTIAL FULFILLMENT OF THE REQUIREMENT OF DEGREE OF MASTER OF SCIENCE IN INFORMATION SCIENCE

2009

# APPLICATION OF MULTILAYER FEED FORWARD ARTIFICIAL NEURAL NETWORK PERCEPTRON IN PREDICTION OF COURT CASE'S TIME SPAN: THE CASE OF FEDERAL SUPREME COURTS'

ESKINDER MESFIN

JANUARY 2009

MASTER OF SCIENCE IN INFORMATION SCIENCE

---

**Principal Advisor: Dr. Rahel B., Addis Ababa University, Department of Information Science**

---

**Associate Advisor: Ato. Yehenew Shiferaw, Addis Ababa University, Department of Information science**

---

**Examiner: Dr. Sebsibe H/Mariam,Addis Ababa University,Department of Computer Science**

---

**Dr. Gashaw kebede, chairman of examining board**

# TABLE OF CONTENTS:

# LIST OF FIGURES

# LIST OF TABLES

## ACRONYMS

ANN Artificial Neural Network

FSC Federal Supreme Courts

FFIC Federal First Instance Court

FHC Federal high Court

FFMP Feed Forward Multilayer Perceptron

CCMS Court Case Management Systems

CCTE Court Case Time Estimations

CBR Case Based Reasoning

MSE Mean Square Error

NMSE Normalized Mean Square Error

R-Value Correlation Coefficient

## ABSTRACT

This research examines and analyzes the use and application of neural networks as a predictive tool. The research was undergone with the assumption to give the Federal Supreme courts in advance estimation of the court case's time span. The significance of the research could possibly benefit a plaintiff and defendants to know their case time length in prior as well the federal courts to perform court room monitoring, ensuring transparency and work efficiency.

A model to address these needs was constructed using a feed forward multilayer neural network perceptron having 9 input neurons to the network and one hidden layer with 20 neurons and finally having a single output neuron, which is the predicted time of the cases in months using MATLAB 7.0 neural network tool box. A selected model was trained with training and validation datasets[67% of the whole datasets], finally tested with the test set reserved for these purpose[33% of the datasets] and a total of more than 33,000 record set was used in building the model.

Based on the performance function, the selected model shows a good performance range of Mean Square error [MSE] which is the difference between the target output and the network output was minimized to fit to the range offering a value of 0.0033 with 94.44% of the error rate was between $\pm$0.2 normalized months. This is the good indication that the developed model could be a reliable predictive model for court cases time span especially for criminal, civil and labor court cases with the assumption that the external factor that affect the court case time span prediction are constant and stable.

Finally when the network is trained with same court case types, the network has show high predictive capability for criminal cases with 95.65% of the data sets residual error minimized between $\pm$ 0.005, 89.54% for civil cases and 91.55% for labor cases. This is the good indication that the developed predictive model can satisfactorily be an alternate choice for predicting court case time span especially court cases related to criminal cases.

.

## ACKNOWLEDGEMENT

I wish to thank Dr. Rahel B. for her support and guidance during the research. I would also like to thank my co-advisor Ato. Yehenew Shiferaw whose advice, knowledge and critical supervision have been invaluable. Special Thanks also goes to Ato. Sintayehu Mitiku, other members of Supreme Court, who give me a constant support during the time of the research.

I would like to thank My brother Dawit and his fiancé, who gave all I need and were with me in all my desperate and hopeful times. My friends: Sisay, Eng.Esayas, Dr.Ephrem, Aynalem, Tessema who tolerate and give me a friendly support for every step of a move.

It would be a shame for me to close up this page without giving my message to my uncles who were a part of my life giving me assistance and guidance during the time of dreadful and unforgettable moments. Finally, these whole thing will not come true without the support of my Mother and Father and of course, Nun Fana [May U rest in Peace]-take this, it is your dedication.

Last Acknowledgment of the page goes to my classmates who were with me for the last two years and have such a fun and sociable times- LOVE YOU ALL.

# CHAPTER ONE: INTRODUCTION

## 1.0 Background

Federal Courts are judicial organs that are established by the constitution. As governmental power is divided in to federal and regional administrative structure, so is jurisdiction of courts divided into regional structures. This means the constitution of the Ethiopian Federal Democratic Republic has facilitated a condition in which both the federal and regional courts perform their judicial activity side by side (Federal Supreme Court, 2007).

According to the proclamation number 25/88 enacted based on the constitution, Federal Courts are established in three layered courts .These are:-The Federal First instance court, the Federal High court and the Federal supreme court.

**Federal First Instance Court [FFIC]** - has a first instance jurisdiction over civil and criminal matters with estimates up to 500,000 birr and other cases that cannot be estimated in terms of money.

**Federal High Court [FHC]**-has a first instance jurisdiction over criminal, tax and labor cases over 500,000 birr claim and entertains appeal from the federal first instance court.

**Federal Supreme Court[FSC]**- is the highest judicial organ which has a jurisdiction to see criminal cases in which government officials are liable; first instance cases; appeal from the high court; final decisions which have basic flaws in the interpretation of law, in its cassation bench[Federal Supreme Court, 2007].

In the three tiers of courts, quite a number of first instance, appellate and classification cases are entertained each year and it is estimated that a total of more than 200,000 unprocessed records are there in the database in an electronic format.

Courts are one of the various areas, where critical data about each case are recorded and kept even after the cases are closed. It has been seen that cases' data, especially in some Federal and regional courts where the number of cases is massive, did not get enough attention to use these recorded cases as a basis for decision-making, performance monitoring and future planning. Identifying and knowing a given pattern of data in a given cases will help the decision makers in deciding up on the specific future activities.

## 1.1 Statement of the problem and justification of the study

These days it is a clear fact that, according to the opinions of legal experts, defendants and plaintiff that court cases do not have a definite time estimate after which the court's issue will be closed. Most court cases are extended year after year without being closed. Such cases occupy most of the courts resources like court benches, human resources, and financial assets for imprecise period of time for we don't have an apparent estimate for monitoring. When these cases will be congested which as result, create reduction on court performances and hindrance in court monitoring and evaluations.

Consequently, the contributory research problem that necessitated this research was the lack of a clear estimate of the time spans/period of court cases which was really affecting the resource allocation and scheduling for courts; such as allocation of court rooms, judges and lawyers,

efficiency; and effectiveness of courts; and as well as create a frustration of the plaintiff and defendants.

The Federal Court implemented a court case management system [CCMS] which could predict the time standard or time span for cases based on the average time of akin cases in the database. Besides, judges and legal experts gather around to decide on the time standard for pending and active court cases. However, such a prediction was not scientifically verified and supported since it was based on general average estimate and the manual system predictions could not put forward the right and expected time estimates over Cases which all these were affecting the resource allocation, in general, for the manipulation of the cases. Accordingly, Federal court requires a model that could predict the time standard for pending/active cases through an implementation of learning mechanism based on previously decided Cases.

Thus in order to plan and implement effectual strategies in helping the decision makers and planners in allocating these financial, human resources and increase the performance monitoring of the court, there should be a need for actionable information which is obviously a result of these research work.

So far no attempt made at the academic level in identifying the time estimate for the active and pending cases in courts to assist decision making and planning in prior and for plaintiff and defendants to know the estimated period of their cases. This research work will be groundwork for the prediction of the time for active and pending Cases based on the pattern of the historic data/cases. Moreover, it will also be an input for other researches in the same area.

### 1.1.1 The Research Question

The following research questions allow the research to meet the objectives proposed:

- Can neural networks accurately forecast Court cases time span?
- What variables are most important for the court case time predictions?
- Can a Neural Network be used by FSC as a Predictive model for court Case time span?
- How Competent a Neural Network predictive model is with that of the Manual based One?

## 1.2 Objective of the study

The general and specific objectives of the proposed study are the following: -

### 1.2.1 General objectives

The general objective of this study centered on exploring the potential applicability of Artificial Neural Network in developing a learning and predictive model that can Pridict the court case time span for both active and pending cases.Generally,The Research centered on developing a model that can make the prediction of time length that the court case would have taken starting from the date the cases brought to the court up to the date the decision on the cases be passed by judges.

### 1.2.2 Specific objectives

In order to achieve the general objective indicated above, the research will have the following specific objectives

- To study the nature and techniques of Artificial neural network in prediction

- To study the current predictive techniques in Federal Supreme Courts'
- To Select and extract the data set required for prediction of the time standard for each active cases
- To perform data making or preparing the data for analysis which includes handling inconsistent and noisy data, accounting for missing values, deriving other fields from existing ones, transformation and integration.
- To Build and test models using the selected tools and technologies.
- To Report results and finding of the research work

## 1.3 Research methodology

The following methods were employed in conducting the proposed study.

### 1.3.1 Literature review

Since having a thorough understanding about a given subject and clearly noticed its application was one of the critical issues to consider, further, literature review was made to get more insight to the concept of the predictive applicability of Artificial Neural Network [ANN], current predictive techniques in Federal Supreme Courts. This was also include having multidimensional view in understanding the data set under investigation through detail and intensive discussion with staff and management of the Federal Supreme Court under study.

### 1.3.2 Data collection

The major task at this stage was to identifying and collecting the relevant dataset for the purpose of the research work at hand. Since the specific steps and tasks at this stage were to determine the final result of the research work, much effort was made in creating the right dataset.

The data on cases at the Federal Supreme Court Office were not combined and kept in a data warehouse. Hence it is necessary to collect data from different sources namely databases, merge relevant attribute from several tables.

The raw data that was used on this research work was a data kept on each of the active/pending and closed cases. The datasets that was used in the research comprises of originally 78 attributes and more than 35,000 records. The attributes are mostly the case information, the defendants and plaintiff details, the opening and the closing date of the cases and several other attributes which was preprocessed in the preprocessing stage of the research works.

### 1.3.4 Data Preprocessing

This is a step where the collected data was arranged into a form that was suitable for the particular Neural Network algorithms for prediction of the time. In other words it was just to prepare dataset for analysis by collecting it to a new database. At this stage pre-processing tasks like handling noisy data, unknown values, missing values, driving new fields from the existing ones and summarization of data was performed by taking into account the selected tool and techniques. This was helpful to effectively apply data mining tools and Neural Network algorithms on the data.

### 1.3.4 Tool Selection

Deferent tools (software and algorithms) supporting these predictive task such as MATLAB, Knowledge studio, Weika Knowledge analysis tools were compared using their capacity to handle such a huge data, convenience, availability and the data visualization of the tool was taken into account then, MATLAB 7.0 Neural network tool box, was selected. In additions, MATLAB

was used in several research papers that the researcher reviews. Using such a tool and selected error back propagation algorithm, *trainlm*, models was built and trained.

After building the model, the next action to be taken was evaluating or assessing and interpreting the results of the selected model. This step was helpful in selecting the best model that finds an interesting pattern. While evaluating a model, its accuracy, the types of errors was taken into consideration.

Finally based on assessment and evaluation of such models, the best model that found an interesting pattern was selected. Furthermore this provided a good ground for result interpretation and recommendation.

## 1.4 Scope of the study

The scope of the research was limited to assessing the potential application of Artificial Neural Network, in supporting the prediction of time span for court cases. The research was majorly performing prediction for the time span of court cases, especial for the active and pending cases.

## 1.5 Organization of Study

The remaining portion of the thesis is broken up into the following chapters: Literature Review and Theoretical Framework, Experimentation and Data Analysis, and Conclusions and Recommendations.

Within the literature review [chapter one], pertinent literature is reviewed and the history and theory of Back propagation neural networks is discussed. Moreover, in these chapters an overall understanding on the data set, variable types and other statistical methodology for the research

was reviewed. The Research work description [Chapter two] describes the different methodologies for addressing the prediction requirement and the current methodology for prediction of court case time spans. Experimentation and Data Analysis [Chapter Three] compiles pertinent tables and charts collected during the research. Data analysis follows with a statistical and graphical review of the information presented. The thesis closes with the conclusions and recommendations [Chapter Five].

# CHAPTER TWO: LITERATURE REVIEW AND THEORETICAL FRAMEWORK

## 2.0 Artificial neural network

Artificial neural network represent a technology that is rooted in many disciplines: neuroscience, mathematics, statistics computer science and engineering. Neural network find application in such diverse fields as modeling, time serious analysis, pattern recognitions, signal processing and control by virtue of an important property; the ability to learn from input data with or without a teacher (Hykin, 1999). Neural Network and its varieties are discussed briefly in following sections.

### 2.0.1 Fundamentals of Neural Network

Artificial neural network is a system that is artificial model of human brain. It can also be referred to as an adaptive statistical model on an analogy with the structure of the brain. Its adaptive behavior comes from the way it can learn to estimate the parameters of some population using a small number of examples at a time.

Neural networks can be considered as a recent development but enjoys a resurgence of interest by many researchers in the field of computer engineering and artificial intelligence. The interest in neural network comes from its potential to find a solution to various problems of application domains. It has been used in large number of application and has proven to be effective in performing complex function in a variety of fields. These include pattern recognition, classification, vision, control system and predictions [Fauset, L, 1994].

Various literatures have given different definitions to neural networks, but one used by Hykin, 1999, will be adopted here:

*"A neural network is a massively parallel distributed processor made up of simple processing units. Which has a natural property for storing experimental knowledge and making it available for use, it resemble human brain in two respects:*

1. *Knowledge is acquired by the network from its environment through a learning process.*
2. *Interneuron connection strength known as synaptic weights, are used to store the acquired knowledge."*

#### 2.0.1.1 The Human brain

The human brain is estimated to contain about $10^{11}$ interconnected neuron. Each neuron is nearly connected to $10^4$ other neurons. The neural model comprises the basic structure of the human brain. The neuron sends out spikes of electricity activity through a long, thin strand known as **axon,** which split into thousands of branches. At the end of each branch, a structure called a **synapse** converts the activities from the axon into electrical effects that inhibits or excites activity in the connected neuron. When a Neuron receives excitatory input that is sufficiently large compared with its inhibitory input, it ends a spike of electrical activity down its axon. Learning occurs by changing the effectiveness of the synapse so that the influence of one neuron on another changes.

The biological neuron model is the foundation of an artificial neuron. It simulates the four basic components of the natural neuron: **Dendrite, soma, axon, and synapses**.

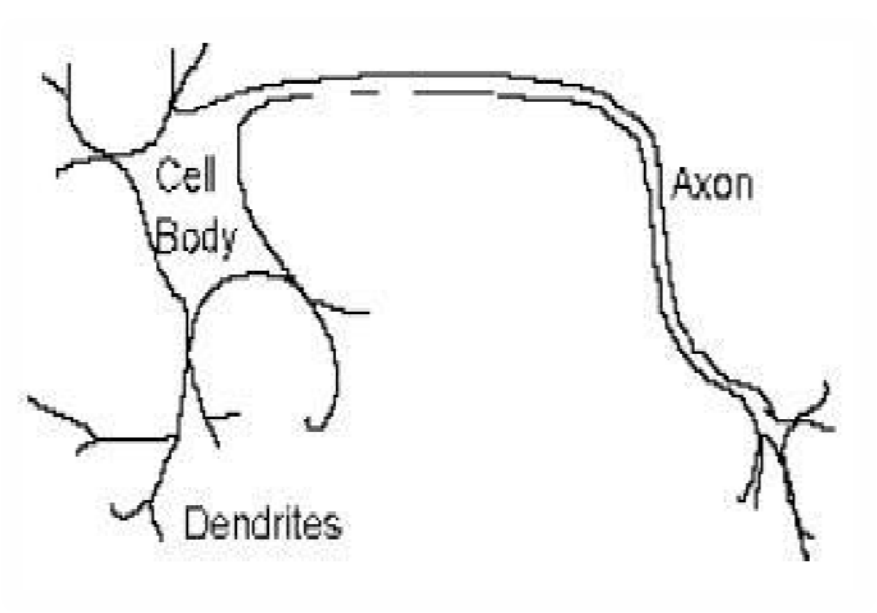


**Figure 2-1: The biological Neuron Model**

### 2.0.1.2 Artificial Neuron

In the neural network literature, the unit analogous to the biological neuron is referred to as a processing elements [**PE**] or units. A processing element has many input path and combines, usually by a simple summation, the values of these input paths.

When creating a functional model of the biological neuron, there are three basic components of importance. First, the synapses of the neuron are modeled as weights. The strength of the connection between an input and a neuron is noted by the value of the weight. Negative weight values reflect inhibitory connections, while positive values designate excitatory connections (Hykin, 1999). The next two components model the actual activity within the neuron cell. An adder sums up all the inputs modified by their respective weights. This activity is referred to as linear combination. Finally, an activation function controls the amplitude of the output of the neuron. An acceptable range of output is usually between 0 and 1, or -1 and 1.

This process is described in the figure below

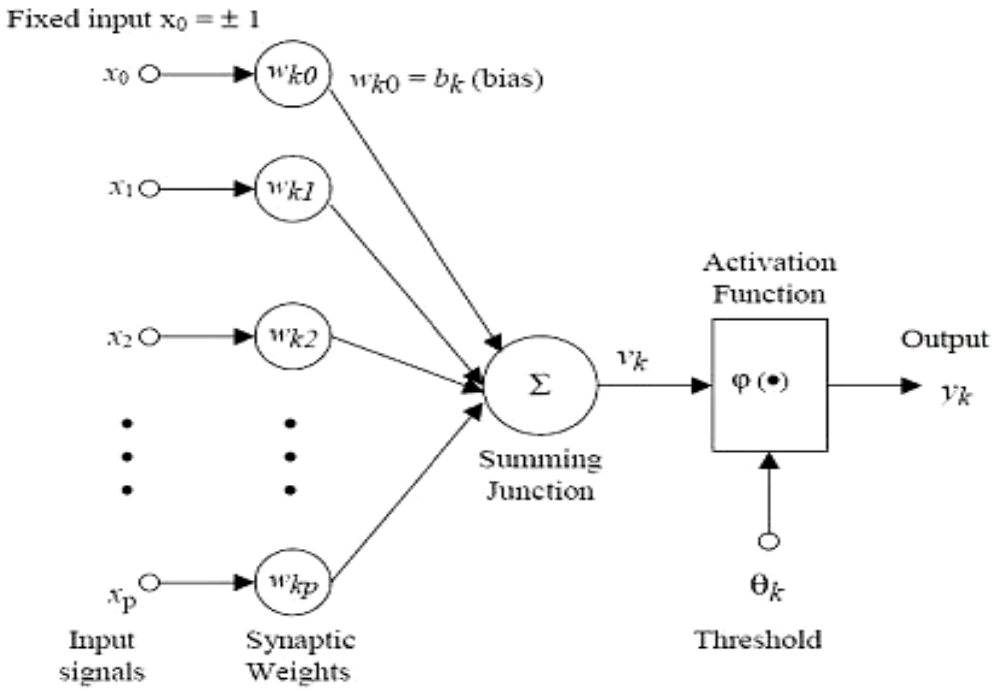


**Figure 2-2: Artificial neural network Model**

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

$$v_k = \sum_{j=1}^{p} w_{kj} x_j$$

The output of the neuron, $V_k$, would therefore be the outcome of some activation function on the value of the weight and the input values.

## 2.1 Component of Neural Network

According to Anderson and McNeil (1992), the seven major components that make up an artificial neural network are described below. These components are valid whether the neuron is used for input, output or is in one of the hidden layers.

### 2.1.1 Weighting Factors

A neuron usually receives simultaneous inputs. Each inputs has its own relative weight, which gives the input the impact that it needs on the processing element's summation function. These weights perform the same type of function similar to the synaptic strength of the biological neuron.

Weights are adaptive coefficient within the network that determines the intensity of the input signal in the artificial neuron. They are the measure of an input's connection strength. This strength can be modified in response to various training sets and through its learning rules.

### 2.1.2 Summation function

The first step in processing element's operation is to compute the weighted sum of all of the inputs. Mathematically, the inputs and the corresponding weight are vectors which can be represented as ($X_1$,$X_2$,……..$X_m$) and ($W_1$,$W_2$…….$W_m$).The total input signal is the dot, or inner product of these two vectors .The result is a single number, not a multi-element vector.

The summation function can be more complex that just the simple input and weight sum of product. The input and weight coefficients can be combined in many different ways before passing onto the transfer function. The specific algorithm for combining neural input is determined by the chosen network architecture and paradigm.

### 2.1.3 Transfer Function

The result of the summation function is transferred to a working output through a logarithmic process known as the transfer function. In transfer function, the sum total can be compared with some threshold to determine the neural input. If the sum is greater than the threshold value, the processing elements generate a signal. If the sum is less than the threshold, no signal is generated.

### 2.1.4 Scaling and Limiting

After the processing element's transfer function, the result can pass through additional processing such as scaling and limiting. This scaling simply multiplies a scale factor times the transfer value, and then adds an offset. Limiting is the mechanism, which insures that the scaled result doesn't exceed an upper or lower bound. This limiting is in addition to the hard limits that the original transfer function may have performed. This type of scaling and limiting is mainly used in topologies to test biological neuron models.

### 2.1.5 Output Function

Each processing element is allowed to output one signal, which it may send to hundreds of other neurons. This is just like the biological neuron, where there are many inputs and only one output. Normally, the output is directly equivalent to the transfer function's result. Some network topologies, however, modify the transfer result to incorporate competition among neighboring processing elements. Neurons are allowed to compete with each other, inhibiting processing elements unless they have great strength. Competition can occur at one or both of two levels. First, competition determines which artificial neuron will be active, or provide an output. Second, competitive inputs help to determine which processing elements will participate in the learning or adaptation process.

### 2.1.6 Error function and Back-propagated value

In most learning networks, the difference between the current output and the desired output is calculated .This raw error is then transferred by the error function to match the particular network architecture. The artificial neuron's error is then typically propagated into learning functions of another processing element. This error term is sometimes called the current error.

The current error is typically propagated backward to a previous layer. Yet, this back propagated value can be either the current error, the current error scaled in some manner or some desired output depending on the network type. Normally, this back-propagated value, after being scaled by the learning function, is multiplied against each of the incoming connection weight to modify them before the next learning cycle.

### 2.1.7 Learning functions

The purpose of the learning function is to modify the variable connection weights on the inputs of each processing elements according to some neural based algorithms. This process of changing the weights of the input connections to achieve some desired result could also be called adaptation functions, as well as learning mode.

## 2.2 Type of Activation Function

The activation function Φ (.) performs a mathematical operation on the signal output. As mentioned earlier, the activation function acts as a squashing function, such that the output of a neuron in a neural network is between certain values (usually 0 and 1, or -1 and 1). In general, there are three types of activation functions, denoted by Φ (.).

First, there is the Threshold Function, also known as Heaviside function, which takes on a value of 0 if the summed input is less than a certain threshold value (v) and the value 1 if the summed input is greater than or equal to the threshold value.

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

Thirdly, there is the sigmoid function. This function can range between 0 and 1, but it is also sometimes useful to use the -1 to 1 range. An example of the sigmoid function is the hyperbolic tangent function. The sigmoid function is strictly continuous function, which can be differentiable. This property makes it appropriate for use in neural network trained by back-propagation.

Some more types of activation function are also used in different areas of application of ANN. For continuous valued targeted with a bound range, the sigmoid function are preferable, provided that either the output or the target to be scaled to the range of the output activation function (Sarles, 1997).

## 2.3 The network Architecture

The principal importance of a neural network lies on their interconnections. Artificial neural networks have simple structures and are designed to mimic the function of the biological neurons. The natural connection of a human brain is too complex to be implemented directly in neural network. The structures, therefore studied so far are simple and easy to be implemented in digital computers.

Neural network involves long training times and are therefore more suitable for application where this is feasible. They require a number of parameters that are typically best determined empirically, such as the network topology or "structure". Neural network has been criticized for their poor interpretability, since it is difficult for human to interpret the symbolic meaning behind the learned weight (Micheline, 2001).

Advantage of neural network, however, includes their high tolerance to noisy data as well as their ability to classify pattern on which they have not been trained. In addition, several algorithms have recently been developed for extraction of rules from trained neural networks.

Architecture of a neural network is the specific arrangement and connection of the neurons that are organized in the form of layers, making up the network. The architecture is defined by a number of layers, the number of units per layers, and the interconnection pattern between layers. In general, Multilayer neural network has a similar structure of topology, basically, these are three types of layers each consisting of a group of neurons.

- The input layers consist of neurons that receive input from the external environment
- The output layers consist of neurons that communicate the output of the system to the user or external environment
- One or more hidden layers between the output and input layers

When the input layer receive the input, its neurons produce output; this become input to the next layer of the system .The manner in which the neural network are structured is internally linked with the learning algorithm used to train the network (Hykin, 1999).

### 2.3.1 Feed Forward Networks

A feed forward network is a network where the neurons on the $i^{th}$ layer send their output to the neurons on the $(i+1)^{th}$ layer, but they do not receive an input back from the neurons of the $(i+1)^{th}$ second layer.

In other words the network is strictly feed forward. A network with only the input and output layer is called single-layered network.

Multilayer feed forward neural network is a network wise at least one hidden layer in addition to its input and output layers. The multilayer feed forward networks or the multilayer perceptron, are generalization of the single layer perceptron. It includes one or more hidden layers each with their respective number of neurons. The function of the hidden neurons is to intervene between the external input and the network output in some useful manner.

### 2.3.2 Recurrent networks

A Recurrent networks, which is sometimes referred to as feedback networks can have a signal traveling in both direction by introducing loops in the network. There may exist one or more such loops. In these networks a neuron of layer after receiving input from another layer, send its output back to the previous layer neurons. Recurrent network are said to be dynamic in that their state is changing continuously until they reach an equilibrium point. In some cases the activation values of the unit's undergo a relaxation process such that the network will evolve to a stable state in which activation does not change further. In other application in which the dynamic behavior constitutes the output of the network, the changes of the activation values of the output units are significant (Perlovsky, 2001).

## 2.4 The learning process

The connection between neurons in a neural network is given adjustable weights or measure of importance. The process of adjusting the weight to make the network learn the relationship between the input and target is called learning. To be more precise a definition of learning that is adapted from Mendal and McClaren [1970] is given by [Hykin, S, 1999] as follows:

> *"Learning is a process by which the free parameters of a neural network are adopted through a process of stimulation by the environment in which the network is embedded. The type of learning is determined by the manner in which the parameters changes take place."*

The process of making the network to learn the solution to a problem follows a set of well defined rules called the learning algorithms. Many learning algorithms have been invented to help find an optimum set of weight that result in the solution of the problem. In general, all the learning methods of an adaptive network can be classified into two major categories.

### 2.4.1 Supervised learning

Most commonly, neural network use supervised training; here the network is trained by providing it with inputs and desired outputs. These input-output pairs are provided by an external teacher or by the system containing the network. For this reason it is also called learning with a teacher. The actual output of the network may or may not match the desired output, depending on the weight at the particular moment. In all cases the training algorithms modifies the weights with the idea that the next time the network sees the same input pattern, it will more closely reproduce the target output pattern as its actual output. The supervised learning model can be depicted in the following figure:

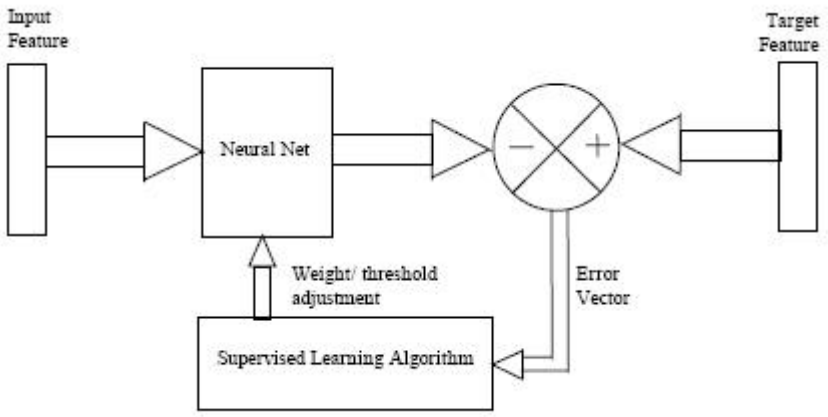


**Figure 2-3: Supervised Learning Model**

### 2.4.2 Unsupervised learning

In unsupervised learning, the network is not trained towards specific outputs. Instead, the net seek to find pattern or regularity in the input data. It is self-organizing learning process that does not require external teacher. Once the network has become tuned to the statistical regularities of the input data; it develops the ability to form internal representation for encoding feature of the input and thereby to create new classes automatically (Becker, 1991).

## 2.5 Learning rate

It is the rate at which the artificial neural networks learn and it depends on the several controllable factors. In selecting the approach, there are many tradeoffs to consider. Obviously; a slower rate means, a lot more time is spent in accomplishing the off-line learning to produce an adequately trained system. With the higher learning rates, however, the network may overshoot the targeted values.

Generally, several factors besides time have to be considered when discussing the offline training and testing. Network complexity, size, architecture, type of learning rule or rules employed and

desired accuracy must be all considered. These factors play a significant role in determining how long it will take to train a network.

Usually learning rate is positive and between zero and one. If the learning rate is greater than one, it is easy for learning algorithms to overshoot in correcting the weights and the network will oscillate. Small values of the learning rate will not correct the current error quickly, but if small steps are taken in correcting errors, there is a good chance of arriving at the best minimum convergence.

## 2.6 Multilayer Feed forward neural networks

Multilayer feed forward neural network or multilayer perceptron network [MLP] is the most popular neural network type. It has been applied successfully to solve difficult and diverse problems by training them in supervised manner with a highly popular algorithm known as *error back-propagation algorithm*.

The typical network has an input layer, at least hidden layer and an input layer. There is no limit on the number of hidden layer but most application use just one or two.

### 2.6.1 Back-propagation

Training a neural network is the process of setting the weight on the inputs of each of the units in such a way that the network best approximates the underlying function. Back-propagation is the most commonly used method for training multilayer feed forward neural network. This training scheme is used for adjusting the connection weight of each unit in such a way that the error between the desired output and the actual output is reduced. More clearly, the neural network adjust the connection weights to each unit, beginning with the connection between the last hidden

layer and the output layer. After the network has made adjustment to this set of connection, it calculates error values for the next pervious layer and makes adjustments. This scheme was developed independently by Webros, Parker, Rumehart, Hinton and Williams (Anderson, D., and McNell, G., 1992).

The role of a training algorithm is to set the network's weight and threshold so as to minimize predictive error by the network. The historical cases gathered are used to automatically adjust the weights and thresholds in order to minimize this error. This process is equivalent to fitting the model represented by the network to the training data available. The error of a particular configuration of the network can be determined by running all the training cases through network, comparing the actual output of generated with the desired or targeted output. The difference is combined together by an error function to give the network error. The most common error function is the sum squared error, where individual error of output on each case squared and summed together (stat soft, 2003).

The error back-propagation method can be applied to any feed forward network with differentiable activation function. The requirement of differentiability of the activation function is based on the following reason; as mentioned before, the difference between the response of an output unit and expected response is the error made by the network. The unit of the output layer uses this error directly to correct their connection weights, but this is not the same for the units of the hidden layer since they are not in direct contact with the error. As a result the unit of hidden layers needs to estimate their error using error back-propagation method. The amount of error made by the network is first converted into an error signal that is proportional to the rate of change of the nonlinear activation function. In fact, this implies that we want the correction to be proportional to the rate of change of the activation function. From the theory of calculus, the

easiest way to fulfill this requirement is to make the correction, which justifies the need for differentiability of the activation function.

The ultimate purpose of error correction learning is to minimize the cost function based on the error signals, such that the actual response of each output neuron in the network approaches the target response for that neuron in some statistical sense. Indeed, once the cost function is selected, error connection learning is strictly an optimization problem to which the usual tools may be brought to bear (Haykin, 1999).

## 2.7 Generalization, over fitting and stopping criteria

Several techniques are available to address the problem of over fitting; One of the most successful methods for overcoming the over fitting problem is to simply provide a set of validation data to the algorithms in addition to the training data (Mitchell, M., 1997). The generalization error will be then estimated. The most commonly used method for estimating generalization error in neural network is split simple or hold-out validation.

A test set is reserved from the training dataset and after training the network it will be run on the test set. The error on the test set provides an unbiased estimate of the generalization error. An improvement of the hold out validation is the cross-validation that allows using all of the data for training, i.e. in a K-fold cross validation, the data is divided into K subsets of equal size. The network will be trained k times, each time leaving out one of the subsets from training for estimating the generalization error and the result are then averaged.

In the above methods of estimating generalization error, the training session is stopped periodically and the network is tested on the validation subset after each period of training .The

validation error rate decrease monotonically to a minimum and starts to increase as the training goes on. Early stopping point is the point at which the validation error rate is minimum. Care should be taken in choosing the minimum point. The minimum reached by the network might not be the global but rather an inflection in the slope after which the error goes toward another minimum. Using lot of hidden units will avoid getting trapped in the local minimum.

## 2.8 Application of neural networks

Neural network are good choices for most classification and prediction tasks when the result of the model are more important than understanding how the model works. Neural network actually represent complex mathematical equation, with lots of summation, exponential function and many parameters. Neural network do not work well if there are many hundreds or thousands of input feature. Large numbers of feature make it more difficult for the network to find patterns and can result in long training phases that never converge to a good solution (Berry and Linoff, 2000).

Neural network have a broad applicability to real world business problems. They have already been successfully applied in many industries. Since neural network are best at identifying patterns or trends in data, they are well suited for prediction, estimation need including: sales forecasting, customer research, data validation, risk management etc… (Stergion and Siganos,1996).

## 2.9 Data Preparation Analysis and Processing

One of the most important components in the success of any neural network solution is the data. The quality, availability and relevance of the data used to develop and run the system are critical

to its success. If inputs has been processed in such a way that it clearly reveals the important information, the task in building a model can be said to go through half way with success. On the other hand, if the necessary input information is complex and confusing, the task of building the model will be difficult or even unsuccessful.

Data processing starts from the data collection and analysis followed by preprocessing and then fed to the neural network. Finally, post-processing is needed to transform the inputs of the network to the required outputs, if necessary. The following section discusses the most important consideration involved in the process.

## 2.9.1 Types of variables

### 2.9.1.1 Categorical variables

Categorical variable do not have a natural ordering and do not have relationships like "greater than" or "less than ". Some of them came from inputs values that do not have numeric values but have to transform to numeric values as an input variables.

According to Berry(2000),Categorical variables can be presented to the network with the *l-of-c* encoding scheme, which has many units as there are values that the variables can take on. Exactly one of the units will be turned on according to the values of the variable, and all the other unit will be turned off. Another way to encode categorical variable is to represent all the possible values to one continuous input variable. The latter case imposes an artificial ordering on the data that does not exist. But for variables with large number of categories, this can dramatically decrease the number of inputs (Micheline, 2001).

#### 2.9.1.2 Ordinal variables

Ordinal variables have a natural ordering; such data can simply be transformed directly in to corresponding values of continuous variable with or without scaling.

### 2.9.2 Data collections

The data collection step comprises of the following tasks:

#### 2.9.2.1 Identifying the data requirement

The first thing to do when planning data collection is to decide what data will be needed to solve the problem. In general, it will be necessary to obtain the assistance of some experts in the field or go through various literatures in the problem area, one need to know:

A. What data are definitely relevant to the problem
B. What data may be relevant
C. What data are collateral? Both relevant and possibly relevant data should be considered as input to the applications

#### 2.9.2.2 Identifying the data source

The next step is to decide from where the data will be obtained. There are two types of data sources:

A. Primary data source obtained from the experiment or through observation and are not published
B. Secondary data source, where the data are available in the form of document, research results etc….

Making use of secondary data source require being consistent, accurate, and applicable. Known sources of data will enable to alleviate difficulties that may arise during data analysis by consulting the expertise available in the source area.

#### 2.9.2.3 Determining the input data quantity

A reasonable estimation of the data required in developing the neural network model is an important task to be considered at large. The amount of data used in training process determines the Neural Network to learn different properties of the data. If the data collected is limited, it may not reflect the full range of properties that the network should be learning and this will limit its performance with unseen data. On the other hand, it is possible to introduce unnecessary expense by collecting too much data. In general, the quantity of data required is governed by the number of training cases that will be needed to ensure the network perform adequately. The intrinsic dimensionality of the data and the required resolution are the main factors determining the number of training cases and therefore, the quantity of the data required.

### 2.9.3 Data Pre-Processing

Today's real world databases are highly susceptible to noisy ,missing and inconsistent data due to the various reasons such as attribute of interest may not always be available, relevant data may not be recorded due to misunderstanding of the subject under consideration, instrument used may be faulty,etc…

There are numbers of data preprocessing techniques; Data cleaning, data integration, data transformation and data reduction are the most commonly used techniques that help to improve the overall quality of the data.

Data cleaning helps to remove the noisy data and correct inconsistencies in the data. Noisy data can be smoothed by the binary method, clustering or regression methods. Some data inconsistencies can be corrected manually by consulting the domain experts. Data integration usually applied to data from multiple sources in order to match up entities having similar properties. The other task in the preprocessing step is the data transformation technique. It is used to transform the data into a form appropriate for analysis. One of the widely used methods of transformation is normalization, which helps to scale the data in to specific ranges.

In the design process of the neural network, the data pre-processing task is vital. Before data's are ready to be used as input to a neural network, the data should be pre-processed in order to make the network process more manageable. There are two basic techniques, which can be used to help us understand the data.

### 2.9.4 Statistical analysis

Neural networks can be regarded as an extension of standard statistical techniques. Most of the above methods are the variant of the statistical techniques, and so such tests can give us an idea of the performance the network is likely to achieve. Some statistical techniques can give useful clues about appropriateness of the data.

#### 2.9.4.1 Data visualization

The distribution of the data can be easily seen when data is presented in a suitable format. Graphical description of data sets is one of the statistical methods used in various applications. Distinguishing feature as valleys and peaks, which characterized the data, can be spotted easily be plotted in a graph.

The combination of visualization and statistical analysis give good results in the pre-processing step. Visualization gives an appraisal of the data, and ideas about the underlying patterns, while statistical analysis enables to test these ideas. Preprocessing, also frequently means partitioning the input space.

Reducing the dimensionality of the input data and minimizing the number of inputs to the network is another task to be considered during data preprocessing. The process of reducing the input data set is called feature extraction. Prior knowledge about the problem area can simplify the selection process which was done in consultation of the domain experts.

### 2.9.5 Data post-processing

Post processing covers any process that is applied to the output of the network. As with pre-processing, it is entirely dependent on the application, and it may include detecting when a parameter exceeds an acceptable range, sometimes it is, just the reverse process of data pre-processing.

# CHAPTER THREE: RESEARCH WORK DESCRIPTIONS

## 3.0 Court case time estimation

Cases are the major inputs to court system where the judges, lawyers, plaintiff, defendant and other legal experts interact up until the cases are closed. There is a tremendous amount of time passed between the date case is brought to the court for the first time and until the final decision passed on these cases.

Federal Supreme Court has implemented several mechanisms to predict or estimate the time span of court cases so as to facilitate court system monitoring and provide information about the age of the active cases to the plaintiff and defendant in a prior.

Court case time estimation is much relevant in several perspective such as in assisting decision makers, planners, monitoring and performance officers, legal experts and to plaintiff and defendants also.

Court case time estimation[CCTE] is one of the court case management system that aid the court system to predict or estimate the age of the cases earlier so that the length of days that the cases would consume or use will be known in advance. As a result, several resources like court room, judges, and legal experts could be allocated.

### 3.0.1 Supreme Court case estimation techniques

Federal Supreme court [FSC] has implemented several techniques to address the problem of predicting the court cases time span in prior before the cases arrived at the hand of judges.

#### 3.0.1.1 Manual Estimation techniques

FSC's first solution to address the problem and need for the prior prediction of the court cases time estimation was using the manual techniques. These techniques estimate the cases time span by using the committee of legal experts, judges who have previous experience in presiding several court cases. These experts will be given a newly non decided case and based on their experience they estimate the time span of the cases.

These manual court case time estimation techniques, even if it laid a foundation for the prediction, was not that much successful for several reasons that there is high manual involvement, subjectivism of the predictors [Manual], quite dominate the prediction and that it was not that much accurate, error prone. As a result, the FSC implemented a computer based prediction techniques so as to address the aforementioned problems.

#### 3.0.1.2 Computer Based Estimation techniques

The second FSC court case time estimation techniques implemented was a computer based which typically avoid the manual interruption and subjectivism. These technique was incorporated along with court case management system [CCMS] which is a locally developed software that deal with the court case management but estimation technique used by the CCMS was not reliable because it simply average the available court case and It was not based on the learning model. Meaning, the computerized methodology was not supported scientifically, does not take into account previously decided cases and was not also address the major need of the FSC.

However, FSC require the court case time estimation techniques which is really based on the prior decided cases, learn, identify time patterns from these case and project the time estimation for newly incoming case.

## 3.1 Conventional approach to Prediction and Neural Network

Several approach and techniques has been implemented by many researchers in prediction and estimation of numerical and class labels i.e. categorical variables. The prediction of class label is known as classification and the prediction of numerical variables is also called Prediction (Micheline, 2001).

Since the researcher was dealing in prediction of court cases time standard for active and new cases which is the numerical variable to be predicted, as a result, the research is more of biased towards prediction than classifications.

There are currently several techniques for implementing prediction over numerical values. The most commonly used are regression and neural network approach. Even thought, these two methods are used to perform prediction, there is quite a big difference between them. Neural network as mentioned massively in the previous chapters is the imitation of the biological neural network which learns and updates its synaptic weight based on the external inputs. Moreover, neural network is capable of generalizing the data set over prediction and in additions it is capable of handling and tolerating noisy data which can affect the outcome of the experimentations (Perlovsky, 2001).

Basically as a predictor, the neuron is functionally equivalent to regression in statistics. However, the regression technique does not encapsulate the above benefits of the neural network even if both offer more or less similar result in tested experimentations (Perlovsky, 2001).

## 3.2 Review of related literature

In recent years, neural network or ANN have become popular as an alternative method to traditional statistical model for prediction. Current ANN application includes forest growth, time serious analysis, dynamic modeling, spatial data analysis and GIS modeling, global and climate change research. Some of the researches conducted elsewhere in areas of Court cases and those prediction employed in Neural Network as a model were reviewed below.

Stamos, 2004, experimented the application of artificial neural network in predicting the death penalty outcomes. In his research, the neural network was developed having three layered perceptron and was trained using the back propagation principles. In the experiment, a sample of 1,366 profiles of death row inmates was used. After optimizing the network's structure and training the network with 1000 epochs, he tested the networks predictive power and achieved a mean square error of 0.0077 and the network was able to correctly classify at a rate of 93.0%.

In other related literature, Ethiopia, 2002, researched on the application of case based reasoning for Amharic legal Precedent Retrieval system for the federal Supreme Court which assist judges to undergo a legal precedent easily. She conducted her research taking the case of the labor law. The explicit request for the reuse of knowledge and experience in the domain called for the application of Case Based Reasoning [CBR].The system described in the thesis, enable the storage and access to legal precedents and extend knowledge in a natural and straight forward manner. Standard case representation to the original knowledge source was used to store legal cases. Legal Cases have a predefined case structure with a fixed number of features. The feature was selected to reflect the important aspects of a legal case. The feature values were used to do the exhaustive search of the case base.

In other works, Mekonnen, 2004, has experimented the application of artificial neural network in performing time serious analysis over electric power overload. In his research, Mekonnen has conducted a neural network approach to predict the next hourly load of electric power consumptions. He constructed a three layer feed forward multilayer perceptron to fit the predictive model. Mekonnen has divided the electric power consumption data in to training, validation and testing datasets. Once after the model was trained, the neural network was able to give a satisfactory prediction over the testing data sets. The selected model for the daily peak load forecast shows an average percentage error of 3.37% and the neural network forecast a total of 101 patterns (above 93%).

In addition, Taffesse, 2004, experimented on the application of artificial neural network in weather forecasting using the Metrological data collected from the Metrological office which are specifically of Weather related. In his research, Taffesse, constructed a three layer multilayer perceptron artificial neural network to predict the next day atmospheric conditions and he had obtained satisfactory results.

Moreover, Helen, 2006, researched on automatic text summarization system for Amharic legal judgment system. The work referred to the problem of processing huge volume of document in legal field, which become more and more difficult to access. The method used in these researches was not a neural network but the domain was in summarizing the court cases and it was more of information retrieval than neural network based predictions.

Seifu, 2005, researched on the application of a hybrid legal expert system. His thesis was a rule based and neural network hybrid legal system focusing on developing a prototype for providing legal advice on criminal cases under Ethiopian Law. This research was a conducted to develop a

system that could assist legal personal, judges and other related legal experts in presiding criminal cases based on the Ethiopian Law.

Fikadu, 2006, experimented on the application of artificial neural network in forecasting tree volume growth using the multilayer perceptron model. In the experimentation, the researcher carried out experimentation to assess the application of artificial neural network for forecasting the tree volume growth. The result obtained are significant to institutions dealing with the forest research in that it enables to predict the growth of the tree against selling and harvesting plans.Fikadu's experimentation generated a forecasting value of a tree and predicted about 86.7% of accuracy.

# CHAPTER FOUR: EXPERIMENTATION AND DATA ANALYSIS

## 4. Court case time span prediction model

Here in this section, the various activity undertaken starting from the preprocessing phase up to the post processing phase of model development methodology will be discussed in detail.

## 4.1. Input data collection and preparation

The primary step in data collection process is the identification of the data requirements. To achieve this mission, different research works have been examined and the knowledge from the domain expert has been taken into account especially in the court case scenarios.

The data that has been collected for this research work originally obtained from the electronic database of Federal Supreme courts. This database held information on court cases, it consist of 78 attributes with 41,173 records.

However, these datasets has been enough to undergo a research work, this data set has to pass through a lot of preprocessing activity so as to eliminate noisy data, outlier, redundant and irrelevant attributes.

Once the redundant attributes are eliminated, the remaining attributes were given to domain area experts to identify relevant attributes which highly affect court case time span. Out of the dataset attributed presented, these experts from their experience were able to select and reduce the attribute list to 9 attributes. These attributes and their data type are described in the table 4.1.

| Attribute name | Attribute type |
|---|---|
| Case type | Categorical |
| Case Rank | Categorical |
| Case Status | Categorical |
| Charge number | Number |
| Witness | Number |
| Case level | Categorical |
| Plaintiff category | Categorical |
| Defendant Category | Categorical |
| Case file type | Categorical |
| Case decided after[output] | Number |

**Table 4-1 : Variable and their data type**

Once these nine inputs have been selected by these experts, these dataset input and output attribute has been transformed in to the way that would be suitable for the neural network training and testing. Elaboration of each data input and their domain value are discussed in detail in the following section.

### 4.1.1 Input Data

The numerical input data type was normalized using the min-max normalization techniques and the categorical variable is encoded in to a continuous variable between 0 and 1 and the representation of such a variable are elaborated below.

For the categorical attributes, the researchers tried to use a dataset where the attribute with a categorical data type represented using the *l-o-c* encoding schema in which one domain will be

on and the remaining will be off. Such encoding schema make the number of input available to the neural network so many that, 27, such representation schema make the network performance poor and took long training Time and offer poor performance value of 0.054 ,where the network showed over prediction for the training data set offered to it rather than trying to minimize the error between the predicted output and the target output, Which was large and training time takes about 2 hrs and 23 minute .As a result, the researcher prefer, as most literature recommend, to represent such categorical variable using a continuous values between 0 and 1, which as a result make the input to the network reasonable.

The first input element is case type attribute which basically describe the type of the actual court case and it comprise of three values. The values and its corresponding representation in the data base is show in the following table.

| VARIABLE VALUES | REPRESENTATIONS |
|---|---|
| **ወንጀል /Criminal** | 0 |
| **የሥራ ክርክር/Labor** | 0.5 |
| **ፍትሐብሔር/Civil** | 1 |

**Table 4-2: Case type Attribute**

The second input to the neural network is case rank; these indicate the level of the actual court case. These inputs have three values, these values and the representation techniques are shown below.

| VARIABLE VALUES | REPRESENTATIONS |
|---|---|
| **ይግባኝ/Appleate** | 0 |
| **የመጀመሪያ ደረጃ/First Instance** | 0.5 |
| **ሰበር/Cassations** | 1 |

**Table 4-3 : Case Rank Attribute**

The third input to the prediction neural network model is case status .These input typically offer information on the appellate case that was appealed to the Federal Supreme courts. Case status is a categorical variable having six domain values.

| VARIABLE VALUES | REPRESENTATIONS |
|---|---|
| **በክርክር ላይ ያለ/On Debate** | 0 |
| **ያልታየ/Unseen** | 0.2 |
| **የማያስቀርብ/Not Reasonable** | 0.4 |
| **የተወሰነ/Decided** | 0.6 |
| **የተዘጋ/Closed** | 0.8 |
| **ወደሌላ ፍ/ቤት የተዛወረ/Transferred** | 1 |

**Table 4-4 : Case Status Attribute**

The fifth and the sixth input presented to the network respectively are charge number over the given case and witness numbers. This charge number will provide to the network the number of charge list that was charged under the case and the witness_number gives a number of witness that are going to be presented to the court during the case presiding. These input are numerical data type and has to be transformed to the desired input format that are suitable for the neural network, the most common transformation technique, the ***min-max*** transformation technique was adopted. These techniques transform based on the formula given below.

$$Y_{NEW} = \frac{Y_{OLD} - Y_{MIN}}{Y_{MAX} - Y_{MIN}}$$

Where $Y_{new}$=new normalized value

$Y_{old}$=Old unnormaized value

$Y_{min}$=Minimum value in the attribute

$Y_{max}$=Maximum value in the attribute

*Case level* is another input to the predictive model used to train the network .This input in general offers information over the appellate case. Case level identifies if the case is from the Federal or Regional level .Since this input has two values it will be encoded as 0 and 1 where 0 indicate a Federal and 1 for Regional.

The other input that offers information to the predictive model about the plaintiff and defendant is *plaintiff_cat* and *defendant_cat* respectively. These inputs categorize the descriptive information about the corresponding entity and contain a numerical representation and which is encoded to a range between 0 and 1 using the min-max techniques.

The ninth input to the network is a *file type* input which describe about the case's file type information and it is categorical in type and is required to be transformed to a format that was suitable to the neural network as follows.

| VARIABLE VALUES | REPRESENTATIONS |
|---|---|
| **የጊዜ ቀጠሮ/Appointment** | 0 |
| **የዋስትና/Trust ship** | 0.25 |
| **የትዕዛዝ/Order** | 0.5 |
| **የፍሬ ጉዳይ/Main** | 0.75 |
| **የአፈጻጸም/Interpretation** | 1 |

**Table 4-5 : Case File Type Attribute**

Finally the outcome of the model as described in previous chapter is to predict the court case's time span ahead based on the trained case time span list. To do these, a derived attribute is generated from the database by deducting case's opening date with that of case's closing date using Microsoft excel 2007's date/time function of **date360 ()** which deduct two input dates and generate a result in number of day. However, the Days generated out of these function was in decimal format and that there are cases that could have taken more than 10,000 days to be closed. In additions, the courts will be closed for 60 days [2 months time] so as to normalize such data the researcher prefers to convert this attribute in to a number of months by dividing the result with 30 and again, this value are un-normalized that the neural network would not be able to project the desired prediction as a result it is transformed in to a normalized form using a Min-max techniques. The sample data set presented to the network depicted below.

| case_type | case_rank | case_status | charge_num | Witness | caselevel | plaintiff_cat | defendant_cat | case_file_ | month_decided_after |
|---|---|---|---|---|---|---|---|---|---|
| 0 | 0 | 0.5 | 0.050761421 | 0.386959665 | 0 | 0.125 | 0 | 0.75 | 0.004261572 |
| 0 | 0 | 0.5 | 0.050761421 | 0.814265836 | 0 | 0.125 | 0 | 0.75 | 0.009845702 |
| 0 | 0 | 1 | 0.050761421 | 0.426576515 | 0 | 0.125 | 0 | 0 | 0.042762675 |
| 0 | 0 | 1 | 0.081218274 | 0.679872647 | 0 | 0 | 0 | 0.75 | 0.003232917 |
| 0 | 0 | 0.5 | 0.081218274 | 0.750949393 | 0 | 0.125 | 0 | 0 | 0.010874357 |
| 0 | 0 | 0.5 | 0.081218274 | 0.981290649 | 0 | 0.125 | 0 | 0.75 | 0.016164585 |
| 0 | 0 | 0.5 | 0.081218274 | 0.566908934 | 0 | 0.125 | 0 | 0.75 | 0.053196179 |
| 0 | 0 | 0.5 | 0.081218274 | 0.96031874 | 0 | 0 | 0.125 | 0.75 | 0.104628949 |
| 0 | 0 | 1 | 0.093908629 | 0.400535166 | 0 | 0.125 | 0 | 0.75 | 0.000440852 |

**Table 4-6 : Sample Preprocessed input dataset**

Finally, the preprocessed attributes and dataset was presented to neural network to develop predictive model that could be trained using the available data set.

## 4.2 Network Architecture and Training

This phase is the main part of the whole process. Several different neural network predictive models are built and tested. The models are based on the most established idea presented in the literature and the finding of this research work.

A fully connected three layer feed forward Artificial neural network [ANN] was used in the development of the network. Generally, neural network with three layers have an approximation capability for a non linear function by using hidden neuron with varying numbers.

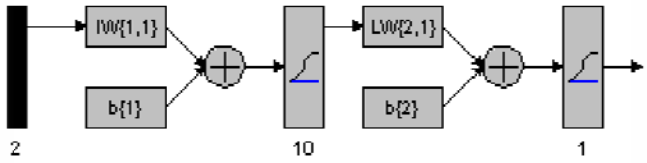


**Figure 4-1: Neural Network Architecture**

From the above figure, figure 4-1, the model consists of input values, one output value and hidden neurons. Determining the number of hidden neuron in the hidden layer is more difficult than determining the size of the input and output layers. Generally, if they are very few, it will not be flexible enough to model the data well, if they are too many, the model will over fit the data. The number of neurons used in the hidden layer in the entire model was not the same. An

attempt was made to observe the performance of the models, testing for different neurons starting from small number of neuron to a large numbers.

The next step was to decide on the type of transfer function to be used in the hidden and output layers. The log-sigmoid [LOGSIG], as briefly discussed in chapter two section 2.2.1, transfer function was used in the output and hidden layers since the target values used in the study were between 0 and 1.

## 4.3 Data organization for model building

### 4.3.1 Partitioning the data set

Multilayer perceptron neural networks are dependent on the data for the prediction of the output. Hence it requires a presence of a huge amount of data sets. In this research work, the total dataset size, 33,411, is assumed to be sufficient for the network to develop accurate model that could predict court case time span. This huge amount of data set was partitioned in to 3 data set that will be used in training, testing and validation the networks performance. Thus, the total number of 33,411 dataset divided in to these sets as follows. Such a classification was undertaken under the percentage classification requirement of the MATLAB neural network.

| SETS | SIZE |
|---|---|
| Testing [33% of the whole] | 8,352 |
| The remaining 67% of the whole dataset[25,059] | |
| Validation[33%] | 6,264 |
| Training [67%] | 18,794 |
| **TOTAL** | **33,411** |

**Table 4-7 : Data Set Partitioning**

Such a division of the whole preprocessed dataset in general is to compromise the two facts; the larger the training set, the better the prediction and the larger the test set, the more accurate the error estimate.

The training set is used to train the network, to compute the gradient and to update the network weight and biases. The validation set is used to monitor the error on a subset of the whole data during the training process. The validation error normally decreases during the initial phase of the training, as does the training set error. However, when the network begins to over fit the data, the error on the validation set will begin to rise, when the validation error increase for specified number of iteration [*max_fail*], the training is stopped. The following figure, figure 4-2, shows this fact. The graph was plotted during the training process for one of the tests attempted. Training stops when the validation set start to rise.

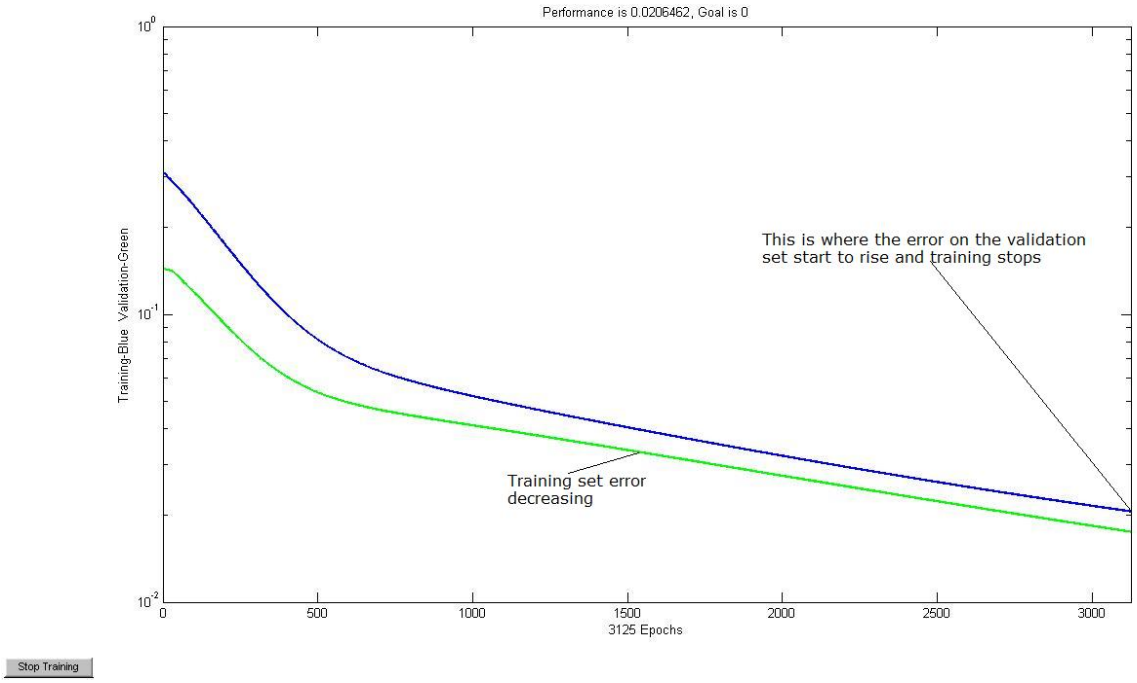


**Figure 4-2 :Training stops time illustrations**

### 4.3.2 Creating and training the network

At this stage, everything was ready to create and train the network .The first action to accomplish this was to import the organized dataset into the work space of the MATLAB neural network toolbox. The MATLAB neural network tool box is selected because it is the most commonly used tool in Neural Network researches. The excel's dataset format had 9 input attribute, one target attribute and a total of 18,794 training records were presented to the network. After these, the feed forward network was created using all the features in the network topology.

After the network was created, the next step was to train the network with the training data set. In creating the network, different parameters in the selected training function were set and tested.

### 4.3.3 Performance Measures

The major performance measurement for a neural network model is the mean square error, one recommended by the MATLAB neural network toolbox, and a best model is the one that tries to minimize this measures. The rate of change in the average mean squared error is typically considered to be small enough if it lies in the range of 0.1 to 1 percent [0.001 to 0.01] per epoch. Sometimes as small as 0.001 percent per epoch is used (Hykin, 1999). So the good neural network is the one that whose mean squared error is between those ranges.

### 4.3.4 Training functions

There are a number of variations of the basic back propagation algorithms. The MATLAB neural network tool box implements a number of these variations. From these functions, three of them were tried for this research work to get the optimal model. These three function are *traingdm* [Gradient descent with momentum], *traindx* [Gradient descent with momentum and adaptive learning rate], and *trainlm* [Levenberg Marquardt algorithms]. The model was trained with these

three different back propagation algorithms with different hidden neuron and with the default learning and momentum. The result is displayed in the following table, table 4-8. Moreover, the experimentation and performance of this model is shown in Appendix [Sample experimentation view].

| **Model** | **Hidden neuron** | **Epoch** | **Performance [MSE]$10^{-2}$** | | |
|---|---|---|---|---|---|
| | | | **Traindm** | **Traingdx** | **Trainlm** |
| 1 | 3 | 1000 | 4.9087 | 6.9845 | 3.7655 |
| 2 | 5 | 1000 | 2.50138 | 4.82493 | 3.7665 |
| 3 | 10 | 1000 | 2.55187 | 0.5097 | 0.364801 |
| 4 | 15 | 1000 | 1.012441 | 0.389383 | 0.396843 |
| 5 | 20 | 1000 | 4.84783 | 0.76875 | 0.3561 |
| 6 | 25 | 1000 | 5.8976 | 2.1233 | 3.4532 |
| 7 | 40 | 1000 | 0.78654 | 2.8765 | 3.9876 |

**Table 4-8 : Network best performance selections**

The number of hidden neuron is selected within these ranges because the network showed poor performance when trained with hidden neuron more than 20 and with neuron number less than 5 but better in between.

Based on the information from the table 4-8, model 4 trained with 20 hidden neuron and *trainlm* back propagation algorithms showed a good performance in compared to the others models because the purpose of the predictive neural network model was to minimize the error between the desired and the target output and hence, based on these, model 4 is selected because it scored the least MSE [Mean Squared Error]. The 20 hidden neuron units were chosen for the network model and successive training were performed for different combination of the learning rate and

momentum. The training process stops upon hitting the error tolerance or reaching epoch limit. The error tolerance was set to 0.001 in all the training situations which is the default error tolerance limit for the MATLAB neural network toolbox.

First the selected network was trained with different combination of the learning rate [*mu*] and *max_fail* with fixed epoch limit of 5000, because the network does not show significant performance improvement beyond this epoch limit .The model with learning rate [*mu*] of 0.5 and *max_fail* 10 showed the least mean square error [performance measure] which is 0.00336577[APPENDIX 8].The overall performance of the training set of the selected model is show in the following graph.

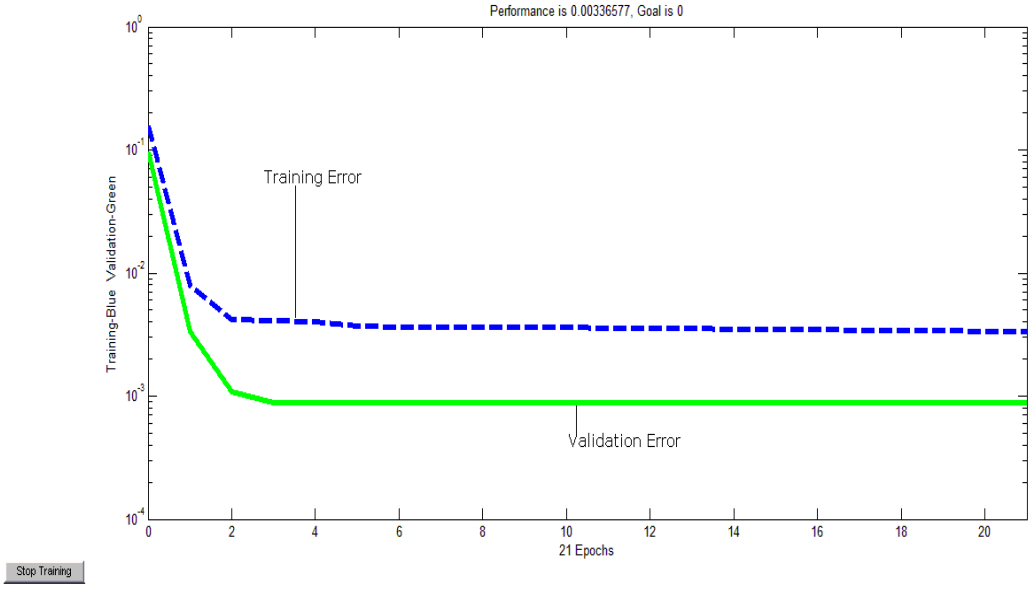


**Figure 4-3: Network training and validation errors graph**

Summarized performance result of the selected model is as follows:

| Measure | Performance |
|---|---|
| MSE | 0.00336577 |
| RMSE | 0.05801 |
| R-values | 0.9435 |

**Table 4-9: Performance measure for the best model**

From table 4-9, it can be seen that the performance function of the neural network is minimized as per requirement of the network. The R-values indicate the coefficient of correlation between the training output and the target output. This value is the measure of how well the variation in output is explained by the targets. According to Howard Demuth et. Al. (2008), if this number is equal to 1, then there is perfect correlation between targets and outputs. In the training result, the number is very close to 1, which indicates a good fit and less variation for the model.

Figure 4-3 on the other hand; graphically show the depiction of the training error and the validation error for the whole training epoch. It is clearly shown in the figure that the training error and that of the validation error are minimized to the range of the required intervals which quite indicate the performance of the network that it has approximated the predicted time span with that of the actual, with satisfactory prediction accuracy according to the performance functions. Moreover, figure 4-3, indicate that the training was stopped after the 21 epochs because it is the point at which the validation error started to increase and the MATLAB neural network tool box stops the training when this validation error start to rise. The predictive capability of the neural network over the training data is elaborated in the following figures and tables.

A histogram of the ANN prediction error that show the distribution of error in the training set over the test period is presented in the figure 4-4.

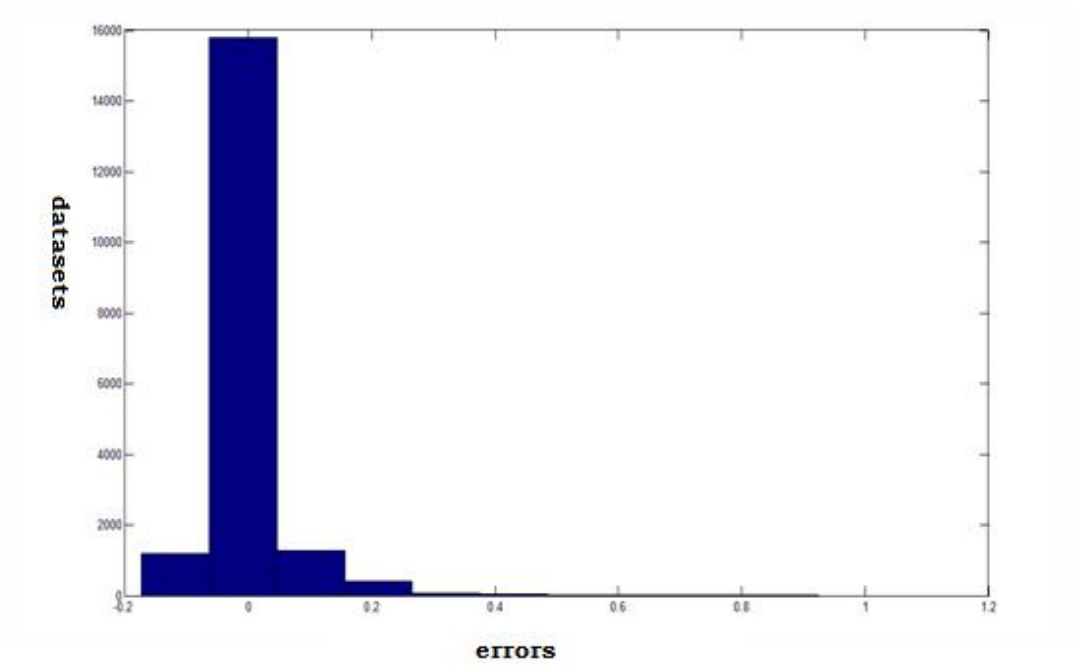


**Figure 4-4 : Histogram of errors for all of the training set**

In the above figure, figure 4-4, shows the distribution of error. The error is the difference between the predicted and the actual value. It is shown in the figure that difference, the error, is between ±0.2 [94.44%] months for most record, more than 17,000 out of the 18,000 records presented to the network and it is the indication of the prediction capability of the network to make the predicted values as close to the actual values.

Figure 4-5 shows the plot of the error of the training the neural network and it is shown clearly from the figure that the error distribution is highly lays between +0.01 and -0.005 months. Positive error value indicates that the network predicted higher values for the input dataset presented to the net. While the negative values indicate that the network predicted smaller but

close values to the actual values for the training datasets presented. It is indicated from the figure, in addition, that the error value for the whole training set presented to the network is between 0 and -0.03.

Unusually error value less than -0.003 could be the result of several factors that could affect the court case time span. Such factors are not captured and not included in the dataset presented to the network. These factors ,according to the networks performance and experts view over the performanc, includes as unavailability of witness on the allotted date, Personal Situation of Judges, Current country's political and social conditions, Calander Effects and etc .

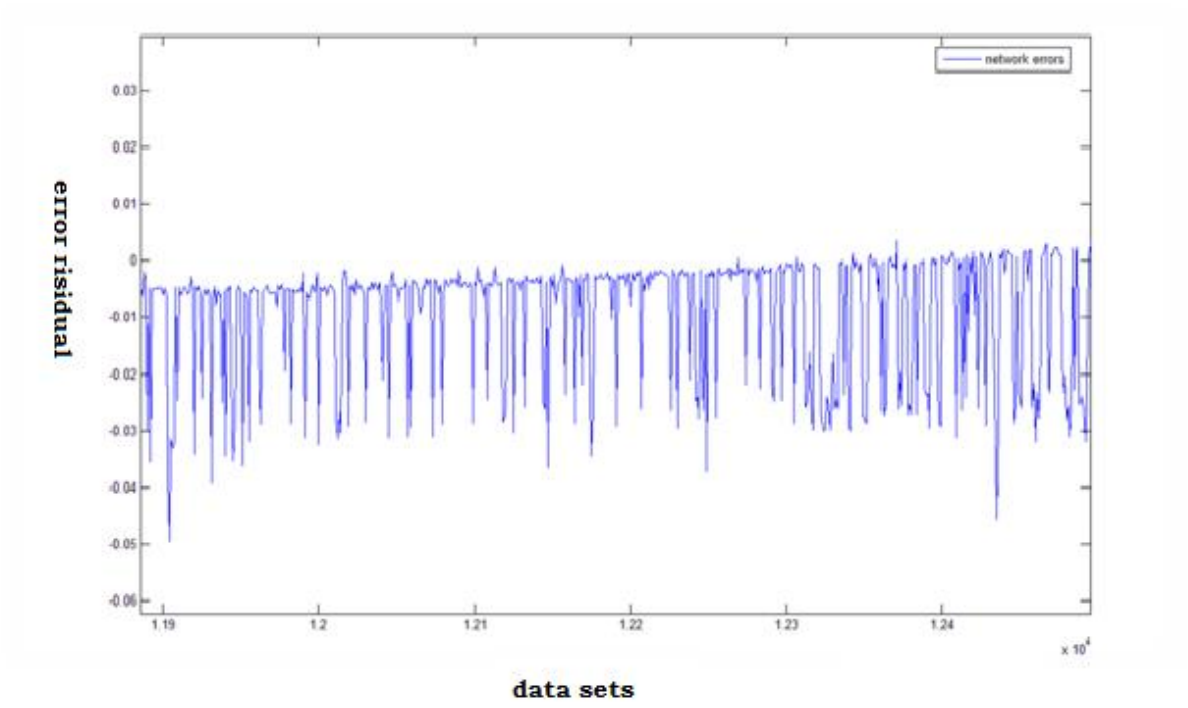


**Figure 4-5 : Network error for each training set presented to the network**

## 4.4 Evaluation and interpretation

After deciding on the optimal network architecture for the application, the time span predictive model has been evaluated with the testing data that was originally reserved to test the performance of the model.

Various predictive error measures between the actual and the predicted time span parameters are defined. However, the most commonly adapted are: Mean Squared Error [MSE], Root Mean squared and Mean absolute percentage error. But the researcher selects MSE because it is the one umostly used and the one that are recommended by the MATLAB Neural Network toolbox.

The mathematical definition of the measure is given by the following formulas (Wikipedia encyclopedia) Mean Squared Error [MSE]

$$MSE = \frac{\sum_{j=1}^{V} E_j^2}{V}$$

Where $\mathbf{E_j = Y_i - y_i}$ and $\mathbf{Y_i}$ is the desired output, **yi** are the forecasted values and ***V*** is the number of test examples.

Root means squared error/deviations [RMSD/E]

$$RMSE = \sqrt{MSE}$$

The Trained model is tested with its test data set to check out its generalization capability and offers the following performance result which is quite close to the training set result performance.

| Measure | Performance |
|---|---|
| MSE | 0.00331238 |
| RMSE | 0.0575532 |
| R-values | 0.9538 |

**Table 4-10: Performance measure of Test result**

From table 4-10, we can deduct that the network offer closure performance rate as compared to the network performance of the network shown in table 4.9. This closer performance of the network is the result of the networks generalization capability in the data set that is not trained before. This testing data set was reserved once the whole data set was divided in to training, testing and validation as show above in table 4-7. Moreover, table 4-10, depict that the testing set offer a good coefficient of correlation which is quite close to the training correlation coefficient that, this statistics strengthen the networks generalization ability again.

When the network is given a data set with same court case types, the network shows the following performance results and their corresponding predictive accuracy in terms of percentage was calculated for the network to predicate the cases with the error range between $\pm$ 0.05.

| Case Type | MSE | Predictive Accuracy |
|---|---|---|
| Criminal | 0.003211 | 95.65% |
| Civil | 0.003343 | 89.54% |
| Labor | 0.003091 | 91.55% |

**Table 4-11: The network performance for various case types**

The little difference in MSE in table 4-11 could possibly result from the fact that the datasets are in the order of, 85% are of Civil Cases and 12 % for criminal and the remaining are labor case, such inequality with the dataset and random division of the data set for training, testing and

validation bring about such a variations. Moreover, such a variation attributed to, according to the experts' opinions, to other factors that are critical for the research work but not captured in the dataset presented to the network such as countries' current situation, absentcizim of witness, invalid evidence, improper witness, in availability of court room/benches, and degree of influence of other cases.

According to table 4-11,The network showed a good predictive for criminal court cases with the capability to minimize MSE[error residual] and make high number of cases' error rate to be between $\pm$ 0.005.The Reason behind the networks various performance for these cases is attributed to the factors mentioned above .

Graphically the training set, validation set and test set for the model is shown below along their performance up on testing.

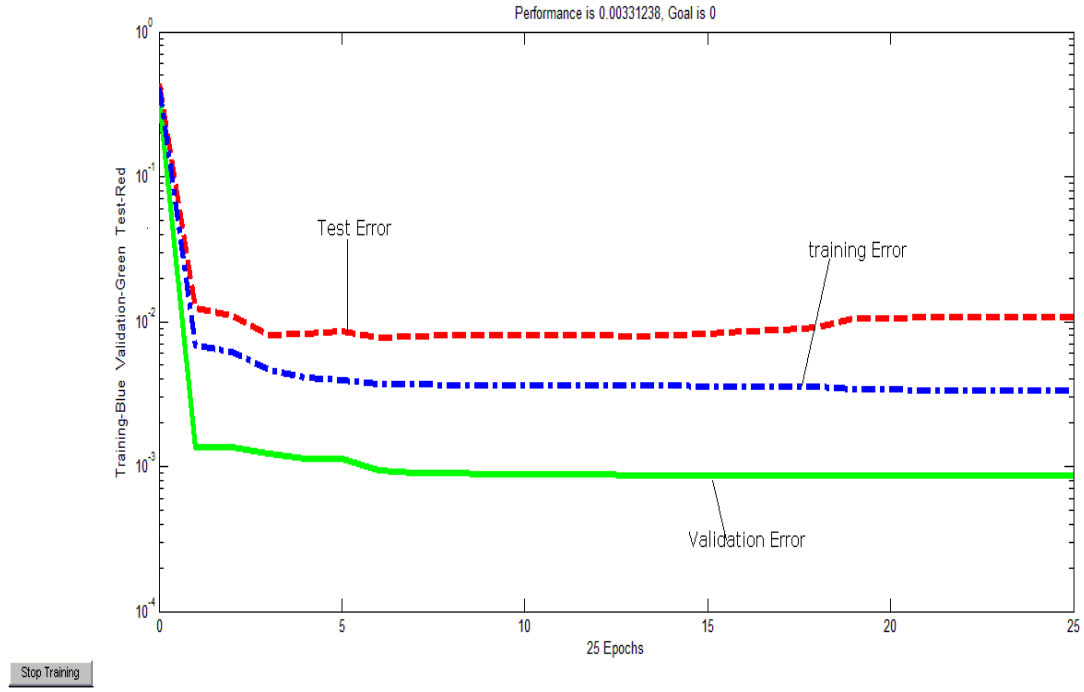


Figure 4-6 : Test set error graph

Figure 4-6 shows the performance of the network presented with the whole data set that is the training set, validation and testing sets. It shows that the network more or less performed closes as the training performance as mentioned above.

Finally, the prediction output of the network is a continuous value which is between 0 and 1 months and it is, as stated in preprocessing section, represented in months. As a result, such value has to be transformed in to a format that is usable and easily understandable. Such a transformation schema adopted from Larose (2005)

***Prediction = output (data range) + minimum***

On the basis of the above equation, we can normalize the networks predictive capability for the data set presented in table 4-6. The data range in the equation refers to the difference between the maximum and the minimum values in the data set presented to the network. Based on the equation the predicted and the normalized output of the network are shown below.

| **PREDICTED OUTPUT** | **NORMALIZED/MONTHS/** |
|---|---|
| 0.11062 | 13.234 |
| 0.076091 | 2.2123 |
| 0.10668 | 3.7611 |
| 0.077588 | 2.789 |
| 0.082658 | 10.896 |
| 0.18964 | 17.21455 |
| 0.14778 | 4.22145 |
| 0.084299 | 11.34255 |

**Table 4-12: The Predicted and Normalized output of the network**

## 4.5 Neural network and Manual prediction comparisons

Here on this section a comparison of the neural network prediction and that of the manual prediction was compared. For this purpose, 10 court cases were given to 7 legal experts and the same data was simulated to the trained net. The legal experts give their prediction of the court case time span for the 10 cases. Each individual's prediction was summed and averaged so as to get the average prediction of these court cases. The overall prediction of these experts and their average and as well the neural's prediction was displayed in the table 4-13.

| | LEP1[1] | LEP2 | LEP3 | LEP4 | LEP5 | LEP6 | LEP7 | AvegP[2] | ANNP[3] |
|---|---|---|---|---|---|---|---|---|---|
| CC1[4] | 2.5 | 2.0 | 3.3 | 2.5 | 2.2 | 3.0 | 3.2 | 5.343 | 6.3321 |
| CC2 | 4.5 | 4.0 | 3.2 | 3.2 | 4 | 6.0 | 3.0 | 3.987 | 4.5611 |
| CC3 | 3.2 | 6.0 | 2.2 | 4.5 | 2.0 | 5.0 | 4.5 | 3.9427 | 4.21 |
| CC4 | 10 | 8 | 6.5 | 7 | 9 | 11 | 12 | 9.071 | 7.900 |
| CC5 | 5 | 8 | 18 | 3 | 10 | 24 | 12 | 11.4285 | 11.4005 |
| CC6 | 8 | 9 | 8 | 9 | 7 | 7 | 12 | 8.5714 | 11.321 |
| CC7 | 10 | 8 | 6 | 8 | 3.20 | 4.5 | 5 | 6.3852 | 5.2300 |
| CC8 | 24 | 15 | 19 | 25 | 21 | 10 | 18 | 18.8571 | 15.045 |
| CC9 | 5 | 4.3 | 3.0 | 6 | 7 | 12 | 15 | 7.5857 | 7.908 |
| CC10 | 10 | 12 | 12 | 10 | 9 | 11 | 10 | 10.5714 | 9.3412 |

**Table 4-13: Comparison of Neural network and human Prediction in Months**

As seen in the table 4-13, the neural network prediction is quite close to the manual prediction of the case's time span .The little variation between the two predictions could possibly result from:

[1] Legal Expert Prediction
[2] Average Predictions
[3] Artificial Neural Network Predictions
[4] Court Case

1. The Experts' perception of a month and the neural network's perception of a month's is quite different i.e. human consider only working days in the months
2. Even if the network is trained with huge datasets,the unequal proportioality of the cases could possible factor these variation as mentioned in earlier sections
3. The experts' prediction takes into account the calendar effects while the neural network did not.
4. The experts' prediction was as a result of the experts' view of several different factors which were not captured and used to train the neural model.

# CHAPTER FIVE: CONCLUSION AND RECOMMENDATIONS

## 5.1 Conclusion

Neural network can learn to approximate any function and behave like associative memories by using just example data, Court case in these experiments, is representative of the desired task. They are model free estimator and are capable of solving complex problems based on the presentation of a large number of training data. This gives them a key advantage over traditional approach such as statistical methods. Neural networks estimate a function without a mathematical description of how the output functionally depend on the inputs .They represent a good approach in that outputs functionally depend on the inputs .

In this thesis, I have investigated the application of Feed Forward neural network with back-propagation learning algorithms in Prediction court Case's time span. The ANN model generate Time period estimation of the court cases a head. The predictive model was developed having 9 inputs and 20 hidden neuron in a single hidden layer and with one output .Once the model is trained with the representative data and validation has performed the network, the model was tested using the representative data that was kept for testing which are independent data sets and the model has shown a satisfactory result as shown above.

The model developed has shown good performance in training session offering a performance value of 0.00336577 and during testing with a data set it shows a performance of 0.00331238.In General ,the model's predictive capability can be depicted with the error range produced and it showed that 94.444% accuracy of prediction performance in $\pm$0.2 months error range.

In the experiment, MATLAB 7.0 was used as a neural network tool. Different functions of the basic back propagation algorithms were tested and one which showed better result as mentioned above was selected.

To conclude, the result of the study can be considered as satisfactory and the proposed model can be used as one component of a decision support system that can provide ahead prediction of court case's time span to the Federal Supreme Courts.

## 5.2 Recommendation

Event thought this research was conducted mainly as an academic Research; its findings have implication far beyond this. It can be used as one component of a decision support system for Federal supreme Courts of Ethiopia. The study can contribute a lot for the further studies conducted in the area at Federal Supreme courts where there are huge data amounts.

Based on the finding of the research and the cases encountered while conducting the study, the researcher would like to recommend on some point that could benefit the future researcher in the area and the FSC.

The first sets of recommendation are mainly targeted to the Federal supreme courts with respect to the recording and keeping data, Even if the data was combined at once place, in order to get optimal advantage from the data, all important attribute of the subject should be included like the age of the defendant and plaintiff as well the nationality of the defendant and plaintiff.

Secondlly, there are a lot of factor that could affect the time period of court cases for example availability of court rooms, availability of Judges, Country's current status and political

situations, these things should be identified and further research can be performed using the Bayesian Uncertainty approach.

Third ,once after the predictive modeling is done the resource allocation that is allocation of court room, available judges and various research could be performed using Genetic Algorithms and could benefit the FSC.

Fourthly, further works can be done with the excluded attributes and other techniques Genetic algorithms and Bayesian approach could be encapsulated with Neural Network predictive capability so as to offer more and advantageous results.

Fifth, Rather than the Feed forward neural network, other neural network approach could be implemented so as to get better and comparative result in the prediction of court case time period.

Sixth, the Federal Supreme Court should expand this thesis's finding and works in such a way that a graphical user interface [GUI] should be developed that could facilitate the time period estimation to novice users.

Seventh, the researchers believe that the experiment could give better result if the prediction is performed using the case based inputs rather than offering all court cases in general.

# 6.0 Reference


1. Anderson D. and McNeil, G. (1992). *Artificial Neural Network Technologies:data anlysis for software.* Rome,NY: Rome Laboratory.
2. Anderson, D. a. (1992). Retrieved 10- 23, 2008, from http://www.dacs.dtlc.mil/techs/Neural/neural.title.html
3. Anderson, J. A. (1995). *An Introduction to Neural Network.* cambridge: MIT press.
4. Becker, S. (1991). Unsupervised learning procedures for neural networks. *International journal of neural system* , 17-33.
5. Berry, M. a. (2000). *Mastering Datamining :The art and science of Customer Relationship Managment.* NewYork: John Wiley and sons inc.
6. Courts, F. S. (2007). *Organization of courts and modern practice.* Addis Abeba.
7. Ethiopia, T. (2002). *Application of Case Based reasoning for amharic legal precedent:a case study with Ethipian labor law.* Addis Abeba: Un published Master Thesis :Addis Abeba University.
8. Fauset, L. (1994). *Foundation of Neural Network.* New Jersey: Prentice-Hall.
9. Fikadu, G. (2006). *Forecasting Tree Volume Growth Using Artificial Neural Network:The case of Cuperssus Lustaica Species.* Addis Abeba: Unpublished Master Thesis :Addis Abeba University.
10. Howard Demuth, M. B. (2008). Neural Network Toolbox, User's Guide. Mathswork Inc.
11. Hykin, S. (1999). *Neural Network-A comperhensive foundation* . Upper Saddle River,NJ: Prentice Hall.
12. J.G, T. (1995). *Neural Networks.* London: AlfreLLd Waller Limited,Publishers.
13. Larose, D. (2005). *Discovering Knowledge In data:An introduction to Data Mining* . New Jearsy: A Jhon Willy and Sons Publications.
14. Mekonen, T. (2004). *Artificial Neural Network Based Short Term Load Forcasting for the Ethiopian Electric and Power Corporation.* Addis Abeba: Unpublished Masters Thesis :Addis Abeba University.

15. Micheline, J. H. (2001). *Data Mininig :Concept and Techniques.* SanFrasisco: Morgan Kaufmann Publishers.
16. Perlovsky, L. I. (2001). *Neural Network and Intellect:Model Based Concept.* NewYork: Oxford University Press.
17. Plate, T. e. (1997). *A comparison between neural networks and other statistical techniques for modeling the relationship between tobacco and alcohol and cancer.* . Retrieved 10-12, 2008, from http://citseer.nj.nec.com/plate96comparison.html
18. Samarasinghe, S. (2007). *Neural Network for Applied Science and Engineering :From fundamental to complex pattern recognition.* New York: Auerbach Publications.
19. Sarles, W. (1997). Retrieved 10-11, 2008, from Neural Network-FAQ ,periodic posting to the usenet news group comp.ai.neural-net: ftp://ftp.sas.com/pub.neural/FAQ.htmlS
20. Seifu, T. (2005). *A rule based /Neural Networ Hybrid Legal Expert System :A prototype for providing legal advice on criminal cases under Ethiopian Law.* Addis Abeba: Unpublished Masters thesis :Addis Abeba Unversity.
21. Stamos, T. K. (2004). Artificial Neural Network for Pridicting Death Penality Outcome. New Orleans.
22. *statsoft*. (2003). Retrieved 11-02, 2008, from Neural Network: http://www.statsoftinc.com/textbook/steneunet.html
23. Stergion, C. a. (1999). Retrieved 11-02, 2008, from http://www.doc.ic.ac.uk
24. Taffesse, Y. (2004). *Application of Artificial Neural Network in Weather Forcasting* **.** Addis Abeba : Unpublished Master Thesis :Addis Abeba University.
25. *Wikipedia encyclopedia*. (n.d.). Retrieved 11-28, 2008, from http://www.wikipedia.org

# 7.0 APPENDIX

APPENDIX 1

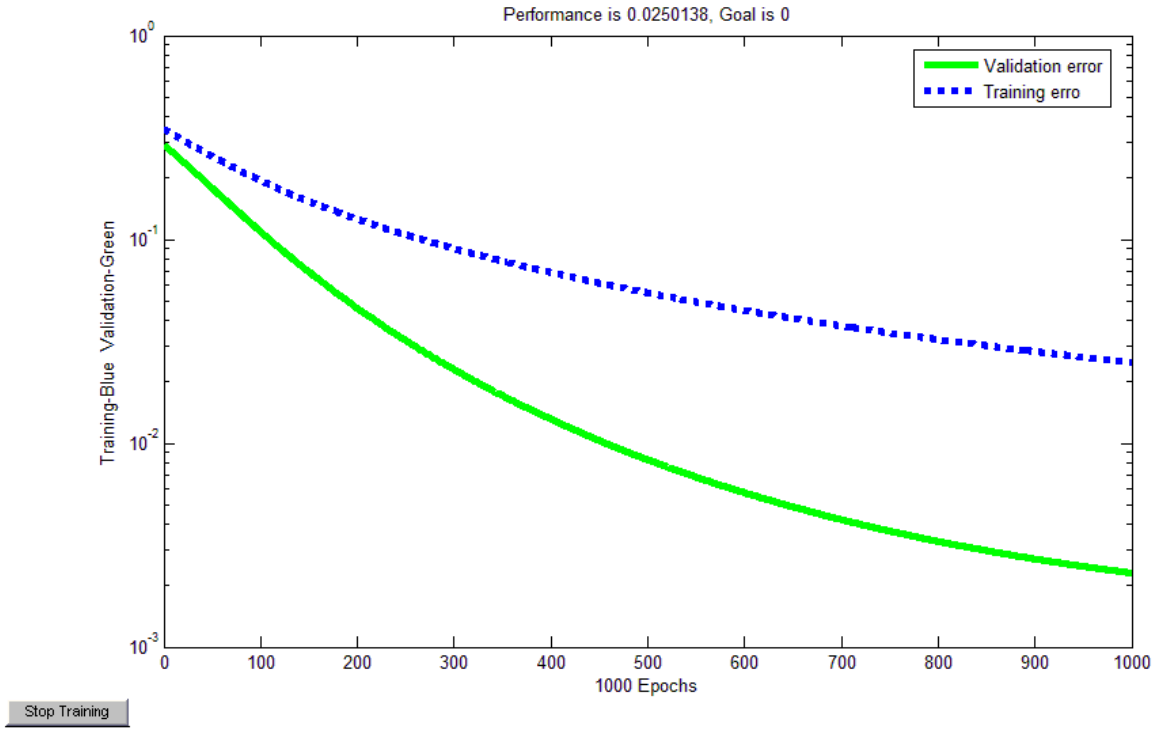


**Figure 7-1 :Network performance of TRINGDM network**

| | |
|---|---|
| MSE | 0.0250138 |
| RMSE | 0.15815 |
| R | 0.65722 |
| Hidden neuron | 5 |
| Training Function | TRINGDM |

**Table 7-1: Performance Function of TRINGDM Network**

APPENDIX 2

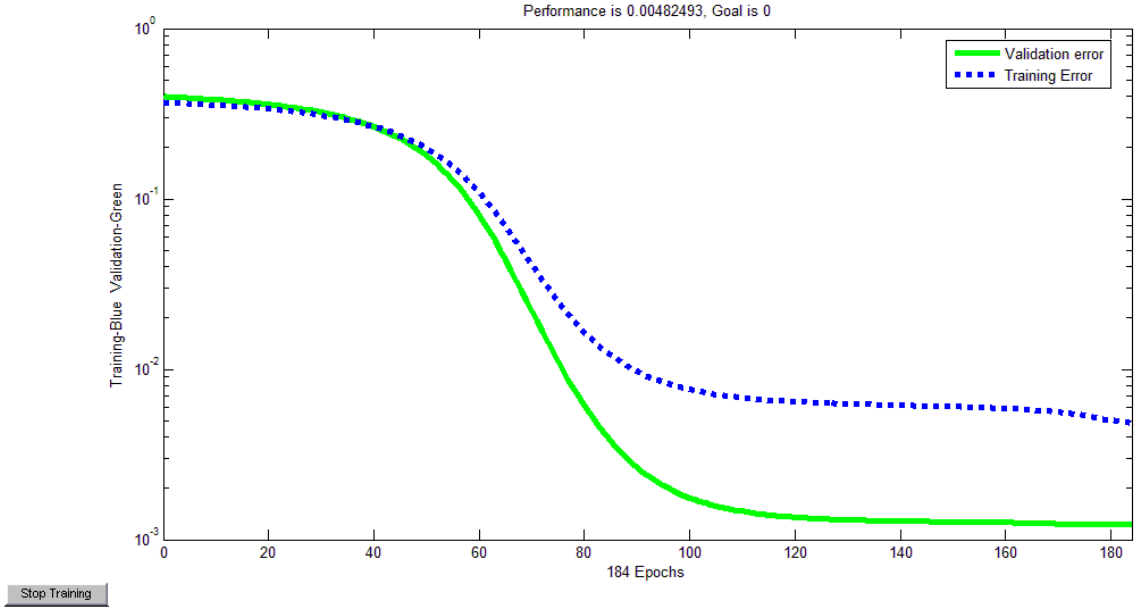


**Figure 7-2 : Performance Function of TRIAINDX network**

| | |
|---|---|
| MSE | 0.00482493 |
| RMSE | 0.0694617 |
| R | 0.6799 |
| Hidden Neuron | 5 |
| Training Function | TRAINDX |

**Table 7-2: Performance Function of TRAINDX network**

APPENDIX 3

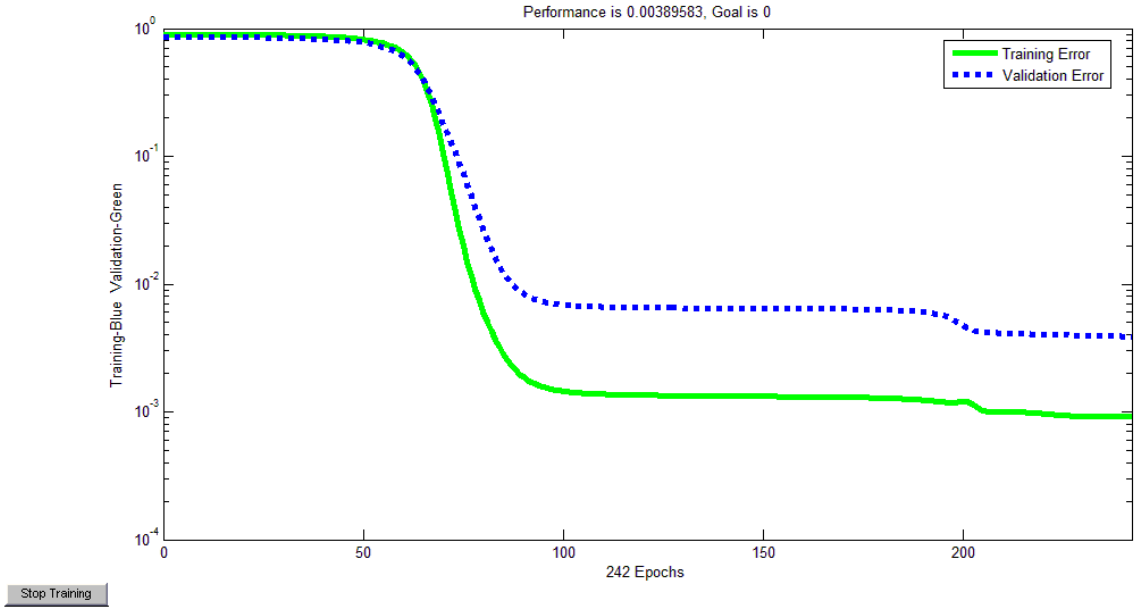


**Figure 7-3 : Performance error curve of TRINDX network with 15 hidden neuron**

| | |
|---|---|
| MSE | 0.00389383 |
| RMSE | 0.0624 |
| R | 0.6923143 |
| Hidden Neuron | 15 |
| Training Function | TRAINDX |

**Table 7-3: Table showing the performance of TRAINDX network**

APPENDIX 4

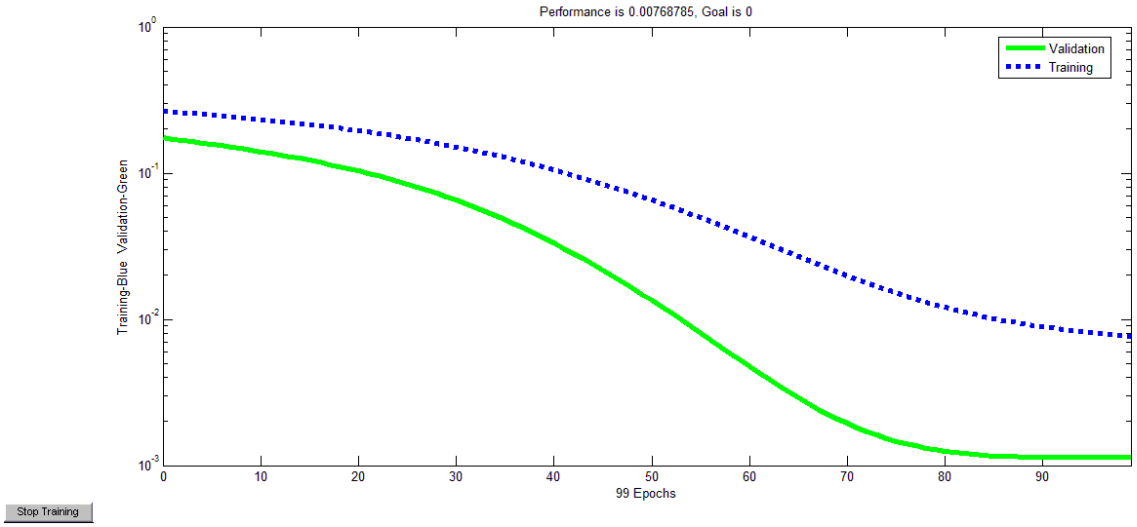


**Figure 7-4 : TRAINDX performance Network with 20 hidden neuron**

| | |
|---|---|
| MSE | 0.0076875 |
| RMSE | 0.08767 |
| R | 0.7899041 |
| Hidden Neuron | 20 |
| Training Function | TRAINDX |

**Table 7-4: Performance statistics for TRAINDX network**

APPENDIX 5

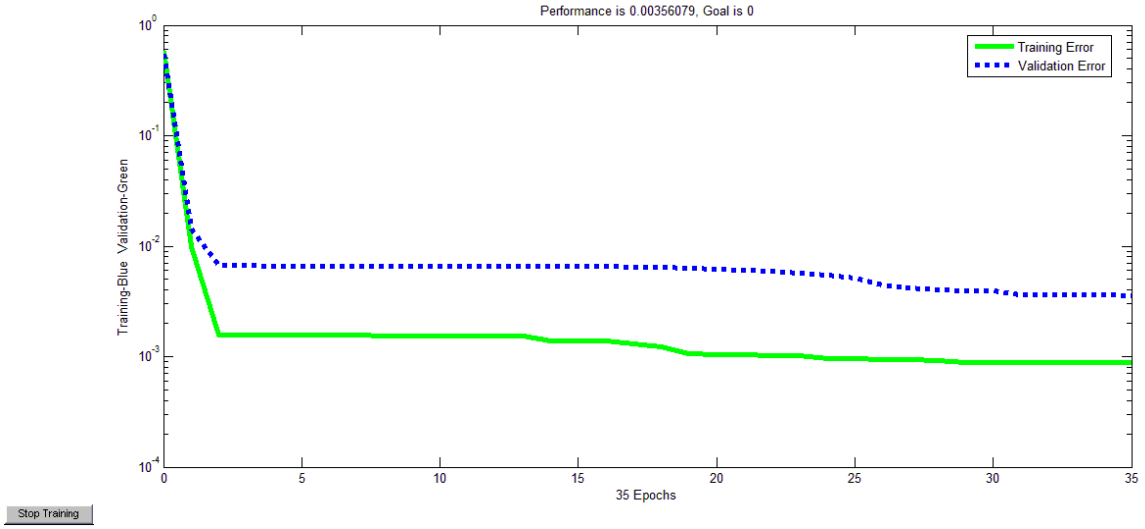


**Figure 7-5 : Performance error figure for TRAINLM function**

| | |
|---|---|
| MSE | 0.0035607 |
| RMSE | 0.0596723 |
| R | 0.88952 |
| Hidden neuron | 20 |
| Training Function | TRAINLM |

**Table 7-5: Performance Statistics for TRAINLM**

APPENDIX 6.

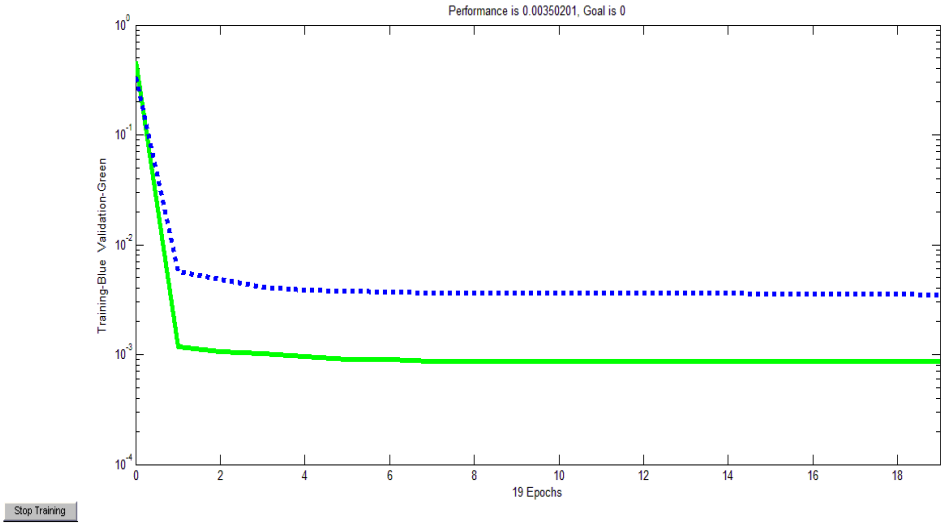


**Figure 7-6 : Performance function for TRAINLM network with learning rate of 0.2**

| | |
|---|---|
| MSE | 0.003502 |
| RMSE | 0.059177 |
| R | 0.88992 |
| Training Function | TRAINLM |
| learning rate [*mu*] | 0.2 |
| *max_fail* | 10 |
| EPOCH | 1000 |

**Table 7-6: Training parameter and performance statistics with learning rate of 0.2**

APPENDIX 7

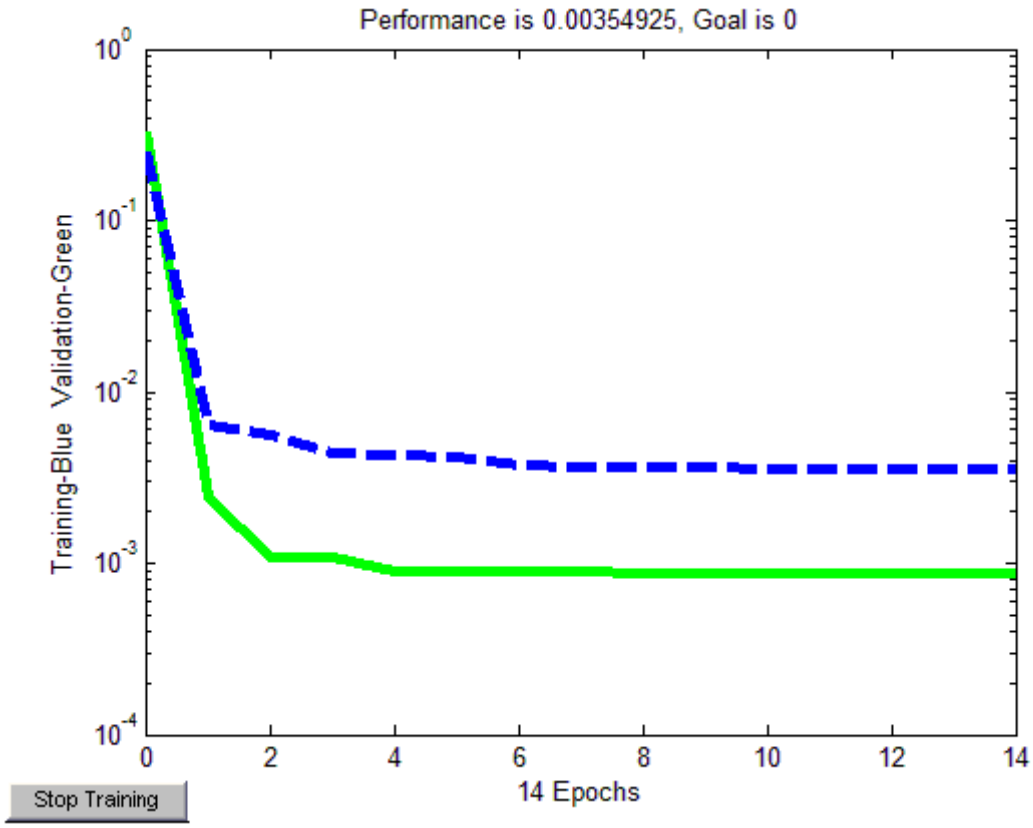


**Figure 7-7 : TRAINLM with learning rate of 0.3**

| | |
|---|---|
| MSE | 0.00354925 |
| RMSE | 0.059937 |
| R | 0.88912 |
| Training Function | TRAINLM |
| learning rate [*mu*] | 0.3 |
| *max_fail* | 5 |
| EPOCH | 1000 |

**Table 7-7: Parameter and statistics for TRAINLM with 0.3 learning rate**

APPENDIX 8

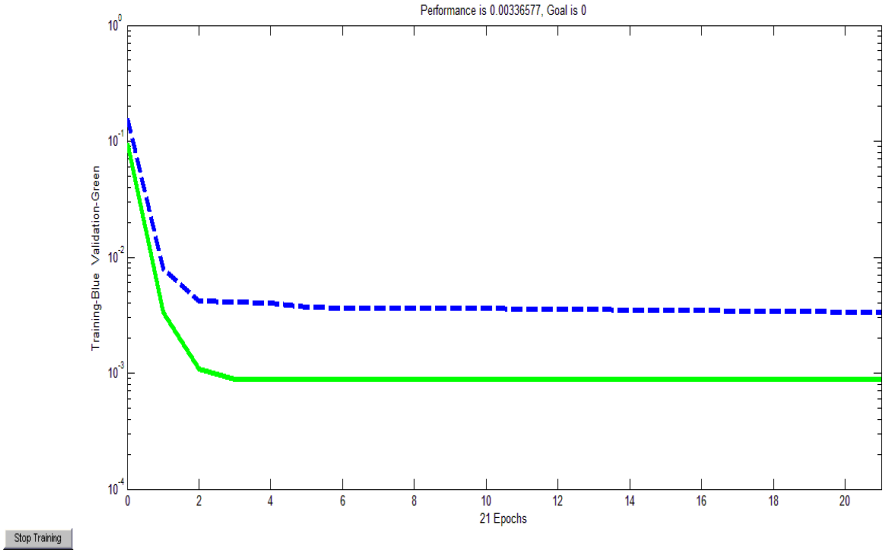


**Figure 7-8 : TRAINLM network with learning rate of 0.5**

| | |
|---|---|
| MSE | 0.00336577 |
| RMSE | 0.05801 |
| R | 0.9435 |
| Training Function | TRAINLM |
| learning rate [*mu*] | 0.5 |
| *max_fail* | 10 |
| EPOCH | 1000 |

**Table 7-8: TRAINLM training and performance statistics**

APPENDIX 9

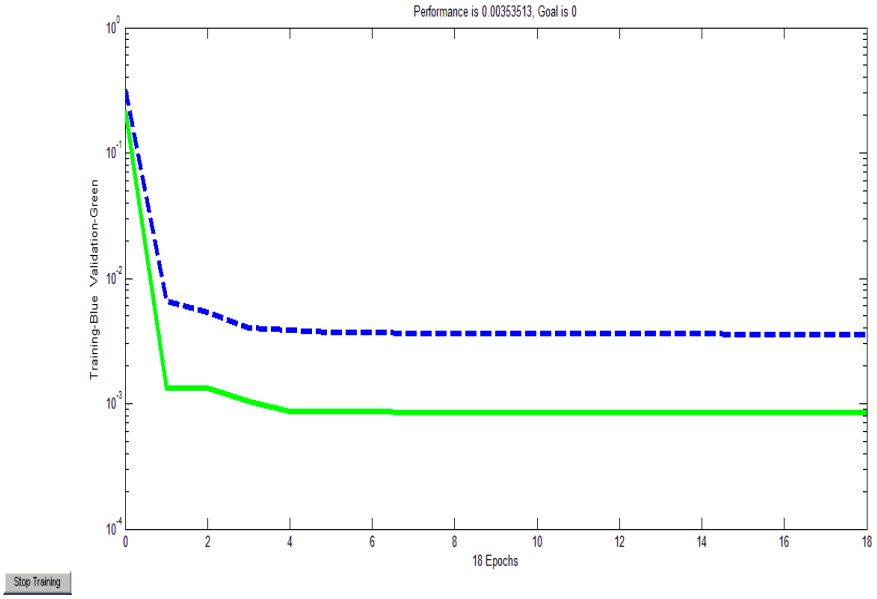


**Figure 7-9 : TRAINLM network with learning rate of 0.65**

| | |
|---|---|
| MSE | 0.00353513 |
| RMSE | 0.0594965 |
| R | 0.88901 |
| Training Function | TRAINLM |
| learning rate [*mu*] | 0.65 |
| *max_fail* | 10 |
| EPOCH | 1000 |

**Table 7-9: Training and Performance Parameter with learning rate of 0.65**

APPENDIX 10

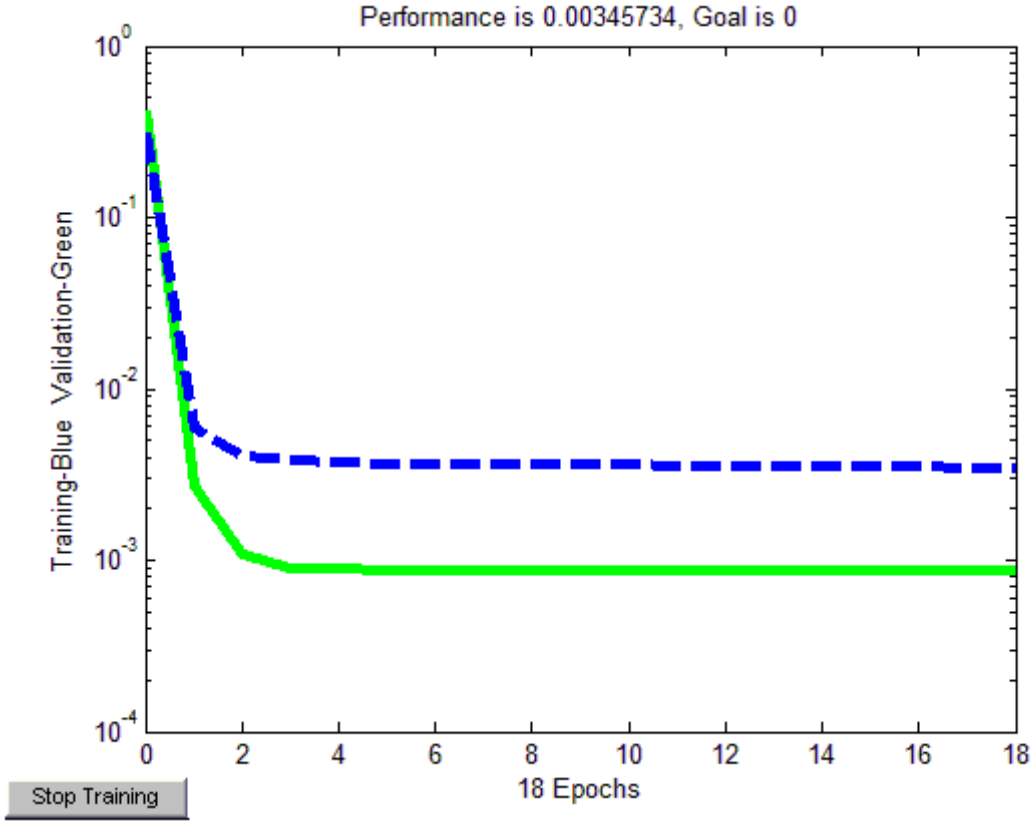


**Figure 7-1 : TRAINLM with learning rate of 0.75**

| | |
|---|---|
| MSE | 0.00345734 |
| RMSE | 0.058799 |
| R | 0.88923 |
| Training Function | TRAINLM |
| learning rate [*mu*] | 0.75 |
| *max_fail* | 10 |
| EPOCH | 1000 |

**Table 7-10: Training and performance statistics of TRAINLM with learning rate of 0.75**

Appendix 11:The datasets

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| 1 | 0 | 1 | 0.159898 | 0.752019 | 0 | 0.125 | 0.125 | 0.75 |
| 1 | 0 | 1 | 0.159898 | 0.842308 | 0 | 0.125 | 0.375 | 0.75 |
| 1 | 0 | 1 | 0.159898 | 0.219848 | 0 | 0.125 | 0.125 | 0.75 |
| 1 | 0 | 0 | 0.159898 | 0.098918 | 0 | 0.125 | 0 | 0.75 |
| 1 | 0 | 0 | 0.159898 | 0.303854 | 0 | 0 | 0.125 | 0.75 |
| 1 | 0 | 0.5 | 0.159898 | 0.144611 | 0 | 0.125 | 0.125 | 0.75 |
| 1 | 0 | 0.5 | 0.159898 | 0.909482 | 0 | 0 | 0.125 | 0.75 |
| 1 | 0 | 0 | 0.159898 | 0.923738 | 0 | 0.125 | 0.125 | 0.75 |
| 1 | 0 | 1 | 0.159898 | 0.289216 | 0 | 0.125 | 0.125 | 0.75 |
| 1 | 0 | 0 | 0.159898 | 0.029019 | 0 | 0.125 | 0.125 | 0.75 |
| 1 | 0 | 0 | 0.159898 | 0.559125 | 0 | 0.125 | 0.5 | 0.75 |
| 1 | 0 | 0 | 0.159898 | 0.374286 | 0 | 0.125 | 0.25 | 0.75 |
| 1 | 0 | 0.5 | 0.159898 | 0.249917 | 0 | 0.25 | 0 | 0.75 |
| 1 | 0 | 1 | 0.159898 | 0.718368 | 0 | 0.125 | 0.125 | 0.75 |
| 1 | 0 | 0.5 | 0.159898 | 0.898153 | 0 | 0.125 | 0.125 | 0.75 |
| 1 | 0 | 0 | 0.159898 | 0.619999 | 0 | 0.375 | 0.125 | 0.75 |
| 1 | 0 | 1 | 0.159898 | 0.271979 | 0 | 0 | 0.125 | 0.75 |
| 1 | 0 | 0 | 0.159898 | 0.843272 | 0 | 0.125 | 0.125 | 0.75 |
| 1 | 0 | 0.5 | 0.159898 | 0.237918 | 0 | 0.125 | 0.125 | 0.75 |
| 1 | 0 | 1 | 0.159898 | 0.39384 | 0 | 0 | 0.125 | 0.75 |
| 1 | 0 | 0.5 | 0.159898 | 0.736061 | 0 | 0.25 | 0.125 | 0.75 |
| 1 | 0 | 0 | 0.159898 | 0.529214 | 0 | 0.125 | 0.25 | 0.75 |
| 1 | 0 | 0 | 0.159898 | 0.881884 | 0 | 0.125 | 0.125 | 0.75 |
| 1 | 0 | 1 | 0.159898 | 0.564491 | 0 | 0.125 | 0.125 | 0.75 |
| 1 | 0 | 1 | 0.159898 | 0.731051 | 0 | 0.125 | 0.125 | 0.75 |
| 1 | 0 | 1 | 0.159898 | 0.284514 | 0 | 0.125 | 0.375 | 0.75 |
| 1 | 0 | 0.5 | 0.159898 | 0.996829 | 0 | 0.25 | 0.375 | 0.75 |
| 1 | 0 | 0.5 | 0.159898 | 0.878513 | 0 | 0.125 | 0.375 | 0.75 |
| 1 | 0 | 0.5 | 0.159898 | 0.16285 | 0 | 0.375 | 0 | 0.75 |
| 1 | 0 | 0.5 | 0.159898 | 0.487663 | 0 | 0.125 | 0.125 | 0.75 |
| 1 | 0 | 0.5 | 0.159898 | 0.178348 | 0 | 0.125 | 0.125 | 0.75 |
| 1 | 0 | 1 | 0.159898 | 0.950175 | 0 | 0.125 | 0.125 | 0.75 |
| 1 | 0 | 1 | 0.159898 | 0.637318 | 0 | 0.125 | 0.125 | 0.75 |
| 1 | 0 | 0.5 | 0.159898 | 0.434344 | 0 | 0.125 | 0.125 | 0.75 |
| 1 | 0 | 0.5 | 0.159898 | 0.144899 | 0 | 0.125 | 0.375 | 0.75 |
| 1 | 0 | 1 | 0.159898 | 0.553015 | 0 | 0.125 | 0.125 | 0.75 |
| 1 | 0 | 0.5 | 0.159898 | 0.195624 | 0 | 0.125 | 0.375 | 0.75 |
| 1 | 0 | 0 | 0.159898 | 0.614239 | 0 | 0.125 | 0.125 | 0.75 |
| 1 | 0 | 0 | 0.159898 | 0.822912 | 0 | 0.125 | 0.125 | 0.75 |

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| 1 | 0 | 1 | 0.159898 | 0.760404 | 0 | 0.125 | 0.125 | 0.75 |
| 1 | 0 | 0 | 0.159898 | 0.323586 | 0 | 0.125 | 0.125 | 0.75 |
| 1 | 0 | 0.5 | 0.159898 | 0.123355 | 0 | 0.125 | 0 | 0.75 |
| 1 | 0 | 0 | 0.159898 | 0.942715 | 0 | 0.125 | 0.375 | 0.75 |
| 1 | 0 | 1 | 0.159898 | 0.394683 | 0 | 0.125 | 0.125 | 0.75 |
| 1 | 0 | 1 | 0.159898 | 0.80263 | 0 | 0.125 | 0.125 | 0.75 |
| 1 | 0 | 1 | 0.159898 | 0.367627 | 0 | 0.125 | 0.125 | 0.75 |
| 1 | 0 | 1 | 0.159898 | 0.905552 | 0 | 0.125 | 0.125 | 0.75 |
| 1 | 0 | 1 | 0.159898 | 0.877426 | 0 | 0.125 | 0.25 | 0.75 |
| 1 | 0 | 1 | 0.159898 | 0.107241 | 0 | 0.125 | 0.125 | 0.75 |
| 1 | 0 | 0.5 | 0.159898 | 0.59748 | 0 | 0.125 | 0 | 0.75 |
| 1 | 0 | 0 | 0.159898 | 0.058075 | 0 | 0.125 | 0.125 | 0.75 |
| 1 | 0 | 0 | 0.159898 | 0.308686 | 0 | 0.25 | 0.375 | 0.75 |
| 1 | 0 | 0.5 | 0.159898 | 0.048705 | 0 | 0 | 0.125 | 0.75 |
| 1 | 0 | 0.5 | 0.159898 | 0.504973 | 0 | 0.125 | 0.125 | 0.75 |
| 1 | 0 | 1 | 0.159898 | 0.069703 | 0 | 0.125 | 0.125 | 0.75 |
| 1 | 0 | 0.5 | 0.159898 | 0.870218 | 0 | 0.375 | 0.25 | 0.75 |
| 1 | 0 | 0.5 | 0.159898 | 0.737872 | 0 | 0.125 | 0.125 | 0.75 |
| 1 | 0 | 0.5 | 0.159898 | 0.673561 | 0 | 0.125 | 0.125 | 0.75 |
| 1 | 0 | 1 | 0.159898 | 0.146903 | 0 | 0.125 | 0.125 | 0.75 |
| 1 | 0 | 1 | 0.159898 | 0.608641 | 0 | 0.125 | 0.375 | 0.75 |
| 1 | 0 | 0 | 0.159898 | 0.191375 | 0 | 0.375 | 0.125 | 0.75 |
| 1 | 0 | 0 | 0.159898 | 0.05482 | 0 | 0.125 | 0.125 | 0.75 |
| | | | | | | | | |

**Table 7-11: Sample training datasets**

## Declaration

The thesis my original work, has not been presented for a degree in only other university and that all sources of material used for the thesis have been duly acknowledged

---

ESKINDER MESFIN

---

**Advisor: Dr. Rahel B.**