PREDICTION & CLASSIFICATION OF WEATHER USING BACK PROPAGATION ALGORITHM

PROJECT REPORT

Submitted in fulfilment for the JComponent of

Soft Computing (SWE1011)

CAL Course

in

M. Tech. – Software Engineering

by

HARISH D (18MIS0163)

NAVEEN KUMAR J(18MIS0395)

Under the guidance of

Dr. S. Hemalatha

SITE

VIT®

Vellore Institute of Technology

(Deemed to be University under section 3 of UGC Act, 1956)

School of Information Technology and Engineering

Fall Semester 2021-22

TABLE OF CONTENTS

Abstract:

In this project, we are going to create a model that predicts the weather and classifies it by using the soft computing technique “Back Propagation Algorithm”. Weather prediction could be vital as a result of they’re used to secure life and property. Weather prediction is that the use of science and innovation to anticipate the state of the temperature for a future time at a given space. Temperature forecasts are made by collecting quantitative data regarding the present state of the atmosphere. It’s the method of recording the parameters of weather like wind direction, wind speed, humidity, rainfall, the temperature then on. Here one parameter states temperature that differs by a couple of units and therefore the variation on alternative parameters saying humidness, temperature, and rain which are going to be foreseen with relevancy temperature. The proposed idea is tested utilizing the ongoing dataset. The outcomes make sure that our model has the potential for effective application for weather prediction.

Keywords

Back Propagation Algorithm, Feed-Forward Neural Network, ANN.

1.Introduction :

Weather Prediction is significant since it decides future atmospheric desires. The impact of weather and climate systems directly or indirectly influence the socioeconomic conditions of the citizens in various
ways that specifically in agriculture, water management, disaster mitigation, early warning system, energy. Weather process could be a dynamic and non-linear development. There's a demand to use applied mathematics post- process techniques on modelled forecast fields to reinforce the prediction quality and value. India is basically an agricultural nation and its economy (in spite of the event of the tertiary areas) is still mostly influenced by what comes from the farm. The environmental condition of any atmospheric region plays an important role in deciding success or failure of crop production
throughout the part of growth and development of plant. Weather forecast support to enhance the production of agriculture in scientific method, cut back risks and losses, increase crop and water use efficiency.

2.Dataset specification:

Our aim is to assemble dataset consisting weather parameters like temperature, humidity, dew point, visibility, atmospheric pressure, sea level, wind speed, wind direction etc. and perform all needed data
pre-processing tasks. Dataset is gathered from kaggle website https://www.kaggle.com/muthuj7/weather-dataset. The collected data is passed to the BPNN to train the neural network.

3.Literature Review

3.1 Review on various schemes:

1.PAPER TITLE: "ANN based Weather Prediction using Back Propagation Technique".

AUTHORS: Saboor Ahmad Kakar, Naveed Sheikh, Saleem Iqbal, Abdul Rehman.

PUBLISHED YEAR: January 2018.

REFERENCE:
https://www.researchgate.net/publication/327400959_Artificial_Neural_Network_based_Weather_Prediction_using_Back_Propagation_TechniqUe.

Authors enclosed the machine learning by change of integrity the neural networks with NWP to access the wind speed. ANN is one amongst the fastest developing ways of machine learning thought-about as non-direct discerning models to perform characterization and forecast. Artificial Neural Network (ANN) has to this point developed intent on be as a superior technique to enhance the preciseness and unwavering quality. During this manner, a multi-layered neural system is structured and ready with the present dataset and bought a affiliation between the present non-linear parameters of weather. However, the big range of meteorological information makes it troublesome or somehow not possible to perform analysis for weather prediction. By considering the classification approach of ANN we will conclude that their approach is way higher than ancient to spot associated patterns for consecutive events. They need planned the new technique with the employment of feed forward artificial neural network for weather prediction that is proved to be a more robust approach as compared to ancient approaches. The results unconcealed that by increasing the quantity of hidden layers, the trained neural network will classify and predict the weather variables with less error.

2.PAPER TITLE: “Weather Forecasting using Neural Network”.

AUTHORS: Priyanka Mahajan, Chhaya Nawale, Siddheshwar Kini, Prof. Krishnanjali Shind.

PUBLISHED YEAR: 2017.

VOLUME: Volume 5, Issue 01.

REFERENCE:
https://www.ijert.org/research/weather-forecasting-using-neural-networkIJERTCONV5IS01197.pdf

In this paper authors states that data mining technique along with neural network gives information for weather prediction which reduces cost as compared to any other prediction models. ANN with backpropagation uses the iterative approach and compares the observed output with targeted output and hence finding the error. This error is used to readjust the values of weights and bias to get a better output helping to minimize the error. Artificial Neural network with Back propagation algorithm seems to be most appropriate method for forecasting weather accurately than numerical differentiation.

3. PAPER TITLE: “Efficient Weather Forecasting Using Artificial Neural Network As Function
Approximator”.

AUTHORS: EI-Feghi, Z.Zubia, S.Abozgaya

PUBLISHED YEAR: January 2014.

REFERENCE:
https://www.sciencedirect.com/science/article/pii/S221201731200326X.

Forecasting is the alluded to as the procedure of estimation in unknown circumstances. Weather forecasting, particularly air temperature, is one of the most significant factors in numerous applications. This paper presents a methodology to create Artificial Neural Network (ANN) to conjecture air temperature. One significant design of neural network namely Radial Basis Function (RBF) will be utilized as a function approximator. To think of suitable places for the RBF neurons, the climate information was bunched into a few gatherings utilizing “k” means grouping calculation. While information required to make temperature forecasts has been accessible for a long while, the mind boggling connections between the information and its impact on the guess of temperature has regularly
end up being troublesome utilizing ordinary PC investigation. In this paper they have introduced a productive strategy for the determining of air temperature dependent on the utilization of perceptions of different meteorological factors. RBF ANN was utilized as a capacity approximator to anticipate the air Temperature.

4.PAPER TITLE: “Daily Weather Forecasting using Artificial Neural Network”.

AUTHORS: Meera Narvekar , Priyanca Fargose.

PUBLISHED YEAR: 2015.

REFERENCE: DOI:10.5120/21830-5088

In this paper, the review is conducted to analyze a more robust approach for forecasting that compares several techniques like Artificial Neural Network, Ensemble Neural Network, Back propagation Network, Radial Basis perform Network, General Regression Neural Network, Genetic algorithm, Multilayer Perceptron, Fuzzy clustering, etc. that area unit used for various kinds of prognostication. Among that neural network with the back propagation rule performs prediction with least error. Neural network could be a advanced network that is self-adaptive in nature. It learns by itself using the coaching information and generates some intelligent patterns that area unit helpful for forecasting the weather.
This paper reviews numerous techniques and focuses principally on neural network with back propagation technique for daily foretelling. The technique uses twenty

eight input parameters to forecast the daily weather in terms of temperature, rainfall, humidity, cloud condition, and weather of the day. The Artificial Neural Network model projected during this paper indicates all the parameters for input and output, coaching and testing information set, range of hidden layers 12 and neurons in every hidden layer, weight, bias, learning rate and activation perform. The Mean square Error between expected output and therefore the actual output is employed to check accuracy.

5.RESEARCH PAPER: "Temporal Weather Prediction using Back Propagation based Genetic Algorithm Technique".

AUTHORS NAME: Shaminder Singh, Jasmeen Gill.

VOLUME: Volume 6, Issue 12.

PUBLISHED YEAR: November 2014.

REFERENCES: http://www.mecs-press.org/ijisa/ijisa-v6-n12/IJISA-V6-N12-8.pdf

In this paper therefore as to show the temperature on specific data on knowledge series, Hybrid back propagation primarily based genetic rule approach is employed because it may be a illustrious technique to organize neural systems for weather prediction. The numerous disadvantages of this technique are that weather parameters were assumed to be freelance of every alternative and their relation with each other wasn't thought of. The authors have featured on the temperature expectation may be a worldly and statistic hooked in to method. A changed statistic primarily based weather prediction model is projected to eliminate the issues incurred in hybrid BP/GA technique. The projected temporal weather prediction model outperforms the previous models whereas playing for dynamic and chaotic weather. Because of nonlinearity in climatic material science, neural systems square measure acceptable to anticipate these earth science procedures. Back Propagation integrated with genetic rule is that the most significant to coach the neural networks. During this paper, back propagation neural network is employed for temperature prognostication.

The technical milestones, that are achieved by the researchers during this field has been reviewed and bestowed during this paper.

6.RESEARCH PAPER: : Machine Learning in Python for Weather Forecast based on Freely Available Weather Data

AUTHORS NAME: E.B. Abrahamsen, O. M. Brastein, B. Lie.

PUBLISHED YEAR: 2018

REFERENCE: https://ep.liu.se/ecp/153/024/ecp18153024.pdf

In this paper, the authors have focused on a new Python API for collecting weather data, and given simple, introductory examples of how such data can be used in machine learning. This data was used to train and fine tune various models. Autoregressive Artificial Neural Networks (ARANN) using Python's TensorFlow. The resulting models were used to predict the temperature of Porsgrunn with forecast horizons of 1, 3, 6 and 12 hours. In this paper the authors are using Error rate and learning rate as metrics to test their system's performance. They have used Metrological Institute data as their datasets. They have used two Experiment models one is auto-regressive neural network (AR-NN) model and second one is autoregressive neural network with exogenous input (ARXNN).ARX model was shown to slightly improve the prediction performance.

7.RESEARCH PAPER: An Efficient Machine Learning Regression Model for Rainfall Prediction

PUBLISHED YEAR: 2015

AUTHORS NAME: R. Usha Rani,T.K.Rama Krishna Rao. R. Kiran Kumar Reddy.

REFERENCE: https://www.semanticscholar.org/paper/An-Efficient-Machine-Learning-Regression-Model-for-Usharani-

Rao/894a61bc0bab8bbb43d8221d34e5861662d094bc

In this paper, the authors proposed a two-level approach for clustering large data set for rainfall data prediction with Self Organized Maps (SOM) and Support Vector Machine (SVM) with ID3. A self-organizing map (SOM) is a type of artificial neural network (ANN) that is trained using unsupervised learning to produce a low-dimensional (typically two-dimensional), discretized representation of the input space of the training samples, called a map, and is therefore a method to do dimensionality reduction. "Support Vector Machine" (SVM) is a supervised machine learning algorithm that can be used for both classification or regression challenges.ID3 algorithm, stands for Iterative Dichotomies 3, is a classification algorithm that follows a greedy approach of building a decision tree by selecting a best attribute that yields maximum Information Gain (IG) or minimum Entropy (H). By using two level architecture they have come up with 82% accuracy.

8.RESEARCH PAPER: WEATHER PREDICTION BY MACHINE LEARNING.

PUBLISHED YEAR: 2018

AUTHORS NAME: Shashank Singh, Faraz Ahmed Nagrami, Aditya Prakash Pillai.

REFERENCE:
https://www.researchgate.net/publication/347511204_WEATHER_PREDICTION_BY_USING_MACHINE_LEARNING

In this paper, the authors are using Random Forest Regression(RFR) regression method since the predicted results are continuous numerical values. In addition, they also did a comparative study with Support Vector Regression(SVR), Multi-layer perceptron, and Extra-Tree Regression(ETR).They are using datasets from wunderground.com(Nashville city).The authors are using root mean squared error (RMSE) to evaluate their models.

9.RESEARCH PAPER: Weather Forecasting Using Machine Learning Algorithm.

PUBLISHED YEAR: 2019

AUTHORS: Nitin Singh, Saurabh Chaturvedi, Shamim Akhter.

REFERENCE: **https://ieeexplore.ieee.org/document/8938211**

In this research paper, the authors have compared the machine learning-based weather forecast model for northwest Bangladesh to increase the accuracy of forecast results in a short period. They have taken the datasets from the Bangladesh Meteorological Department(BMD). The datasets have information about weather details like year, month, day, maximum temperature, minimum temperature, average temperature. The authors are using Root Mean Square Error (RMSE) ,Mean Absolute Error (MAE) as their metrics to test their models. 80% of data is used for training and 20% of data used for the testing from the total dataset.The authors concluded that The Extreme Machine Learning (ELM) model performs better than Artificial Neural Network with an accuracy rate of 95%.

10.RESEARCH PAPER: "Classification and Prediction of Future Weather by using Back Propagation Algorithm-An-Approach".

AUTHORS NAME: Sanjay D. Sawaitul, Prof. K. P. Wagh , Dr. P. N. Chatur.

PUBLISHED YEAR: January 2012.

VOLUME: Volume 2, Issue 1

.

REFERENCE:

http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.414.778.

In this paper, completely different models that were utilized in the past for weather forecasting are mentioned. This paper is specializing in 3 elements, 1st is completely different models which were utilized in meteorology, second half is introducing a replacement wireless kit use for meteorology and third half includes the rear Propagation formula will be applied on completely different parameters of forecast. This paper concludes that the new technology of wireless medium is used for meteorology method. In past, numbers of models were used for weather forecasting supported Artificial Neural Network, Soft Computing or data processing concept. Their system will increase the responsibility, accuracy and consistency of identification and interpretation of weather pictures. They conjointly conclude that the rear Propagation formula can even be applied on the meteorology information. Neural Networks are capable of modelling a forecast system. The neural network signal processing approach for meteorology is capable of yielding sensible results and may be thought-about as an alternate to ancient meteorological approaches.

3.2. Literature Survey: Comparative study of the papers reviewed

Title, Year, Authors	**Methodology or Techniques used**	**Advantages**	**Issues**	**Metrics used**
1.ANN based Weather Prediction using Back Propagation Technique Year: 2018 Authors: 1. Saboor Ahmad Kakar, 2. Naveed Sheikh, 3. Saleem Iqbal, 4. Abdul	Feed-forward backpropagation network with hyperbolic tangent sigmoid transfer function	1.Satisfactory rainfall prediction any time. 2. Adaptable in real time. 3. Fast computation 4. Lowest error and correlation Coefficient.	Accuracy is less. 2. Requires larger bandwidth. 3. Performance less.	1. Mean Square Error (MSE). 2.Tangent hyperbolic sigmoid transfer function. 3.Correlat -ion coefficient R 2.

Rehman.				
2.Weather Forecasting using Neural Network. Year: 2017 Authors: 1.Priyanka Mahajan, 2.Chhaya Nawale, 3.Siddheshwar Kini, 4.Prof. Krishnanjali Shind	1. Artificial Neural Network and Back Propagation Algorithm. 2. Data mining Technique	1.Highly efficient in predicting Non-linear data. 2. Increased Accuracy.	1. It needs full knowledge of atmospheric Dynamics. 2. Involves calculations with a large number of variables & datasets.	1. Mean Squared Error (MSE).
3.Efficient Weather Forecasting Using Artificial Neural Network As Function Approximator. Year: 2014 Authors: 1.EI-Feghi, 2. Z.Zubia, 3. S.Abozgaya.	1.Back Propagation Network (BPN). 2. Radial Basis Function Networks (RBFN).	1. The network learns very fast with back propagation Algorithm. 2. The results are more accurate for predicting the Future weather.	1. The approximate error in the Output. 2. No linearity	1. Accuracy. 2. Training 3. Testing
4.Daily Weather Forecasting using Artificial Neural Network. Year: 2015 Authors: 1. Meera	1. Ensemble Neural Network. 2.Back propagation Network. 3. Radial Basis Function Network. 4. Genetic	1. Self adaptive. 2. Better approximation. 3. Greater accuracy.	1. Input data should be cleaned. 2.Performance is slower.	1.Maximum temperature. 2.Minimum temperature. 3. Relative Humidy.

Narvekar, 2. Priyanca Fargose, 3. Assistant Professor	Algorithm. 5. Multilayer Perceptron. 6. Fuzzy clustering.			
5.Temporal Weather Prediction using Back Propagation based Genetic Algorithm Technique. Year : 2014 Authors: 1. Shaminder Singh, 2. Jasmeen Gill.	1. Back Propagation Algorithm. 2. Genetic Algorithms. 3. Support Vector Machine (SVM). 4.SelfOrganizing Map (SOM)	1. Genetic Algorithms is good for "noisy" environment 2.SVM is relatively memory efficient.	1.Feed forward architecture need to useone or more than one hidden layers.	1.Time Series Prediction 2.Weather Forecasting.
6.An Efficient Machine Learning Regression Model for Rainfall Prediction Year:2015 Authors: 1. R. Usha Rani 2. T.K.Rama Krishna Rao. 3. R. Kiran Kumar Reddy.	Regression Model: 1. Self Organized Maps (SOM) 2. Support Vector Machine (SVM) with ID3)	hierarchical algorithms is to reduce computational cost for each cluster. Second, it gives rough visualization for each cluster.	1.SOM-equires necessary and sufficient data in order to develop meaningful clusters. 2.SVM does not perform very well when the data set has more noise i.e. target classes are overlapping	1.Accuray rate: 82%
7.WEATHER PREDICTION BY MACHINE LEARNING Year:2020 Authors: 1. Shashank Singh 2. Faraz Ahmed	1. Random Forest Regression (RFR) 2. Ridge Regression (Ridge) **3.** Support Vector	1.It reduces overfitting in decision trees and helps to improve the accuracy. 2.It is flexible to both classification and regression	1.It requires much computational power as well as resources as it builds numerous trees to combine their outputs. 2. It also requires	1.Root mean squared error (RMSE).

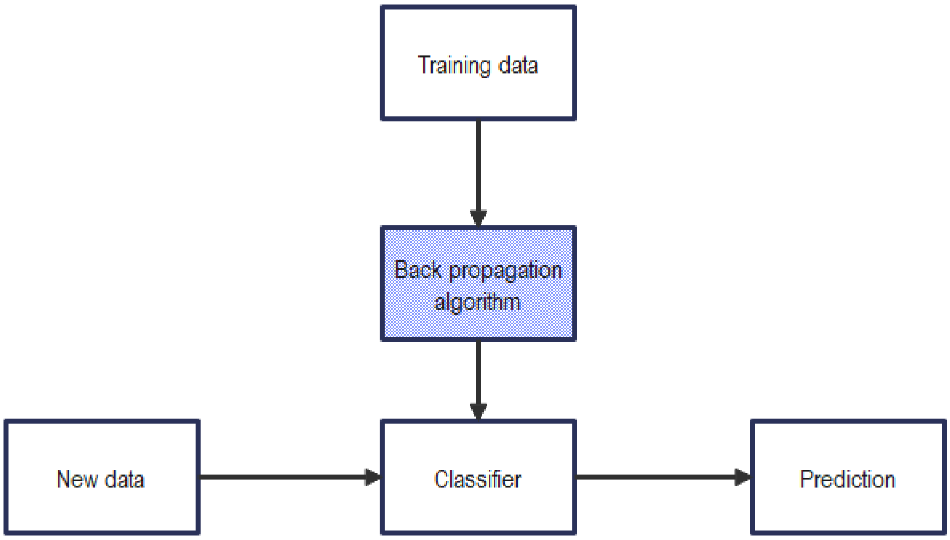
Nagrami 3. Aditya Prakash Pillai.	(SVR) **4.** Multi-layer Perceptron (MLPR), and Extra-Tree Regression (ETR).	problems. 3.It works well with both categorical and continuous values. 4.It automates missing values present in the data. 5.Normalising of data is not required as it uses a rule-based approach.	much time for training as it combines a lot of decision trees to determine the class.	
8.Machine Learning in Python for Weather Forecast based on Freely Available Weather Data. Year: 2018 Authors: 1.E.B.Abrahamsen 2.O. M. Brastein, B. Lii	1.Auto-regressive neural networks	Timeseries modeling :A common method for modeling discrete timeseries data is the use of auto-regressive models with exogenous inputs (ARX).	1.Hardware Dependence	1.Error Rate 2.Accuray.
9.Weather Forecasting Using Machine Learning Algorithm. Year: 2019 Authors: 1. Nitin Singh 2. Saurabh Chaturvedi 3. Shamim Akhter.	1.Random forest classification	1.Lowcost,reliable, and efficient weather forecasting application using the machine learning concept in python on Raspberry pi board. 2.Advantage of RFC is Missing values are substituted by the variable appearing the most in a particular node. Among all the	1. Complexity 2. Longer training period.	Accuracy 87.90%. confusion matrix.

		available classification methods, random forests provide the highest accuracy.		
10.Classification and Prediction of Future Weather by using Back Propagation AlgorithmAn Approach. Year: 2012 Authors: Sanjay D. Sawaitul, 2. Prof. K. P. Wagh, 3. Dr.CHATUR	1. Back Propagation Algorithm. 2. Wireless kit and wireless weather forecasting device.	1.More consistent. 2. Great accuracy. 3. Precise Calculation.	1. Less range sensors are used. 2.Transmission range is less. 3. Prediction of large areas at a time is not possible.	1.Quantization.

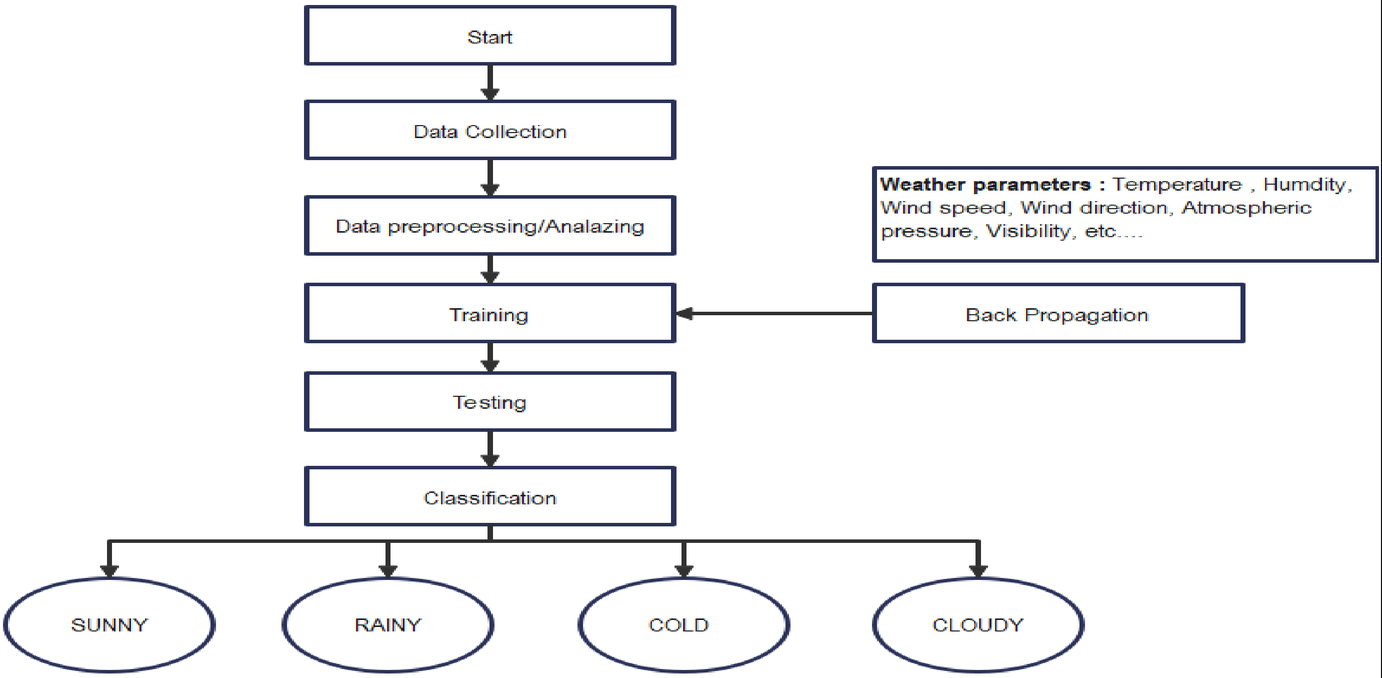
4.System Design:

4.1Detailed Architecture:

First we need to train the model using dataset obtained from the Kaggle website using Back propagation soft computing technique. While training we will be splitting dataset into two parts. One part for training the model and another part is for testing the model. After generating the weather prediction model we can give input parameters obtained from the sensors like max humidit,min humidity, max temperature, min temperature, etc . weather prediction model will classify the weather based the input values like sunny or foggy or cold or thunderstorm.



4.2 Detailed Description of Modules:



MODULES INVOLVED:

Components of a modern weather forecasting system include the following modules:

- Data Collection
- Data Pre-processing
- Training
- Testing
- Results/Output

Data Collection:

In this step numerous sensors are used to collect the data like wind sensors, rain sensor, pressure sensor and temperature sensors. The data is collected repeatedly when short interval of time so we’ve enough input for the process. Large dataset helps in increasing the accuracy of the output. After this the data is send for pre-processing.

Data Pre-processing:

The Pre-processing step is employed to remove the unwanted data or noise recorded by the sensors during transmission or it may refer to the selection of a specific area for consideration for prediction purpose.

Training:

Training of the network is done with the help of back propagation algorithm. During this we’ve to choose learning rate and therefore the momentum value. Learning rate control the speed of the network. As we have a tendency to increase the learning rate it’ll speed up the training. We are using tan h activation function as a result of it provides more recognition and accuracy. We can use any number of hidden layers or can increase it if the network is not learning well. The input of the back propagation network is temperature, rainfall, humidity, pressure and precipitation.

Testing:

Testing is completed by providing with numerous different dataset as input and obtaining the specified result. When we don't get correct results than the network is once more trained. Network is checked with completely different situations which will occur in future.

Output:

When the testing step is complete the output result is provided. On the basis of this result we classify that how will be the future weather. In Classification technique, it'll show what's going to be the future weather, whether it will be sunny day, rainy or cloudy day what's going to be the change in wind speed, humidity etc. The Classification Technique can facilitate for taking some prevention from the climatical hazards.

5.Software Requirement Specifications:

S/w and H/w Requirements

Software Requirements:

- Programming languages used: python
- Libraries used: Keras, numpy,matplotlib
- Code editor:Visual studio
- Operating System: windows

Hardware Requirements:

- Processor i5 (for efficient usage)
- 4 GB RAM
- 512 GB storage
- Must Support virtual box

6.Experimental Results & Discussion

6.1 Source Code:

Train.py:

```
from keras.models import Sequential
from keras.layers import Dense, Activation
from keras.utils import np_utils
import numpy as np



np.random.seed(7)
# Load dataset
data = np.loadtxt("data.txt")

# Splitting Dataset
X_train = data[:4646,:12]
Y_train = data[:4646,12:13]
X_test = data[4646:6936,:12]
Y_test = data[4646:6936,12:13]
len_y_train=len(Y_train)
len_y_test=len(Y_test)
```

```
# Data preprocessing
for i in range(0,len_y_train):
    if(Y_train[i]==1000):
        Y_train[i] = 3
        Y_train[i] = int(Y_train[i])
    elif(Y_train[i]==100):
        Y_train[i] = 2
        Y_train[i] = int(Y_train[i])
    elif(Y_train[i]==10):
        Y_train[i] = 1
        Y_train[i] = int(Y_train[i])
    elif(Y_train[i]==1):
        Y_train[i] = 0
        Y_train[i] = int(Y_train[i])

for i in range(0,len_y_test):
    if(Y_test[i]==1000):
        Y_test[i] = 3
        Y_test[i] = int(Y_test[i])
    elif(Y_test[i]==100):
        Y_test[i] = 2
        Y_test[i] = int(Y_test[i])
    elif(Y_test[i]==10):
        Y_test[i] = 1
        Y_test[i] = int(Y_test[i])
    elif(Y_test[i]==1):
        Y_test[i] = 0
        Y_test[i] = int(Y_test[i])



Y_train = Y_train.astype('int32')
Y_train = np_utils.to_categorical(Y_train,4)
Y_test = Y_test.astype('int32')
Y_test = np_utils.to_categorical(Y_test,4)
```

```
# Defining Network
model = Sequential()
model.add(Dense(100, input_dim=12, kernel_initializer='uniform', activation='relu'))
model.add(Dense(80, kernel_initializer='uniform', activation='relu'))
model.add(Dense(60, kernel_initializer='uniform', activation='relu'))
model.add(Dense(60, kernel_initializer='uniform', activation='relu'))
model.add(Dense(4))
model.add(Activation('softmax'))
model.summary()
model.compile(loss='binary_crossentropy', optimizer='adam', metrics=['accuracy'])
model.fit(X_train, Y_train, epochs=50, batch_size=10, verbose=2,
validation_data=(X_test,Y_test))
scores = model.evaluate(X_test, Y_test, verbose=0)

# Printing Accuracy
print("\n")
print("%s: %.2f%%" % (model.metrics_names[1], scores[1]*100))

# Saving Weights
json_string = model.to_json()
open('model_architecture.json', 'w').write(json_string)
model.save_weights('weights.h5',overwrite=True)



# This is backpropagation network running with 50 epochs
# Accuracy increases with increase in no. of epochs
```

Predict.py:

```
from keras.models import model_from_json
import numpy as np
import matplotlib.pyplot as plt

# Loading Prediciton data
```

```
pre_data = np.loadtxt('predict.txt')
model = model_from_json(open('model_architecture.json').read())
model.load_weights('weights.h5')
model.compile(loss='binary_crossentropy', optimizer='adam', metrics=['accuracy'])
pre_data = pre_data.astype('int32')
pre_data = pre_data.reshape(1,12)


print('\n \t \t \tInputs')
print('\n',pre_data)

ans = model.predict(pre_data)
np.set_printoptions(suppress=True)

# Data Representation
print('\n \tSUNNY \t RAINY \t CLOUDY \tCOLD')
print('\n',ans)
yy=["ThunderStorm","Rainy","Foggy","Sunny"]
xx=[i*100 for i in ans[0]]

# Visualization
fig, ax = plt.subplots()
ax.barh(yy,xx)
ax.set(xlim=[0, 100], xlabel='Probability', ylabel='',title='Weather Prediction')
plt.show()
```

Predict.txt:

```
1 20 7 9 4 93 35 1020 1016 3 0 11
```

COMPARISON WITH OTHER MODEL:

model 2 Naive bayes.py

```
# importing required libraries
import numpy as np
from sklearn.naive_bayes import GaussianNB
from sklearn.metrics import accuracy_score
from keras.utils import np_utils

# Loading dataset
data = np.loadtxt("data.txt")

# Splitting dataset
X_train = data[:4646,:12]
Y_train = data[:4646,12:13]
X_test = data[4646:6936,:12]
Y_test = data[4646:6936,12:13]
len_y_train = len(Y_train)
len_y_test = len(Y_test)

print('Shape of training data :',X_train.shape)
print('Shape of testing data :',Y_test.shape)



# Data Preprocessing
for i in range(0,len_y_train):
    if(Y_train[i]==1000):
        Y_train[i] = 3
        Y_train[i] = int(Y_train[i])
    elif(Y_train[i]==100):
        Y_train[i] = 2
        Y_train[i] = int(Y_train[i])
```

```
    elif(Y_train[i]==10):
        Y_train[i] = 1
        Y_train[i] = int(Y_train[i])
    elif(Y_train[i]==1):
        Y_train[i] = 0
        Y_train[i] = int(Y_train[i])

for i in range(0,len_y_test):
    if(Y_test[i]==1000):
        Y_test[i] = 3
        Y_test[i] = int(Y_test[i])
    elif(Y_test[i]==100):
        Y_test[i] = 2
        Y_test[i] = int(Y_test[i])
    elif(Y_test[i]==10):
        Y_test[i] = 1
        Y_test[i] = int(Y_test[i])
    elif(Y_test[i]==1):
        Y_test[i] = 0
        Y_test[i] = int(Y_test[i])

def res(x):
    if(x==0):
        return 'ThunderStorm'
    elif(x==1):
        return 'Rainy'
    elif(x==2):
        return 'Foggy'
    else:
        return 'Sunny'

model = GaussianNB()
```

```
# fit the model with the training data
model.fit(X_train,Y_train.ravel())

# predict the target on the train dataset
predict_train = model.predict(X_train)
print('Target on train data',predict_train)

# Accuray Score on train dataset
accuracy_train = accuracy_score(Y_train,predict_train)
print('accuracy_score on train dataset : ', accuracy_train*100)

# predict the target on the test dataset
predict_test = model.predict(X_test)

# print('Target on test data',[predict_test for predict_test in predict_test])

# Accuracy Score on test dataset
accuracy_test = accuracy_score(Y_test,predict_test)
print('accuracy_score on test dataset : ', accuracy_test*100)

# Load Prediction Data
pre_data = np.loadtxt('predict.txt').reshape(1, -1)
pre_data=pre_data.astype('int32')




# Print Input & Prediction
print('Input: ',pre_data)
print ("\n \t \t \t Weather would be",res(model.predict(pre_data)))



# This is Naive bayes network
```

model_architecture.json:

{"class_name": "Sequential", "config": {"name": "sequential", "layers": [{"class_name": "InputLayer", "config": {"batch_input_shape": [null, 12], "dtype": "float32", "sparse": false, "ragged": false, "name": "dense_input"}}, {"class_name": "Dense", "config": {"name": "dense", "trainable": true, "batch_input_shape": [null, 12], "dtype": "float32", "units": 100, "activation": "relu", "use_bias": true, "kernel_initializer": {"class_name": "RandomUniform", "config": {"minval": -0.05, "maxval": 0.05, "seed": null}}, "bias_initializer": {"class_name": "Zeros", "config": {}}, "kernel_regularizer": null, "bias_regularizer": null, "activity_regularizer": null, "kernel_constraint": null, "bias_constraint": null}}, {"class_name": "Dense", "config": {"name": "dense_1", "trainable": true, "dtype": "float32", "units": 80, "activation": "relu", "use_bias": true, "kernel_initializer": {"class_name": "RandomUniform", "config": {"minval": -0.05, "maxval": 0.05, "seed": null}}, "bias_initializer": {"class_name": "Zeros", "config": {}}, "kernel_regularizer": null, "bias_regularizer": null, "activity_regularizer": null, "kernel_constraint": null, "bias_constraint": null}}, {"class_name": "Dense", "config": {"name": "dense_2", "trainable": true, "dtype": "float32", "units": 60, "activation": "relu", "use_bias": true, "kernel_initializer": {"class_name": "RandomUniform", "config": {"minval": -0.05, "maxval": 0.05, "seed": null}}, "bias_initializer": {"class_name": "Zeros", "config": {}}, "kernel_regularizer": null, "bias_regularizer": null, "activity_regularizer": null, "kernel_constraint": null, "bias_constraint": null}}, {"class_name": "Dense", "config": {"name": "dense_3", "trainable": true, "dtype": "float32", "units": 60, "activation": "relu", "use_bias": true, "kernel_initializer": {"class_name": "RandomUniform", "config": {"minval": -0.05, "maxval": 0.05, "seed": null}}, "bias_initializer": {"class_name": "Zeros", "config": {}}, "kernel_regularizer": null, "bias_regularizer": null, "activity_regularizer": null, "kernel_constraint": null, "bias_constraint": null}}, {"class_name": "Dense", "config": {"name": "dense_4", "trainable": true, "dtype": "float32", "units": 4, "activation": "linear", "use_bias": true, "kernel_initializer": {"class_name": "GlorotUniform", "config": {"seed": null}}, "bias_initializer": {"class_name": "Zeros", "config": {}}, "kernel_regularizer": null, "bias_regularizer": null, "activity_regularizer": null, "kernel_constraint": null, "bias_constraint": null}}, {"class_name": "Activation", "config": {"name": "activation", "trainable": true, "dtype": "float32", "activation": "softmax"}}]}, "keras_version": "2.5.0", "backend": "tensorflow"}

6.2 Screenshots with Explanation

Dataset targeted output Explanation:

Labels mean as per the following......

0 = Thunder Storm 0001

1 = Rainy 0010

2 = Foggy 0100

3 = Sunny 1000

Train.py:

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
dense (Dense)                (None, 100)               1300
_________________________________________________________________
dense_1 (Dense)              (None, 80)                8080
_________________________________________________________________
dense_2 (Dense)              (None, 60)                4860
_________________________________________________________________
dense_3 (Dense)              (None, 60)                3660
_________________________________________________________________
dense_4 (Dense)              (None, 4)                 244
_________________________________________________________________
activation (Activation)      (None, 4)                 0
=================================================================
Total params: 18,144
Trainable params: 18,144
Non-trainable params: 0
```

Total number of parameters is trained.

```
Epoch 1/10
465/465 - 15s - loss: 0.4009 - accuracy: 0.6535 - val_loss: 0.3757 - val_accuracy: 0.6920
Epoch 2/10
465/465 - 0s - loss: 0.3238 - accuracy: 0.7277 - val_loss: 0.3341 - val_accuracy: 0.6981
Epoch 3/10
465/465 - 0s - loss: 0.3087 - accuracy: 0.7305 - val_loss: 0.3426 - val_accuracy: 0.6863
Epoch 4/10
465/465 - 0s - loss: 0.3027 - accuracy: 0.7413 - val_loss: 0.3586 - val_accuracy: 0.6706
Epoch 5/10
465/465 - 0s - loss: 0.2961 - accuracy: 0.7462 - val_loss: 0.3449 - val_accuracy: 0.7003
Epoch 6/10
465/465 - 0s - loss: 0.2935 - accuracy: 0.7462 - val_loss: 0.3207 - val_accuracy: 0.7187
Epoch 7/10
465/465 - 0s - loss: 0.2900 - accuracy: 0.7471 - val_loss: 0.3576 - val_accuracy: 0.6780
Epoch 8/10
465/465 - 0s - loss: 0.2852 - accuracy: 0.7503 - val_loss: 0.3116 - val_accuracy: 0.7204
Epoch 9/10
465/465 - 0s - loss: 0.2832 - accuracy: 0.7471 - val_loss: 0.3590 - val_accuracy: 0.6806
Epoch 10/10
465/465 - 0s - loss: 0.2845 - accuracy: 0.7495 - val_loss: 0.3254 - val_accuracy: 0.6994


accuracy: 69.94%
```

We have to load the dataset and train with number of epochs.

Epoch table for Training phase:

- For each epochs the accuracy differs accordingly.
- If we increase the number of epochs the accuracy also increases along with it.

EPOCHS	ACCURACY
10	69.94%
20	72.87%
50	76.12%
100	80.76%
200	91.21%

Predict.txt:

1 28 13 10 4 77 23 1016 1011 4 1 14

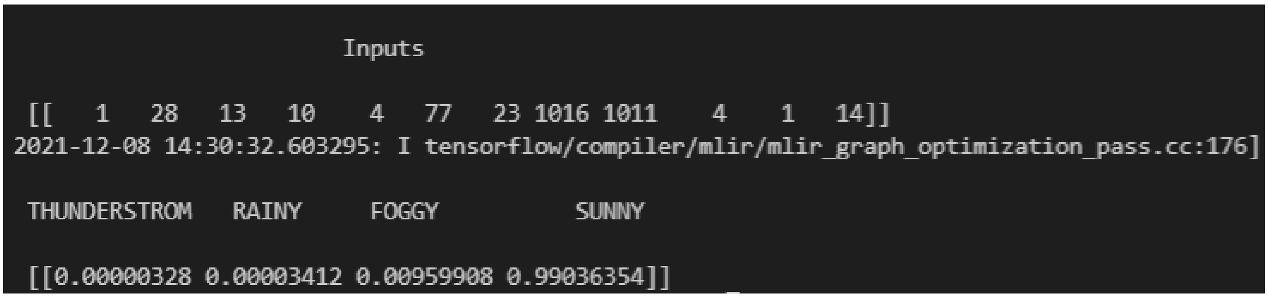
We have to give certain parameters in the text.

Attributes considered are Max Temperature, Min Temperature, Max Dewpoint, Min Dewpoint, Max Humidity, Min Humidity, Max Pressure, Min Pressure, Max Visibility, Min Visibility and Mean Wind Speed.

Prediction is performed using neural nets with backpropagation.

Library: Keras.

Predict.py:



```
                         Inputs

 [[   1   28   13   10    4   77   23 1016 1011    4    1   14]]
2021-12-08 14:30:32.603295: I tensorflow/compiler/mlir/mlir_graph_optimization_pass.cc:176]

 THUNDERSTROM   RAINY     FOGGY          SUNNY

 [[0.00000328 0.00003412 0.00959908 0.99036354]]
```

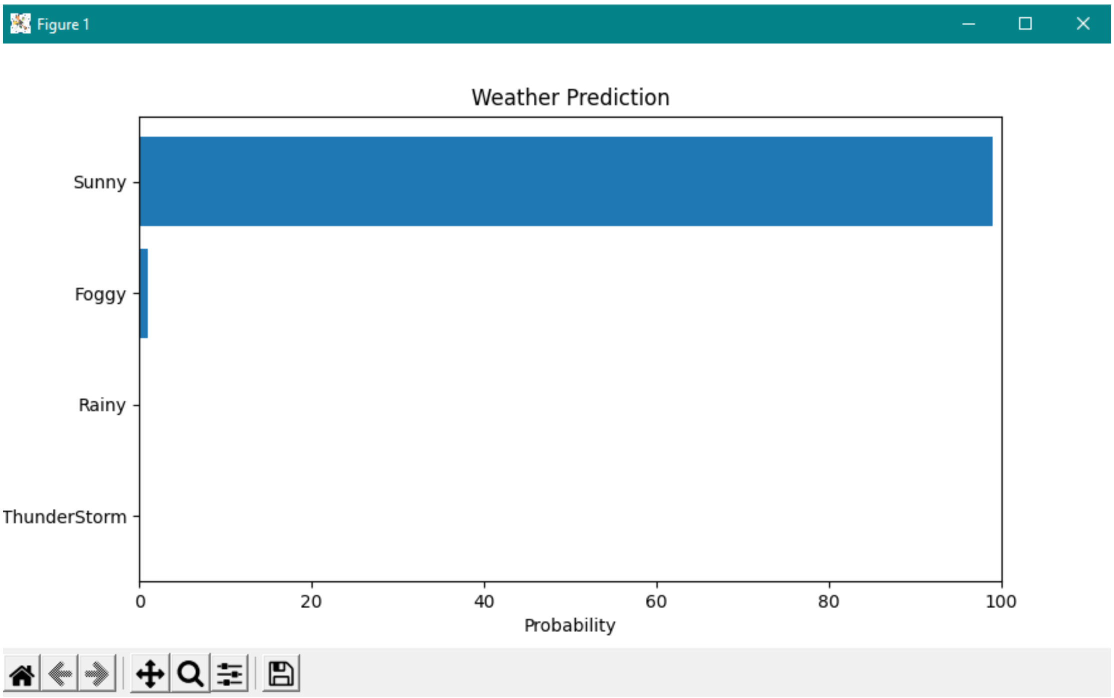
The input file predict.txt is passed through this predict.py where the result is obtained for the given parameters.

The accuracy for sunny is high and predicted the weather would be sunny for the given parameters.

```
THUNDERSTROM    RAINY      FOGGY          SUNNY

[[0.00000328 0.00003412 0.00959908 0.99036354]]
```

OUTPUT:



For the given parameters the output is shown Foggy which is the expected output.

Comparison Model:

Model 2 naïve bayes:

```
Shape of training data : (4646, 12)
Shape of testing data : (2289, 1)
Target on train data [2. 2. 2. ... 3. 0. 3.]
accuracy_score on train dataset :  65.06672406371072
accuracy_score on test dataset :  61.9921363040629
Input:  [[   1   28   13   10    4   77   23 1016 1011    4    1   14]]

                         Weather would be Foggy
```

Same Input is passed through this Model 2 Naïve bayes (comparison model) where we can see the accuracy is less when compared with sequential model.

REFERENCES:

1. https://www.researchgate.net/publication/327400959_Artificial_Neural_Network_based_Weather_Prediction_using_Back_Propagation_Technique

2. https://www.ijert.org/research/weather-forecasting-using-neural-network-IJERTCONV5IS01197.pdf

3. https://www.sciencedirect.com/science/article/pii/S221201731200326X

4. https://www.researchgate.net/publication/281953396_Daily_Weather_Forecasting_using_Artificial_Neural_Network

5. http://www.mecs-press.org/ijisa/ijisa-v6-n12/IJISA-V6-N12-8.pdf

6. https://ep.liu.se/ecp/153/024/ecp18153024.pdf

7. https://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.695.3985&rep=rep1&type=pdf

8. https://www.researchgate.net/publication/347511204_WEATHER_PREDICTION_BY_USING_MACHINE_LEARNING

9. https://ieeexplore.ieee.org/document/8938211

10. http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.414.778.

An Approach for Prediction of Weather System by Using Back propagation Neural Network

Y.Md.Riyazuddin, Dr.S.Mahaboob Basha, Dr.K.Krishna Reddy

[1]Research Scholar, [2,3]Professor
[1,2]Department of CSE, [3]Department of Physics
[1]Rayalaseema University, [2]Al Habeeb College of Engineering, [3]Yogivemana University

***Abstract*: This paper utilizes artificial neural networks for temperature forecasting. Our study based on back propagation neural network which is trained and tested based on dataset provided. In formulating the ANN-based predictive model; three-layer network has been constructed. Suitable air temperature predictions can provide farmers and producers with valuable information when they face decisions regarding the use of mitigating technologies such as orchard heaters or irrigation. The research presented in this thesis developed artificial neural networks models for the prediction of air temperature. In this paper, back propagation neural network is used for temperature forecasting. The technical milestones, that have been achieved by the researchers in this field has been reviewed and presented in this paper. From the past decades there are various models are developed for weather forecasting using artificial neural network, and by using soft computing, which are discussed in this paper. Artificial neural networks and the back propagation algorithm used for temperature forecasting in general are explained.**

1. Introduction

Air temperatures prediction is of a concern in environment, industry and agriculture. The climate change phenomenon is as the first environmental problem in the world threatening the human beings. The industrial activities are so effective in this problem and cause the global warming which the world has been faced with, lately. Knowing the variability of ambient temperature is important in agriculture because extreme changes in air temperature may cause damage to plants and animals [4].Air temperature forecasting is useful in knowing the probability of tornado, and flood occurrence in an area. Prediction of the energy consumption, soil surface temperature and solar-radiation are also related to ambient air temperature forecasting.

Artificial Neural Network (ANN), a component of soft computing, is highly suitable for the situations where the underlying processes exhibit chaotic features. The concept of ANN is originated from the attempt to develop a mathematical model capable of recognizing complex patterns on the same line as biological neuron work. It is useful in the situations where underlying processes / relationships display chaotic properties. ANN does not require any prior knowledge of the system under consideration and are well suited to model dynamical systems on a real-time basis. It is, therefore, possible to set up systems so that they would adapt to the events which are observed and for this, it is useful in real time analyses, e.g., weather forecasting, different fields of predictions, etc.

ANN provides a methodology for solving many types of non-linear problems that are difficult to solve by traditional techniques. Most meteorological processes often exhibit temporal and spatial variability, and are further plagued by issues of non-linearity of physical processes, conflicting spatial and temporal scale and uncertainty in parameter estimates. With ANN, there exists the capability to extract the relationship between the inputs and outputs of a process. Thus, these properties of ANN are well suited to the problem of weather forecasting under consideration [2].

One type of network sees the nodes as 'artificial neurons'. These are called artificial neural networks

.An artificial neuron is a computational model inspired in the natural neurons. Natural neurons receive signals through synapses located on the dendrites or membrane of the neuron. When the signals received are strong enough, the neuron is activated and emits a signal though the axon. This signal might be sent to another synapse, and might activate other neurons. The complexity of real neurons is highly abstracted when modeling artificial neurons. These basically consist of inputs, which are multiplied by weights, and then computed by a mathematical function which determines the activation of the neuron [1].Another function computes the output of the artificial neuron. ANN combines artificial neurons in order to process information.

The higher a weight of an artificial neuron is, the stronger the input which is multiplied by it will be. Weights can also be negative, so we can say that the signal is inhibited by the negative weight. Depending on the weights, the computation of the neuron will be different. By adjusting the weights of an artificial neuron we can obtain the output we want for specific inputs. But when we have an ANN of hundreds or thousands of neurons, it would be quite complicated to find by hand all the necessary weights. But we can find algorithms which can adjust the weights of the ANN in order to obtain the desired output from the network. This process of adjusting the weights is called learning or training [Fig. 1].

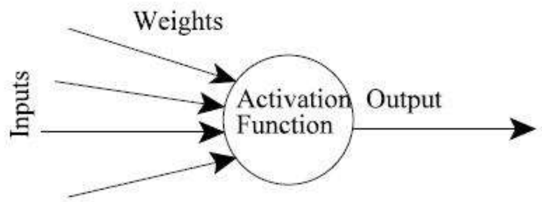


Fig.1 An Artificial Neuron.

2. Materials and Methods

A. Data Sets

The data used in this study are daily and monthly
for air temperature prediction were collected from wireless kit and wunderground.com, with the help of these parameters, forecast temperature using statistica software as a platform. The different sensors like rain sensor, wind sensor, and thermo-hydro sensor records different parameters like rainfall, wind, temperature and humidity. The recorded data is present in the form of datasheet [9]. This data set is send for pre-processing and then to the statistica software. These data were considered in three different data sets including training, test and validation in artificial neural network.

B. Back Propagation Learning

The simple perceptron is just able to handle
linearly separable or linearly independent problems. By taking the partial derivative of the error of the network with respect to each weight, we will learn a little about the direction the error of the network is moving. In fact, if we take the negative of this derivative (i.e. the rate change of the error as the value of the weight increases) and then proceed to add it to the weight, the error will decrease until it reaches local minima. This makes sense because if the derivative is positive, this tells us that the error is increasing when the weight is increasing. The obvious thing to do then is to add a negative value to the weight and vice versa if the derivative is negative. Because the taking of these partial derivatives and then applying them to each of the weights takes place, starting from the output layer to hidden layer weights, then the hidden layer to input layer weights (as it turns out, this is necessary since changing these set of weights requires that we know the partial derivatives calculated in the layer downstream), this algorithm has been called the back propagation algorithm [5]. A neural network can be trained in two different modes: online and batch modes. The number of weight updates of the two methods for the same number of data presentations is very different. The online method weight updates are computed for each input data sample, and the weights are modified after each sample. An alternative solution is to compute the weight update for each input sample, but store these values during one pass through the training set which is called an epoch. At the end of the epoch, all the contributions are added, and only then the weights will be updated with the composite value. This method adapts the weights with a cumulative weight update, so it will follow the gradient more closely. It is called the batch-training mode. Training basically involves feeding training samples as input vectors through a neural network, calculating the error of the output layer, and then adjusting the weights of the network to minimize the error.
The previously mentioned back-propagation learning algorithm works for feed-forward networks with continuous output. Training starts by setting all the weights in the network to small random numbers. Now, for each input example the network gives an output, which starts randomly. We measure the squared difference between this output and the desired output—the correct class or value. The sum of all these numbers over all training examples is called the total error of the network. If this number was zero, the network would be perfect, and the smaller the error, the better the network. By choosing the weights that minimize the total error, one can obtain the neural network that best solves the problem at hand. This is the same as linear regression, where the two parameters characterizing the line are chosen such that the sum of squared differences between the line and the data points is minimal. This can be done analytically in linear regression, but there is no analytical solution in a feed-forward neural network with hidden units. In back-propagation, the weights and thresholds are changed each time an example is presented, such that the error gradually becomes smaller. This is repeated, often hundreds of times, until the error no longer changes. In back-propagation, a numerical optimization technique called gradient descent makes the math particularly simple; the form of the equations gave rise to the name of this method. There are some learning parameters (called learning rate and momentum) that need tuning when using back-propagation, and there are other problems to consider. For instance, gradient descent is not guaranteed to find the global minimum of the error, so the result of the training depends on the initial values of the weights. However, one problem overshadows the others: that of over-fitting [3]. Over-fitting occurs when the network has too many parameters to be learned from the number of examples available, that is, when a few points are fitted with a function with too many free parameters. Although this is true for any method for classification or regression, neural networks seem especially prone to over parameterization. A network that over fits the training data is unlikely to generalize well to inputs that are not in the training data. There are many ways to limit over-fitting (apart from simply making small networks), but the most common include averaging over several networks, regularization and using methods from Bayesian statistics. To estimate the generalization performance of the neural network, one needs to test it on independent data, which have not been used to train the network. This is usually done by cross-validation, where the data set is split into, for example, and ten sets of equal size. The network is then trained on nine sets and tested on the tenth, and this is repeated ten times, so all the sets are used for testing. This gives an estimate of the generalization ability of the network; that is, its ability to classify inputs that it was not trained on. To get an unbiased estimate, it is very important that the individual sets do not contain examples that are very similar.

C. Statistica Software

I. Select the Variables for the Analysis
Statistica data miner distinguishes between categorical and continuous variables and dependent and predictor. Categorical variables are those that contain information about some discrete quantity. Continuous variables are measured on a continuous scale [10]. Dependent variables are those we want to predict. Predictor variables are those that we want to use for prediction or classification.

II. Feature Selection and Variable Screening

The Feature selection and variable screening module will automatically select subsets of variables from extremely large data files or databases connected for in-place processing .The module can handle a practically unlimited number of variables. Literally millions of input variables can be scanned to select predictors for regression or classification. Specifically, the program includes several options for selecting variables that are likely to be useful or useful or informative in specific subsequent analysis [11]. The unique algorithms implemented in the feature selection and variable screening module will select continuous or categorical predictor variables that show a relationship to the continuous or categorical dependent variables of interest, regardless of weather that relationship is simple or complex.

III. Factor Analysis

The Factor Analysis module contains a wide range of statistics and options, and provides a comprehensive implementation of factor analytic techniques with extended diagnostics and a wide variety of analytic and exploratory graphs. It performs principal components and common and hierarchical factor analysis and can handle extremely large analysis problems.

IV. Statistica Neural Network

Statistica Neural Network is a comprehensive, state-of-the-art, powerful and extremely fast neural networks data analysis package that contains the following features: integrated pre and post processing, including data selection, nominal-value encoding, scaling, normalization and missing value substitution with interpretation for classification, regression and time series problems[11]. Statistica Neural Network has numerous facilities to aid in selecting appropriate network architecture. Statistica Neural Network statistical and graphical feedback includes bar charts, matrices and graphs of individual and overall case errors, vital statistics such as regression error ratios, which are all automatically calculated.

3. Training and Testing Neural Network

The best training procedure is to compile a wide range of examples (for more complex problems, more examples are required), which exhibit all the different characteristics of the problem. To create a robust and reliable network, in some cases, some noise or other randomness is added to the training data to get the network familiarized with noise and natural variability in real data [7]. Poor training data inevitably leads to an unreliable and unpredictable network. Usually, the network is trained for a prefixed number of epochs or when the output error decreases below a particular error threshold. Special care is to be taken not to over train the network. By overtraining, the network may become too adapted in learning the samples from the training set, and thus may be unable to accurately classify samples outside of the training set.[Fig. 2]

A. Choosing the Number of Neurons

The number of hidden neurons affects how well the network is able to separate the data. A large number of hidden neurons will ensure correct learning, and the network is able to correctly predict the data it has been trained on, but its performance on new data, its ability to generalize, is compromised [5]. With too few hidden neurons, the network may be unable to learn the relationships amongst the data and the error will fail to fall below an acceptable level. Thus, selection of the number of hidden neurons is a crucial decision.

B. Choosing the Intial Weights

The learning algorithm uses a steepest descent technique, which rolls straight downhill in weight space until the first valley is reached. This makes the choice of initial starting point in the multidimensional weight space critical. However, there are no recommended rules for this selection except trying several different starting weight values to see if the network results are improved.

C. Choosing the Learning Rate

Learning rate effectively controls the size of the step that is taken in multidimensional weight space when each weight is modified. If the selected learning rate is too large, then the local minimum may be overstepped constantly, resulting in oscillations and slow convergence to the lower error state[6]. If the learning rate is too low, the number of iterations required may be too large, resulting in slow performance.

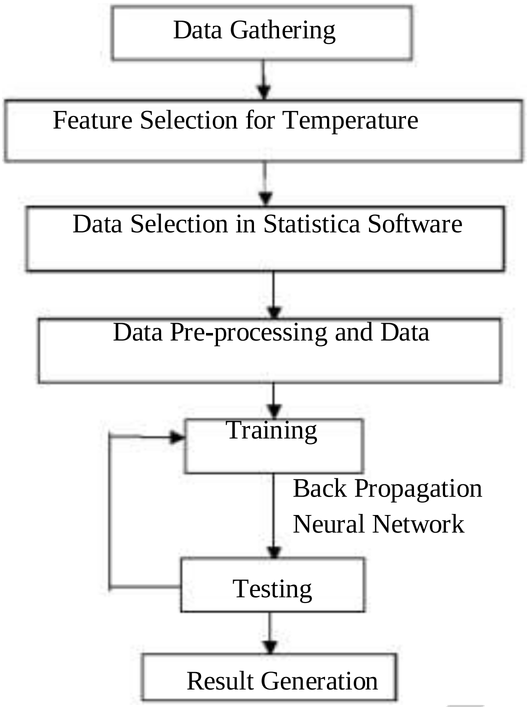


Figure 2: Flowchart of temperature prediction system

4. Result and Discussion

The obtained results indicate that satisfactory prediction accuracy has been achieved through back propagation neural network. A back propagation neural network with gradient descent method minimizes the error rate and it is a promising approach for temperature forecasting.

Mean, minimum and maximum air temperatures were considered as input and output of the network. Three different data sets were extracted from the input and target data for training, validation and test phases. While training set consists of 50 percent of data to build the model and determine the parameters such as weights and biases, validation data set includes 25 percent to measure the performance of network by holding constant parameters. Finally, 25 percent of data is used to increase the robustness of model in the test phase.

The validation and testing phases are very important due to misleading of small error in the training phase. If the network is not trained well due to the irrelevant data of the individual cases such as over fitting, it leads to the small error in the training set and makes large error during validation and test phases. While the purpose of training phase is based on learning, it is not a good metric for the performance of network in validation phase.

5. Conclusion

Neural-networks-based ensemble models were developed and applied for hourly temperature forecasting. The experimental results show that the ensemble networks can be trained effectively without excessively compromising the performance. The ensembles can achieve good learning performance because one of the ensemble's members is able to learn from the correct learning pattern even though the patterns are statistically mixed with erroneous learning patterns.

References

[1]S. Mathur" A feature based neural network model for weather forecasting" World academy of science, engineering and technology 34, 2007.
[2]R.A. Pielke, "A comprehensive meterological modeling system RAMS," Meteorology and Atmosphe-ric Physics, Springer-Verlag Vol. 49, 69-91p,1992.
[3]B.A. Smith " Improving air temperature prediction with artificial neural networks" International journal of computational intelligence, 2007.
[4]J. Gill "Training back propagation neural networks with genetic algorithm for weather forecasting" IEEE 8th international symposium on intelligent systems and informatics serbia, 2010.
[5]Senduru Srinivasulu, "Extracting Spatial Semantics in Association Rules for Weather Forecasting Image",
Research Scholar Department of Information Technology, Sathyabama University Chennai, India IEEE 2010
[6]S.S. Baboo and I.K Shereef. " An efficient weather forecasting system using artificial neural network"
International journal of environmental science and development, vol. 1, no. 4, 2010.
[7]Y.Wang and S.Banavar "Convective Weather Forecast Accuracy Analysis at center and sector levels", NASA Ames Research center, Maffett Field, Califonia.

[8]I.S. Isa “Weather forecasting using photovoltaic system and neural network” Second international conference on computational intelligence, communication systems and networks, 2010.
[9]Weather.com,http://www.weather.com
[10]AccuWeather.com,http://www.accuweather.com [11]Statistica SANN-v9_training_manual.