**Chapter 3**

**Short-Term Forecasting of Wind Speed based on Artificial Intelligent Techniques**

*This chapter presents Artificial Intelligent Techniques for short term forecasting of wind speed. Neural Network (NN), Fuzzy Logic (FL), Adaptive Neuro Fuzzy Inference System (ANFIS) and Support Vector Machine (SVM) models are investigated. Levenberg Marquardt, Bayesian Regulation and Scaled Conjugate Gradient algorithm are used in Neural Network. Rule based algorithm framed in FL model using variance of wind speed data. Trapezoidal membership function is used to train ANFIS model. RBF, linear, quadratic, and polynomial are used to train SVM model. Two paired combinational models using NN, FL, ANFIS and SVM models are tested with actual measured wind speed data of three sites. A case study for Wind Speed Forecasting for Yearly, Monthly, Daily Average Wind Speed using ANFIS Model is carried out.*

**3.1 Neural Networks**

A neural network is a structure where it looks like a biological neuron. It is defined as a “massively parallel distributed processor made up of simple processing units, which has a natural bias for storing experimental information and making it accessible for use”. Consideration of human central nervous system is encouraged by idea of artificial neural network. Simple artificial [nodes](https://en.wikipedia.org/wiki/Node_(neural_networks)" \o "Node (neural networks)) are identified as "neurons", "processing elements" and "units" associated together to form a biological Neural Network [28,30,35-36].

**3.1.1 Neural Network Architecture**

There are many types of neural networks, beginning with comparatively simple to very complex. Feed-forward neural network contains layers of mechanisms for processing and make independent computations on given data set. Results are received and passed to next layer and finally, a section of one or additional processing elements regulate output from network. Every component computed on a weighted sum of given inputs. Layers placed between first and last layer are known as hidden layers. Elements are similar to neurons in a human brain, and therefore, they are called as cell or artificial neurons. Neuron processes its statistics which is a fundamental operation of neural network [30].

Bias

Summing Function

Weights

Inputs

Output

**Fig. 3.1: Basic Structure of a Neuron [26]**

Fig. 3.1 presents basic structure of a Neuron. Three components in neuron model are adder, synapses and activation function. Connections between synapses are characterized by directed edges of graph. Adder represented by sigma is used to summing inputs. A threshold function is employed to qualify neurons in an output layer. Internally realistic bias denoted by is included inneuronal model and consequence of changing net input value of activation function. A neuron is described by two equations [35]:

|  |  |  |
| --- | --- | --- |
|  |  | (3.1) |
|  |  | (3.2) |

Where,

are signal in input layer

are synaptic weight

is linear combined output from input signals

is bias

is activation function

is neuron output signal

Activation function is denoted by, defines output of a neuron in terms of induced local field. Sigmoid function is used as activation function. With slope factor of sigmoid function, activation function is defined by [36]:

|  |  |  |
| --- | --- | --- |
|  |  | (3.3) |

Curve traced sigmoid function structure represented in Fig. 3.2. Curve resembles hysteresis curve is helpful in describing wind speed values as function of past values.

**Fig. 3.2: Sigmoid Function**

**Multithreaded Back Propagation**

Back propagation is an iterative process which takes a great deal of time to compute. When multi-core systems are used in multithreaded technique then it may significantly decrease time to converge. It is relatively simple to adapt back propagation algorithm to operate in a multithreaded manner if batching is used. Training data is divided equally in large batches of thread. Each thread accomplishes backward and forward disseminations. Weight and threshold deltas are summed for each thread. At each end of iteration all threads temporarily pause for weight and threshold [35].

**3.1.2 Design of a Neural Network**

Design of neural network includes designing input, output, and hidden processing elements. Two sets of weights in activations of hidden layer neurons help to determine output neuron activation function. Weights are modified to reduce MAPE between network’s prediction and actual target value. Adjustments are prepared in inverse direction from hidden layer to output layer till terminating condition is reached. A three layer neural network with two inputs and one output is given in Fig. 3.3.

F5(z)

F6(z)

F4(z)

F3(z)

F2(z)

F1(z)

X1

X2

**Y**

**Fig. 3.3: Three layer Neural Network**

Training data set consists of input signals (and) with corresponding target. At each iteration weight coefficients of nodes are updated using new data from and input of neuron in next layer. Symbols represents output signal of neuron. Functions at each layer are given by:

|  |  |  |
| --- | --- | --- |
|  |  | (3.4) |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

Where,

y1-y6 are function output of each layer

wij are weights assigned to connections between neurons j to neuron i

x1 and x2 are input variables

Error signal *δ* of output layer neuron is given by:

|  |  |  |
| --- | --- | --- |
|  |  | (3.5) |
|  |  |
|  |  |
|  |  |
|  |  |

Where,

is gradient at ith layer

Correction applied for weight connecting to neuron to neuron is defined by delta rule as:

|  |  |  |
| --- | --- | --- |
|  | | (3.6) |
|  |  | (3.7) |

Where,

is corrected weight

n is number of neurons

is local gradient

Local gradient depends on neuron in a hidden node. If neuron is an output node, is derivative of and error signal If neuron is a hidden node, is associated with derivative and weighted sum of .

**Training Algorithms**

Algorithms for training neural network are broadly classified into three types: Levenberg-Marquardt, Bayesian regulation and Scaled conjugate gradient.

**Levenberg-Marquardt**: This algorithm normally takes extra memory, but less time to compute. Training will stops automatically when simplification stops improving, as designated by an increase in mean square error of validation samples.

**Bayesian regulation:** This algorithm normally takes more time, but it can result good simplification for problematic, small or noisy datasets. Training stops permitting to adaptive weight minimization.

**Scaled conjugate gradient**: This algorithm takes less memory. Training automatically stops when simplification stops improving, as specified by an increase in mean square error of validation samples.

Levenberg Marquardt, Bayesian Regularization and Scaled Conjugate Gradient algorithms are used to train NN using MATLAB 2014(a) Neural Network Toolbox. Initial random weights are assigned for all neurons and updated at every iteration. Input pattern is divided in to three parts as training, testing and validating with a ratio of 70:15:15% of entire wind speed data. Various patterns are analyzed to match output patterns. Fitting error is used as stopping criteria. After Then forecasting results are presented for all three algorithms. Fitting error is estimated for each of neurons and weights are adjusted accordingly in all nodes. Cycle is repeated until reaching of least MSE. Process is summarized by a flowchart as shown in Fig. 3.3. Steps for a 3-layer Neural Network with one hidden layer are listed as follows:

Step 1: Randomly initializing weights and assigning of training set data in input node

Step 2: Forward pass T = Actual output for a network

Step 3: Calculate error (T - O) at output units

Step 4: Compute Δwh for all weights from hidden layer to output layer ; backward pass

Step 5: Estimate Δwi for all weights from input layer to hidden layer

Step 6: Update weights in network until stopping criterion satisfied and return network

|  |
| --- |
| NO  NO  NO  YES  YES  YES  Get desired Equation  Initialize weights W for all neurons  Get pattern X and Feed Forward  Compute cycle error E  Adjust weights for output layer using F(w,x,d)  E=0  More hidden layers?  E<Emax  Adjust weights for hidden layer  More Patterns ?  STOP  START |
| **Fig. 3.4: Flowchart for NN Architecture** |

**3.1.3 Result of Neural Network**

Simulation results of neural network based short term forecasting of wind speed with two hidden layers using Levenberg-Marquardt (LM) training algorithm with MSE as performance criteria are shown in Fig. 3.5 to 3.12. Fig. Performance curves, training state, error histogram, ACF for fitting error, regression coefficients, correlation and response of output elements are presented using Matlab Neural Network Toolbox. It is found that best performance is found in6th epoch with MSE of 1.8337, gradient of 0.7499 at 11th iteration and regression of 0.8911 for target data. Error histogram indicates that, most error values range between -0.6155 to 0.8517. Forecasting results of three training algoriths are presented in Fig. 3.13 to 3.30.

|  |  |  |
| --- | --- | --- |
| **Simulation Result of Neural Network Tool Box for Levenberg Algorithm** | | |
|  |  |  |
| **Fig. 3.5: NN Architecture with 2 hidden layers** |
|  |
| **Fig. 3.6: Performance of LM Algorithm for h 1-12 h2- 6** | **Fig. 3.7: Training State of NN** |
|  |  |
|  |  |
| **Fig. 3.8: Distribution of Error for training, validation and test of NN** | **Fig. 3.9: Autocorrelation of Error with confidence interval of 95%** | **Fig. 3.10: Regression Coefficient of Training, Validation, test and regression** |
|  |  |  |
|  |  | |
| **Fig. 3.11: Correlation between Input and Error with 95% confidence interval** | **Fig. 3.12: Response of Output Element for input element for training, validation and test states** | |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  | | --- | --- | |  | | |  |  |   **Fig. 3.13:(a) Actual Vs Forecasted wind speed for 24 hour using Neural Network using (LMA) Training model (b) Regression plot of Actual Vs Forecasted (c) Error Distribution for Bagalkot Site** | |  |  | | --- | --- | |  | | |  |  |   **Fig. 3.14:(a) Actual Vs Forecasted wind speed for 24 hour using Neural Network using (LMA) Training model (b) Regression plot of Actual Vs Forecasted (c) Error Distribution for Vijayapura Site** | |  |  | | --- | --- | |  | | |  |  |   **Fig. 3.15:(a) Actual Vs Forecasted wind speed for 24 hour using Neural Network using (LMA) Training model (b) Regression plot of Actual Vs Forecasted (c) Error Distribution for Bengaluru Site** |
|  | | |
| |  |  | | --- | --- | |  | | |  |  |   **Fig. 3.16:(a) Actual Vs Forecasted wind speed for 168 hour using Neural Network using (LMA) Training model (b) Regression plot of Actual Vs Forecasted (c) Error Distribution for Bagalkot Site** | |  |  | | --- | --- | |  | | |  |  |   **Fig. 3.17:(a) Actual Vs Forecasted wind speed for 168 hour using Neural Network using (LMA) Training model (b) Regression plot of Actual Vs Forecasted (c) Error Distribution for Vijayapura Site** | |  |  | | --- | --- | |  | | |  |  |   **Fig. 3.18:(a) Actual Vs Forecasted wind speed for 168 hour using Neural Network using (LMA) Training model (b) Regression plot of Actual Vs Forecasted (c) Error Distribution for Bengaluru Site** |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  | | --- | --- | |  | | |  |  |   **Fig. 3.19: (a) Actual Vs Forecasted wind speed for 24 hour using Neural Network using BR Algorithm Training model (b) Regression plot of Actual Vs Forecasted (c) Error Distribution for Bagalkot Site** | |  |  | | --- | --- | |  | | |  |  |   **Fig. 3.20: (a) Actual Vs Forecasted wind speed for 24 using Neural Network using BR Algorithm Training model (b) Regression plot of Actual Vs Forecasted (c) Error Distribution for Vijayapura Site** | |  |  | | --- | --- | |  | | |  |  |   **Fig. 3.21: (a) Actual Vs Forecasted wind speed for 24 using Neural Network using BR Algorithm Training model (b) Regression plot of Actual Vs Forecasted (c) Error Distribution for Bengaluru Site** |
|  | | |
| |  |  | | --- | --- | |  | | |  |  |   **Fig. 3.22: (a) Actual Vs Forecasted wind speed for 168 hour using Neural Network using BR Algorithm Training model (b) Regression plot of Actual Vs Forecasted (c) Error Distribution for Bagalkot Site** | |  |  | | --- | --- | |  | | |  |  |   **Fig. 3.23: (a) Actual Vs Forecasted wind speed for 168 using Neural Network using BR Algorithm Training model (b) Regression plot of Actual Vs Forecasted (c) Error Distribution for Vijayapura Site** | |  |  | | --- | --- | |  | | |  |  |   **Fig. 3.24: (a) Actual Vs Forecasted wind speed for 168 using Neural Network using BR Algorithm Training model (b) Regression plot of Actual Vs Forecasted (c) Error Distribution for Bengaluru Site** |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  | | --- | --- | |  | | |  |  |   **Fig. 3.25: (a) Actual Vs Forecasted wind speed for 24 hour using Neural Network using SCG Algorithm Training model (b) Regression plot of Actual Vs Forecasted (c) Error Distribution for Bagalkot Site** | |  |  | | --- | --- | |  | | |  |  |   **Fig. 3.26: (a) Actual Vs Forecasted wind speed for 24 hour using Neural Network using SCG Algorithm Training model (b) Regression plot of Actual Vs Forecasted (c) Error Distribution for Vijayapura Site** | |  |  | | --- | --- | |  | | |  |  |   **Fig. 3.27: (a) Actual Vs Forecasted wind speed for 24 hour using Neural Network using SCG Algorithm Training model (b) Regression plot of Actual Vs Forecasted (c) Error Distribution for Bengaluru Site** |
|  | | |
| |  |  | | --- | --- | |  | | |  |  |   **Fig. 3.28: (a) Actual Vs Forecasted wind speed for 168 hour using Neural Network using SCG Algorithm Training model (b) Regression plot of Actual Vs Forecasted (c) Error Distribution for Bagalkot Site** | |  |  | | --- | --- | |  | | |  |  |   **Fig. 3.29: (a) Actual Vs Forecasted wind speed for 168 hour using Neural Network using SCG Algorithm Training model (b) Regression plot of Actual Vs Forecasted (c) Error Distribution for Vijayapura Site** | |  |  | | --- | --- | |  | | |  |  |   **Fig. 3.30: (a) Actual Vs Forecasted wind speed for 168 hour using Neural Network using SCG Algorithm Training model (b) Regression plot of Actual Vs Forecasted (c) Error Distribution for Bengaluru Site** |

Fig. 3.19 to 3.25 presents wind speed forecasting results for 24 hour to 168 hour using BR training algorithm. Table 3.2 presents comparison of wind speed forecasting results for Bayesian Regularization Algorithm for 24 hour and 168 hour ahead. For 24 hour prediction, regression coefficient R2 is 0.485 for Bagalkot, 0.317 for Vijayapura and 0.364for Bengaluru site and MAPE is 12.309% for Bagalkot, 13.058% for Vijayapura and 12.408% for Bengaluru site. Similarly for 168 hour prediction regression coefficient R2 is 0.436 for Bagalkot, 0.309 for Vijayapura and 0.423 for Bengaluru site and MAPE is 17.112% for Bagalkot, 17.457% for Vijayapura and 17.257% for Bengaluru site.

Fig. 3.26 to 3.30 presents wind speed forecasting results for 24 hour to 168 hour using BR training algorithm. Table 3.1 presents comparison of wind speed forecasting results for Scaled Conjugate Gradient Training Algorithm for 24 hour and 168 hour ahead. For 24 hour prediction, regression coefficient R2 is 0..893 for Bagalkot, 0.412 for Vijayapura and 0.467for Bengaluru site and MAPE is 11.202% for Bagalkot, 12.218% for Vijayapura and 12.518% for Bengaluru site. Similarly for 168 hour prediction regression coefficient R2 is 0.803 for Bagalkot, 0.471 for Vijayapura and 0.546 for Bengaluru site and MAPE is 13.370% for Bagalkot, 16.969% for Vijayapura and 15.118% for Bengaluru site.

Table 3.1 gives comparison of wind speed forecasting results for Levenberg-Marquardt Algorithm for 24 hour and 168 hour ahead. For 24 hour prediction, regression coefficient R2 is 0.372 for Bagalkot, 0.317 for Vijayapura and 0.256 for Bengaluru site and MAPE is 12.218% for Bagalkot, 12.871% for Vijayapura and 12.081% for Bengaluru site. Similarly for 168 hour prediction regression coefficient R2 is 0.336 for Bagalkot, 0.311 for Vijayapura and 0.496 for Bengaluru site and MAPE is 16.720% for Bagalkot, 16.807% for Vijayapura and 16.165% for Bengaluru site.

**Table 3.1: Comparison of forecasting result for LM, BR and SCG algorithms**

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Method** | **LM Algorithm** | | | | **BR Algorithm** | | | | **SCG Training Algorithm** | | | |
| **Site** | **24 hour** | | **168 hour** | | **24 hour** | | **168 hour** | | **24 hour** | | **168 hour** | |
| **R2** | **MAPE in %** | **R2** | **MAPE in %** | **R2** | **MAPE in %** | **R2** | **MAPE in %** | **R2** | **MAPE in %** | **R2** | **MAPE in %** |
| Bagalkot | 0.372 | 12.218 | 0.336 | 16.720 | 0.485 | 12.309 | 0.436 | 17.112 | 0.893 | 11.202 | 0.803 | 13.370 |
| Vijayapura | 0.317 | 12.871 | 0.311 | 16.807 | 0.327 | 13.058 | 0.309 | 17.457 | 0.412 | 12.218 | 0.471 | 16.969 |
| Bengaluru | 0.256 | 12.081 | 0.496 | 16.165 | 0.364 | 12.408 | 0.423 | 17.257 | 0.467 | 12.518 | 0.546 | 15.118 |

**Fig. 3.31: Comparison Neural Network Training Algorithms for Bagalkot Site for 24 hour**

**Fig. 3.32: Comparison Neural Network Training Algorithms for Vijayapura Site for 24 hour**

**Fig. 3.33: Comparison Neural Network Training Algorithms for Bengaluru Site for 24 hour**

**Fig. 3.34: Comparison Neural Network Training Algorithms for Bagalkot Site for 16824 hour**

**Fig. 3.35: Comparison Neural Network Training Algorithms for Vijayapura Site for 168 hour**

**Fig. 3.36: Comparison Neural Network Training Algorithms for Bengaluru Site for 168 hour**

Fig. 3.31 to 3.36 depicts comparison of forecasted results of three algorithms. It is found that curves of LM algorithm are very close to actual values. Further peak values in this algorithm are at lesser rate. Forecasted curves of BR and SCG algorithm are almost same.

**3.2 Fuzzy Logic**

Fuzzy logic is becoming popular in dealing with data analysis problems that are normally handled by statistical approaches. Ability to handle imprecise data and to interpret analysis model built has made fuzzy logic as an important tool for many applications. Fuzzy logic is a system which is an expansion of multi valued logic. Fuzzy logic is an approach to computing based on “degrees of truth” rather than usual “true or false” Boolean logic. It relates to classes of objects in which membership is a matter of degree with un-sharp boundaries. “In almost every case same product built without fuzzy logic but fuzzy is faster and cheaper”. A linguistic variable values are words rather than numbers. Using words closer to human intuition than using numbers is preferred. Theory of fuzzy logic deals with two problems of Fuzzy set theory, which deals with ambiguity found in semantics [20,40,68-69].

.

LOGIC

FUZZY

SYMBOL

MANIPULATION &

NUMERICAL

COMPUTATION

APPROXIMATE REASONING

POSSIBILITY

**Fig. 3.37: Flow of logic in fuzzy techniques [70].**

**Features of Fuzzy Logic**

* It is conceptually easy to understand and flexible.
* It can model non-linear functions of arbitrary complexity.
* It is blended with conventional control techniques.
* It is based on natural language (Human communication).

Because of many advantages of fuzzy logic techniques, it is implemented in most of present applications [70].

**3.2.1 Fuzzy Logic System, Algorithm and Fuzzy Rules:** Richness of FL is that there are enormous numbers of possibilities that lead to lots of different mappings. This richness does require a careful understanding of FL and elements that comprise a FLS. One can, of course, challenge validity of some of these possibilities. Algorithm for fuzzy logic system is given in Fig. 3.38. Steps for fuzzy system are listed below:

Step 1: Define terms and linguistic variables; construct MF (MF), construct rule base (initialization).

Step 2: Converting input crisp data to values of fuzzy using MF (fuzzification).

Step 3: Evaluating rule base, combining results of rule base (inference).

Step 4: Converting output data to non-fuzzy values (defuzzification).

|  |
| --- |
| RULES  INTERFERENCE  FUZZIFIER  DIFUZZIFIER  Crisp input  Crisp output  Fuzzy Input  Fuzzy Output  **Fig. 3.38: General fuzzy logic system [18]** |

**Linguistic Variable**

Linguistic variable plays a key role in fuzzy logic for approximate reasoning with higher order variables. Values of a linguistic variable are words or sentences in an ordinary language.

**Membership Functions (MF)**

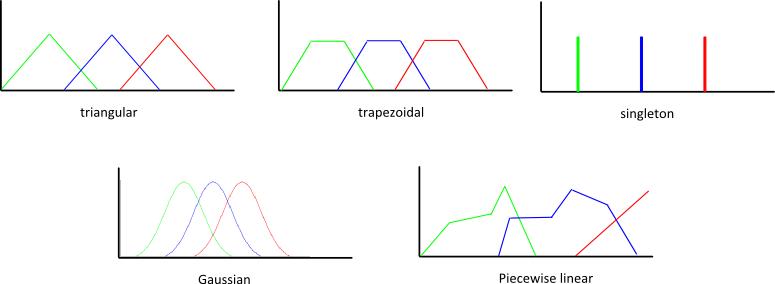
MFs are used in fuzzification and defuzzification steps of an FLS, to map non-fuzzy input values to fuzzy linguistic terms. There are different forms of MF such as triangular, trapezoidal, and piecewise linear, Gaussian or singleton as shown in Fig. 3.39. Most common types of MF are triangular, trapezoidal and Gaussian shapes.

**Fuzzy Rules**

Fuzzy variables are processed by fuzzy logic rules, with MIN and MAX operators. Fuzzy logic is interpreted as extended Boolean logic. For binary “0” and “1”, MIN and MAX operators in fuzzy logic perform same calculations as AND and OR operators in Boolean logic respectively. Binary Operation using Boolean Logic and Fuzzy Logic is presented in Table 3.3.

**Table 3.2: Binary Operation using Boolean Logic and Fuzzy Logic**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **A** | **B** | **A AND B** | **MIN (A,B)** | **A OR B** | **MAX(A,B)** |
| 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 0 | 1 | 1 |
| 1 | 0 | 0 | 0 | 1 | 1 |
| 1 | 1 | 1 | 1 | 1 | 1 |



**Fig. 3.39: Different forms of membership functions**

**Fuzzy System Modeling**

### In proposed system, inputs are fuzzified first and then entered in framework containing fuzzy rules based on If-Then statements. These statements make fuzzy outputs of system which are defuzzified.

**Rule Formation**

For this application of wind forecasting, fuzzy rules are formed as shown in Table 3.5 rules are created with five variables for each of two inputs.

**Table 3.3: Rule Formation**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Avg. wind Speed** | **S. Deviation** | | | | |
| **VL** | **L** | **M** | **H** | **VH** |
| VL | L | L | L | M | M |
| L | L | L | M | M | H |
| M | L | M | M | M | H |
| H | M | M | H | H | H |
| VH | M | H | VH | VH | VH |

Four rules are formed using minimum, maximum and mean wind speed values of training pattern which are given as:

Rule 1: If x1(t) is about 0 THEN

Rule 2: If x1(t) is above 0 and below minimum wind speed THEN

Rule 3: If x1(t) is above minimum and below mean wind speed THEN

Rule 4: If x1(t) is above mean and below maximum wind speed THEN

Where, A1, A2, A3, A4, B1, B2, B3 and B4 are given by:

, , ,,

, ,

, ,

Where,

is mean wind speed

is minimum wind speed

is maximum wind speed

YES

NO

Initialization

Convert to Crisp

Wind Speed Data

MFs (MF)

Rule Base

Evaluate rules in Base

Combine results of each Rule

Convert Output data to Non-Fuzzy Values

Convert Crisp input data to Fuzzy values Using MF

Convert Output data to Non-Fuzzy Values

Inference

Inference

Fuzzification

Defuzzification

Initialization

Forecast with non-fuzzy values for Test values

Is min(MAPE), max(R2) ?

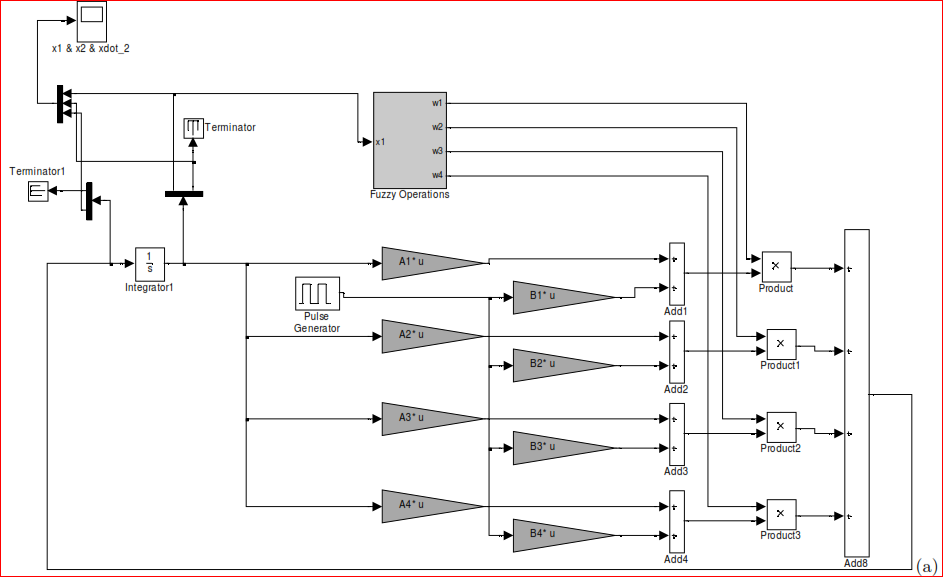
A

A

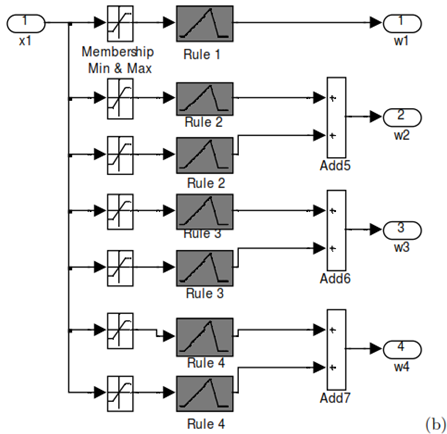
Forecast Future values

**Fig. 3.40: Flow Chart for Fuzzy Logic based Wind Speed Forecasting**

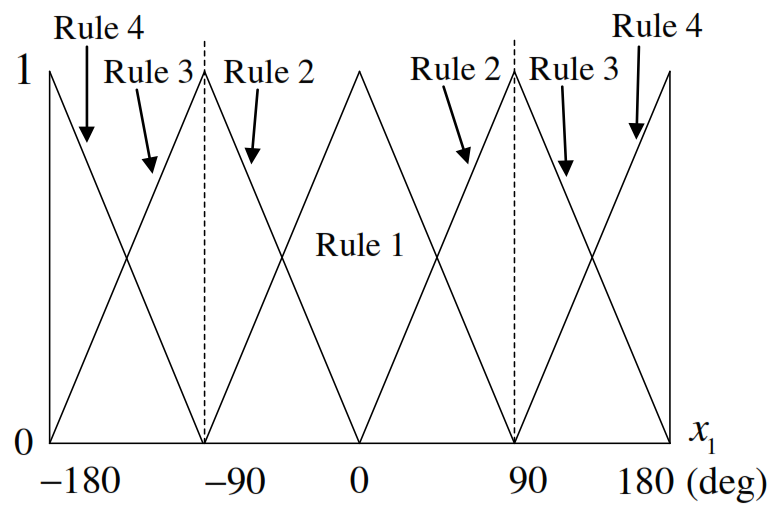
Flow chart for fuzzy logic system for forecasting of wind speed is given in Fig. 3.40. It has four stages namely, initialization, fuzzification, inference and defuzzification. Input data is converted to crisp values. This crisp value fed to MF to frame rule base. Crisp inputs are converted to fuzzy rules using MF. These rules are evaluated in rule base and results of each rule are combined and these are again inverted to non-fuzzy values. Prediction is done by using non-fuzzy values of data and compared with actual wind speed data. Rule base is continuously updated at each iteration until reaching global optimum MAPE and R2value. Rule formation in Table 3.5 is used as rule base for wind speed input values to set range in data. Construction of fuzy model with 4 rules using matlab simulink is shown in Fig. 3.41 (a) and respective Fuzzy operation block is represented in Fig. 3.41 (b). Four rule membership function is presented in Fig. 3.42.



**Fig. 3.41(a): Matlab - Simulink implementation of Fuzzy using rules**



**Fig. 3.41(b): Fuzzy Operation Block**



**Fig. 3.42: Four rule membership function [65]**

**3.2.2 Results of Fuzzy Logic Model**

Table 3.7 presents comparison of Wind Speed Forecasting results for 24 hour and 168 hour ahead for three sites. MAPE and R2are used to measure performance of models. It is found that MAPE is directly proportional to R2. Fuzzy Logic model result indicates there is an improvement in performance as compared to Neural Network model. Expected curves are very close to actual values. MAPE values range from 8.556% to 9.658% for 24 hour and 11.778% to 12.983% for 168 hour horizon. R2 values range from 0.624 to 0.824 for 24 hour and 0.450 to 0.783 for 168 hour horizon.

Forecasting results, regression curve and error distribution for fuzzy logic based forecasting for 24 hour and 168 hour are presented in Fig. 3.43 to 3.48. It is observed that forecasted curve is very close to actual wind speed curve. Forecasted curve nearly follows actual curve throughout forecast horizon. Regression curve linear and distribution error is uniformly distributed in negative and positive values.

**Table 3.4: Comparison of Wind Speed Results of Fuzzy Logic Model**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Site** | **24 hour** | | **168 hour** | |
| **MAPE** | **R2** | **MAPE** | **R2** |
| Bagalkot | 9.477 | 0.824 | 12.713 | 0.783 |
| Vijayapura | 8.556 | 0.624 | 11.778 | 0.450 |
| Bengaluru | 9.658 | 0.751 | 12.983 | 0.651 |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  | | --- | --- | |  | | |  |  |   **Fig. 3.43: (a) Actual Vs Forecasted wind speed for 24 hour using FL model (b) Regression plot of Actual Vs Forecasted (c) Error Distribution for Bagalkot Site** | |  |  | | --- | --- | |  | | |  |  |   **Fig. 3.44: (a) Actual Vs Forecasted wind speed for 24 hour using FL model (b) Regression plot of Actual Vs Forecasted (c) Error Distribution for Vijayapura Site** | |  |  | | --- | --- | |  | | |  |  |   **Fig. 3.45: (a) Actual Vs Forecasted wind speed for 24 hour using FL model (b) Regression plot of Actual Vs Forecasted (c) Error Distribution for Bengaluru Site** |
|  | | |
| |  |  | | --- | --- | |  | | |  |  |   **Fig. 3.46: (a) Actual Vs Forecasted wind speed for 168 hour using FL model (b) Regression plot of Actual Vs Forecasted (c) Error Distribution for Bagalkot Site** | |  |  | | --- | --- | |  | | |  |  |   **Fig. 3.47: (a) Actual Vs Forecasted wind speed for 168 hour using FL model (b) Regression plot of Actual Vs Forecasted (c) Error Distribution for Vijayapura Site** | |  |  | | --- | --- | |  | | |  |  |   **Fig. 3.48: (a) Actual Vs Forecasted wind speed for 168 hour using FL model (b) Regression plot of Actual Vs Forecasted (c) Error Distribution for Bengaluru Site** |

**3.3 Adaptive Neuro Fuzzy Inference Approach (ANFIS)**

System Modeling based on conventional mathematical tools (e.g., differential equations) is not well suited for dealing with ill-defined and uncertain systems. By contrast, a fuzzy inference system employing fuzzy if-then rules can model qualitative aspects of human knowledge and reasoning processes without employing precise quantitative analyses. This fuzzy modeling or fuzzy identification, first explored systematically by Takagi and Sugeno has found numerous practical applications in control prediction and inference [37]. However, there are some basic aspects of this approach in need for better understanding. More specifically:

* No standard methods exist for transforming human knowledge or experience into rule base and database of a fuzzy inference system.
* There is a need for effective methods for tuning membership functions (MF’s) so as to minimize output error measure or maximize performance index.

Adaptive-Network-based Fuzzy Inference System, or simply ANFIS, can serve as a basis for constructing a set of fuzzy if-then rules with appropriate membership functions to generate stipulated input-output pairs. Due to adaptive capability of ANFIS, its applications to adaptive control and learning control are immediate. Most of all, it can replace almost any neural networks in control systems to serve same purposes. Moreover, four of generic designs (i.e., supervised control, direct inverse control, neural adaptive control and back-propagation of utility) of neural networks in control, as proposed by Werbos [66], are also directly applicable schemes for ANFIS. Active role of neural networks in signal processing also suggests similar applications of ANFIS. Nonlinearity and structured knowledge representation of ANFIS are primary advantages over classical linear approaches in adaptive filtering and adaptive signal processing, such as identification, inverse modeling, predictive coding, adaptive channel equalization, adaptive interference (noise or echo) canceling, etc [75].

**3.3.1 Proposed ANFIS Model**

ANFIS works to tune parameters and structure of a fuzzy inference system is recognized by guidelines of adaptive neural learning guidelines. Network is regarded both as with capabilities of learning fuzzy rules from data and an adaptive fuzzy inference system, as connectionist design provided with linguistic significance.

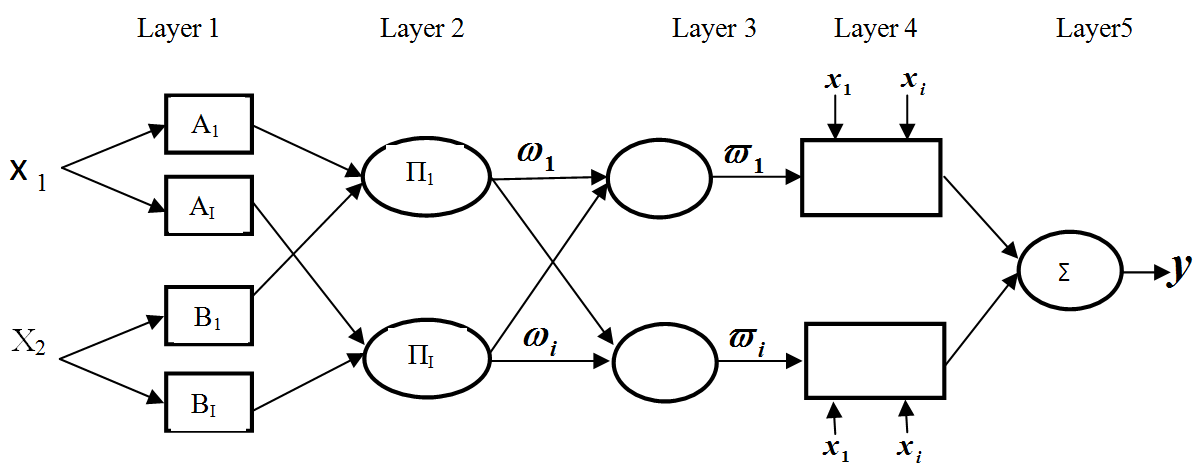
|  |
| --- |
| model dia.jpg |

**Fig. 3.49: ANFIS block diagram [75]**

ANFIS Block diagram is shown Fig. 3.49. Using given input information, ANFIS technique builds up a framework. FIS structure maps commitments through information enrollment work and connected parameters. ANFIS displaying process starts by obtaining an informational collection and separating it into preparing and testing informational indexes. Preparation of informational index is used to find hidden reason parameters for participation capacities by consistently dispersing each of enrollment capacities [61,66]. A threshold value for a difference between desired output and actual is determined. Then an error for each data pair is found to compare a model with actual system with testing data set. ANFIS facility with neural learning set of laws is used to recognize, tune parameters and configure fuzzy inference system. System is viewed both as an adaptive fuzzy inference system with abilities to take in learning a fuzzy set of laws from data and as a connectionist structural design provided with linguistic meaning.

**Architecture of ANFIS Model**

ANFIS is constructed by 5 layers FFNN which includes fuzzification, rule, normalization, defuzzification layers and single summation neuron. Architecture for ANFIS is presented in Fig. 3.50.



**Fig. 3.50: ANFIS Architecture diagram [75]**

**Layer 1:** In this layer each node is adjusted to a function constraint. Output of each node is a grade of membership value that is provided by contribution of MF is given by:

|  |  |  |
| --- | --- | --- |
|  |  | (3.8) |
|  |  | (3.9) |

Where,

is output of layer 1

X and y are inputs of node i

Ai and Bi  are fuzzy sets

μAi(x) and μBi-2(y) are grades of MF

**Layer 2**: This layer uses AND operator to fuzzify inputs and output at every node. Increasing signal came from this node and conveyed to further node. Firing strength of each regulation is represented by every node in this layer. In second layer, T standard operator with general execution for example, AND is connected to acquire output is written as:

|  |  |  |
| --- | --- | --- |
|  |  | (3.10) |

Where,

is Firing strength of each rule

**Layer 3**: Normalization is done to get firing strength from past layer. Every node in this layer is fixed or non adaptive. Each node is a figuring of proportion between rules firing strength and sum of all rules firing strengths. This outcome is called standardized firing strength. Output of 3rd layer is normalization firing potency is given by:

|  |  |  |
| --- | --- | --- |
|  |  | (3.11) |

**Layer 4**: Normalized product of firing strength and a first order is output polynomial in this layer is given by:

|  |  |  |
| --- | --- | --- |
|  |  | (3.12) |

Where,ωi is output of layer 3. pi , qi and ri are consequents parameters.

**Layer 5**: In this layer, distinct node is fixed that estimates on whole output as summary of all received signals from earlier node.

Overall output of model is given by:

|  |  |  |
| --- | --- | --- |
|  |  | (3.13) |

**Generalized Bell Shaped Membership Function**

Generalized Bell shaped MF is used in several applications is given by [75]:

|  |  |  |
| --- | --- | --- |
|  |  | (3.14) |
|  |  | (3.15) |
|  |  | (3.16) |
|  |  | (3.17) |

Where,

ai, bi, ci are states of membership function

**Sugeno Fuzzy Inference System**

Sugeno fuzzy inference is similar to Mamdani system in many regards initial two sections of FIS is applying fuzzy operator and fuzzifying data sources are precisely identical. Sugeno output MFs are either linear or steady output but in case of Mamdani are different. It is computationally efficient, optimization, works well with linear techniques and adaptive techniques. Continuity of output surface and mathematical analysis, sugeno is well suited. For a 0th order Sugeno model, output point O is a constant (a=b=0). Output point Oi of each rule is biased by firing strength wi of rule is given by:

|  |  |  |
| --- | --- | --- |
|  |  | (3.18) |

Where F1(.) and F2 (.) are Membership functions for Inputs 1 and 2. Final output given by:

|  |  |  |
| --- | --- | --- |
|  |  | (3.19) |

Where,

N is number of rules

Zi is output at final layer

**3.3.2 Mathematical Representation of ANFIS**

Mathematical representation of ANFIS is framed using output of each layer functions. ANFIS is considered as a parameterized nonlinear mapping of *f* which is given by [75]*:*

|  |  |  |
| --- | --- | --- |
|  |  | (3.20) |

Where,  *is* a position of output singleton if Sugeno way of thinking is applied. MF corresponds to input of regulation l. Connective basis is carried out by a defuzzification and product by center of gravity method. This is written as:

|  |  |  |
| --- | --- | --- |
|  |  | (3.21) |

Where and is given by:

|  |  |  |
| --- | --- | --- |
|  |  | (3.22) |

If *F* is a continuous, nonlinear map on a solid plane, then *f* can be estimated by *F* to any accuracy given by:

|  |  |  |
| --- | --- | --- |
|  |  | (3.23) |

Where is Fuzzy system function

**Theorem 1:** Let F be a enclosed function of [m, n], E={v1...vk} and set of points in [m, n]. Then least squares polynomial of grade ≤l, which is minimized by function:

|  |  |  |
| --- | --- | --- |
|  |  | (3.24) |

Where,

f(vi) is a enclosed function

p(vi) is lease square polynomial

**Theorem 2:** If FϵC[a b], then for any n≥0, there exist a best polynomial similar to πn of degree ≤n such that over all polynomials *p* of degree ≤n is given by:

|  |  |  |
| --- | --- | --- |
|  |  | (3.25) |

Where,

F is enclosed function

un is nth polynomial

p is least square polynomial

**Gradient Decent Learning Rule**

|  |  |
| --- | --- |
|  | i=1, 2, 3,….R  F is estimated output value by ANFIS |
|  | D is actual output |
|  | Gradient of ANFIS’s output builds ANFIS’s output nearer to actual output |
|  | This is done by updating values of  parameters (e.g., a, c,…) over n (iteration/step)  η is learning rate |

**Least Squares Estimator Method**

Least square estimator method is commonly used to estimate unidentified parameters of functions. Linearly parameterized expression of linear model output y is given by:

|  |  |  |
| --- | --- | --- |
|  |  | (3.26) |

Where,

is input vector of model.

are identified functions of u

are unknown parameters to be estimated.

To recognize unidentified parameters, generally need an examination to get a planning informational index gathered of sets. They compare to information yield sets of reason framework. Substituting coordinate into conditions yield an arrangement of m direct conditions:

|  |  |  |
| --- | --- | --- |
|  |  | (3.27) |

Simplifying eq.(3.27) in a short form using matrix notations as given by:

|  |  |  |
| --- | --- | --- |
|  |  | (3.28) |

Where,

B is a m x n matrix

is an n x 1 unidentified vector of parameters

y is an output vector

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  |  |  | (3.29-3.31) |

row of combined data matrix denoted by is associated to ith Input output data pair is given by:

|  |  |  |
| --- | --- | --- |
|  |  | (3.32) |

To recognize individually unknown vector, it is essential that If A is square (m=n) and non singular, then can solve X by:

|  |  |  |
| --- | --- | --- |
|  |  | (3.33) |

Now, finding exact solution to search for a , which minimize amount of square error define by:

|  |  |  |
| --- | --- | --- |
|  |  | (3.34) |

Where, is error vector formed by a particular option of

**Learning Algorithm for ANFIS**

ANFIS uses Hybrid Algorithm to categorize MF Parameters of single output Sugeno type. Combination of Least square and back propagation gradient descent methods are used for training Fuzzy inference Systems. Two way hybrid learning algorithm for ANFIS is presented in Table 3.5.

**Table 3.5: Two ways of hybrid learning algorithms for ANFIS**

|  |  |  |
| --- | --- | --- |
| **Function**  **Parameters** | **Forward** | **Backward** |
| Basis | Constant | Gradient Descent |
| Resultant | Least squares estimator | Constant |
| Signals | Node outputs | Error signals |

**Error Measure**

Error in ANFIS is measured using amount of squared error defined by Pth entry of error estimate is given by:

|  |  |  |
| --- | --- | --- |
|  |  | (3.35) |

Where,

Tm,p is mth component of Pth target

is Element mth actual output vector

Complete error by summing individual function output error is given by:

|  |  |  |
| --- | --- | --- |
|  |  | (3.36) |

Execution of gradient descent E error rate for Pth training data for all node output O is given by:

|  |  |  |
| --- | --- | --- |
|  |  | (3.37) |

Where,

is partial differential of individual error Ep with respect to output of kth element.

Update of formula is provided by modified input output pairs is given by:

|  |  |  |
| --- | --- | --- |
|  |  | (3.38) |

Off line wisdom alter rule is based on derivative of general error with respect to α [68]:

|  |  |  |
| --- | --- | --- |
|  |  | (3.39) |

**Hybrid Learning Rule**

Hybrid learning rule framed using Least Square estimator method. Output of hybrid learning rule is a function of input variable and number of parameters used in function is given by:

|  |  |  |
| --- | --- | --- |
|  |  | (3.40) |

Where,

I is Vector of input variables

S is set of parameters

F is ANFIS function implementation

Function of is complex function is linear in some elements of S since these components is distinguished by LSM. Set parameter S is decayed into twice sets direct sum, S2 is linear elements of (I,S) is given by:

|  |  |  |
| --- | --- | --- |
|  |  | (3.41) |

Where S is a linear element

**Combination of Gradient Descent Backward Pass and LSE**

Every output is computed by derived error and feedback from output to input is given by [66].

|  |  |  |
| --- | --- | --- |
|  |  | (3.42) |
|  |  | (3.43) |

Parameter in S2 is modified by gradient method is given by:

|  |  |  |
| --- | --- | --- |
|  |  | (3.44) |

**Fig. 3.52: Flow chart for wind forecast using ANFIS**

START

Load Train, Test Data for FIS Generation

Input Parameters and MFs FIS Model Optimization Method, Number of Epochs for Training and Testing Data

Training Data into ANFIS System

Training Finished ?

Results after Training

Checking Data into ANFIS

Testing Finished ?

View ANFIS Structure: Error Curve, Generated Rules, Adjusted MF and Forecast Output

Stop

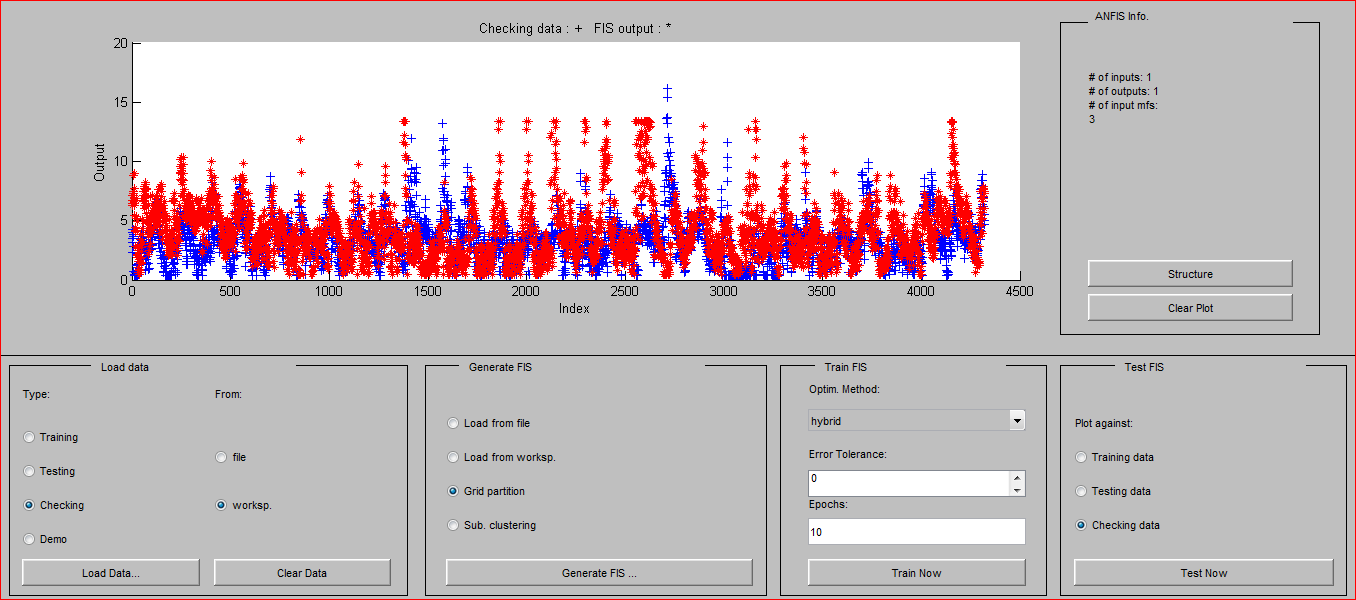
NO

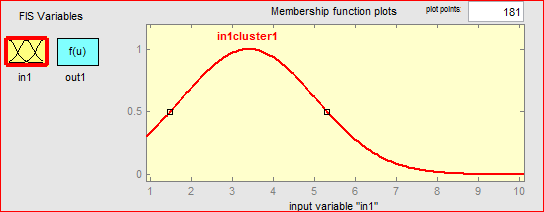
Yes

No

Yes

Flow chart for wind speed forecasting using ANFIS is presented in Fig. 3.52. Inference systems with Sugeno type fuzzy are recognized by hybrid algorithm for learning with use of ANFIS. Training data set is followed by grouping of least square method and back propagation method training for gradient descent FIS MFs parameters. Model support with this choice of inspection for model over an informational collection is known as inspection data set. Initial condition of ANFIS training used by GENFIS 1 creates a Sugeno kind of FIS Structure. Grid data is separated by using a single output Sugeno fuzzy type inference system was generated by Genfis1.



****

**Fig. 3.53: ANFIS Editor and Membership Function**

**3.3.3 Results of ANFIS Model**

Historical wind speed data of three sites for period of 7 years are used to train ANFIS model. Table 3.6 give comparison of forecasted results for 24 hour and 168 hour for three sites. It is found that MAPE range from 8.24% to 9.61% for 24 hour and 10.00% to 10.82% for 168 hour. There is remarkable improvement in forecasting accuracy as compared to NN and FL models.

**Table 3.6: Comparison of Wind Speed Results of ANFIS Model**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Site** | **24 hour** | | **168 hour** | |
| **MAPE** | **R2** | **MAPE** | **R2** |
| Bagalkot | 8.2488 | 0.910 | 10.0035 | 0.875 |
| Vijayapura | 8.8870 | 0.528 | 10.8248 | 0.492 |
| Bengaluru | 9.6179 | 0.662 | 10.0610 | 0.699 |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  | | --- | --- | |  | | |  |  |   **Fig. 3.53: (a) Actual Vs Forecasted wind speed for 24 hour using ANFIS model (b) Regression plot of Actual Vs Forecasted (c) Error Distribution for Bagalkot Site** | |  |  | | --- | --- | |  | | |  |  |   **Fig. 3.54: (a) Actual Vs Forecasted wind speed for 24 hour using ANFIS model (b) Regression plot of Actual Vs Forecasted (c) Error Distribution for Vijayapura Site** | |  |  | | --- | --- | |  | | |  |  |   **Fig. 3.55: (a) Actual Vs Forecasted wind speed for 24 hour using ANFIS model (b) Regression plot of Actual Vs Forecasted (c) Error Distribution for Bengaluru Site** |
|  | | |
| |  |  | | --- | --- | |  | | |  |  |   **Fig. 3.56: (a) Actual Vs Forecasted wind speed for 168 hour using ANFIS model (b) Regression plot of Actual Vs Forecasted (c) Error Distribution for Bagalkot Site** | |  |  | | --- | --- | |  | | |  |  |   **Fig. 3.57: (a) Actual Vs Forecasted wind speed for 168 hour using ANFIS model (b) Regression plot of Actual Vs Forecasted (c) Error Distribution for Vijayapura Site** | |  |  | | --- | --- | |  | | |  |  |   **Fig. 3.58: (a) Actual Vs Forecasted wind speed for 168 hour using ANFIS model (b) Regression plot of Actual Vs Forecasted (c) Error Distribution for Bengaluru Site** |

|  |  |  |
| --- | --- | --- |
|  |  |  |
|  |  |  |
| **Fig. 3.59: One Year Wind Speed Forecast using ANFIS** | **Fig. 3.60: January month Wind Speed Forecasted outputs** | **Fig. 3.61: 1st Week of January month Wind Speed Forecasted outputs** |
|  |  |  |
|  |  |  |
|  |  |  |
| **Fig. 3.62: 1st Day of January month Wind Speed Forecasted outputs** | **Fig. 3.63: February month Wind Speed Forecasted outputs** | **Fig. 3.64: March month Wind Speed Forecasted outputs** |

|  |  |  |
| --- | --- | --- |
|  |  |  |
| **Fig. 3.65: April month Wind Speed Forecasted outputs** | **Fig. 3.66: May month Wind Speed Forecasted outputs** | **Fig. 3.67: June month Wind Speed Forecasted outputs** |
|  |  |  |
|  |  |  |
| **Fig. 3.68: July month Wind Speed Forecasted outputs** | **Fig. 3.69 (a): Agaust month Wind Speed Forecasted outputs** | **Fig. 3.69(b): Agaust month Wind Speed Forecasted outputs** |

**3.3.4 Case Study: Wind Speed Forecasting for Yearly, Monthly, Daily Average Wind Speed using ANFIS Model**

A case study conducted for forecasting yearly, monthly, daily average wind speed data. Graphical results presented in Fig. 3.59 to 3.69 and comparison of MAPE values are presented in Table 3.7 to 3.13 for different training algorithm.

**Table 3.7: January month error value for each MFs**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Sl.** | **Type of MFs** | **MAE** | **MAPE** | **Time Taken**  **t in sec** |
| 1. | Generalized bell(gbell) | 0.2037 | 6.3100 | 0.2794 |
| 2. | Triangular(tri) | 0.2058 | 6.4337 | 0.3690 |
| 3. | Trapezoidal(trap) | 0.2030 | 6.3163 | 0.3351 |
| 3. | Gaussian(gauss) | 0.2033 | 6.3320 | 0.3227 |

**Fig. 3.70: January month MAE and MAPE v/s MFs**

**Table 3.8: 1st Week of January month error value for each MFs**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Sl.** | **Type of**  **MFs** | **MAE** | **MAPE** | **Time Taken**  **t in sec** |
| 1. | Generalized bell(gbell) | 1.4728×10-4 | 2.0804 | 0.1146 |
| 2. | Triangular(tri) | 0.1549 | 3.3729 | 0.2230 |
| 3. | Trapezoidal(trap) | 0.1486 | 3.2095 | 0.2551 |
| 3. | Gaussian(gauss) | 0.1495 | 3.2359 | 0.2087 |

**Fig. 3.71: 1st Week of January month MAE and MAPE v/s MFs**

**Table 3.9: 1st Day of January month error value for each MFs**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Sl.** | **Type of Membership**  **Functions** | **MAE** | **MAPE** | **Time Taken**  **t in sec** |
| 1. | Generalized bell(gbell) | 0.0998 | 3.0428 | 0.2352 |
| 2. | Triangular(tri) | 0.1033 | 3.1632 | 0.2049 |
| 3. | Trapezoidal(trap) | 0.1011 | 3.0851 | 0.2187 |
| 3. | Gaussian(gauss) | 0.1001 | 3.0512 | 0.2031 |

**Fig. 3.72: 1st Day of January month MAE and MAPE v/s MFs**

**Table 3.10: 1st hour 1st Day of January month error value for each MFs**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Sl.** | **Type of**  **MFs** | **MAE** | **MAPE** | **Time Taken**  **t in sec** |
| 1. | Generalized bell(gbell) | 0.0998 | 3.0428 | 0.1997 |
| 2. | Triangular(tri) | 0.1033 | 3.1632 | 0.2091 |
| 3. | Trapezoidal(trap) | 0.0988 | 3.0428 | 0.2063 |
| 3. | Gaussian(gauss) | 0.0990 | 3.0430 | 0.2979 |

**Fig. 3.73: 1st hour 1st Day of January month MAE and MAPE v/s MFs**

ANFIS with Generalized Bell, Triangular, Gaussion and Trapezoidal membership functions are compared and presented in Table 3.10 to 3.13 and Fig. 3.70 to 3.73. It is found that performance of g-bell MF is greater than remaining MFs. MAPE for gbell-6.31%, Tri-6.44%, Trap-6.32% and gauss-6.33%. Using ANFIS tool in matlab2014a, training and testing data are loaded. FIS structure is generated by selecting type of MF. In ANFIS tool got error value for gbell, tri mf, trap mf and gauss mf. In these MFs triangular MF got error value compared to other MF.

**3.4 Support Vector Machine (SVM) models**

In machine learning, SVMs are supervised learning models with connected learning algorithms that analyze data used for regression analysis and classification. Each marked as belonging one or other two categories, SVM training algorithm builds a model that assigns new examples to one category or other, making it a non-probabilistic [binary](https://en.wikipedia.org/wiki/Binary_classifier" \o "Binary classifier) [linear classifier](https://en.wikipedia.org/wiki/Linear_classifier" \o "Linear classifier) although methods such as [Platt scaling](https://en.wikipedia.org/wiki/Platt_scaling" \o "Platt scaling) exist to use SVM in a probabilistic classification setting. An SVM model is a representation of examples as points in space, mapped so that examples of separate categories are divided by a clear gap that is as wide as possible. New examples are then mapped into that same space and predicted to belong to a category based on which side of gap they fall [76-81].

**3.3.1 Mathematical Modeling of Support Vector Machine**

Support vector machine implements SRM (Statistical Risk Minimization) principle [82].

Let, feature spaceand  be a vector of weights deciding a hyper plane is written as:

|  |  |  |
| --- | --- | --- |
|  |  | (3.45) |

Set of hyper planes satisfying conditions is given by:

|  |  |  |
| --- | --- | --- |
|  |  | (3.46) |

Where,

is smallest sphere radius containing vectors

is weights norm and

is an estimation of **VC** (**V**apnik-**C**hervonenkis) dimension

SVM separates training data using VC dimension is given by[82]:

|  |  |  |
| --- | --- | --- |
|  |  | (3.47) |

Where,

is between {-1 , +1}

is elements of training data,

is bias function

Feature space equality is given by:

|  |  |  |
| --- | --- | --- |
|  |  | (3.48) |

To control generalization ability of SVM in separating hyper planes that minimizes functional is given by:

|  |  |  |
| --- | --- | --- |
|  |  | (3.49) |

With probability 1-η hyper planes that separates data with bound on test error given by:

|  |  |  |
| --- | --- | --- |
|  |  | (3.50) |

Where  is dimension VC of set of hyper planes. Now approximating **VC** maximal margin hyper plane is obtained by:

|  |  |  |
| --- | --- | --- |
|  |  | (3.51) |

Where,

|  |  |  |
| --- | --- | --- |
|  |  | (3.52) |
|  |  | (3.53) |

**Radial Basis Function Machines**

Classical radial basis function machines use set of decision rule given by:

|  |  |  |
| --- | --- | --- |
|  |  | (3.54) |

Where,

is Euclidian distance

Coefficient of expansion

Function is a monotonic nonnegative function is given by:

|  |  |  |
| --- | --- | --- |
|  |  | (3.55) |
|  |  | (3.56) |

Where,

is width parameter of kernel function

are support vectors

**Polynomial Learning Machine**

Function for convolution is used to construct polynomial decision rules of degree, inner product is given by:

|  |  |  |
| --- | --- | --- |
|  |  | (3.57) |

It contains all products up to degree. Using system, decision function given by:

|  |  |  |
| --- | --- | --- |
|  |  | (3.58) |

Where  is dimensional polynomial in -dimensional input space

### Two-Layer Neural Networks

Two-layer neural network is defined by choosing kernels as:

|  |  |  |
| --- | --- | --- |
|  |  | (3.59) |

Where,

is sigmoid function

v, c are parameters VC Dimension

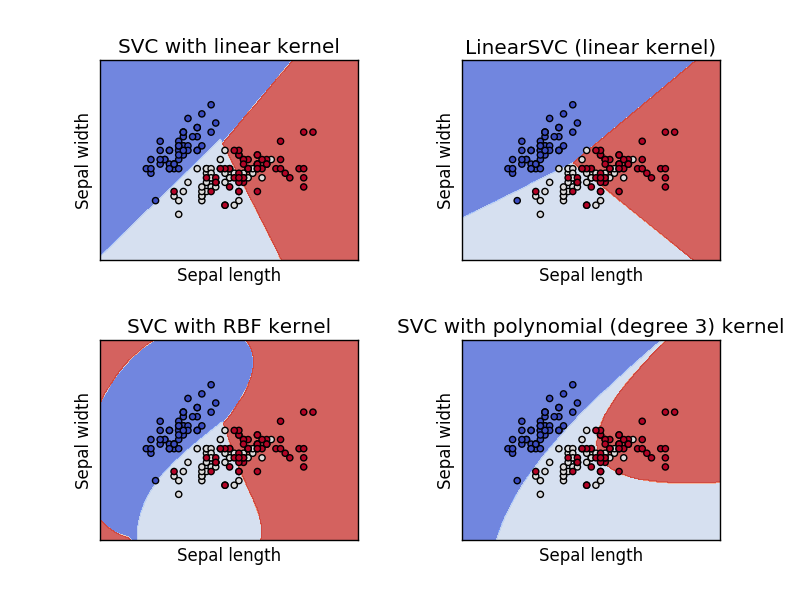
SV machine is constructed by implementing rules:

|  |  |  |
| --- | --- | --- |
|  |  | (3.60) |

Different types of kernel functions are presented in Fig. 3.75. Four types of Kernel functions namely SVC with Linear kernel, Linear SVC, SVC RBF Kernel, and SVC polynomial kernel are tested.

**3.3.2 Structure and Features of SVM model**

Architecture of two layer machine, determining number *N* of hidden units ( number of support vectors), Vectors of weights  in neurons of first layer and Vectors of weights for second layer (values of). Structure of SVM model is given in Fig. 3.75. Support vectors x1 to xn are transformed non-linearly using kernel functions K(x1,x2,..,xn) with weights α1 y1, α2 y2, α3 y3,… αn yn. Finally high dimension feature space is converted to low dimension feature space given by f(x) by choosing of proper hyper-plane for maximizing margin which is presented in Fig. 3.76 and Fig. 3.77. Optimal hyper-plane is chosen by properly selecting support vectors as shown in Fig. 3.77.



**Fig. 3.74: Types of Kernel Function [81]**

K(x1,x)

K(x2,x)

K(x2,x)

Y

α1 y1

X1

X2

X3

X2

α2 y2

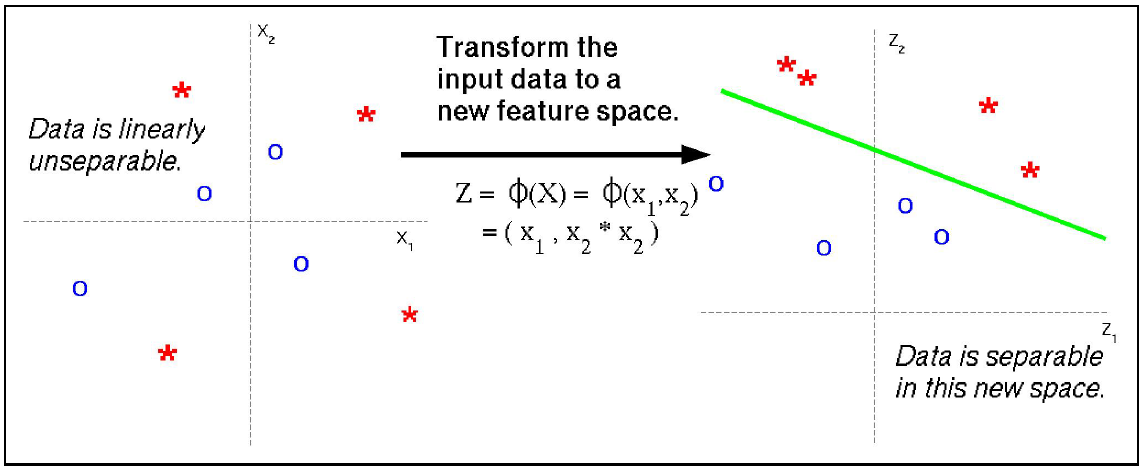
αn yn

Weights αi yi

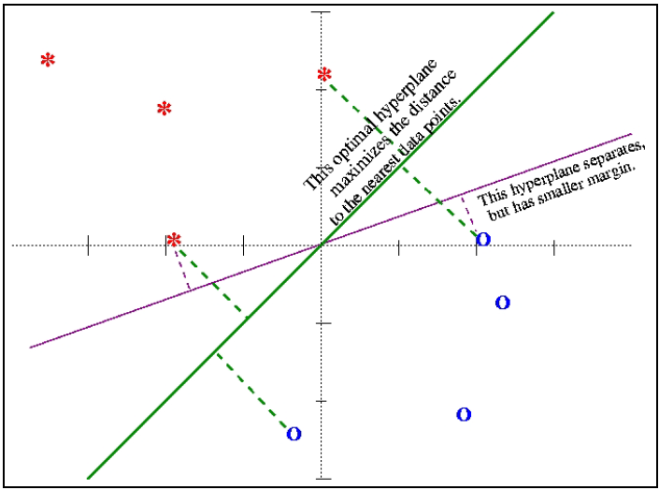
Nonlinear transformation based on Support Vectors x1 to xn

Decision Rule

**Fig. 3.75: Structure of SVM Model [82]**

****

**Fig. 3.76: Data Separation in a Feature Space**

****

**Fig. 3.77: Choosing hyper plane for maximizing margin**

**Steps for SVM model**

Detailed procedure for short-term forecasting of wind speed using SVM model is presented in Fig. 3.77 is summarized as:

* Loading wind speed data to SVM model, normalization of data into training, testing and validation, select number of Iteration for training.
* Unique (group train) to avoid repetition of same values
* Empty matrix formation for data size. SVM training, trains a SVM classifier on data taken from two groups.
* Kernels functions are RBF, linear, quadratic, and polynomial.
* Check group is a vector or a character array, orientation of data, grouping data, Sizing of data.
* Converting group index from 2 to (-1 to +1). Calculate bias by applying indicator function to support vectors with largest alpha to maximize margin
* Build up compound matrix for SVM solver. Calculate pivots of separating lines from support vectors again, save variables
* Get probability estimates of test instance using each model, classify data using SVM
* Forecast wind speed and obtain MSE and MAPE values along with graphs

Raw wind speed time series data is not suitable to develop mathematical model for SVM unlike other models. Classified data is used to train SVM model. Prior to processing and training SVM model, data is normalized by:

|  |  |  |
| --- | --- | --- |
|  |  | (3.61) |

This normalized data is divided into three parts, training, testing and validation in ratio of 50%:25%:25%. Training data further classified as two groups, one containing normalized data and another containing group data in {-1,+1}. This classification is done on averaged classification. Below average data is grouped as {-1} and above average data is grouped as {+1}. Kernel functions are selected using characteristics of data. Three kernel functions are used to train SVM model. Bias is calculated by applying indicator function to SVM with largest alpha to maximize margin. Detailed procedure for SVM based wind speed forecasting is presented in Fig. 3.78.

|  |
| --- |
| Training finished  Start  Load data  Bias calculation with largest alpha to maximize margin  Normalization of data into train & test  No. of epoch N=100  Selection of kernel function  Trains a SVM classifier on data taken from 2 groups  Probabilistic estimates of test instance using each model  MAPE &MSE, Largest alpha to maximize margin  NO  YES |
| **Fig. 3.78. Flow Chart for Wind Speed forecast with SVM** |

**3.3.3 Results of SVM Model**

Function flow chart of Support Vector Machine is used for wind speed forecast of 24 hour and 168 hour prediction horizon with three site data at Bagalkot, Vijayapura and Bengaluru sites. Initially data is classified by using linear classifier, and then this data is used frame kernel functions. Comparison of MAPE values for 24 hour and 168 hour is given in Table 3.11 and Fig. 3.79 to 3.83.

**Table 3.11: Comparison of Wind Speed Results of SVM Model**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Site** | **24 hour** | | **168 hour** | |
| **MAPE** | **R2** | **MAPE** | **R2** |
| Bagalkot | 7.938 | 0.905 | 9.3881 | 0.884 |
| Vijayapura | 6.842 | 0.871 | 9.7930 | 0.831 |
| Bengaluru | 6.531 | 0.737 | 9.6230 | 0.709 |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  | | --- | --- | |  | | |  |  |   **Fig. 3.79: (a) Actual Vs Forecasted wind speed for 24 hour using SVM model (b) Regression plot of Actual Vs Forecasted (c) Error Distribution for Bagalkot Site** | |  |  | | --- | --- | |  | | |  |  |   **Fig. 3.80: (a) Actual Vs Forecasted wind speed for 24 hour using SVM model (b) Regression plot of Actual Vs Forecasted (c) Error Distribution for Vijayapura Site** | |  |  | | --- | --- | |  | | |  |  |   **Fig. 3.81: (a) Actual Vs Forecasted wind speed for 24 hour using SVM model (b) Regression plot of Actual Vs Forecasted (c) Error Distribution for Bengaluru Site** |
|  | | |
| |  |  | | --- | --- | |  | | |  |  |   **Fig. 3.82: (a) Actual Vs Forecasted wind speed for 168 hour using SVM model (b) Regression plot of Actual Vs Forecasted (c) Error Distribution for Bagalkot Site** | |  |  | | --- | --- | |  | | |  |  |   **Fig. 3.83: (a) Actual Vs Forecasted wind speed for 168 hour using SVM model (b) Regression plot of Actual Vs Forecasted (c) Error Distribution for Vijayapura Site** | |  |  | | --- | --- | |  | | |  |  |   **Fig. 3.84: (a) Actual Vs Forecasted wind speed for 168 hour using SVM model (b) Regression plot of Actual Vs Forecasted (c) Error Distribution for Bengaluru Site** |

**3.5 Comparison of AI Techniques**

Neural Network, Fuzzy Logic, Adaptive Neuro Fuzzy Inference System and SVM models are compared for three sites with three seasonal patterns.

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Table 3.12: Comparison of MAPE values for AI Techniques for combined single and two paired models** | | | | | | | | | |
| **24 hour pattern** | **Bagalkot** | | | **Vijayapura** | | | **Bengaluru** | | |
| **P1** | **P2** | **P3** | **P1** | **P2** | **P3** | **P1** | **P2** | **P3** |
| NN | 12.37 | 17.38 | 15.90 | 17.92 | 13.80 | 12.17 | 15.15 | 16.09 | 13.04 |
| FL | 13.10 | 13.33 | 15.57 | 19.33 | 13.43 | 13.42 | 15.21 | 13.38 | 13.99 |
| ANFIS | 13.61 | 16.44 | 18.20 | 17.95 | 13.57 | 13.72 | 15.78 | 15.00 | 15.96 |
| SVM | 11.33 | 13.95 | 16.93 | 20.39 | 16.57 | 10.98 | 15.86 | 15.76 | 13.95 |
| NN-FL | 13.79 | 15.70 | 16.44 | 13.93 | 15.28 | 13.39 | 11.86 | 15.49 | 13.92 |
| NN -ANFIS | 11.45 | 16.70 | 17.48 | 21.32 | 13.96 | 13.62 | 16.39 | 15.33 | 15.55 |
| NN -SVM | 11.07 | 16.36 | 17.23 | 17.78 | 16.54 | 13.42 | 13.77 | 16.45 | 15.32 |
| FL - ANFIS | 12.28 | 16.39 | 17.07 | 18.71 | 13.80 | 13.20 | 15.49 | 15.59 | 15.13 |
| FL -SVM | 11.05 | 16.98 | 17.07 | 20.79 | 12.70 | 15.51 | 15.92 | 13.84 | 16.29 |
| ANFIS -SVM | 12.41 | 15.31 | 15.28 | 12.22 | 13.79 | 12.57 | 12.32 | 15.05 | 13.93 |
| **168 hour Pattern** | **Bagalkot** | | | **Vijayapura** | | | **Bengaluru** | | |
| **P1** | **P2** | **P3** | **P1** | **P2** | **P3** | **P1** | **P2** | **P3** |
| NN | 17.50 | 13.92 | 18.52 | 17.50 | 17.86 | 17.74 | 17.50 | 16.39 | 18.13 |
| FL | 17.63 | 16.04 | 18.02 | 17.63 | 16.49 | 17.63 | 17.63 | 16.27 | 17.82 |
| ANFIS | 18.60 | 13.66 | 17.24 | 18.60 | 16.62 | 17.47 | 18.60 | 15.64 | 17.35 |
| SVM | 16.29 | 16.47 | 17.54 | 18.29 | 17.49 | 16.36 | 18.29 | 16.98 | 16.95 |
| NN- FL | 18.58 | 17.15 | 19.06 | 18.58 | 18.46 | 17.46 | 18.58 | 17.81 | 18.26 |
| NN- ANFIS | 17.12 | 16.98 | 19.25 | 17.12 | 17.62 | 17.66 | 17.12 | 17.30 | 18.45 |
| NN -SVM | 15.26 | 17.69 | 18.74 | 18.26 | 17.00 | 16.36 | 18.26 | 17.35 | 17.55 |
| FL- ANFIS | 19.38 | 17.11 | 19.14 | 19.38 | 17.19 | 16.68 | 19.38 | 17.15 | 17.91 |
| FL -SVM | 15.44 | 16.85 | 18.89 | 18.44 | 18.02 | 18.14 | 18.44 | 17.43 | 18.51 |
| ANFIS -SVM | 15.02 | 17.25 | 18.12 | 18.02 | 16.81 | 17.01 | 18.02 | 17.03 | 17.57 |

Comparison of MAPE for combinational single pair and two paired model for AI techniques are summarized in Table 3.12. These single and combinational models are tested with 18 set of patterns of 24 hour and 168 hour horizon for three sites. Each site having three patterns for both 24 hour and 168 hour patterns. 6 combinations formed with 4 type of models. It is found that, single SVM model with MAPE of 11.33% for 24 hour and 16.29% for 168 hour has outperformed remaining AI techniques. Further SVM combined with ANFIS has lesser MAPE of 11.05% for 24 hour and 15.02% for 168 hour. Comparison of Actual vs Forecasted wind speed for different AI models is presented in Fig. 3.86 to Fig. 3.91 for 24 hour and 168 hour prediction horizon. Also aggregate of all models is plotted. There is a close agreement between actual and forecasted wind speed in aggregated AI model.

**Fig. 3.86: Comparison forecasted with actual wind speed from AI models for Bagalkot Site for 24 hour**

**Fig. 3.87: Comparison forecasted with actual wind speed from AI models for Vijayapura Site for 24 hour**

**Fig. 3.88: Comparison forecasted with actual wind speed from AI models for Bengaluru Site for 24 hour**

**Fig. 3.89: Comparison forecasted with actual wind speed from AI models for Bagalkot Site for 168 hour**

**Fig. 3.90: Comparison forecasted with actual wind speed from AI models for Vijayapura Site for 168 hour**

**Fig. 3.91: Comparison forecasted with actual wind speed from AI models for Bengaluru Site for 168 hour**

**3.5 Statistical Tests on Forecasted Results for AI Techniques**

Tests conducted in previous sections are also used to validate forecasted wind speed result of artificial intelligent techniques. Paired t-test and solution time for Actual Vs Forecasted Wind Speed for 4 AI models is presented in Table 3.13. Response time for AI methods is represented graphically in Fig. 3.92. It is found that, time for solution from SVM range from 11.1 sec to 12.9 sec which is lowest among ANN, FL, ANFIS. ANFIS takes longer time for solution range from 23.6 sec to 25.1 sec. Further, Pearson correlation for SVM range from 0.859 to 0.877 which is also higher among AI models. Inference this solution time and Pearson correlation indicates superiority of SVM in responding for uncertain wind speed data.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Table 3.13: Paired t-test and solution time for Actual Vs Forecasted Wind Speed for 4 AI models** | | | | | | |
| **Site** | **Method** | **Pearson**  **Correlation** | **t-stat** | **t-critical one tail** | **t-critical two tail** | **Solution Time in Sec** |
| Bagalkot | ANN | 0.661 | -55.232 | 1.406 | 1.359 | 15.6 |
| FL | 0.736 | -98.958 | 1.553 | 1.674 | 13.9 |
| ANFIS | 0.734 | -48.844 | 1.378 | 1.305 | 23.6 |
| SVM | **0.873** | -73.695 | 1.283 | 1.132 | **12.8** |
| Vijayapura | ANN | 0.614 | -56.910 | 1.015 | 0.709 | 16.2 |
| FL | 0.743 | -121.905 | 1.403 | 1.352 | 13.2 |
| ANFIS | 0.773 | -76.167 | 1.278 | 1.121 | 25.1 |
| SVM | **0.859** | -79.422 | 1.362 | 1.180 | **12.9** |
| Bengaluru | ANN | 0.710 | -116.025 | 1.520 | 1.996 | 17.1 |
| FL | 0.755 | -202.545 | 1.387 | 1.323 | 13.8 |
| ANFIS | 0.773 | -76.167 | 1.278 | 1.121 | 25.1 |
| SVM | **0.877** | -53.674 | 1.368 | 1.285 | **11.1** |

**Fig. 3.92: Time required by methods to solve forecasting problem**

Two Sample F-Tests for variances assuming equal variance for actual vs forecasted values for 4 AI models is presented in Table 3.13. Results reveal that SVM model with highest F value ranging from 0.772 to 0.789 and F-critical value ranging from 0.608 to 0.649 has outperformed remaining AI models. F and F-critical value are plotted in Fig. 3.93. SVM’s quick response indicates its suitability short-term forecasting. This characteristic of SVM is further tested for 18 patterns and found that response time for all patterns is similar to that in 24 hour forecast.

**Table 3.14: Two Sample F-Test for variances (Assuming equal variance for actual vs forecasted values) for 4 AI models**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Site** | **Method** | **F** | **P Critical**  **one-tail** | **F-critical** |
| Bagalkot | ANN | 0.640 | 0.001 | 0.667 |
| FL | 0.681 | 0.123 | 0.742 |
| ANFIS | 0.666 | 0.008 | 0.654 |
| SVM | 0.772 | 0.000 | 0.608 |
| Vijayapura | ANN | 0.462 | 0.001 | 0.481 |
| FL | 0.664 | 0.100 | 0.666 |
| ANFIS | 0.678 | 0.002 | 0.606 |
| SVM | 0.764 | 0.000 | 0.646 |
| Bengaluru | ANN | 0.500 | 0.000 | 0.721 |
| FL | 0.655 | 0.146 | 0.658 |
| ANFIS | 0.672 | 0.002 | 0.606 |
| SVM | 0.789 | 0.000 | 0.649 |

**Fig. 3.93: Variation of F and F-Critical value in F-test for AI methods**

**Regression Test between Actual and Forecasted Wind Speed Values**

Two Sample F-Tests for variances (Assuming equal variance) for actual vs forecasted values for 4 AI Models is presented in Table 3.15. Regression response from four AI models is presented graphically in Fig. 3.93. Regression value for ANN model is found lowest among four models. R2value for ANN models range from 0.366 to 0.602. This shows failure of ANN model to response for uncertain data. Further regression response for SVM model range from 0.870 to 0.884 which is highest among AI models. This indicates superiority of SVM model for uncertain data. Regression test results of Bagalkot Site (a)Residual Plot, (b)Line Fit Plot (c)Normal Probability Plot for ANN, FL, ANFIS and SVM moels of Bagalkot , Vijapyapur and Bengaluru sites are presented in Fig. 3.95 to 3.97.

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Table 3.15: Regression and Two Sample F-Test for variances (Assuming equal variance) for actual vs forecasted values for 4 AI Models** | | | | | | | | | | |
| **Site** | **Method** | **Multiple**  **R2** | **R2** | **Adjusted**  **R2** | **SE** | **SS Regr** | **F** | **t-stat** | **Lower**  **95%** | **Upper**  **95%** |
| Bagalkot | ANN | 0.667 | 0.602 | 0.699 | 1.469 | 641.176 | 340.035 | 9.903 | 2.207 | 3.403 |
| FL | 0.664 | 0.668 | 0.867 | 1.087 | 793.737 | 769.266 | 1.748 | 0.097 | 1.034 |
| ANFIS | 0.750 | 0.754 | 0.671 | 1.524 | 615.250 | 303.869 | 11.652 | 2.644 | 3.809 |
| SVM | **0.881** | 0.884 | 0.581 | 1.061 | 211.528 | 213.910 | 10.007 | 1.748 | 2.757 |
| Vijayapura | ANN | 0.626 | 0.366 | 0.362 | 1.177 | 112.283 | 112.286 | 15.396 | 3.034 | 2.244 |
| FL | 0.669 | 0.698 | 0.696 | 0.855 | 213.212 | 335.445 | 3.841 | 0.583 | 2.008 |
| ANFIS | 0.788 | 0.579 | 0.576 | 1.063 | 206.649 | 209.560 | 10.037 | 1.763 | 2.700 |
| SVM | **0.889** | 0.879 | 0.601 | 1.029 | 217.317 | 233.683 | 9.276 | 1.587 | 2.523 |
| Bengaluru | ANN | 0.637 | 0.488 | 0.485 | 0.614 | 49.412 | 150.068 | 13.506 | 1.694 | 2.322 |
| FL | 0.655 | 0.683 | 0.681 | 0.501 | 69.090 | 315.361 | 2.312 | 0.036 | 0.850 |
| ANFIS | 0.788 | 0.579 | 0.576 | 1.063 | 206.649 | 209.560 | 10.037 | 1.763 | 2.017 |
| SVM | **0.879** | 0.870 | 0.661 | 1.547 | 606.658 | 290.098 | 8.835 | 2.080 | 2.837 |

**Fig. 3.94: Regression response of AI methods**

|  |  |  |
| --- | --- | --- |
|  |  |  |
| **Neural Network** | | |
|  | | |
|  |  |  |
| **Fuzzy Logic** | | |
|  | | |
|  |  |  |
| **Adaptive Neuro Fuzzy Inference System** | | |
|  | | |
|  |  |  |
| **Support Vector Machines** | | |
|  | | |

**Fig. 3.95: Regression test results of Bagalkot Site (a)Residual Plot, (b)Line Fit Plot (c)Normal Probability Plot for ANN, FL, ANFIS and SVM moels of Bagalkot Site**

|  |  |  |
| --- | --- | --- |
|  |  |  |
| **Neural Network** | | |
|  | | |
|  |  |  |
| **Fuzzy Logic** | | |
|  | | |
|  |  |  |
| **Adaptive Neuro Fuzzy Inference System** | | |
|  | | |
|  |  |  |
| **Support Vector Machines** | | |
|  | | |

**Fig. 3.96: Regression test results of Bagalkot Site (a)Residual Plot, (b)Line Fit Plot (c)Normal Probability Plot for ANN, FL, ANFIS and SVM moels of Vijayapura Site**

|  |  |  |
| --- | --- | --- |
|  |  |  |
| **Neural Network** | | |
|  | | |
|  |  |  |
| **Fuzzy Logic** | | |
|  | | |
|  |  |  |
| **Adaptive Neuro Fuzzy Inference System** | | |
|  | | |
|  |  |  |
| **Support Vector Machines** | | |
|  | | |

**Fig. 3.97: Regression test results of Bagalkot Site (a)Residual Plot, (b)Line Fit Plot (c)Normal Probability Plot for ANN, FL, ANFIS and SVM moels of Bengaluru Site**

**3.7 Salient Outcomes of AI Techniques based Forecasting of Wind Speed**

Four AI techniques are investigated for three site data with 3 patterns each for 24 hour and 168 hour patterns. Further two pair combinational model using 4 AI techniques are compared with this data. Methodologies and results analysis are presented in chapter. Salient outcomes of work are listed below:

* Four AI Techniques namely Neural Network, Fuzzy Logic, Adaptive Neuro Fuzzy Inference System and Support Vector Machine models are developed and tested for Bagalkot, Vijayapura and Bengaluru Sites.
* In Neural Network, Levenberg-Marquardt (LM), Bayesian Regularization (BR) and Scaled Conjugate Gradient (SCG) Algorithm are used to train model and tested for three sites. It is found that SCG algorithm performed better remaining two algorithms. Forecasting performance improved with for both 24 hour and 168 hour prediction. Further investigation of regression results indicates remarkable improvement for correlation between actual and forecasted wind speed values.
* A Fuzzy Logic model is developed for wind speed forecasting model using Rules framed from input data training pattern. It is found that performance is improved as compared to Neural Network in both 24 hour and 168 hour prediction.
* Combination of Neural Network and Fuzzy Logic based model namely Adaptive Neuro Fuzzy Inference System (ANFIS) is developed for wind speed forecasting. Wind speed forecasted is tested with Generalized bell, Triangular, Trapezoidal and Gaussian type MFs for three sites. Forecasting performance is improved as compared with Trapezoidal MF.
* Support Vector Machine model is developed using Radial Basis Function training algorithm. Forecasting results of SVM indicates a remarkable improvement over NN, FL and ANFIS model.
* Single and combinational models are tested with 18 set of patterns of 24 hour and 168 hour horizon for three sites. Each site having three patterns for both 24 hour and 168 hour pattern. 6 combinations formed with 4 types of models.
* It is found that, single SVM model with MAPE of 11.33% for 24 hour and 16.29% for 168 hour has outperformed remaining AI techniques. Further SVM combined with ANFIS has lesser MAPE of 11.05% for 24 hour and 15.02% for 168 hour.
* Test results reveal that regression response is highest and solution time is lowest in SVM model. Quick response SVM model is most suited for short-term forecasting. Further response of SVM model in uncertain situation is consistent. Further SVM model is adaptable to all type of training and test patterns.
* Validation of this SVM model is further tested in combination approach.