“Application of Machine Learning for Climate Modeling”

B.E. Project Report

Submitted in partial fulfillment of the requirements

For the degree of

Bachelor of Engineering
in
Computer Engineering

By
Mr. Sarthak Langde
Ms. Tejal Tari
Mr. Abhishek Brahmachary
Mr.Prathmesh Mundhe

Supervisor
Prof. Tushar Ghorpade

Department of Computer Engineering
Ramrao Adik Institute Of Technology

Dr. D. Y. PatilVidyanagar,Sector 7, Nerul, Navi Mumbai 400 **706.**
(Affiliated to University of **Mumbai)**

April - 2017

Ramrao Adik Institute of Technology

(Affiliated to the University of **Mumbai)**

Dr. D. Y. PatilVidyanagar, Sector 7, Nerul, Navi Mumbai 400 **706.**

CERTIFICATE

This is to certify that, the project 'A' titled

<u>"Application of Machine Learning for Climate Modeling"</u>

is a bonafide work done by

Mr. Sarthak Langde
Ms. Tejal Tari
Mr. Abhishek Brahmachary
Mr.Prathmesh Mundhe

and is submitted in the partial fulfillment of the requirement for the degree of

Bachelor of Engineering
in
Computer Engineering
to the
University of Mumbai

Supervisor
Prof.Tushar Ghorpade

Project Co-ordinator	Head of Department	Principal
Prof. Aditi Chhabria	Dr. Leena Ragha	Dr.Ramesh Vasappanavara

Project Report Approval for B.E.

This is to certify that the project ‘A’ entitled *“Application of Machine Learning for Climate Modeling”* is a bonafide work done by*Sarthak Langde, Tejal Tari, Abhishek Brahmachary, Prathmesh Mundhe* under the supervision of *Prof .Tushar Ghorpade***.** This dissertation has been approved for the award of *Bachelor’s Degree in Computer Engineering, University of Mumbai*.

Examiners :

1. . . . .

2. . . . .

Supervisors :

1. . . . .

2. . . . .

Principal :

. . . .

Date :

Place :

DECLARATION

I declare that this written submission represents my ideas in my own words and where other's ideas or words have been included, I have adequately cited and referenced the original sources. I also declare that I have adhered to all principles of academic honesty and integrity and have not misrepresented or fabricated or falsified any idea/data/fact/source in my submission. I understand that any violation of the above will be cause for disciplinary action by the Institute and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been taken when needed.

Sarthak Langde	13CE1018	______________
Tejal Tari	13CE1096	______________
Abhishek Brahmachary	13CE3007	______________
Prathmesh Mundhe	13CE1100	______________

ACKNOWLEDGEMENT

It gives us great pleasure in presenting this project on "**Application of Machine Learning for Climate Modeling**". Our project is a result of dedicated team effort. We wish to express our sincere thanks to **Prof.Tushar Ghorpade** for being our project guide and for guiding us throughout the project.

We would also like to recognize the sincere efforts of our professor especially **Dr. Leena Ragha** (Head of the Department, Computer Science),**Prof. Aditi Chhabria** and **Prof. Bhavana T**(Project Co-ordinators) for their step by step guidance , as well as **Dr. Ramesh Vasappanavara** (Principal) for their suggestions, time to time help which helped us achieving goals and overcoming difficulties.

We would also like to thank all the faculty and staff who have helped us directly or indirectly.
Sincere thanks for all the support.

ABSTRACT

Climate systems are essentially chaotic systems. The impacts of potential climatic changes pose important scientific and societal challenges. Increasing temperatures due to global warming pose a big threat to current hydrological processes. The South-West Summer Monsoons are one of the most unpredictable monsoons. The risk posed to the industrial sectors and the agricultural sectors due to this unpredictability cannot be underestimated. There has been a rise in extreme conditions of temperature and pressure in various parts of the Indian subcontinents due to these chaotic climate conditions. These have led to disasters like cyclone and heavy rainfall on one end and droughts and famine on the other. These disasters cause a destruction of the crop produce and lead to severe famines. Countless number of farmers commit suicide every year to escape this calamity. In this project, an attempt has been made to bring a certain factor of predictability of these extreme conditions. In this project we are using the advanced artificial intelligence algorithms of neural networks and take advantage of high performance computing to predict climatic conditions of some of the regions of India and make an attempt to predict the occurrence of rainfall, which in turn will help farmers with crop modelling.

INDEX

Chapter 1

INTRODUCTION

1.1 Application

Developing a rainfall forecasting and flood warning system for an extended period of time is a difficult task. Both internal and external characteristics of rainfall field depend on many factors including: vapour pressure, temperature, wind speed and its direction. A model based on statistical mechanics, Artificial Neural Networks (ANNs) is an efficient alternative. ANNs are computationally robust which have the ability to learn and generalize from examples to produce meaningful solutions to problems even when the input data contain errors or are incomplete [1].

In this project we have selected multi-layer feed-forward backpropagation ANN to build a system to efficiently predict the rainfall conditions of some of the regions in India so that this can be used further for crop modelling. A survey shows that nearly 89.7 lakh farmers in Maharashtra have been impacted by the drought. The figure is almost on a par with the population of Sweden. Maharashtra is already known for its farm crisis and reports the highest number of farmer's suicides in the country [2]. The primary objective of this paper is to analyse various climatic conditions, study the past available data and design a Neural Network for predicting the conditions leading to the disasters. The secondary objective is to focus on how the predictions can be helpful to farmers with their crop production. This paper discusses computational techniques like machine learning, statistics and data mining, also it enables research on climate informatics which will accelerate discovery and answer pressing questions in climate science.

1.2 Motivation

There are several motivations to do this project. One of them being the Chennai floods of 2015, where 470 people lost their life and over 18 lakh were displaced [3] due to the lack of awareness and prediction of weather and that prompted us to do the project that would predict the climatic conditions using the past data of various years and would help alert the people well in advance. Climate change and agriculture are interrelated processes, both of which take place on a global scale. Climate change affects agriculture in a number of ways, including through changes in average temperatures, rainfall, and climate extremes.

Despite technological advances, such as improved varieties, genetically modified organisms, and irrigation systems, weather is still a key factor in agricultural productivity, as well as soil properties and natural communities. The effect of climate on agriculture is related to variability's in local climates rather than in global climate patterns. The Earth's average surface temperature has increased by 1.5 °F (0.83 °C) since 1880.

Another motivation for proper climate prediction has been to help farmers with types of crops to grow, the season to grow them, etc. Proper prediction would help in the following ways:

1. **Improve crop production planning --** Climate predictions of seasonal weather patterns (e.g., drought, heat waves, cool planting season) help farmers decide which crops are most likely to flourish in the predicted growing season.
2. **Improve production input planning --** Farmers purchase a variety of crop inputs (e.g., fertilizers, pesticides, fungicides) that are utilized during various times of the growing seasons.
3. **Improved crop field operations planning --** Crop yields are sensitive to the timing of field operations. Climate predictions will provide lead time to help farmers mitigate the effects of adverse weather during the planting season.
4. **Mitigate negative soil impact --** Climate predictions of extreme weather events can help farmers plan their cropping and cultural programs to minimize the erosive impact of these events.

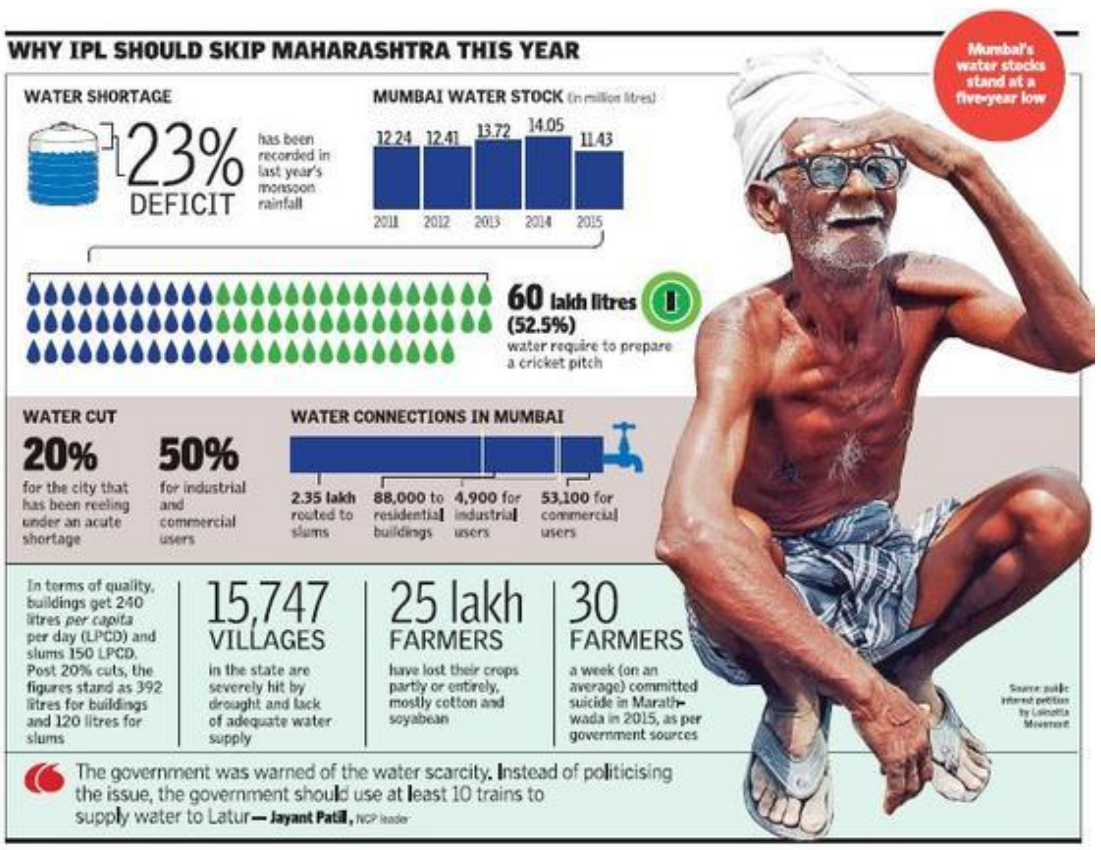


Figure 1.1: Severity of drought conditions in Maharashtra highlighted by the Times of India

As we can see from the figures, Figure 1.1 and Figure 1.2, nearly 89.7 lakh farmers in Maharashtra have been impacted by the drought that has devastated the kharif crop, official data shows. The figure is almost on a par with the population of Sweden. Maharashtra is already known for its farm crisis and reports the highest number of farmer's suicides in the country. The drought — brought on by a delayed and inadequate monsoon — is set to deepen the distress for its cultivators [4].

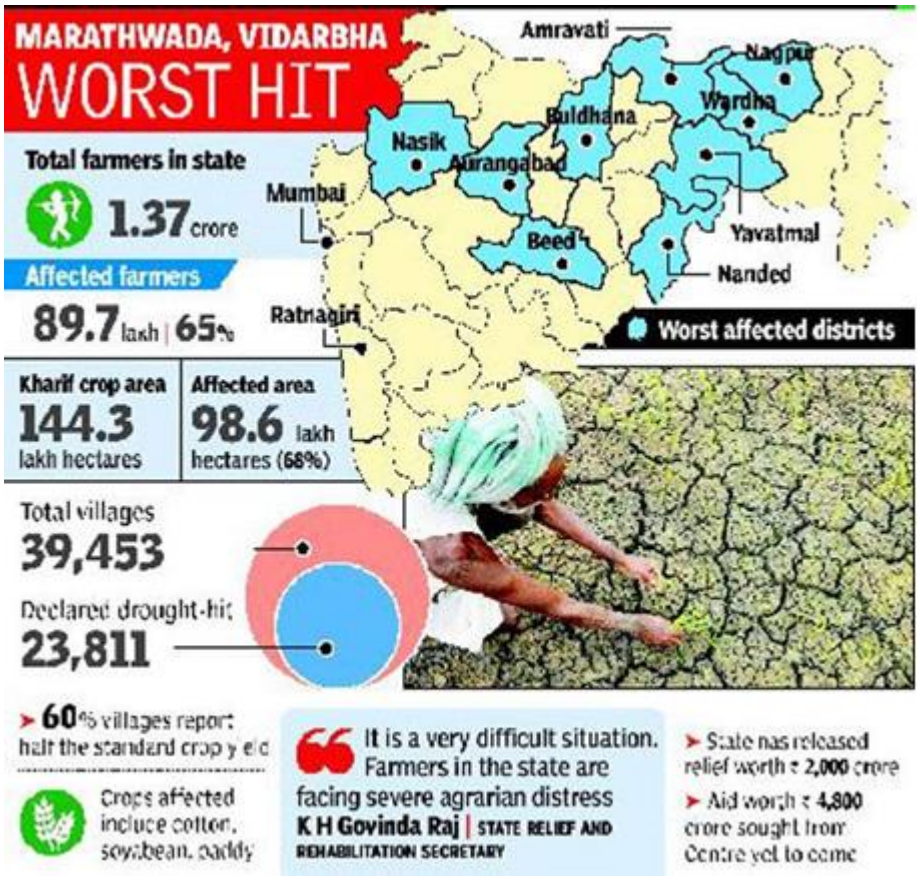

MARATHWADA, VIDARBHA
WORST HIT
Total farmers in state
1.37 crore
Affected farmers
89.7 lakh | 65%
Kharif crop area
144.3 lakh hectares
Affected area
98.6 lakh hectares (68%)
Total villages
39,453
Declared drought-hit
23,811
60% villages report half the standard crop yield
Crops affected include cotton, soybean, paddy
It is a very difficult situation. Farmers in the state are facing severe agrarian distress
K H Govinda Raj | STATE RELIEF AND REHABILITATION SECRETARY
State has released relief worth ₹ 2,000 crore
Aid worth ₹ 4,800 crore sought from Centre yet to come
Amravati
Nagpur
Wardha
Buldhana
Nasik
Aurangabad
Mumbai
Yavatmal
Beed
Nanded
Ratnagiri
Worst affected districts

Figure 1.2: Implications of lack of rainfall in the Vidharbhha Region of Maharashtra

1.3 Scope

The primary objective of this project will be to analyze the climatic conditions of the Marathwada region and build a prediction model that can accurately predict the rainfall in that region. The secondary objective will be focusing on how the predictions can be used effectively to make farmers lives easier.

1.4 Problem Definition

Understanding the climate data and bridging the gap between data and understanding by using climate informatics and computational techniques like machine learning, statistics and data mining. This study will enable research on climate informatics which will accelerate discovery and answer pressing questions in climate science.

1.5 Organization of the report

The flow of report goes as follows:-

- Chapter 2 presents a literature survey done on situational factors affecting climate, list of technical parameters and a comparison of various forecasting models developed previously.
- Chapter 3 presents proposed work and methodology for climate modelling using Artificial Neural Networks.
- Chapter 4 deals with the design of the system and the overall system model.
- Chapter 5 gives the overview of the proposed results of the system.

Chapter 2

LITERATURE SURVEY

2.1 Research Paper Survey

B.D.Kulkarni, S.D.raskar, R.N.Keshavamurty , C.venkatesan, S.S.Tambe (1997) [5] started stated that for predicting rainfall conditions of various regions of india proved beneficial when trained with error back propogation (EBP) algorithm on multilayered feedforward NN [3]. Thus they found that the results gained,strongly note the similarity with the statistical models and the predictions obtained from the network models shows that they can provide a powerful mechanism for Indian Summer Monsoon Rainfall (ISMR) predictions.

S.Weesakul, N.Q.Hung, N.K.Tripathi, and M.S.Babel (2009)[6]developed ANN model that predicted rainfall and helped flood management in Bangkok, Thailand [4][5]. They gathered 4 years of hourly data from 75 rain gauge stations of few areas to develop the ANN model and used a combination of meteorological parameters like cloudiness, wet bulb temperature, air pressure, relative humidity, rainfall at the point of forecasting and rainfall at the surrounding stations, as an input data, advanced ANN model to apply with continuous data containing rainy and non-rainy period and allowed model to issue forecast at any moment. Results show that the rainfall forecasts for Bangkok from 1 to 3 h ahead were highly satisfactory. The ANN model was found to be efficient in fast computation and capable of handling the noisy and unstable data that are typical in the case of weather data. The predicted values of all 75 rain gauge stations matched well with the observed rainfall for forecasts with short lead times of 1 or 2h.

A.J.Litta, U.C.Mohanty, and Sumam Mary Idicula (2013)[7]developed ANN model with Levenberg-Marquardt (LM algorithm to derive thunderstorm forecasts from 1 to 24 h ahead at Kolkata [6]. The objective of their study was to use ANNs to predict temperature and relative humidity during thunderstorm days from 1 to 24 h ahead using prior weather data as inputs. LM algorithm appears to be the best learning algorithm for mapping the different chaotic relationships. The developed ANN model with LM algorithm was used to predict surface temperature and relative humidity at hourly intervals with 1, 3, 6, 12, and 24 h ahead during same severe thunderstorm cases. Analysis of the results reveals that the 1, 3, and 24 h ANN models were able to predict hourly temperature and relative humidity adequately with sudden fall and rise. Hence they concluded that the ANN model with LM algorithm has well predicted the hourly temperature and relative humidity in terms of sudden fall of temperature and rise of humidity during thunderstorm hours.

Lubna. B.Mohammed, Eman A. Abdelhafez, WalidShaheen, and Mohammad. A.Hamdan (2013) [8]developed NARX model using different training algorithms to estimate hourly solar radiation from meteorological data sets. The model was constructed and tested using MATLAB software. The comparative analysis between the estimated data and measured data showed that NARX model has the ability to recognize the relationship between the input and output variables and predict hourly solar radiation accurately.Some important qualities about NARX networks with gradient-descending learning gradient algorithm have been reported: (1) learning is more effective in NARX networks than in other neural network (the gradient descent is better in NARX) and (2) these networks converge much faster and generalize better than other networks. NARX networks are often much better at discovering long time – dependences than conventional recurrent neural networks. The statistical error analysis shows the prediction accuracy based on NARX model. Different training algorithms were compared to select the best suited algorithm. The Marquardt–Levenberg learning algorithm with a minimum root mean squared error (RMSE) and maximum coefficient of determination (R) was found as the best in both training and validation period when applied in NARX model.

The review of literature in this chapter has concentrated largely on Artificial neural network algorithm which is an attractive inductive approach in rainfall prediction owing to their highly nonlinearity, flexibility and data driven learning in building models without any prior knowledge about catchment behavior and flow processes.Also, NARX network can be easily and efficiently applied to prediction of time series using embedding theory to reconstruct the input of NARX network.Not only are NARX neural networks computationally powerful in theory, but they have several advantages in practice. For example, it has been reported that gradient-descent learning can be more effective in NARX networks than in other recurrent architectures with “hidden states”.It was found that LM algorithm was faster and achieved better performance than the other algorithms in learning.

Table 2.1.1: Comparison of various machine learning techniques used for climate modelling

Name	**Year**	**System**	**Implementation**	**Advantages**
Wong	2003	Soft computing technique that uses ANN and Fuzzy Logic.	They have used SOM first to divide the data into sub-population and hopefully reduce the complexity of the whole data space to something more homogeneous. After classification, they have used BPNNs to learn the generalization characteristics from the data within each cluster. They extracted fuzzy rules for each cluster. The fuzzy rule base is then used for rainfall prediction.	The advantage of allowing analyst to understand and interact with the model using fuzzy rules
Christodoulou	2004	Weather radar instead of rain-gauges measuring rainfall on the ground.	The neural SOM and the statistical KNN classifier were implemented for the classification task using the radar data as input and the rain-gauge measurements as output. The rainfall rate on the ground was predicted based on the radar reflections with an average error rate of 23%.	They have observed that the prediction of rainfall rate based on weather radar measurements is possible
Lin	2005	ANN	have developed a neural network with two hidden layers to forecast typhoon rainfall	found that, the forecasting model can produce reasonable forecasts
Guhathakurta, Kumarasiri	2006	ANN for Long-Range Monsoon Rainfall	Two fundamentally different approaches for designing a	The authors believed that each of these models was

		Prediction for the Districts and Sub-Division Kerala based on the area weighted value of all district forecast	model, the statistical method based on autoregressive integrated moving average (ARIMA) and the emerging computationally powerful techniques based on ANN. feed-forward back-propagation architecture.	extended to make predictions several time steps into the future, where accuracies were found to be decreasing with the number of time steps. They have also studied and presented the success rates and rainfall trends within the monsoon seasons
Paras, Chattopadhyay	2007	ANN with parameters like maximum temperature, minimum temperature and relative humidity using the features extracted over different periods as well as from the weather parameter time-series itself. The learning rate parameter was fixed at 0.4 and the momentum rate was chosen 0.9.	Back propagation for supervised learning using the data recorded at a particular station. Genetic Optimizer (GO) was used by them to optimize the ANN architecture	The model could be suitably adapted for making forecasts over larger geographical areasthey have compared the performance of the neural net model with conventional persistence forecast and found that the Neural Net, in the form of Multilayer Perceptron was adroit in the prediction of monsoon rainfall over India
Chattopadhyay	2008	ANN model	Supervised back propagation learning procedure.	They have concluded that the eleven-hidden-nodes three-layered neural network has more efficacy than asymptotic regression in the present forecasting task
Xinia, Karmakar	2009	Model based on empirical mode decomposition (EMD) and the RBF neural network (RBFN) for rainfall prediction.	Developed a three layer perception feed forward back propagation deterministic and probabilistic artificial neural network models to predict long-range monsoon rainfall over the subdivision EPMB	The method had a high accuracy in denoising and prediction of the rainfall sequencethe performance of the model in probabilistic forecast was better evaluated over deterministic forecast
Baboo and Shareef	2010	ANN model temperature, wind speed, humidity, etc.,	Used back propagation neural network for predicting the temperature based on the training set provided to the neural network.Other techniques used are linear correlation analysis (LCA), modular artificial neural network (MANN), K-nearest-neighbors (K-NN), and linear regression (LR).	Ultimately, they have obtained 99.79% of accuracy in the training and 94.28% of accuracy in testing. From these results the authors could able to predict the rainfall for the future
Chadwick, El-Shafie, Geethaand Selvaraj	2011	ANN model and Multi Regression model (MLR) Multi-Layer Perceptron Neural	Support Vector Machine (SVM), back propagation neural network model. They have used statistical	Found that the ANN model shows better performance than the MLR model they

		network (MLP-NN), Radial Basis Function Neural Network (RBFNN) and Input Delay Neural Network (IDNN).	parameters such as the Root Mean Square Error, Mean Absolute Error, Coefficient Of Correlation and BIAS to make the comparison between the two models, Wavelet transform was used for extraction of approximate and detail coefficient of the rainfall data series.	concluded that IDNN could be suitable for modeling the temporal dimension of the rainfall pattern, thus, provides better forecasting accuracy finally they concluded that the feed forward back propagation ANN can describe the behavior of rainfall-runoff relation more accurately than the classical regression model
Abhishek	2012	ANN model with the performance measures such as Mean square error, and Normalized mean square error on testing as well as training data set for short term prediction	three algorithms were tested in multi-layer architecture: Back Propagation Algorithm (BPA), Layer Recurrent Network (LRN) and Cascaded Back-Propagation (CBP)	the authors have found that BPA is the best algorithm out of the three tested They have seen that the performance measures such as Mean square error, and Normalized mean square error on testing as well as training data set for short term prediction were optimal in comparison with other network such as Jordon Elmann Neural Network, SOFM (Self organized feature map), RNN (Recurrent neural network)
Chau	2013	ANN model	The proposed preprocessing techniques included moving average (MA) and singular spectrum analysis (SSA). The modular models were composed of local support vectors regression (SVR) models or/and local artificial neural networks (ANN) models.	they have showed that the MA was superior to the SSA when they were coupled with the ANN
Singh and Borah	2013	ANN model	Used feed-forward back-propagation neural network algorithm for Indian Summer Monsoon Rainfall (ISMR) forecasting. Based on this algorithm, they have proposed the five neural network architectures designated as BP1, BP2, BP3, BP4, BP5 using three layers of neurons (one input layer, one hidden layer and one output layer). The data	Results of the proposed method clearly exhibit superiority of their model over the considered existing model. The seasonal rainfall values over India for next 5 years have also been predicted

			set was trained and tested separately for each of the neural network architecture (BP1–BP5). The forecasted results obtained for training and testing data are compared with existing model	

Chapter 3
PROPOSAL

3.1 Problem Statement

Study the climate change in terms of large spatio-temporal data of rainfall and then quantify it accordingly. Analyze the rainfall data at multiple spatial scales of district and multiple temporal scales of monthly rainfall. The use of machine learning will be made to predict the rainfall as well as its change on other sectors like agriculture.

3.2 Proposed Work

- Satellite data analysis
- Understanding the data format and data structures used
- NetCDF data analysis (in Java and C language)
- Long term data analysis (111 years rainfall)
- Development of algorithms for spatial and temporal analysis
- Development of visualization tools
- Prediction Model using ANN

3.3 Proposed Methodology

3.3.1 Neural Networks

A neural network (Artificial Neural Network) is a data processing system consisting of many of simple highly interconnected processing elements and from the past few years, they have been studied very intensively for their capability to solve a large domain of problems. [9] Since problems with neural networks learning processes emerged, their mathematical models have also been created. For every problem statement, a minor modification to the original definition of neural networks yielded better results. A three layer neural network has been proved to be a universal function approximator [10] and finds its use in a number of fields like control theory, robotics, speech recognition, pattern recognition, data compression, expert systems and many others [11].

In this section, we define the mathematical model of the neural network and the learning process thus used to arrive at an efficient method to forecast the stock market.

The developed neural network model is based on one of the neural network architecture called Multi-Layer Perceptron (MLP) network model (also known as multilayer feedforward network). This is the most popular network architecture in use today. In this type of network, the units each perform a biased weighted sum of their inputs and pass this activation level through a transfer function to produce their output, and the units are arranged in a layered feedforward topology. The network thus has a simple interpretation as a form of input-output model, with the weights and thresholds (biases), the free parameters of the model. Such networks can model functions of almost arbitrary complexity with the number of layers and the number of units in each layer, determining the function complexity.

Fig. 1 shows a generalised flowchart adopted for using neural networks to solve problems. Raw data is first collected in whatever reliable form available. Then, this data is screened for redundancy, inaccuracy and is normalised to function as the input vector for training a particular neural network. The normalisation range for data is specific to the activation function used in the neural network. Building the network comprises of understanding the complexity of the problem and narrowing down the network topology and other characteristics for the efficient working of the model. Training is again specific to network topology and the data set available. Once trained, a network maybe used for actual computation. The flowing sections discuss in brief about the model adopted in this project.

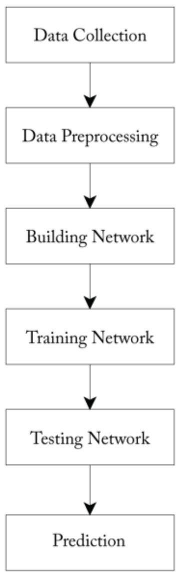


Fig. 1 – Modelling a Neural Network

Perceptron

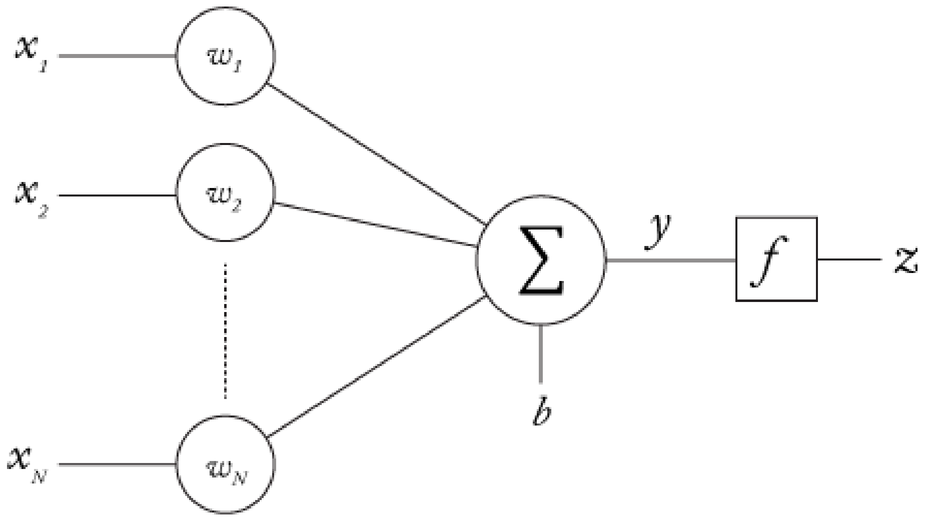


Fig. 2 – Diagram of a Perceptron

A perceptron is a processing unit that takes in several inputs and produces a single output. Fig.2 shows a perceptron with N inputs denoted by$x_1, x_2 \cdots\ x_N$. Each of these inputs is associated with weights denoted by$w_1, w_2 \cdots\ w_N$. y is the sum of all weighted inputs with an additional variable called the bias, b which then used to activate the perceptron using an activation function. The MLP neural network uses a non-linear activation function produces an output z.

$$y = \sum_{i=1}^{N}(x_i \,.\, w_i) + b$$

Eq. 1

$$z = f(y) = \frac{1}{1+e^{-y}} \;, f(y)\ \epsilon\ [0,1]$$

Eq. 2

The activation function, f described here is a sigmoidal function which has an input range of $[-\infty, \infty]$ and maps it to $[0,1]$. For an error backpropagation algorithm, the activation function needs to be differentiable, smooth, monotonic, and bounded. A sigmoid function seems to be best suited since it has all properties required. Here, both the hidden and output layers have the same activation function to ensure a uniform learning rate for all nodes.

Layers

Each network has got exactly one input layer, K hidden layers and one output layer. The number of inputs and outputs of the neural network is heavily dependent of the size of the data set available for training. The input corresponds to the dimension of input matrix. The number of neurons in the output corresponds to the problem at hand (In this case, Length of forecast of stock market). The number of hidden layers as well as the dimension of neurons in each hidden layer is proportional to the ability of the network to approximate more complicated functions. A neural network with a complicated topology is required to solve problems with higher interdependencies of system elements. But, it is not always true that a complicated topology is more efficient than a simple one, in producing results. This is because a complicated network is more sensitive to noise and minor fluctuations in the system. [11] The most efficient topology would be one that requires minimal computational power and a produces reliable and accurate result consistently, for a small training data set.

Weights

The time taken to train the weights of a neural network subject to a particular data set may be used as a parameter to discuss the efficiency of a neural network and its topology [12].Training here, refers to the systematic updating of weights within a given range to produce a certain output. This project makes use of a supervised learning algorithm. Supervised learning is a machine learning technique that sets parameters (weights) of a neural network from training data. Supervised learning requires the training set to have two parts, the input set and the target set. The input set is presented to the neural network, which computes the output for a particular set of randomly initialised weights. The output (set) of the system is then compared with the target set and the error is used to update weights. This iteration runs till the correlation between the target set and output set is theoretically, one.

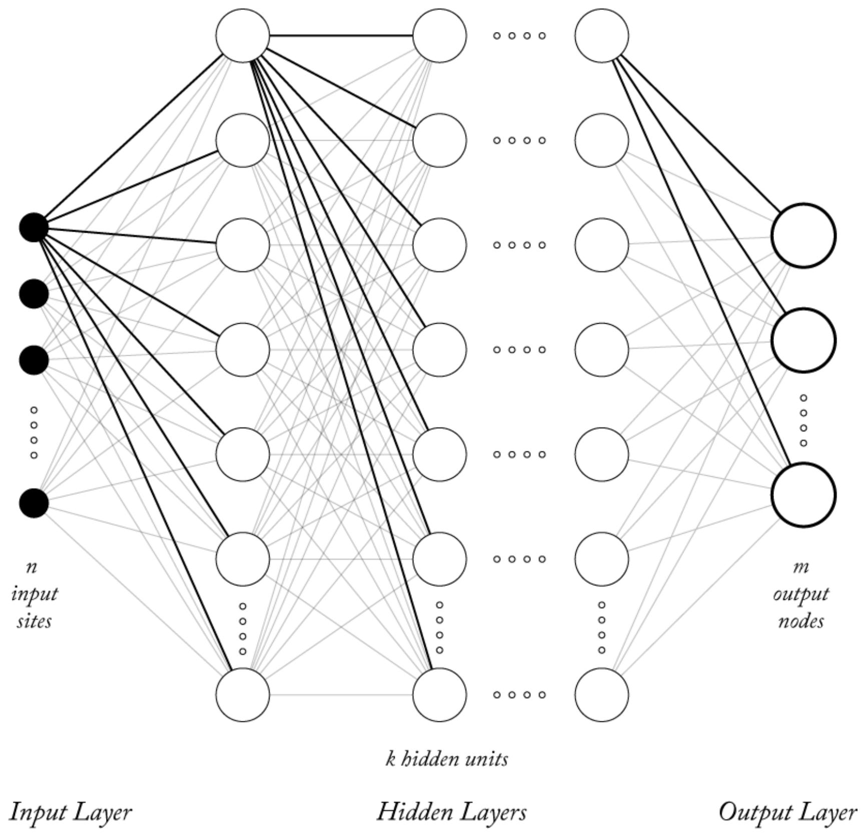


Fig. 3 – Feedforward Neural Network Model

Training Algorithm

In this project, we discuss feed forward neural networks with error backpropogation using gradient descent as the training algorithm. The task of the learning neural network is to set the value of its weights for any valid input value after having seen output values. The backpropagation algorithm looks for the minimum of the error function in weight space using the method of gradient descent. The combination of weights which minimizes the error function is considered to be a solution of the learning problem. Since this method requires computation of the gradient of the error function at each iteration step, the error function used should be continuous and differentiable for the entirety of its domain. The main objective here is to minimize the error function.

Consider a neural network which has n inputs and m output units. Here, we may assume that the network consists of any number of hidden units and is capable of exhibiting any desired feed-forward connection pattern. It is required to have a training set $\{(x_1, t_1), \dots, (x_p, t_p)\}$ consisting of p ordered pairs of n- and m-dimensional vectors, which are called the input and output patterns. The activation functions at each node of the network are sigmoids and hence are continuous and differentiable. The weights of the edges are real numbers selected at random.

When the input pattern x_i from the training set is presented to this network, it produces an output o_i different in general from the target t_i. The objective here, is to make o_i and t_i identical for $i = 1,\ldots,p$, by using a learning algorithm. The average error function to be minimized is given by

$$\varepsilon_{av} = \frac{1}{N}\sum_{n=1}^{N}\varepsilon(n)$$

Eq. 3

Where,

$$\varepsilon(n) = \frac{1}{2}\sum_{i=1}^{P}\left|\left|o_i - t_i\right|\right|^2$$

The weights in the network are the only parameters that can be modified to make the quadratic error ε as low as possible. Because ε is calculated by the network exclusively through composition of the node functions, it is a continuous and differentiable function of the l weights $w_1, w_2,\ldots, w_l$in the network. Thus ε can be minimized by using an iterative process of gradient descent. The gradient of ε is calculated as shown below

$$\nabla\varepsilon = \left(\frac{\delta\varepsilon}{\delta w_1}, \frac{\delta\varepsilon}{\delta w_2}, \cdots, \frac{\delta\varepsilon}{\delta w_l}\right)$$

Each weight is then updated using the increment

$$\Delta W_i = -\gamma\frac{\delta\varepsilon}{\delta w_i}, i = 1,2\cdots l$$

Where γ represents a learning constant, i.e., a proportionality parameter which defines the step length of each iteration in the negative gradient direction. This extension of the original network now reduces the whole learning problem to the question of calculating the gradient of a network function with respect to its weights. The method to compute this gradient, is discussed in the section below [15].

Formal Definition -

Fig. 4 shows a network with n input sites, k hidden, and m output nodes. Let the weight between input site i and hidden unit j be denoted by $w_{ij}^{(1)}$. And the weight between hidden unit i and output unit *j* be denoted by $w_{ij}^{(2)}$. It may be assumed that the bias b of each unit is implemented as the weight of an additional edge. The sum of weight between the additional weight and the hidden unit *j* is denoted by$w_{nj}^{(1)}$. Similarly, the weight between the other additional weight and the output unit *j* is denoted by $w_{kj}^{(1)}$.

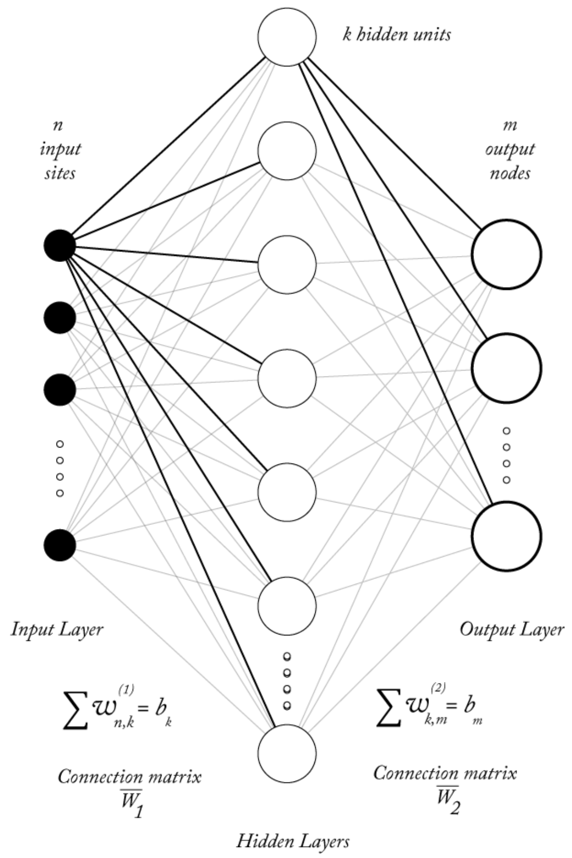


Fig. 4 – 3 Layer Neural Network

There are $(n \times k)$ weights between input sites and hidden units and $(k \times m)$ between hidden and output units. Let $\overline{\boldsymbol{W_1}}$ denote the $(n \times k)$ matrix with component $w_{ij}^{(1)}$at the i-th row and the j-th column. Similarly, let $\overline{\boldsymbol{W_2}}$denote the $(k \times m)$ matrix with components $w_{ij}^{(2)}$. The matrix of weights will be needed in the backpropagation step. Let the n-dimensional input vector be,$\boldsymbol{o} = (o_1, \dots, o_n)$. Then, the excitation net_j of the j-th hidden unit is given by

$$net_j = \sum_{i=1}^{n} w_{ij}^{(1)} o_i + b_j$$

Eq. 4

And the activation function is a sigmoid and the output $o_j^{(1)}$ of this unit is given by

$$o_j^{(1)} = s\left(\sum_{i=1}^{n} w_{ij}^{(1)} o_i + b_j\right)$$

Eq. 5

The excitation of all units in the hidden layer can be computed with the vector-matrix multiplication $(\boldsymbol{o}.\overline{\boldsymbol{W_1}})$. The vector $o^{(1)}$ whose components are the outputs of the hidden units is given by

$$o^{(1)} = s(\boldsymbol{o}\overline{\boldsymbol{W}}_{\boldsymbol{1}})$$

using the convention of applying the sigmoid to each component of the argument vector. The excitation of the units in the output layer is computed using the vector$\boldsymbol{o}^{(\boldsymbol{1})} = (o_1^{(1)}, \dots, o_k^{(1)})$. The output of the network is the m-dimensional vector $o^{(2)}$, where

$$o^{(2)} = s(\boldsymbol{o}^{(\boldsymbol{1})}\overline{\boldsymbol{W}}_{\boldsymbol{2}})$$

Algorithm –

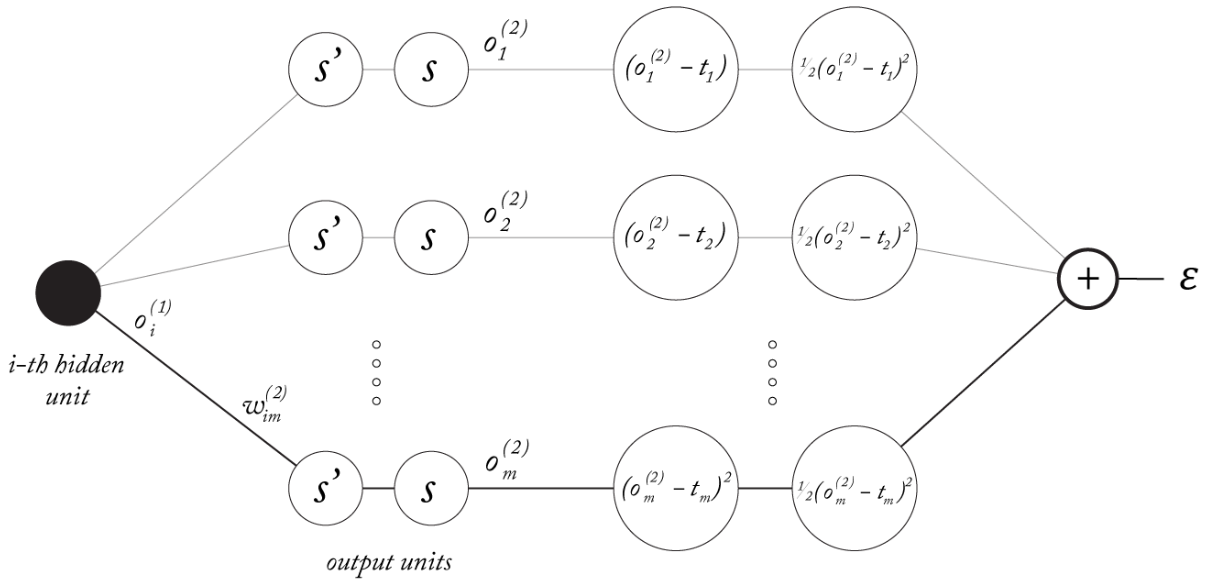


Fig. 5 – Error Function Computation

Fig. 5 shows the network for computation of the error function. Although this project deals with usage of more than one training pair, to simplify discussion, we deal with a single input-output pair (o, t) and generalize later to p training examples. The right sides compute the quadratic deviation $\frac{1}{2}(o^{(2)}{}_i - t_i)$ for the i-th component of the output vector and the left sides store $(o^{(2)}{}_i - t_i)$. Each output unit i in the original network computes the sigmoid s and produces the output $o^{(2)}{}_i$. Addition of the quadratic deviations gives the error ε. The error function for p input-output examples can be computed by creating p networks like the one shown, one for each training pair, and adding the outputs of all of them to produce the total error of the training set.

After choosing the weights of the network randomly, the backpropagation algorithm is used to compute the necessary corrections. The algorithm can be decomposed in the following four steps:

1) Feed-forward computation

The vector o is presented to the network. The vectors $o^{(1)}$ and $o^{(2)}$ are computed and stored. The evaluated derivatives of the activation functions are also stored at each unit.

2) Backpropagation to the output layer

Now, the first set of partial derivatives $\partial E / \partial w_{ij}^{(2)}$ are computed. The backpropagation path from the output of the network up to the output unit j is shown in the Fig below.

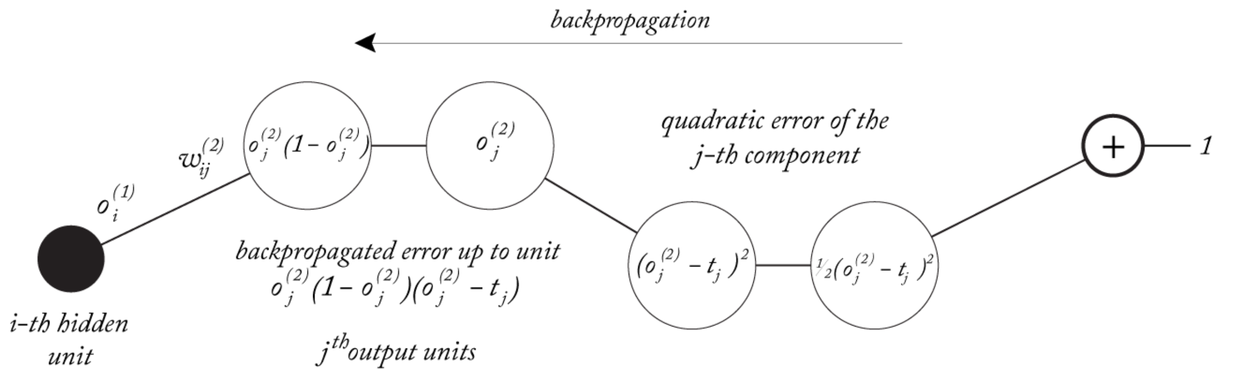

backpropagation
$w_{ij}^{(2)}$
$o_i^{(1)}$
$o_j^{(2)}(1-o_j^{(2)})$
$o_j^{(2)}$
quadratic error of the j–th component
$(o_j^{(2)}-t_j)^2$
$\frac{1}{2}(o_j^{(2)}-t_j)^2$
$+$
1
backpropagated error up to unit
$o_j^{(2)}(1-o_j^{(2)})(o_j^{(2)}-t_j)$
i–th hidden unit
j^{th}output units

Fig. 6 – Backpropagation to the Output Layer

A simple inspection of the backpropogation path gives us all the multiplicative terms which define the backpropagated error $\delta_j^{(2)}$. Therefore

$$\delta_j^{(2)} = o_j^{(2)}(1 - o_j^{(2)})(o_j^{(2)} - t_j)$$

And the partial derivative is

$$\frac{\partial E}{\partial w_{ij}^{(2)}} = \left[o_j^{(2)}\left(1 - o_j^{(2)}\right)\left(o_j^{(2)} - t_j\right)\right] o_i^{(1)} = \delta_j^{(2)} o_i^{(1)}$$

3) Backpropagation to the hidden layer

Now, we have to compute the partial derivatives $\partial E / \partial w_{ij}^{(1)}$. Each unit j in the hidden layer is connected to each unit q in the output layer with an edge of weight $w_{jq}^{(2)}$for $q = 1, \dots, m$. The backpropagated error up to unit j in the hidden layer must be computed taking into account all possible backward paths, as shown in Fig. 7. The backpropagated error is then

$$\delta_j^{(1)} = o_j^{(1)}\left(1 - o_j^{(1)}\right) \sum_{q=1}^{m} w_{jq}^{(2)} \; \delta_q^{(2)}$$

Eq. 6

Therefore the partial derivative we are looking for is

$$\frac{\partial E}{\partial w_{ij}^{(1)}} = \delta_j^{(1)} o_i$$

The backpropagated error can be computed in the same way for any number of hidden layers and the expression for the partial derivatives of ε keeps the same analytic form.

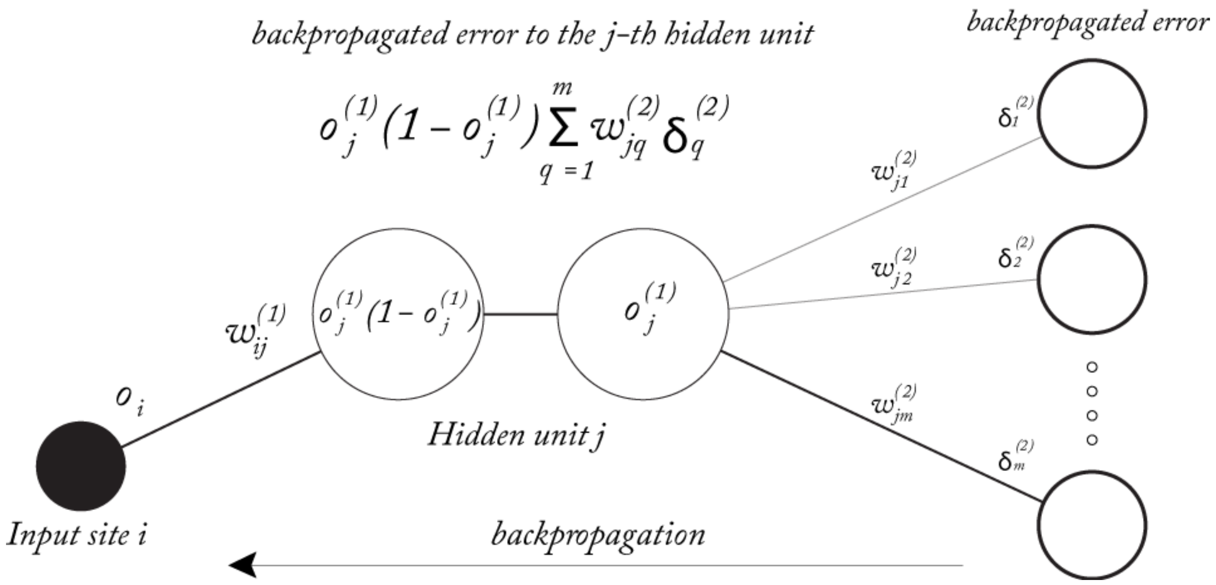


Fig. 7 – Backpropagation to the hidden layer

4) Weight updates

After computing all partial derivatives the network weights are updated in the negative gradient direction. A learning constant γ defines the step length of the correction. The corrections for the weights are given by

$$\Delta w_{ij}^{(2)} = -\gamma o_i^{(1)}\,\delta_j^{(2)}, for\ i = 1,\dots,k;\ j = 1,\dots,m,$$

$$\Delta w_{ij}^{(1)} = -\gamma o_i\,\delta_j^{(1)}, for\ i = 1,\dots,n;\ j = 1,\dots,k,$$

Here, $o_{n+1} = o_{k+1}^{(1)}$. In this case, where $p > 1$ input-output patterns, an extended network was used to compute the error function for each of them separately. The weight corrections

$$\Delta_1 w_{ij}^{(1)}, \Delta_2 w_{ij}^{(1)}, \dots, \Delta_p w_{ij}^{(1)}$$

The necessary update in the gradient direction used was

$$\Delta w_{ij}^{(1)} = \sum_{q=1}^{p} \Delta_q w_{ij}^{(1)}$$

Eq. 7

The algorithm is stopped when the value of the error function has become sufficiently small or the network runs for a few epochs.

3.4 Data Source

The dataset used was the High-resolution gridded dataset from the Climate Research Unit of the University of East Anglia. The data is available with a resolution of 0.5 degrees both latitude and longitude wise for six parameters. The data is monthly averaged and is available for a period of 1900 to 2015. The parameters considered are cloud cover (CLD), diurnal temperature range (DTR), wet-day frequency (FRE), vapour pressure (VAP) and precipitation (PRE).

The Marathwada region consists of the districts Aurangabad, Nanded, Latur, Beed, Jalna, Hingoli, Osmanabad and Parbhani of the Maharashtra state in India. The data was sampled at 17 spatial points spread over the region and averaged to obtain a single value for all parameters of entire Marathwada region.

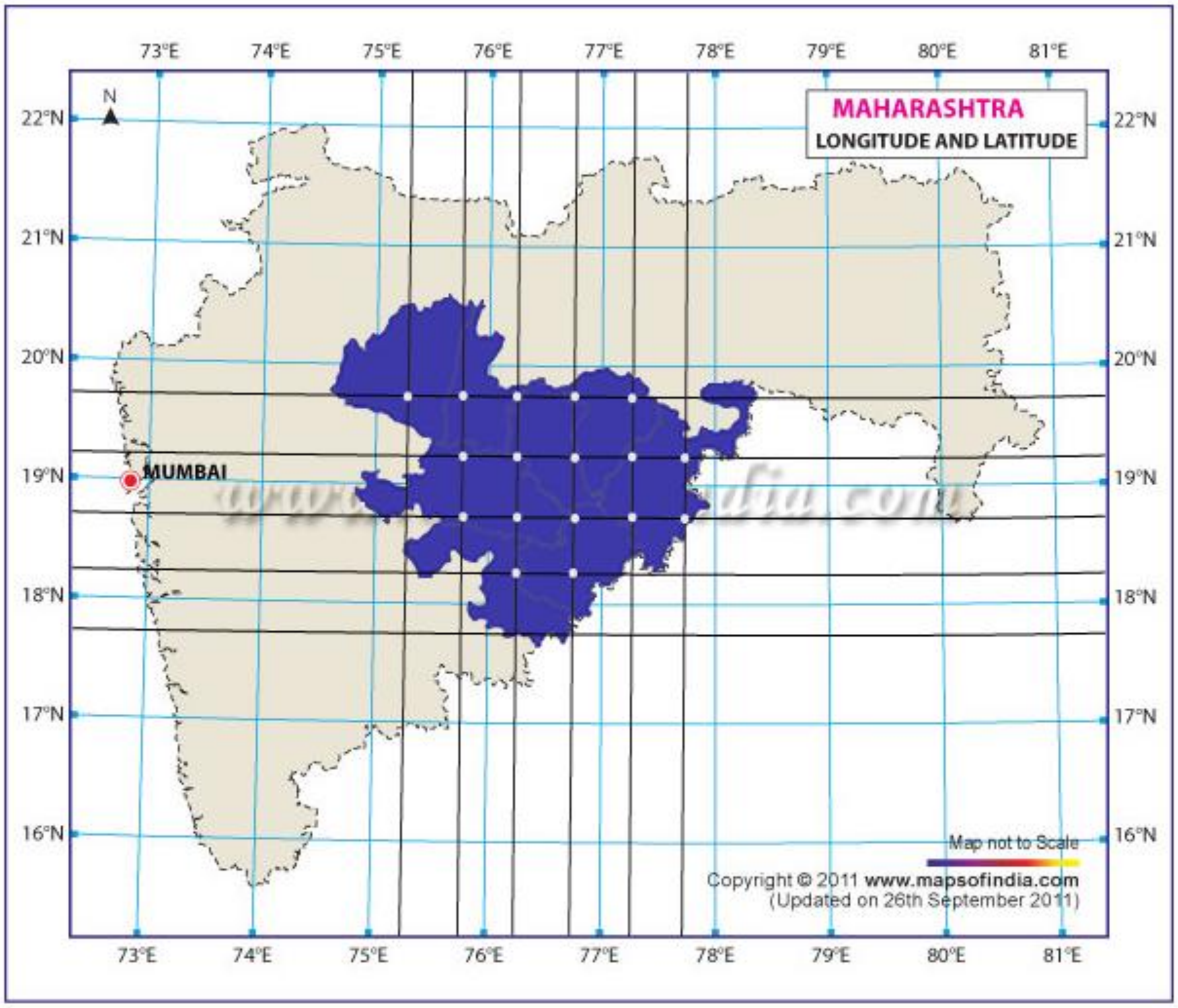


Figure 8: Grid points from which data was used as input

3.5 Data Pre-processing

Data normalization is usually done because values of raw data can vary widely and may lead to some machine learning algorithms not converging properly. Normalizing the data between the ranges of 0 to 1 is necessary for our activation function as it results in faster convergence for gradient descent. The range of [0, 1] is required for the activation function used is a unidirectional sigmoid. The formula used for normalization was:

$$x_{norm} = \frac{(x - x_{min})}{(x_{max} - x_{min})}$$

Eq. 8

where, x_{norm} is normalized value of the current data value x

$x_{min}$is the minimum value of the selected feature

$x_{max}$is the maximum value of the selected feature.

From Fig 3.1, the general steps involved for prediction of rainfall using the ANN model can be seen. For trend analysis, curve fitting using Levenberg-Marquardt algorithm was used to predict the seasonal trends of the monsoon.

3.6 Training Algorithm

The Levenburg-Marquardt (LM) method shows the most able convergence during the Back Propagation (BP) training process as it acts as a trade-off between the first-order optimization method (steepest-descent method) which has stable but quite slow convergence and the second-order optimization method (Gauss-Newton method) which has opposite characteristics [I].

The Levenberg-Marquardt algorithm:In the Back Propagation algorithm, the performance index F(w) to be minimized is defined as the sum of squared errors between the target outputs and the network's simulated outputs, viz.

$F(w)=e^Te$ …(1)

where $w=[w_1, w_2, ..., w_N]^T$ consists of all weights of the network, e is the error vector comprising the errors for all the training examples.

When training with the LM method, the increment of weights Δw can be calculated as follows

$\Delta w = [J^T J + \lambda I]\ J^T e$ …(2)

where, J is the Jacobian matrix, λ is the training parameter which is to be updated using the decay rate β depending on the outcome. In particular, λ is multiplied by the decay rate β ($0< \beta < I$) whenever F(w) decreases, while λ is divided by β whenever F(w) increases in a new step.

The following pseudo-codes explain the training process of the standard Levenberg-Marquardt algorithm,

```
Initialize the weights and parameter $\lambda$ ($\lambda = 0.01$ is suitable).
    1. Calculate the addition of squared errors of all inputs, F(w).
    2. Calculate the Jacobian matrix J.
    3. Solve Equation (2) to get the increment of weights $\Delta w$.
    4. Recalculate the addition of squared errors F(w) using $w+\Delta w$ as the trial w, and examine
       IF trial F(w) < F(w) in Step 2, THEN
              $w = w+\Delta w$
              $\lambda = \lambda . \beta$ ($\beta = 0.1$)
              go back to Step 2.
       ELSE
              $\lambda = \lambda / \beta$
              go back to Step 4.
              END IF [2]
```

3.6 Nonlinear Autoregressive Network with Exogenous Inputs

The nonlinear autoregressive network with exogenous inputs (NARX) is a recurrent dynamic network, with feedback connections enclosing several layers of the network. The NARX model is based on the linear ARX model, which is commonly used in time-series modeling.

The defining equation for the NARX model is

$$y(t)=f(y(t-1),y(t-2),\ldots,y(t-ny),u(t-1),u(t-2),\ldots,u(t-nu))$$

where, the next value of the dependent output signal $y(t)$ is regressed on previous values of the output signal and previous values of an independent (exogenous) input signal. We implement the NARX model by using a feedforward neural network to approximate the function *f*. This implementation also allows for a vector ARX model so the input and output can be multidimensional.

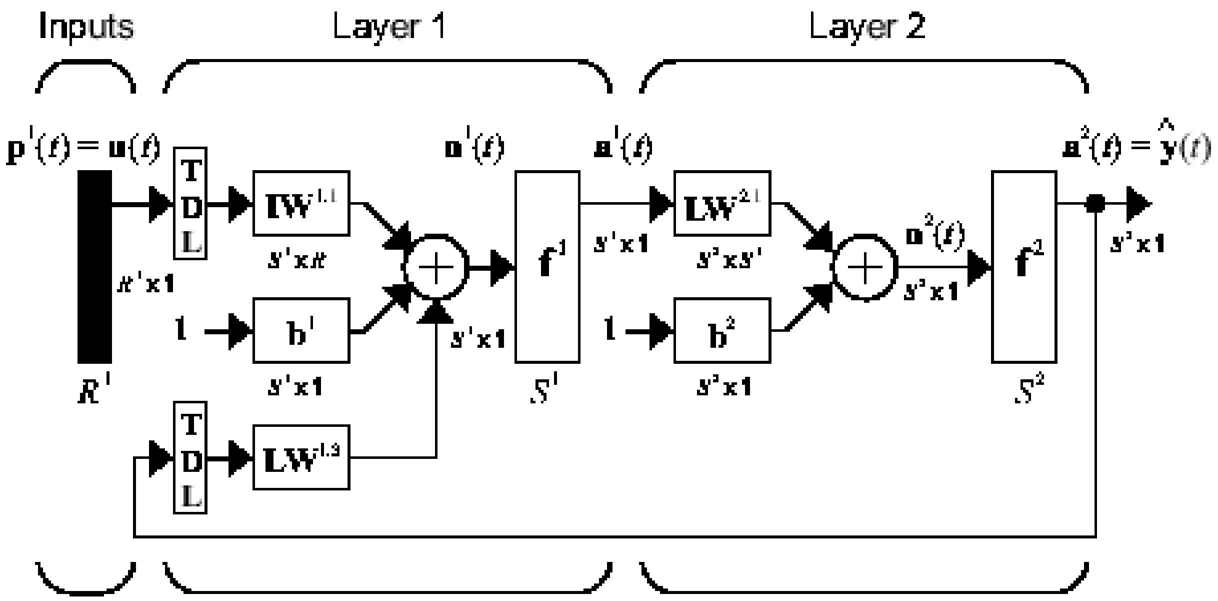


Figure 10: A two-layer feedforward network used for the approximation.

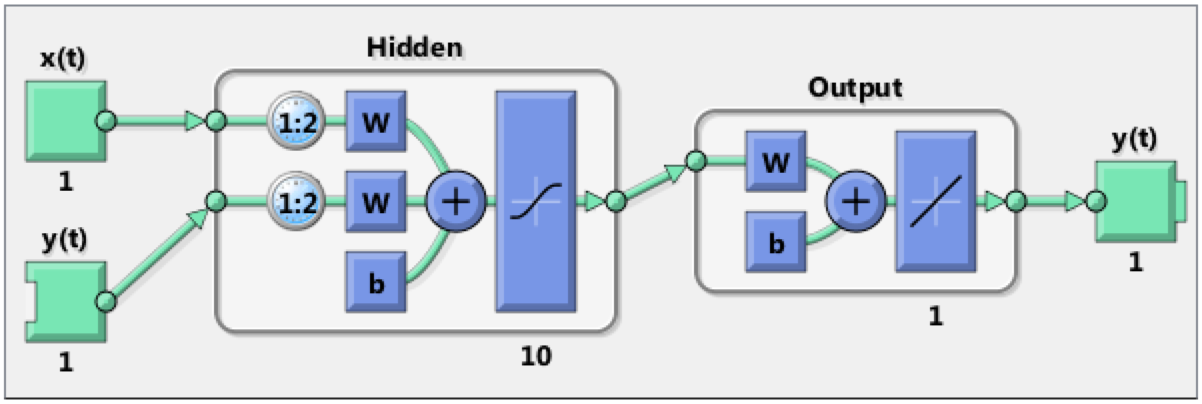


Figure 11:Open Loop Configuration of NARX

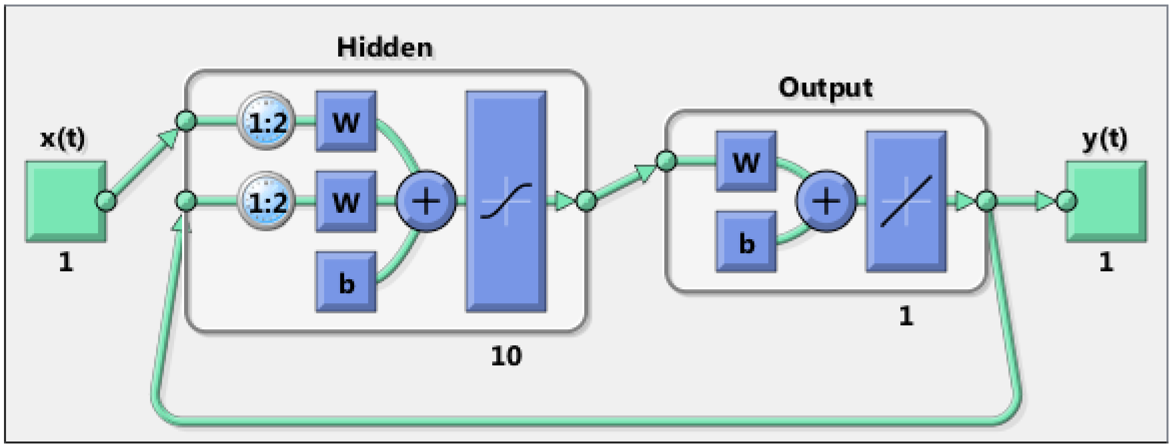


Figure 12:Closed Loop Configuration of NARX

All of the training is done in open loop (also called series-parallel architecture), including the validation and testing steps. The typical workflow is to fully create the network in open loop, and only when it has been trained (which includes validation and testing steps) is it transformed to closed loop for multistep-ahead prediction. Likewise, the R values in the GUI are computed based on the open-loop training results.

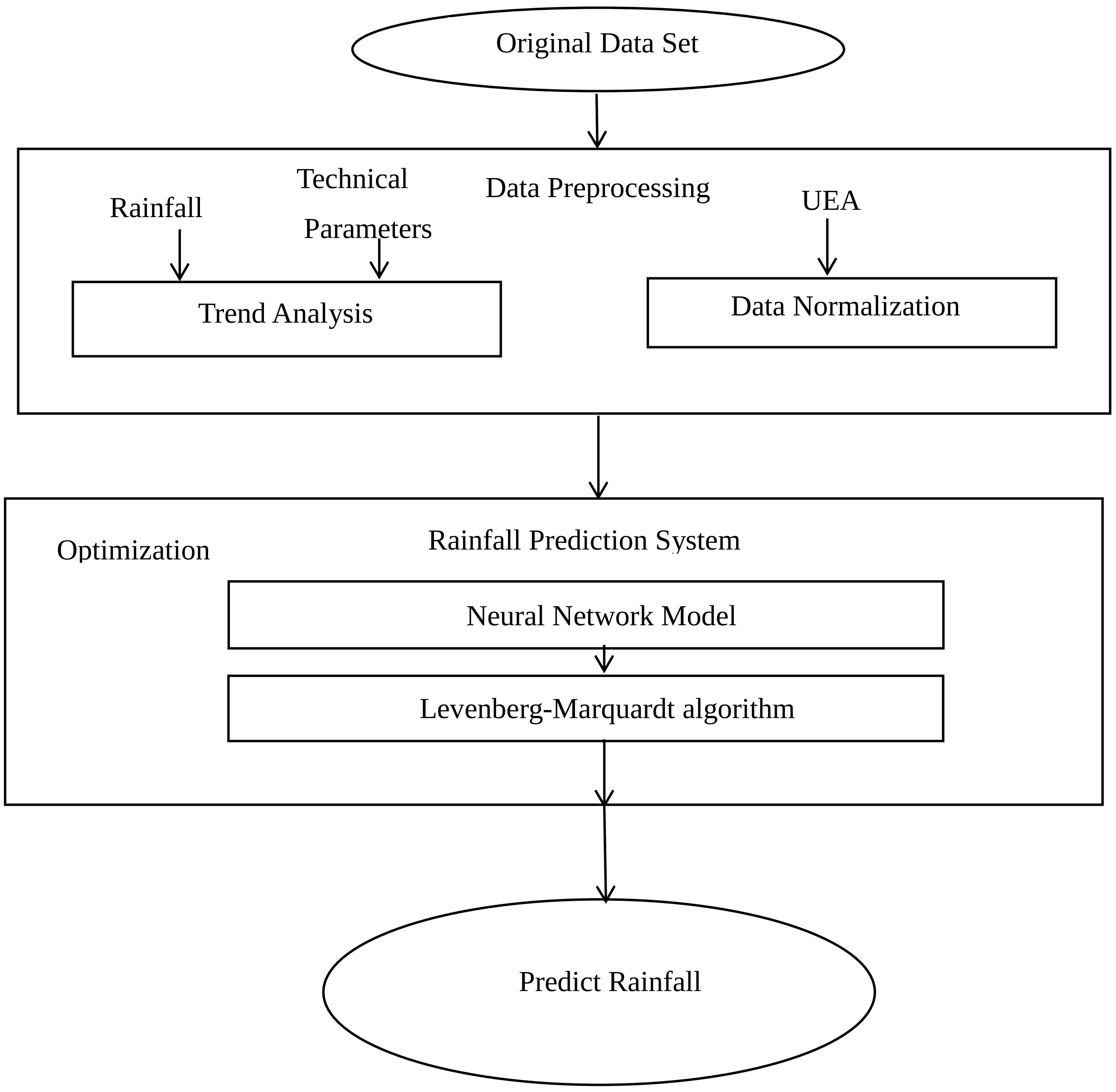


Fig 3.1 System model for rainfall prediction

3.6 HARDWARE AND SOFTWARE REQUIREMENTS

3.6.1 Software Requirements

1. Linux
2. MATLAB
3. R
4. Python
5. NetCDF

3.6.2 Hardware Requirements

1) 288 cores of Intel Itanium2 Processor at 1.66 GHz clock speed
2) 18 MB L3 cache per Dual core Processor
3) 608 GB Global Shared Memory
4) 2 x 300 GB System Disks
5) Gigabit Ethernet Interfaces
6) 4 GbpsFibre Channel Interfaces

Chapter 4
PLANNING AND FORMULATION

4.1 Project Schedule

Since the project will be done in the period from July 2016 to April 2017 the estimated time will be as follows
Semester VII: 14 weeks
Semester VIII: 15 weeks
The estimated time period plan would be:

Table 4.1. Schedule of the project

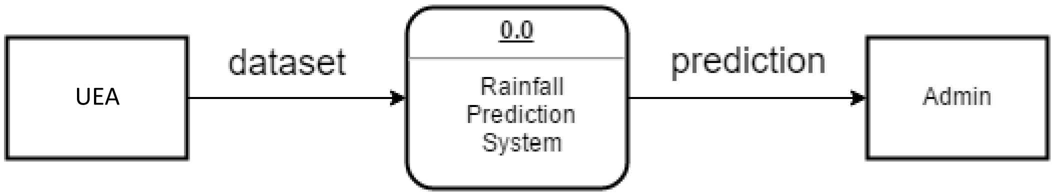
DURATION	**WORK**
July 2016-August 2016	Studying the existing system and Collecting the information.
August 2016-September 2016	Analysis of project requirements
September 2016-October 2016	Establishing the Project scope & Design of DFD,
December 2016-January 2016	Study &Analyze the Algorithms
January 2017-February 2017	Database Design & creation of GUI
February 2017-March 2017	Implementation of algorithm
March 2017-April 2017	Result Generation, Testing & documentation

Chapter 5

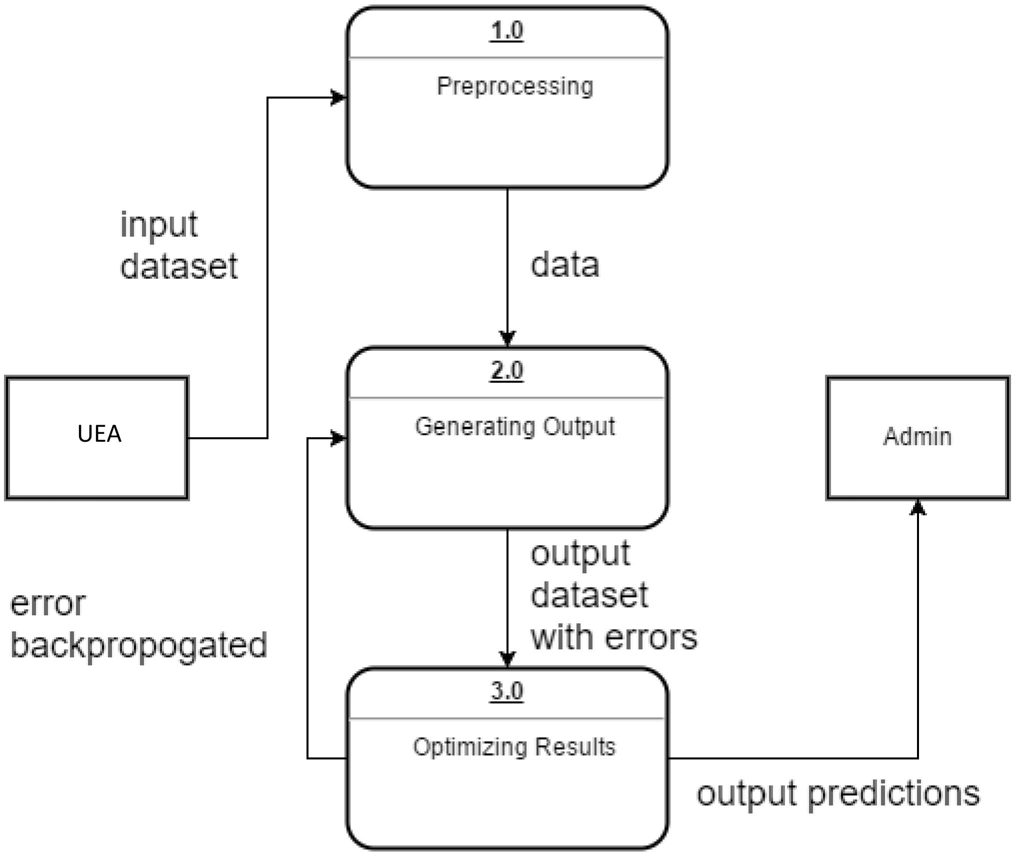
DESIGN OF THE SYSTEM

4.1. DFD

4.1.1 LEVEL 0 DFD



4.1.2 LEVEL 1 DFD



Chapter 6

RESULTS

5.1 Results

A thorough understanding of the climatic conditions of Marathwada region of Maharashtra, India was obtained. A network prediction model was built with different combinations of parameters as input. The table 6.1 illustrates the results obtained with maximum correlation between the network predictions and actual data going up to 90.56%.

Evaluation Parameters:

Correlation: Correlation is a statistical measure that indicates the extent to which two or more variables fluctuate together. In our project, the correlation has been shown between source data and predicted data. The formula used for calculating correlation is:

$$\mathrm{r} = \frac{N\sum xy - (\sum x)(\sum y)}{\sqrt{[N\sum x^2 - (\sum x)^2][N\sum y^2 - (\sum y)^2]}}$$

Eq. 9

Where, N is the total number of pairs of source -predicted data

$\sum xy$ is the product of pairs of source- predicted data

$\sum x$ is the sum of source data

$\sum y$ is the sum of predicted data

$\sum x^2$ is the sum of squared source data

$\sum y^2$ is the sum of squared predicted data

Mean Square Error: The mean squared error (MSE) measures the average of the squares of the errors (the difference between the source data and the predicted data). Our system's performance is evaluated based on the following formula:

$$MSE = \frac{\sum_{i=0}^{N}(x_i - y_i)^2}{N}$$

Eq. 10

Table 6.1. Results in terms of statistics

Model No.	Parameters	Mean Squared Error	Correlation
1	WET	4091	90.56
2	CLD	4542	73.85
3	DTR	4288	88.96
4	TMP	4525	74.92
5	VAP	4306	83.3
6	PET	4158	86.21
7	WET,CLD,DTR,TMP1,VAP,PET	5520	64.25

An open source portal named- “Unnati Kissan Kendra” was developed for farmers to have access to all kinds of information with regards to farming. An Android app was developed for the same portal for enabling farmers to have easier access to the information provided by the portal.For a majority of farmers who don’t have an internet connection or smartphone, an IVR system is developed to communicate important information via a landline or feature phone.

5.2 Graphs:

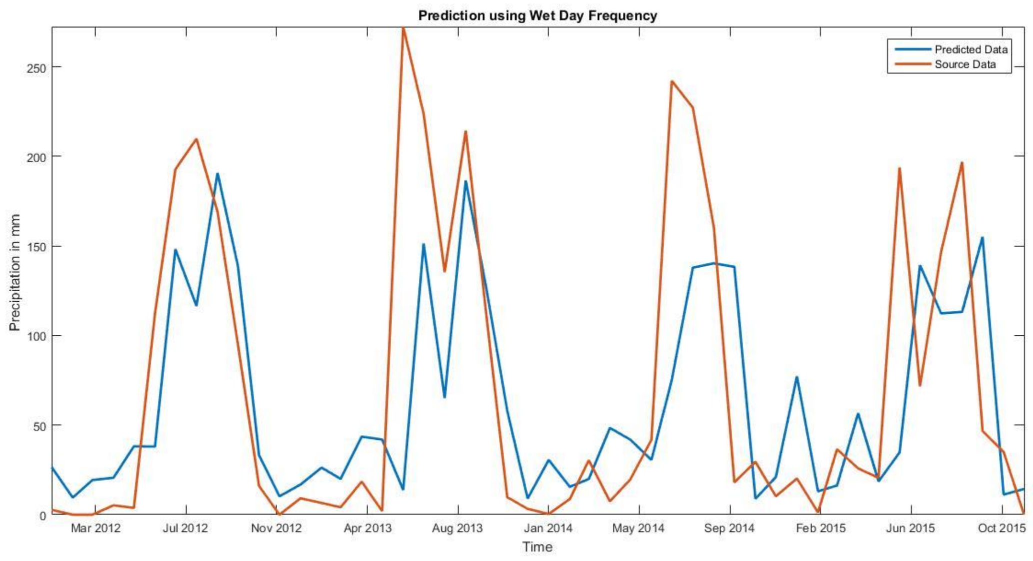


Fig 6.1 Prediction using model 1

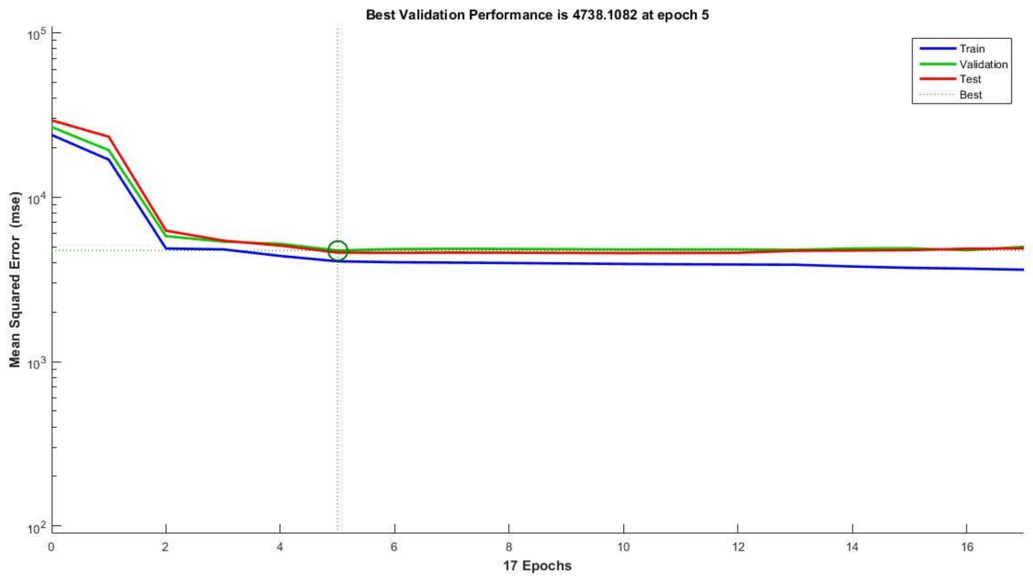


Fig 6.2 Performance of model 1

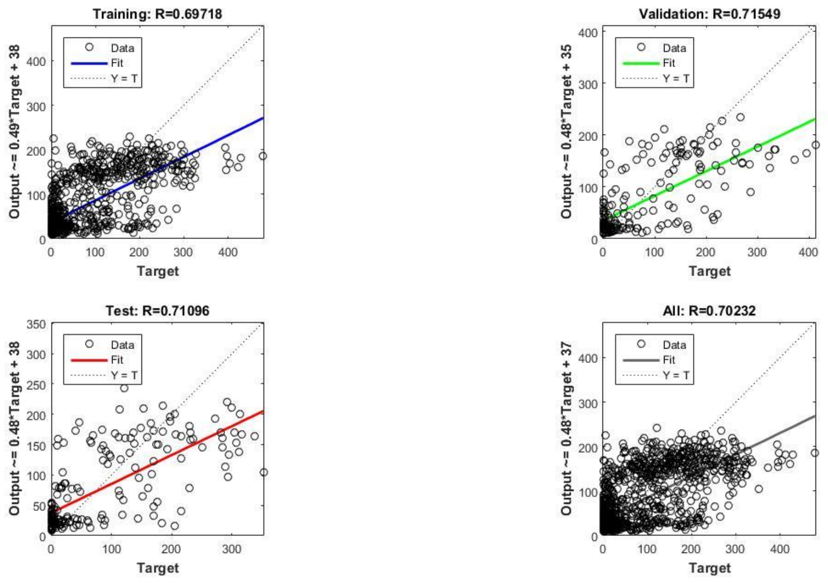


Fig 6.3 Regression results of model 1

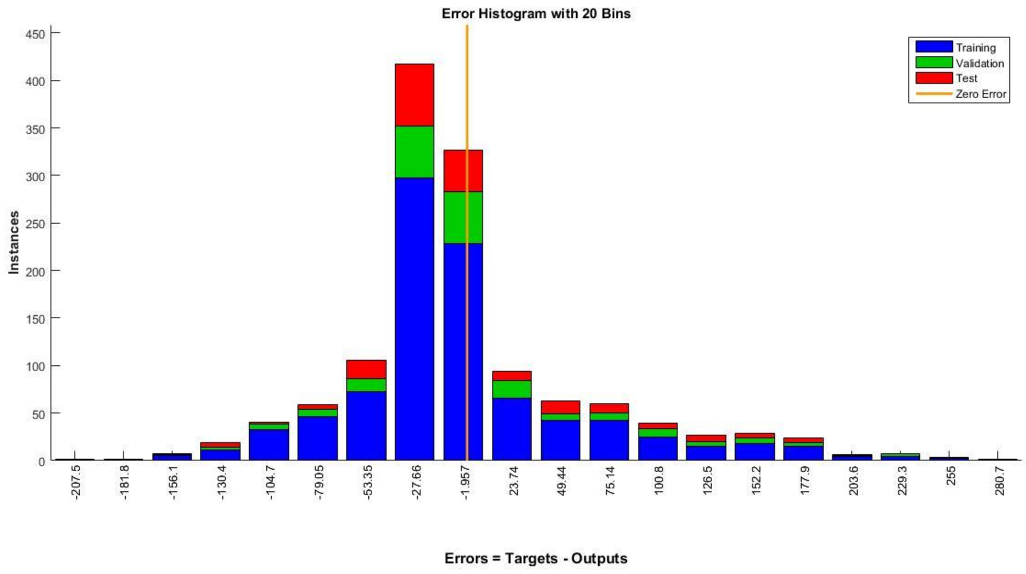


Fig 6.4 Error Histogram of model 1

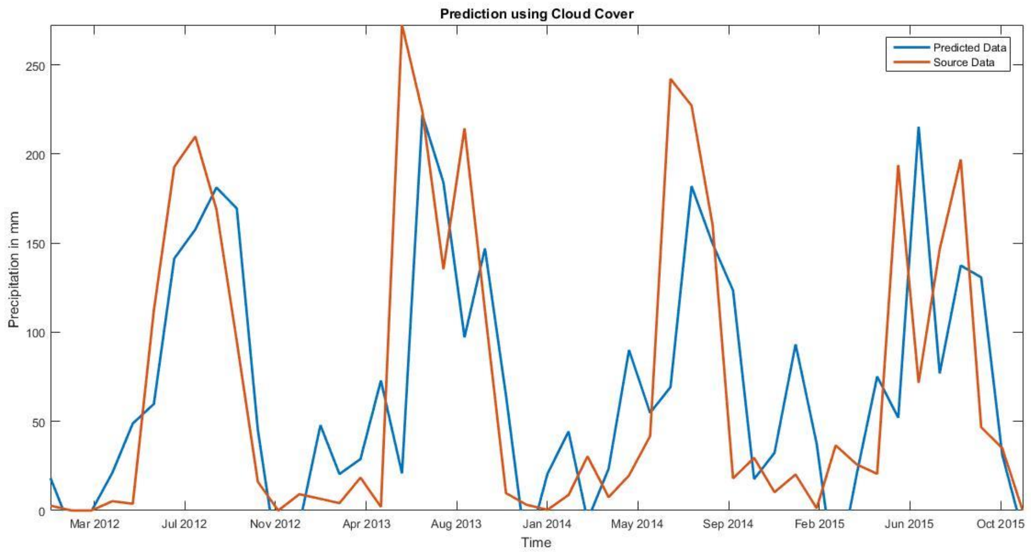


Fig 6.5 Prediction using model 2

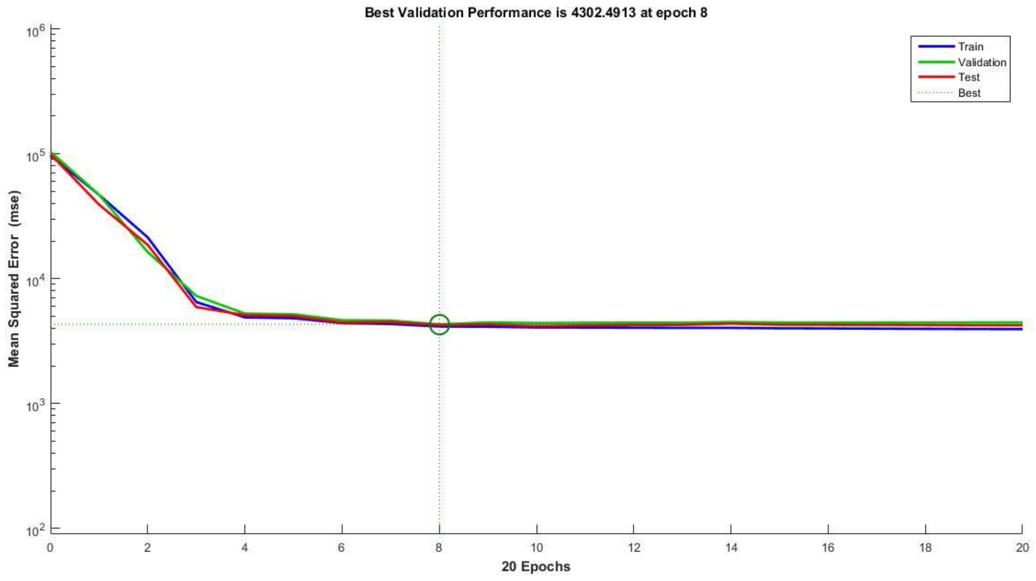


Fig 6.6 Performance of model 2

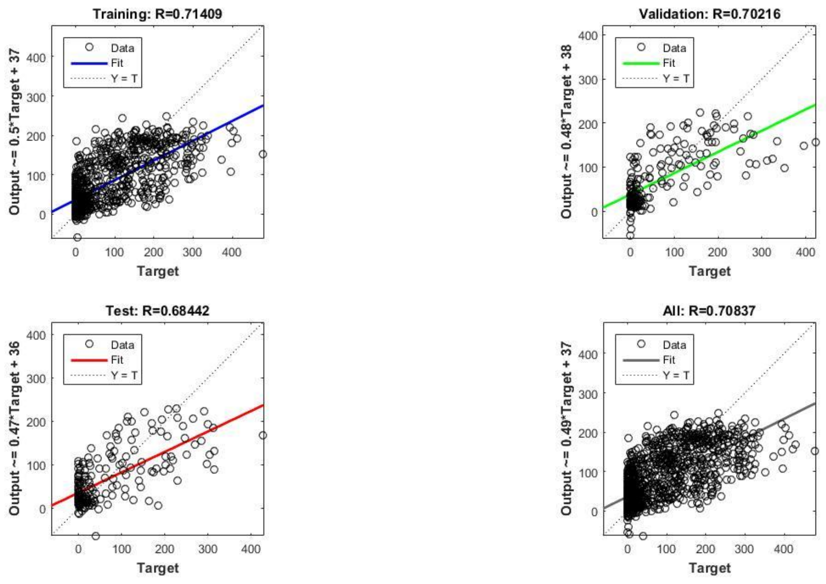


Fig 6.7 Regression results of model 2

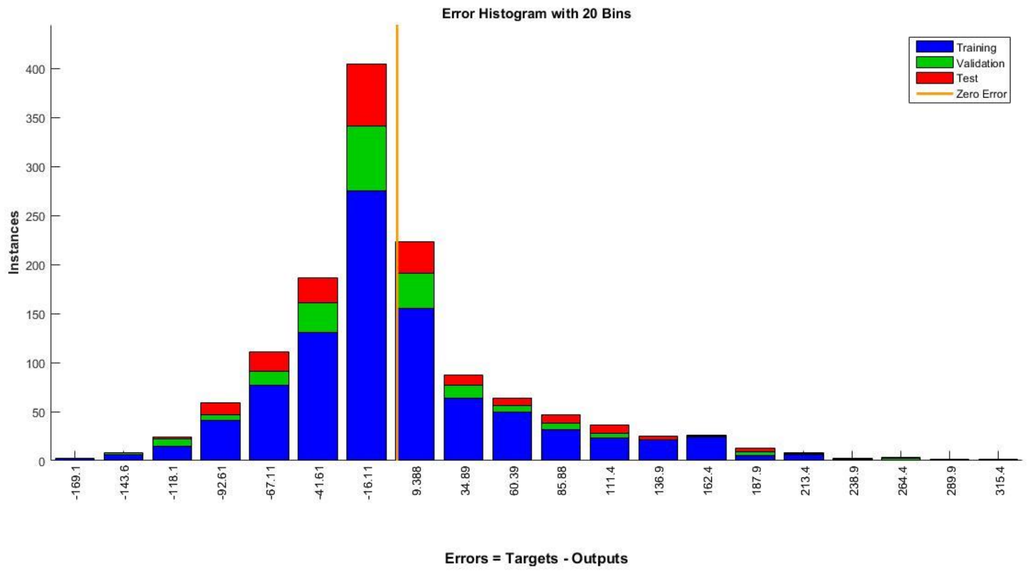


Fig 6.8 Error Histogram of model 2

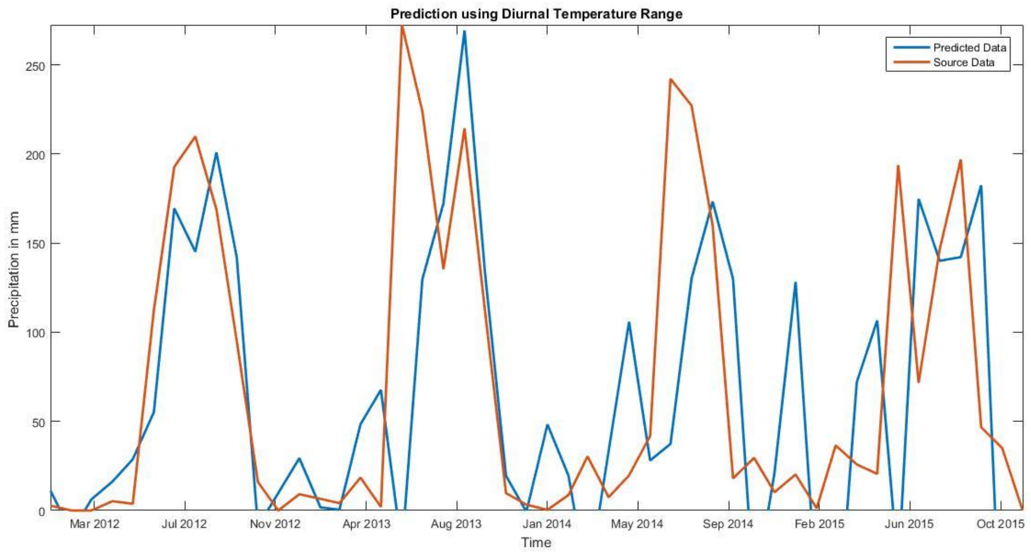


Fig 6.9 Prediction using model 3

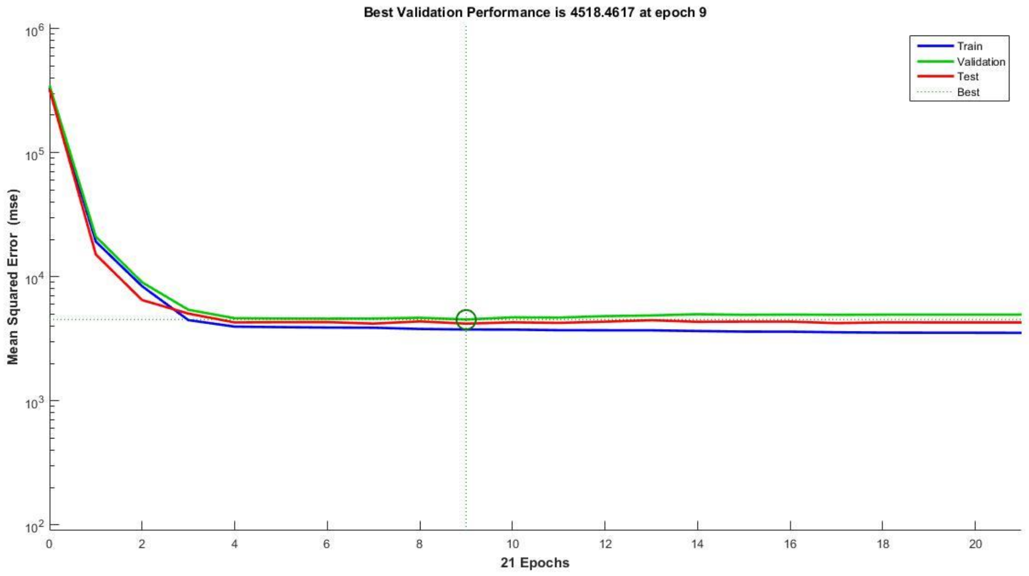


Fig 6.10 Performance of model 3

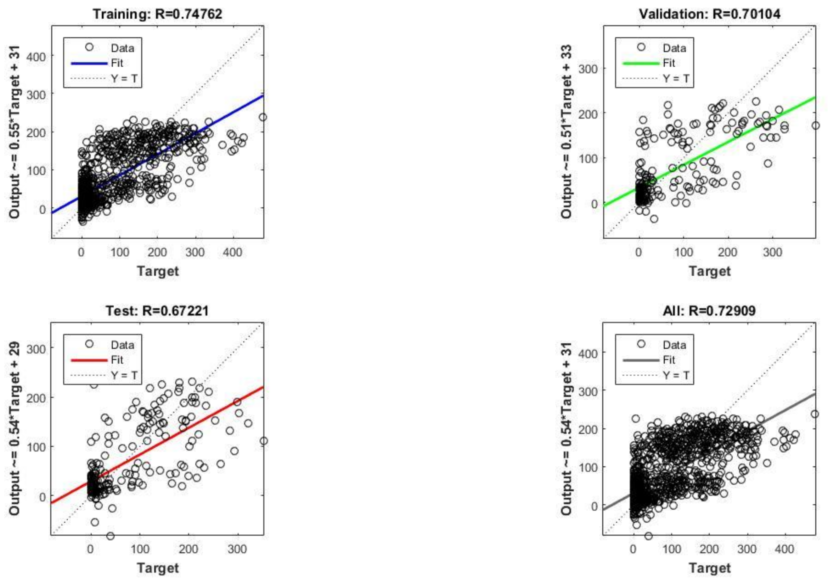


Fig 6.11 Regression results of model 3

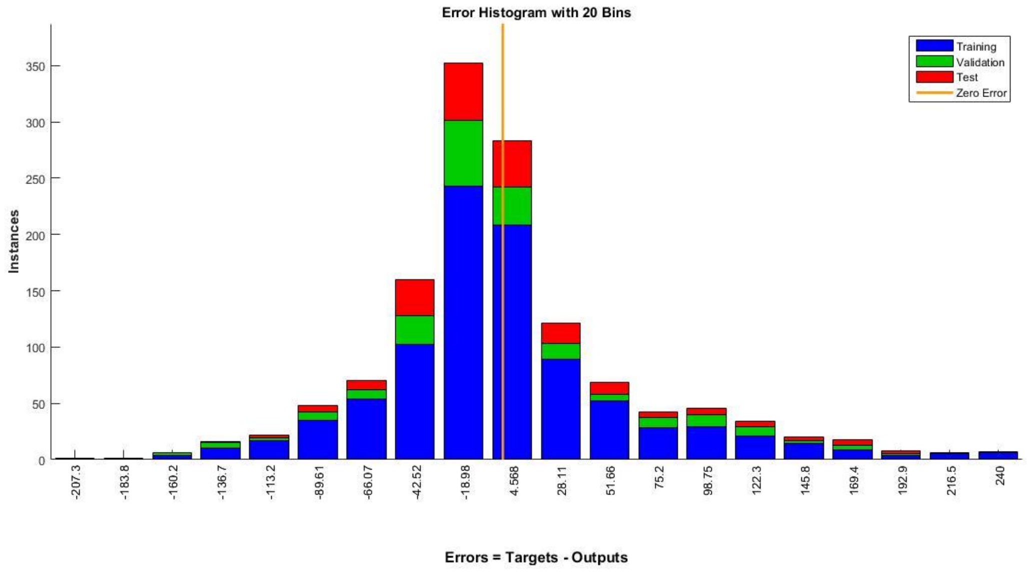


Fig 6.12 Error Histogram of model 3

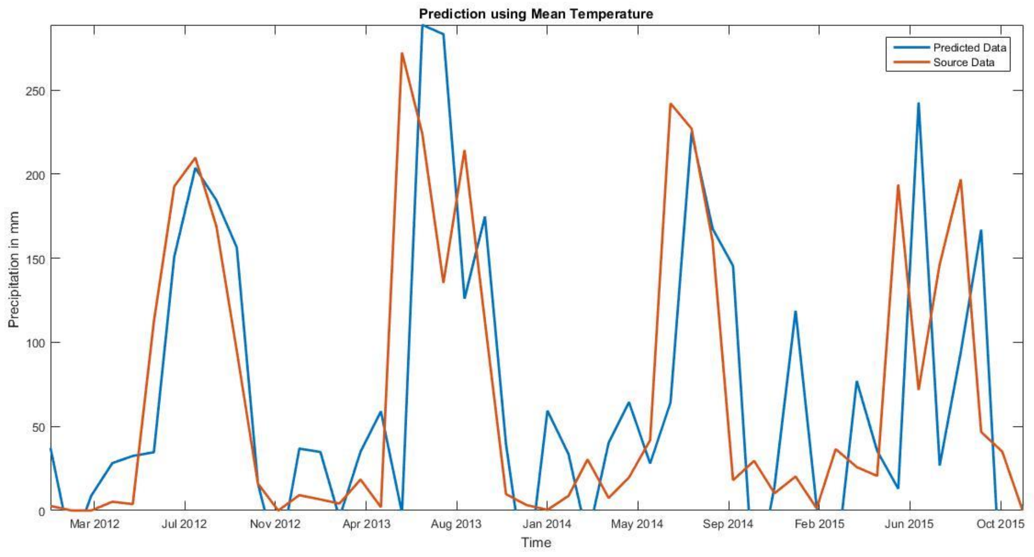


Fig 6.13 Prediction using model 4

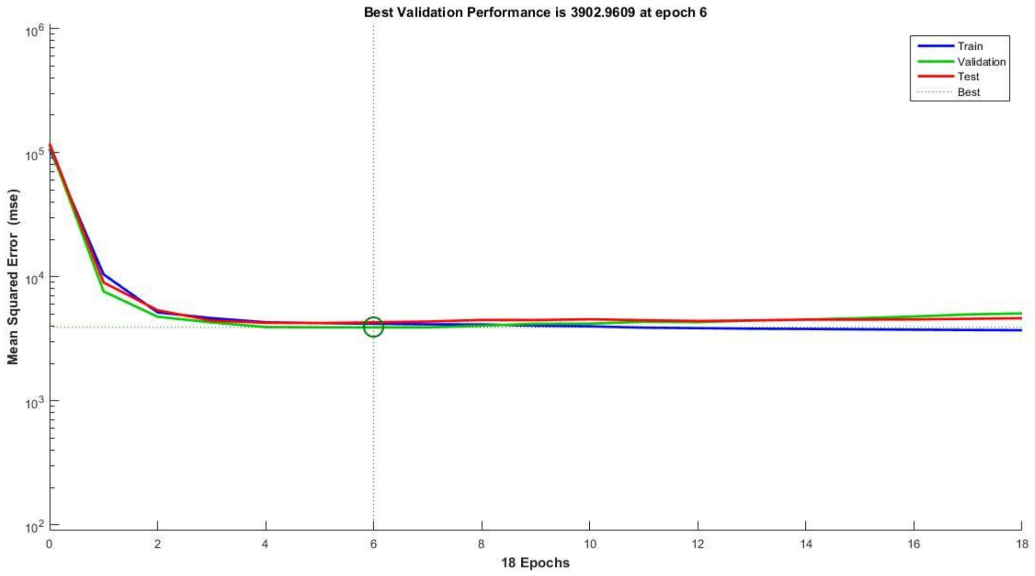


Fig 6.14 Performance of model 4

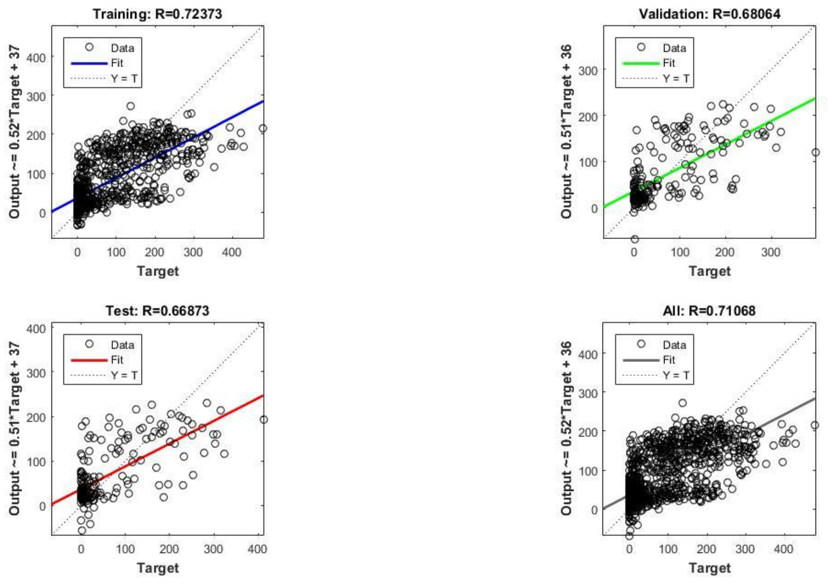


Fig 6.15 Regression results of model 4

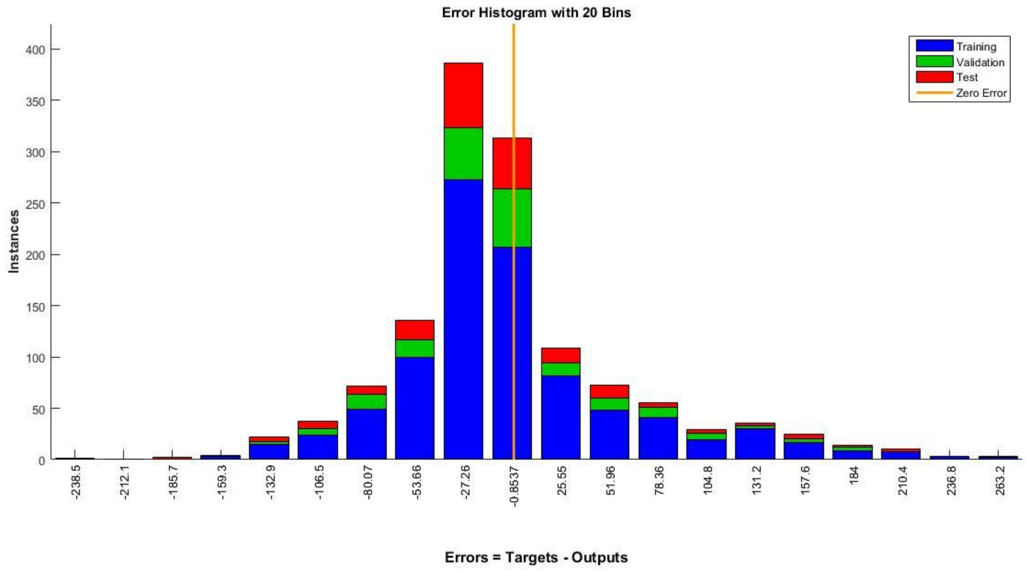


Fig 6.16 Error Histogram of model 4

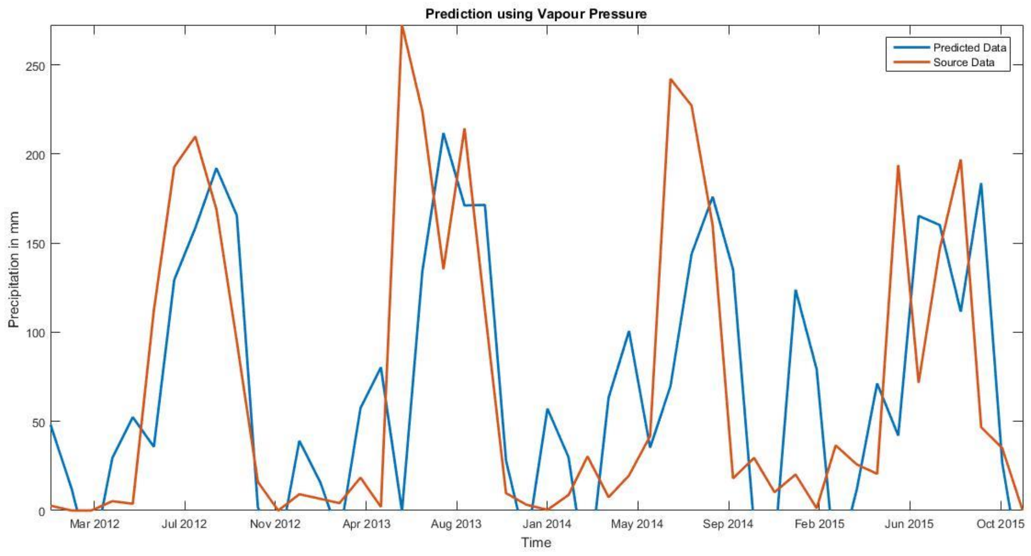


Fig 6.17 Prediction using model 5

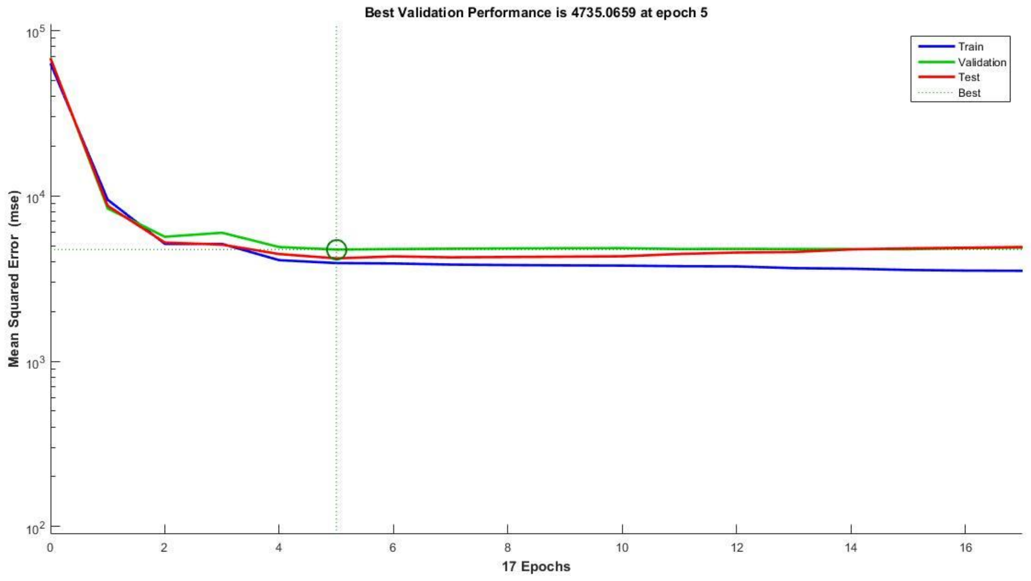


Fig 6.18 Performance of model 5

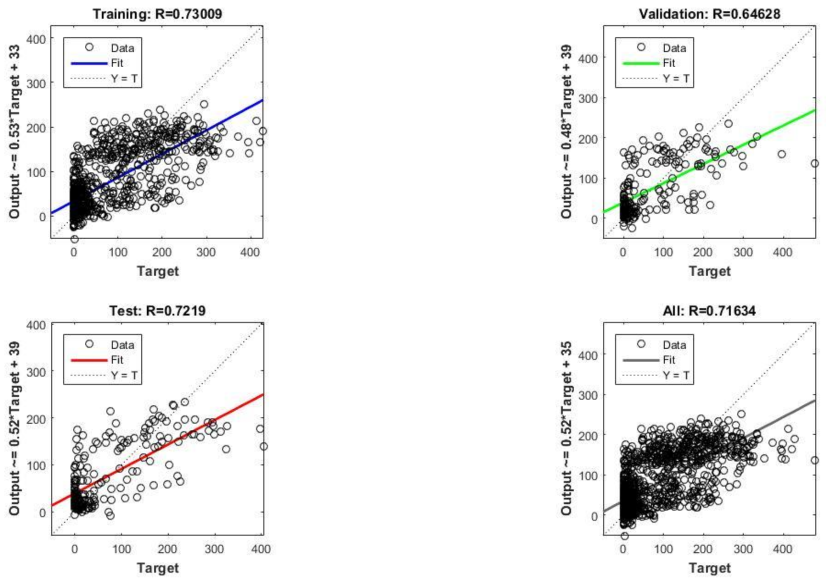


Fig 6.19 Regression results of model 5

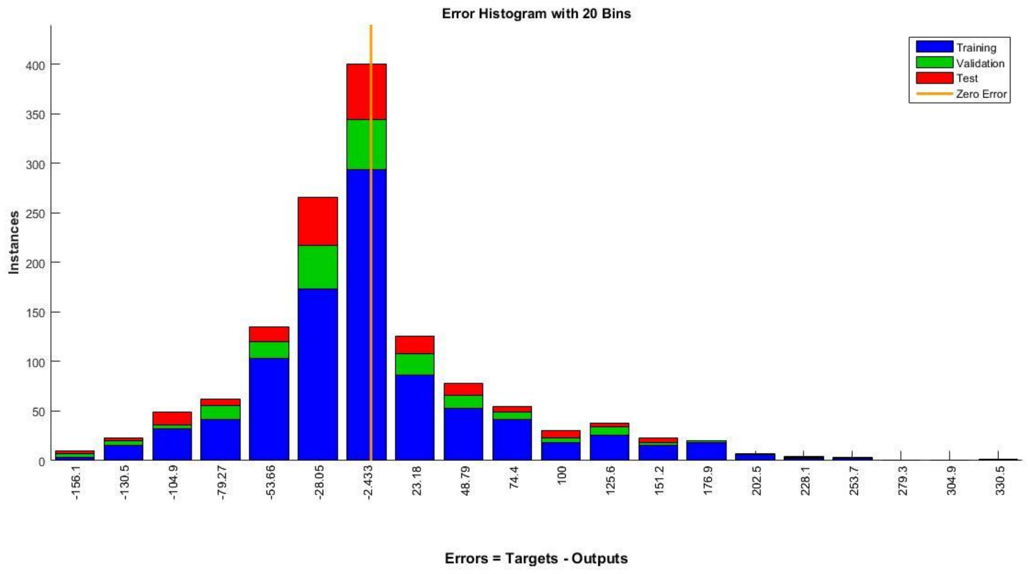


Fig 6.20 Error Histogram of model 5

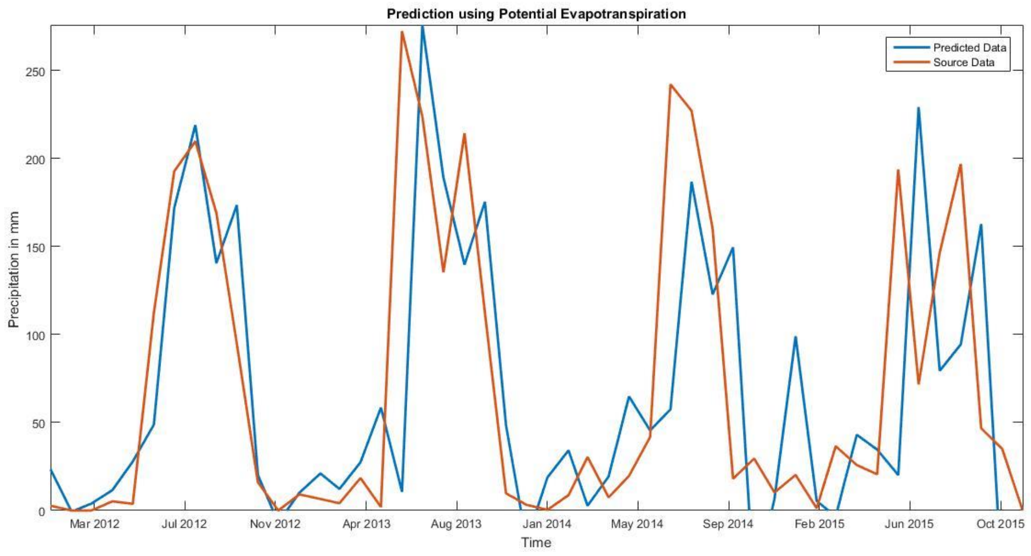


Fig 6.21 Prediction using model 6

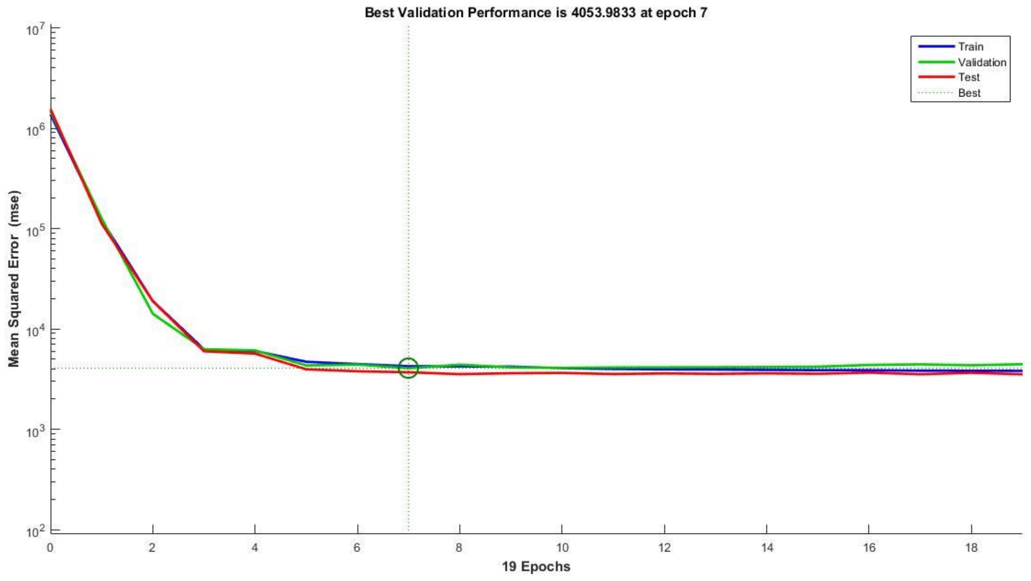


Fig 6.22 Performance of model 6

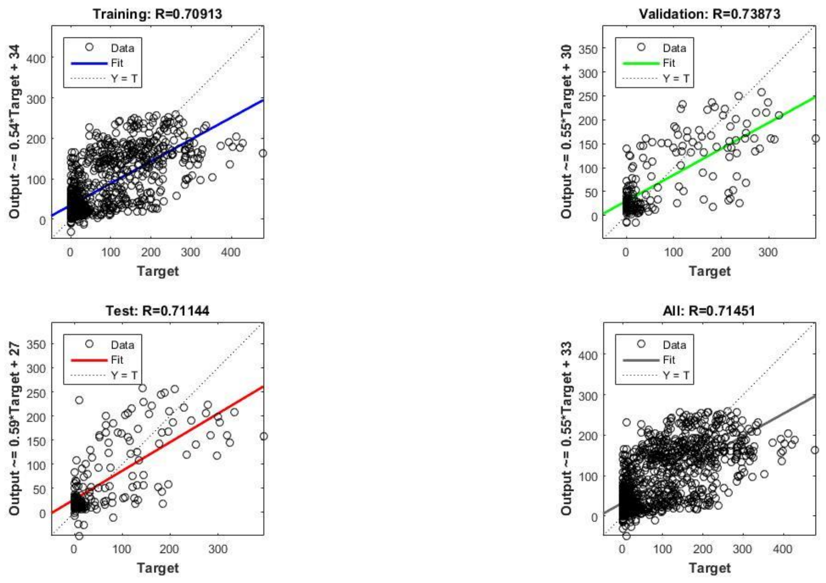


Fig 6.23 Regression results of model 6

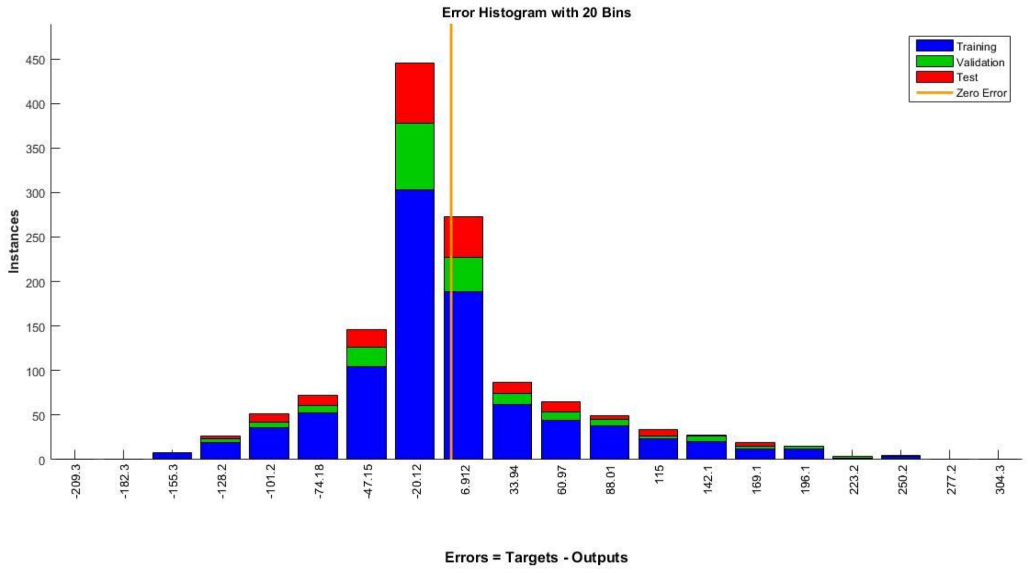


Fig 6.24 Error Histogram of model 6

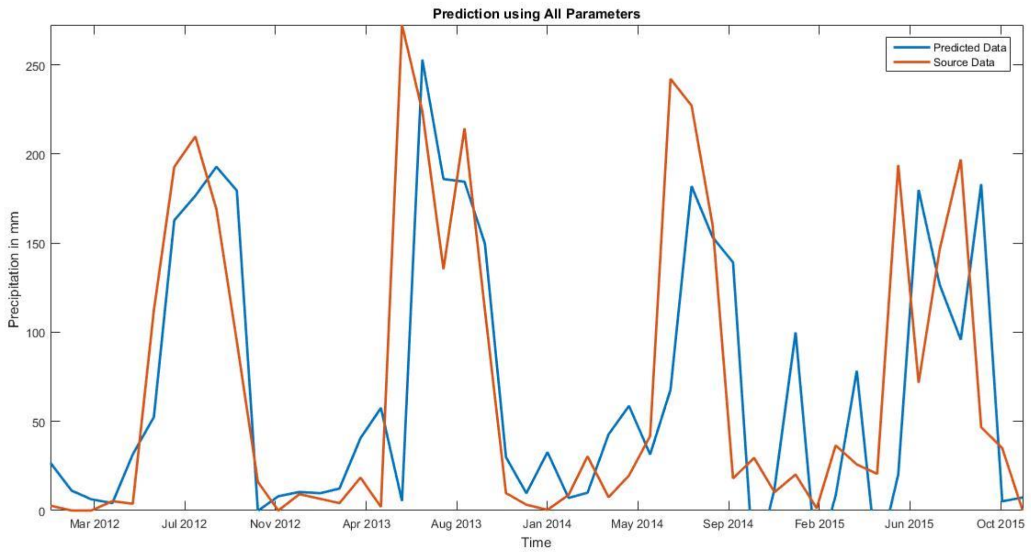


Fig 6.25 Prediction using model 7

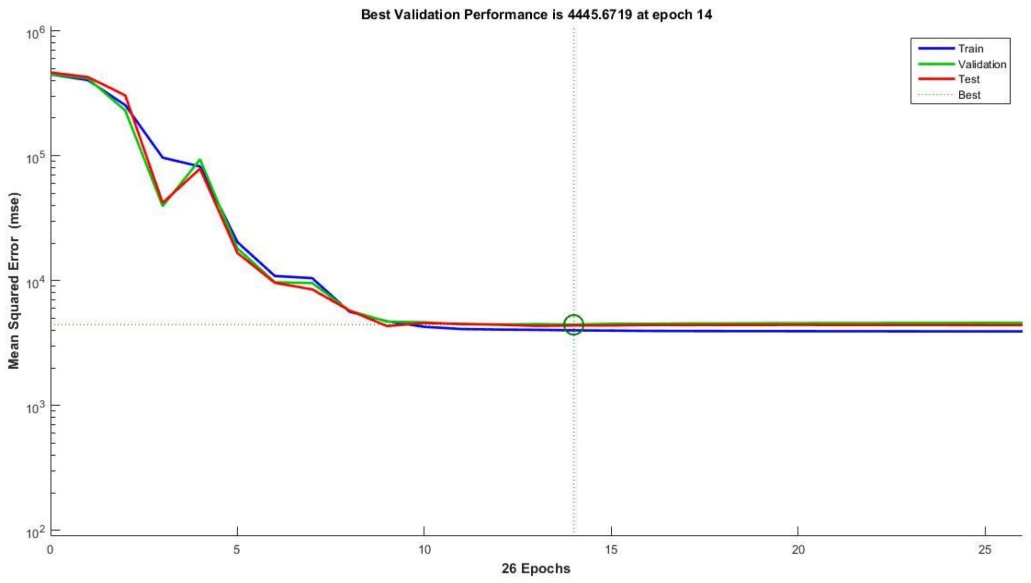


Fig 6.26 Performance of model 7

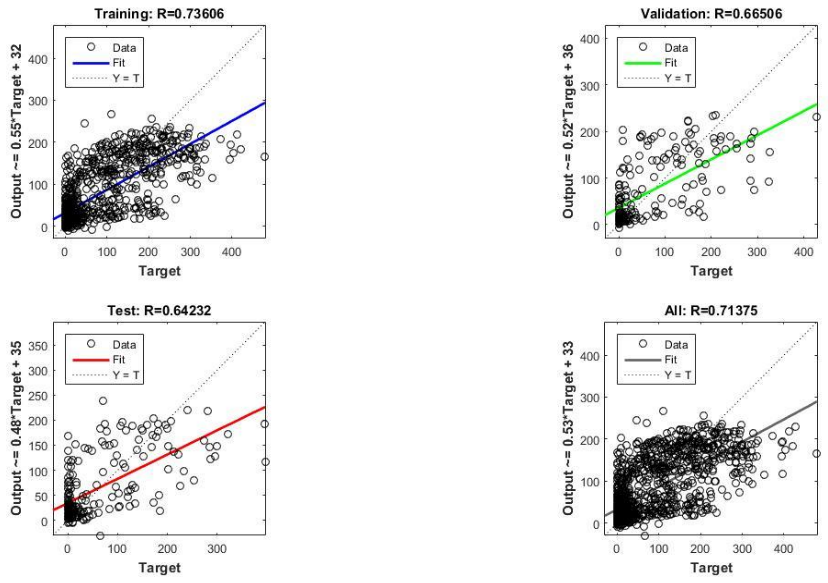


Fig 6.27 Regression results of model 7

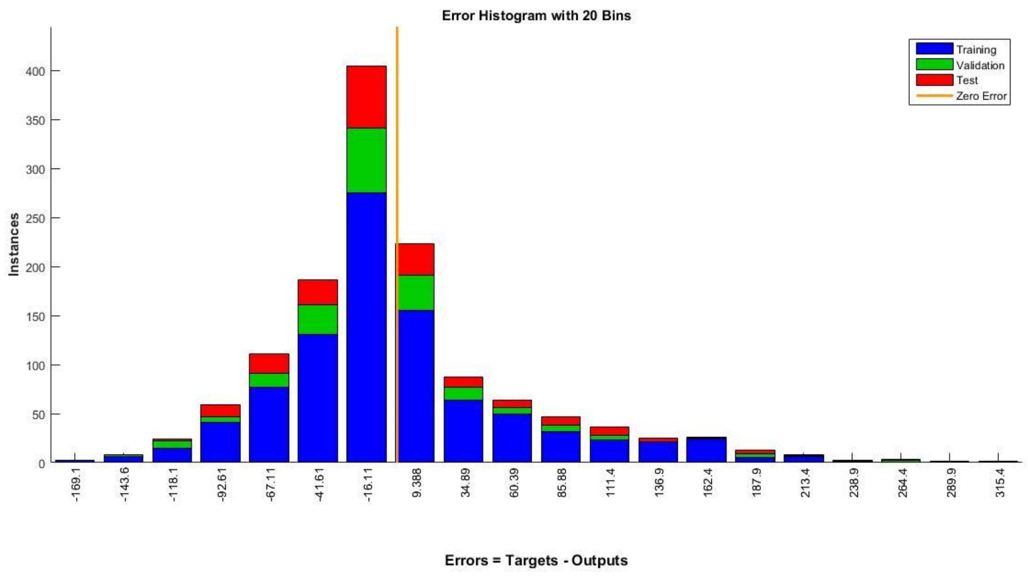


Fig 6.28 Error Histogram of model 7

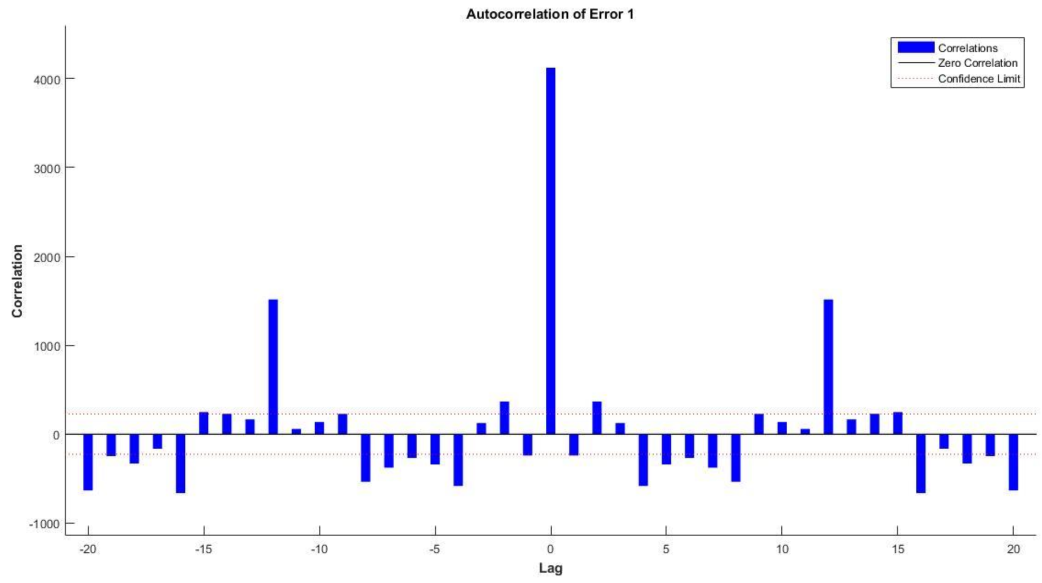


Fig 6.29 Error Autocorrelation of model 7

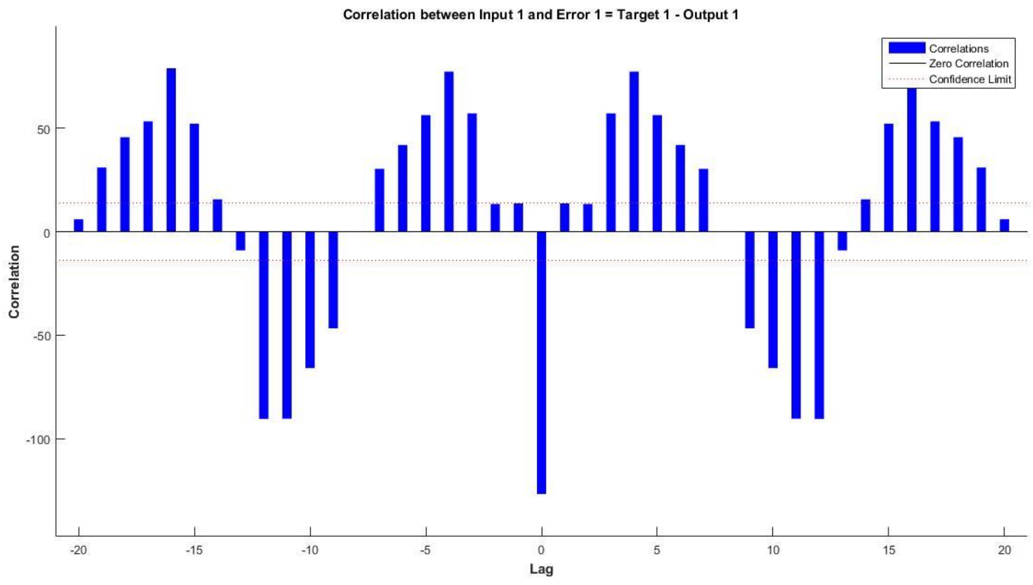


Fig 6.30 Input Error Cross Correlation of model 7

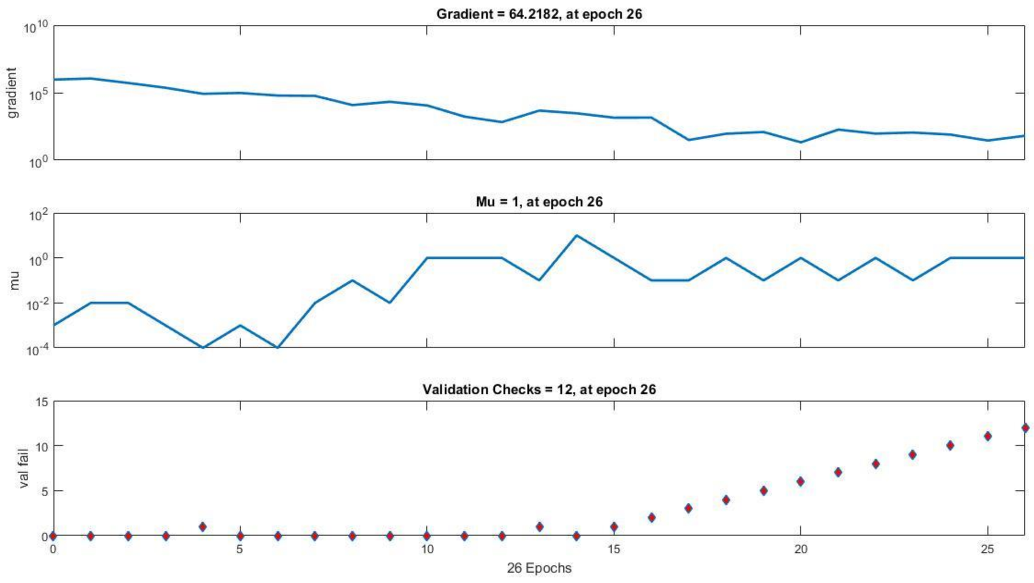


Fig 6.31 Training State of model 7

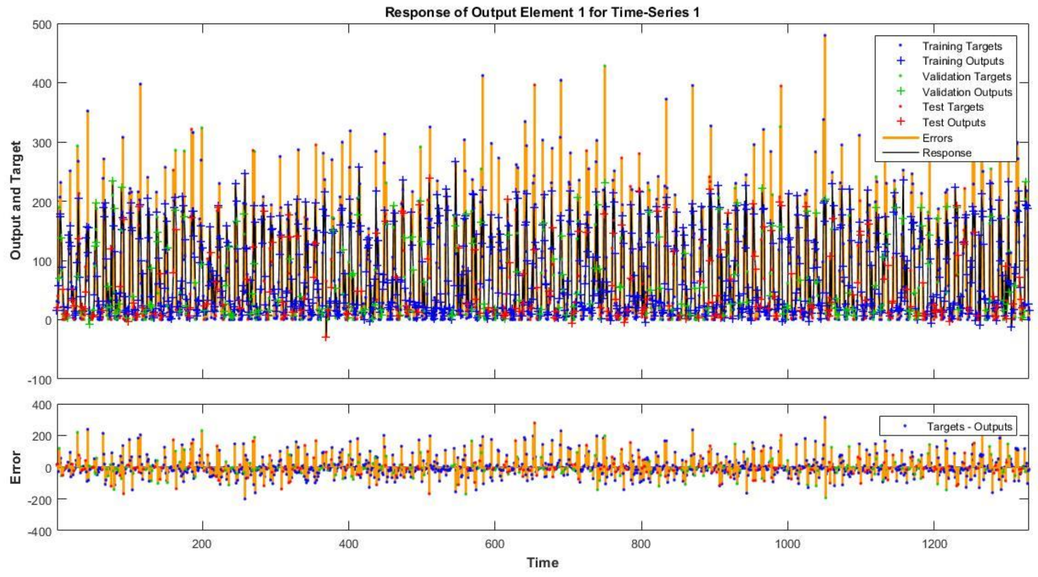


Fig 6.32 Time Series Response of model 7

Chapter 7
CONCLUSION

Analytical models had an upper-hand in predictive analytics till some years ago. With the sheer amount of data available for various parameters of climate, it becomes imperative that we try to use soft computing techniques to predict the onset of natural disasters.

Over thousands of farmers commit suicide every year due to poor planning and lack of awareness regarding the onset and amount of precipitation. With proper planning using crop modelling, classification algorithms can provide a methodological and scientific approach to the farmers to plant crops which will yield them with maximum profit.

Millions of people are displaced every year due to natural disasters like floods, typhoons, cyclones, famines, etc. Although it is not in our hand to completely circumvent them, we have an option to be ready to help mitigate them. Our project allows us to have an upper-hand for predicting the rainfall in a region and helps farmers decide which crop to grow and gives related information regarding loans, economic policies, buyers, sellers, modern techniques, etc. to provide a one stop solution to ease their life.

REFERENCES

[1] Nasseri, M., Keyvan Asghari, and M. J. Abedini. "Optimized scenario for rainfall forecasting using genetic algorithm coupled with artificial neural network." Expert Systems with Applications 35.3 (2008): 1415-1421.

[2] Govindaraju, R. S. (2000a). Artificial neural network in hydrology, I: Preliminary concepts. Journal of Hydrologic Engineering, 5(2), 115–123.

[3] "Northeast monsoon claimed 470 lives in Tamil Nadu: Jayalalithaa". *Business Line. 4 January 2016*. Retrieved 22 January 2016

[4] Times of India - Feb 3, 2015.

[5] Venkatesan, C., et al. "Prediction of all India summer monsoon rainfall using error-back-propagation neural networks." Meteorology and Atmospheric Physics 62.3-4 (1997): 225-240.

[6] Hung, Nguyen Q., et al. "An artificial neural network model for rainfall forecasting in Bangkok, Thailand." Hydrology and Earth System Sciences 13.8 (2009): 1413-1425.

[7] Litta, A. J., Sumam Mary Idicula, and U. C. Mohanty. "Artificial neural network model in prediction of meteorological parameters during Premonsoon thunderstorms." International Journal of atmospheric sciences 2013 (2013).

[8] Mohammed, Lubna B., et al. "Hourly solar radiation prediction based on Nonlinear Autoregressive Exogenous (NARX) neural network." JJMIE 7.1 (2013).

[9] Bielecki, Andrzej. "Dynamical properties of learning process of weakly nonlinear and nonlinear neurons." Nonlinear Analysis: Real World Applications 2.2 (2001): 249-258.

[10] Hornik, Kurt, Maxwell Stinchcombe, and Halbert White. "Multilayer feedforward networks are universal approximators." Neural networks 2.5 (1989): 359-366.

[11] Patrick K. Simpson, “Neural Networks Applications”(1997), ISBN:0780325664

[12] Adeli, Hojjat, and Shih-Lin Hung. Machine learning: neural networks, genetic algorithms, and fuzzy systems. John Wiley & Sons, Inc., 1994.

[13] Rojas, Raúl. “*Neural Networks: A Systematic Introduction.*” Springer Science & Business Media 2013.