VIT

Vellore Institute of Technology

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

School of Computer Science and Engineering

J Component report

Programme : B.Tech (CSE)

Course Title : IOT Domain Analyst

Course Code : ECE-3502

Slot : A1

Title: Forest Fire Detection using Fuzzy Logic

Team Members: Mihir Antwal (19BCE1641)

Sam Methuselah (19BCE1698)

Faculty: Prof. Shola Usha Rani

Sign: S.U. [S.usharani],

Date: 26/04/22

DECLARATION

I hereby declare that the report titled "**Forest Fire Detection and Prediction using Fuzzy Logic"** submitted by me to VIT Chennai is a record of bona-fide work undertaken by us under the supervision of **: Prof. Shola Usha Rani**, School of Computer Science and Engineering, Vellore Institute of Technology, Chennai.

Signature of the Candidate

Sam Methuselah

Reg. No. 19BCE1698

Signature of the Candidate

Mihir Antwal

Reg. No. 19BCE1641

CERTIFICATE

Certified that this project report entitled "**Forest Fire Detection and Prediction using Fuzzy Logic"** is a bonafide work of **Mihir Antwal (19BCE1641) and Sam Methuselah (19BCE1698)** and they carried out the Project work under my supervisionand guidance for ECE3502 – IoT Domain Analyst.

Shola Usha Rani

SCOPE, VIT Chennai

ACKNOWLEDGEMENT

We wish to express our sincere thanks and deep sense of gratitude to our project guide, **Prof. Shola Usha Rani** for her consistent encouragement and valuable guidance offered to us in a pleasant manner throughout the course of the project work.

We are extremely grateful to **Dr. Ganesan R.**, Dean of School of Computer Science Engineering, VIT Chennai, for extending the facilities of the School towards our projectand for his unstinting support.

We express our thanks to our Head of the Department **D**r. **Nithyanandam P** for his support throughout the course of this project.

We also take this opportunity to thank all the faculty of the School for their support andtheir wisdom imparted to us throughout the course.

We thank our parents, family, and friends for bearing with us throughout the course of our project and for the opportunity they provided us in undergoing this course in such aprestigious institution.

Signature of the Candidate
Sam Methuselah
Reg. No. 19BCE1698

Signature of the Candidate
Mihir Antwal
Reg. No. 19BCE1641

ABSTRACT

Forest is considered as one of the most important and indispensable resource. However, forest fire, affected by some human uncontrolled behavior in social activities and abnormal natural factors, occurs occasionally. These are prevalent forest dangers that inflict significant harm to wildlife and the environment. It is critical to recognize these types of flames as soon as possible in order to minimize the risk of harm to the biological system.

According to the Global Warming Report 2008, one of the real causes of an increase in global temperature is fast spreading fire. More than 4000 hectares of forestland were burned in Uttarakhand's slopes in late 2016. Lightning, extremely hot and dry weather, and human negligence are all common causes of wildfire. It may be avoided if a reliable technology could be established in forest areas to detect fires and warn firefighting authorities, who could then take rapid action. One of the strategies for early wildfire detection is presented in this study.

This project's motive is to predict the occurrence of forest fire by continuous monitoring of factors that are responsible for forest fire like temperature, smoke and flame in forests. We use latest technologies like IoT for collecting data frequently through the sensors. All these nodes containing sensors are deployed at various locations of the forest publish the data to ThingSpeak that is easily accessible. Further the nodes can be classified as- no forest fire, mild, moderate and severe based on the threshold values and fuzzy logic is applied on this data to predict the chance of occurrence of forest fire and is expressed in percentage. In case of severe chance, a priority alert notification will be send to the connected devices through IFTTT platform with the help of PushSafer. This will help the authorities to take necessary measures to prevent it and to avoid the loss of human life.

Keywords: IoT; MQTT broker; Fuzzy logic; Forest fire; Temperature; Smoke; Flame; Sensors

CONTENTS

1. INTRODUCTION

1.1 OBJECTIVE AND GOAL OF THE PROJECT

Several million acres of forest are destroyed every year due to forest fire. Forest fire not only destroys many valuable trees but also destroys the vegetation in that area. The fire will burn the trees and also the soil is burnt and so many acres of land become water repellent. Forest fire is one of the major causes of global warming as tones of greenhouse gases are emitted into the atmosphere. Nowadays the prediction mechanisms used are watching through towers, satellite imaging, long distance video recording, etc. But these do not provide quicker response which is most important in forest fire prediction. The causes of forest fires are many: natural i.e. rise in temperature, change in wind direction, moisture level etc. and human related, both intentional and unintentional like the burning by grazers and gatherers of forest produce, shifting cultivation, fires to ward off wild animals or by visitors to forests by way of smoking etc. The cost of such disaster may be millions of trees, in addition to losses of structures, animals (wild and farm), and human life. Forest fires lead to global warming, soil erosion, ozone layer depletion and the loss of livelihood of those dependent on forest products. The only way of protecting forest from wild fires is their early prediction. This project predicts the occurrence of forest fire by making use of IoT where temperature, smoke and flame sensors deployed in the forest classify it as no, mild, moderate and severe forest fire using the sensor values and the percentage of chance of occurrence of forest fire is calculated using fuzzy logic and necessary measures can be taken based on the predicted output

1.2 INTERNET OF THINGS

1.2.1 What is IoT?

The internet of things, or IoT, is a system of interrelated computing devices, mechanical and digital machines, objects, animals or people that are provided with unique identifiers (UIDs) and the ability to transfer data over a network without requiring human-to-human or human-to-computer interaction. An IoT ecosystem consists of web-enabled smart devices that use embedded processors, sensors and communication hardware to collect, send and act on data they acquire from their environments. IoT devices share the sensor data they collect by connecting to an IoT gateway or other edge device where data is either sent to the cloud to be analyzed or analyzed locally. Sometimes, these devices communicate with other related devices and act on the information they get from one another. The devices do most of the work without human intervention, although people can interact with the devices -- for instance, to set them up, give them instructions or access the data. The connectivity, networking and communication protocols used with these web-enabled devices largely depend on the specific IoT applications deployed.

1.2.2 APPLICATIONS OF IOT

Consumer Applications

A growing portion of IoT devices are created for consumer use, including connected vehicles, home automation, wearable technology (as part of Internet of Wearable Things (IoWT), connected health, and appliances with remote monitoring capabilities.

Commercial applications

Medicine and health care: IoT devices can be used to enable remote health monitoring and emergency notification systems. These health monitoring devices can range from blood pressure and heart rate monitors to advanced devices capable of monitoring specialized implants, such as pacemakers, Fit-bit electronic wristbands, or advanced hearing aids.

Transportation: Application of the IoT extends to all aspects of transportation systems (i.e. the vehicle, the infrastructure, and the driver or user). Dynamic interaction between these components of a transport system enables inter and intra vehicular communication, smart traffic control, smart parking, electronic toll collection systems, logistic and fleet management, vehicle control, safety and road assistance.

Industrial applications

Manufacturing: The IoT intelligent systems enable rapid manufacturing of new products, dynamic response to product demands, and real-time optimization of manufacturing production and supply chain networks, by networking machinery, sensors and control systems together.

Agriculture: There are numerous IoT applications in farming such as collecting data on temperature, rainfall, humidity, wind speed, pest infestation, and soil content. This data can be used to automate farming techniques, take informed decisions to improve quality and quantity, minimize risk and waste, and reduce effort required to manage crops.

1.2.3 ROLE OF IOT IN ENVIRONMENTAL MONITORING AND DISASTER MANAGEMENT

Environmental monitoring is a broad application for the Internet of Things. It involves everything from monitoring levels of ozone in a meat packing facility to monitoring national forests for smoke. Using IoT environment sensors for these various applications can take an otherwise highly labor-intensive process and make it simple and efficient. Some of the use cases are:

1. Monitoring air for quality, carbon dioxide and smog-like gasses, carbon monoxide in confined areas, and indoor ozone levels.
2. Monitoring water for quality, pollutants, thermal contaminants, chemical leakages, the presence of lead, and flood water levels.
3. Monitoring soil for moisture and vibration levels in order to detect and prevent landslides.
4. Monitoring forests and protected land for forest fires.
5. Monitoring for natural disasters like earthquake and tsunami warnings.
6. Monitoring fisheries for both animal health and poaching.
7. Monitoring snowfall levels at ski resorts and in national forests for weather tracking and avalanche prevention.
8. Monitoring data centers for air temperature and humidity.

2. LITERATURE SURVEY

2.1 FOREST FIRE PREDICTION AND ALERT SYSTEM USING WIRELESS SENSOR NETWORK

In this method the sensor module is deployed in the forest manually or through a helicopter. The sensor module consists of multiple sensors like temperature sensor, humidity sensor, etc. They collect the target environment in formation and continuously transfer it to the control center where the necessary process is carried out.

It uses Zigbee which is a specification for communication in a wireless personal area network (WPAN). Zigbee is based on an IEEE 802.15 standard. It consumes low power with transmission distance of 10 to 100 meters line of sight. It can transmit data over long distance through intermediate devices such as by forming mesh network. Zigbee has a defined rate of 250 Kbit/s, best suited for intermittent data transmissions from a sensor or input device. It is simple and much less expensive than other WPANs such as Bluetooth and Wi-Fi.

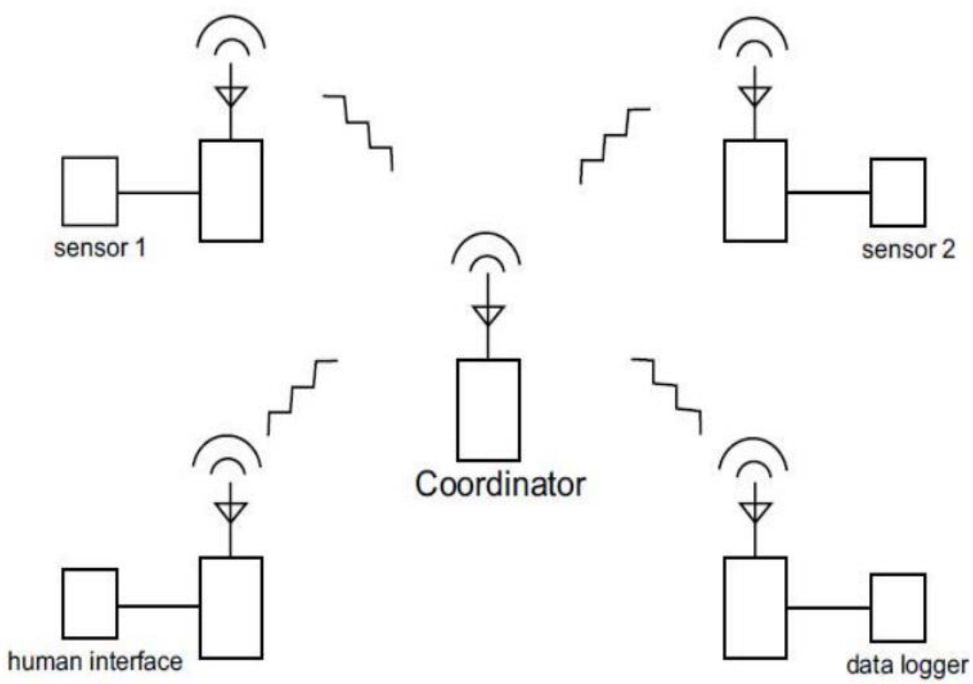


Figure 1- Zigbee based sensor network

Sensor nodes are less costly and even if it gets damaged in fire it won't be a great loss. WSN has the property of self-configuration and hence need not be organized manually. Using GPS the exact location of the fire can be easily obtained and the nearest fire service can be easily informed using GSM. When the temperature in a particular node gets increased over affixed threshold value then the alert is sent to the control center. The threshold value is always fixed above the maximum temperature which is experienced in that particular region to avoid any false alarm due to the increase in the atmospheric temperature. As soon as the fire is detected in a particular node the alert is sent to the control center and also to the neighboring nodes. Once the nearer nodes get the alert, timer is started and it is run till the nearer node detects the fire. This is to find the rate of spread of the fire in the forest. When the rate of spread is known then the necessary action can be taken quickly. All the nodes are equally spaced in order to easily find the rate of spread of fire. The rate of spread directly depends on the speed of air blowing and also the fire usually spreads upwards in a hilly area. These are taken into considerations while designing the detection system.

$$\text{Rate of spread of fire} = \frac{\text{Distance between two nodes}}{\text{Time interval between reception of alert and fire detection}}$$

2.2 FOREST FIRE DETECTION THROUGH VARIOUS MACHINE LEARNING TECHNIQUES USING MOBILE AGENT INWSN

This paper suggests that regression works best for forest fire detection with high accuracy by dividing the forest fire dataset. It also presents comparison of various machine learning techniques like as SVM (support vector machines), neural network, decision tree, regression, so on for detection of forest fires and new approach perform better as compared to other machine learning techniques.

Support Vector Machine (SVM)

In machine learning support vector machine is supervised learning model with associated learning algorithms that analyze data and pattern matching. Support Vector Machine (SVM) models are a close cousin to classical alternative training method for polynomial, radial basis function and multi-layer perception classifiers in which the weights of the network are found by solving a quadratic programming problem with linear constraints, rather than by solving a non-convex, unconstrained minimization problem as in standard neural network training.

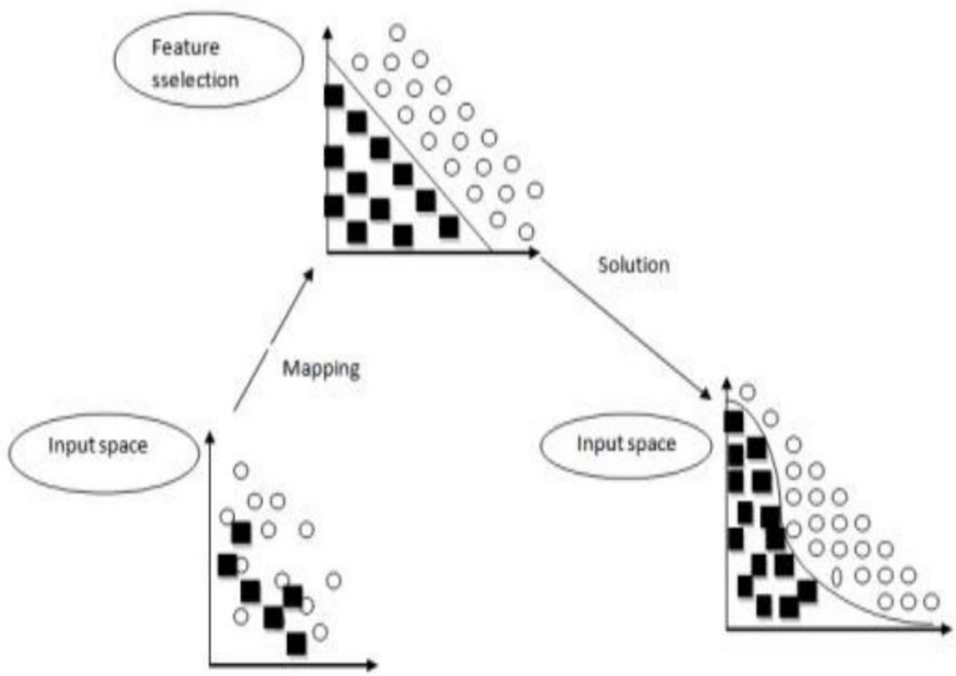


Figure 2- SVM algorithm

Artificial Neural Networks (ANN)

ANN is a system that has the ability of learning by itself and learning is an adaptive process. These basically work on principle of neuron. The first model of neuron contained two inputs and one output. Both the inputs should be active for correct output. The weights for both the inputs were equal and output was binary.

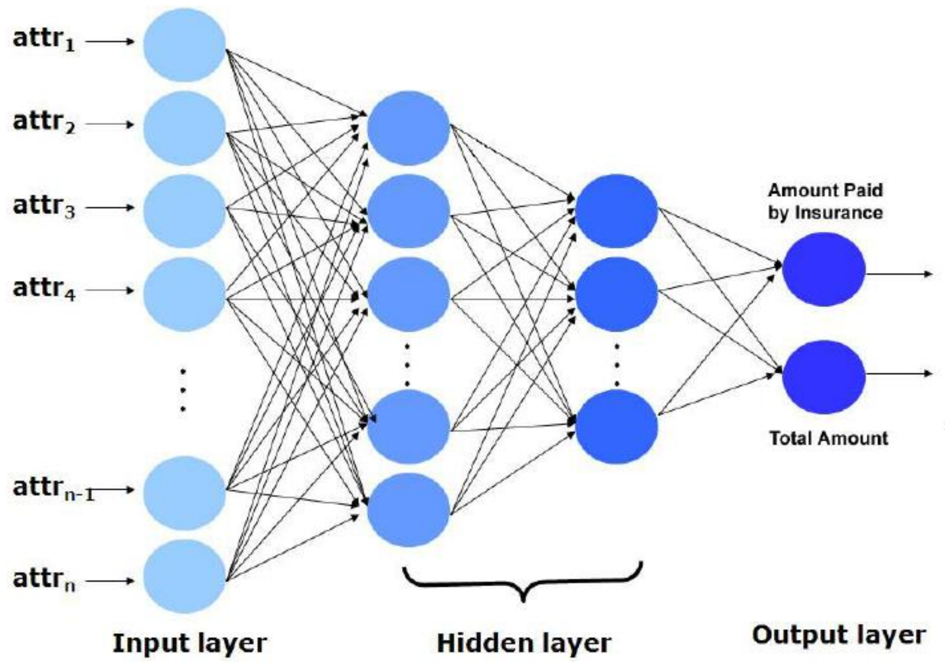


Figure 3- ANN model

Decision tree

Decision tree is a form of multiple variable analyses. They allow predicting, explaining, and classifying an outcome. Decision tree classify instances by sorting them down the tree from the root to some leaf node. The final result is a decision tree in which each branch represents a possible scenario of decision and its outcome.

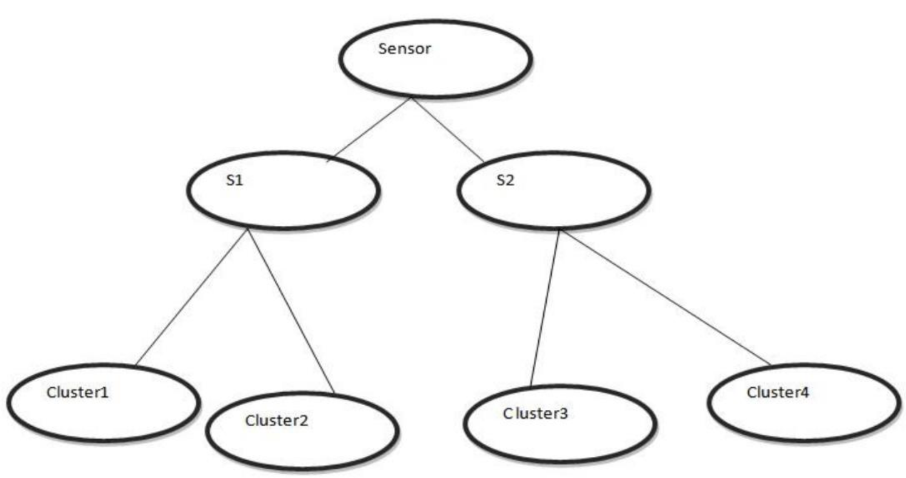


Figure 4- Decision tree

Feed Forward Neural Networks (FFNN)

Feed-forward networks can be seen as cascaded squashed linear functions. The inputs feed into a layer of hidden units, which can feed into layers of more hidden units, which eventually feed into the output layer. Each of the hidden units is a squashed linear function of its inputs. Neural networks of this type can have as inputs any real numbers, and they have a real number as output. For regression, it is typical for the output units to be a linear function of their inputs. For classification it is typical for the output to be a sigmoid function of its inputs (because there is no point in predicting a value outside of [0,1]). For the hidden layers, there is no point in having their output be a linear function of their inputs because a linear function of a linear function is a linear function; adding the extra layers gives no added functionality. The output of each hidden unit is thus a squashed linear function of its inputs.

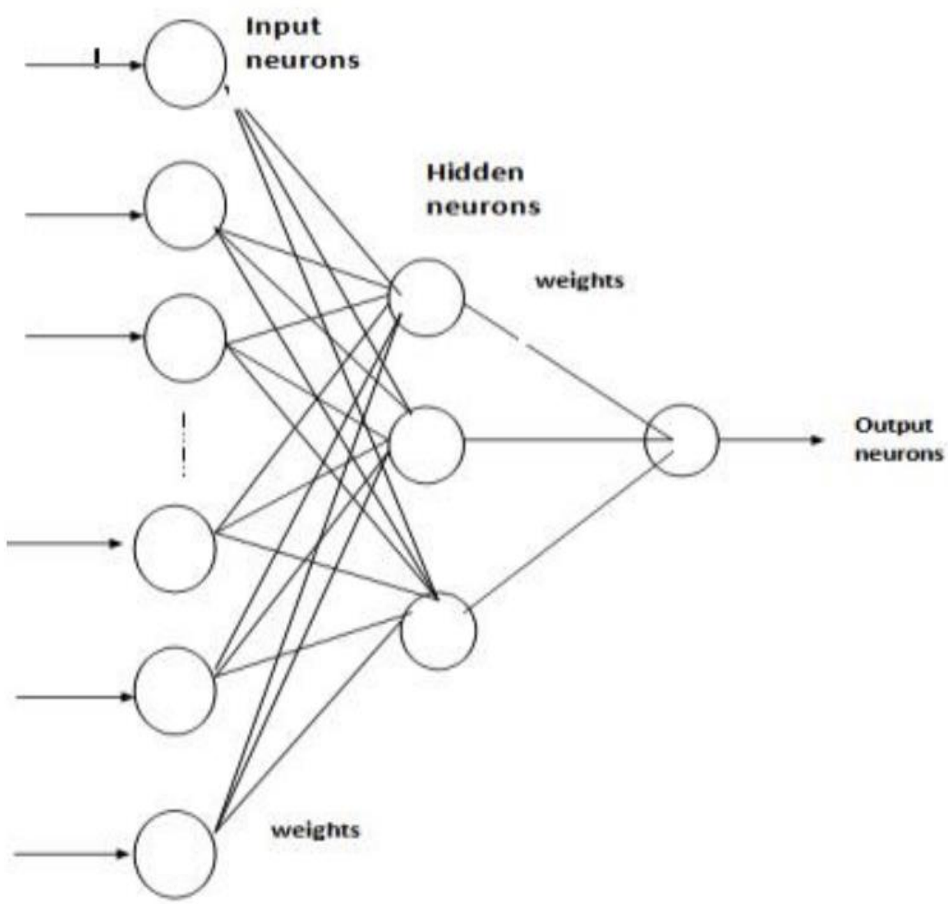


Figure 5- Architecture of FFNN

2.3 PREDICTION OF FOREST FIRE USING SVM– ANOVEL APPROACH

In this approach, video cameras are used for monitoring the forest fire. Camera data are sufficient only for the detection of forest fire, but for the prediction some more information is needed. For this purpose, sensor nodes are deployed in the forest area. Sensor nodes are capable of measuring the parameters like temperature and humidity. These parameters are important for the prediction of natural forest fire. From there both the sensor reading and the camera data are sent to base station (BS). But for the classification purpose data only before and during the forest fire are taken. SVM belong to the class of supervised learning algorithms in which the learning machine is given a set of examples (or inputs) with the associated labels (or output values). SVMs construct a hyper plane that separates two classes (this can be extended to multi-class problems). While doing so, SVM algorithm tries to achieve maximum separation between the classes.

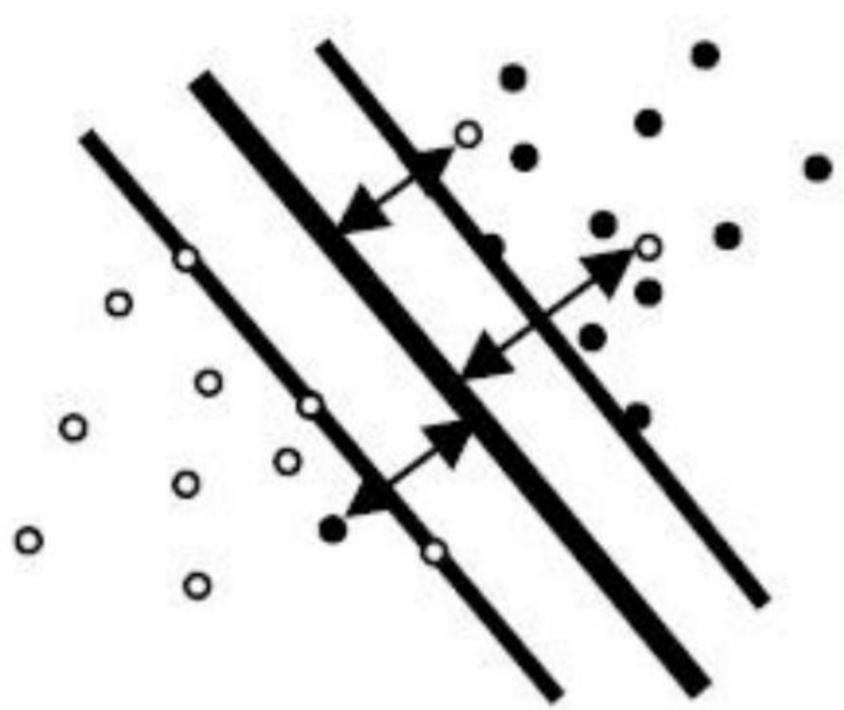


Figure 6- SVM linear classification

In support vector classifiers we have a maximally separating hyper plane of the form $wTx - \gamma = 0$ and two bounding hyper planes of the form $wTx - \gamma = 1$ and

$w^T x - \gamma = -1$. The data points belonging to +1 class satisfy the constraint $w^T x - \gamma \geq 1$ and the data point belonging to -1 class satisfy the constraint $w^T x - \gamma \leq -1$. However, in our case some of the data points will be deviated from their respective bounding plane, such deviation of data points from their respective bounding planes are called as error. A positive quantity called ξ is added or subtracted to the data points that constitutes to error to satisfy the constraints. The new constraints are $w^T x - \gamma + \xi \geq 1$ and $w^T x - \gamma - \xi \leq -1$. Our aim is to try for maximum margin between the bounding planes and minimize the number of data points contributing to error. Maximum margin can be achieved by minimizing the quantity $1/2 w^T w$ which is the reciprocal of the distance between the two bounding hyper planes from the origin. Feature values i.e. Humidity and Temperature values prevailing inside the forest will be sent to the base station from the observation spot periodically. These feature values are given to the learned SVM classifier to predict the class labels as 'YES 'corresponding to a 'chance for fire occurrence' and 'NO' corresponding to 'no chance of fire occurrence'.

3 PROJECT DESCRIPTION

3.1 SYSTEM FLOW

The sensors used for measuring parameters such as temperature, smoke and flame are deployed in the forest as nodes and monitors the environmental conditions continuously. Then each node is classified into -no, mild, moderate and severe forest fire categories and the percentage of chance of occurrence of forest fire is calculated using fuzzy logic. The sensor data and the classification output is then published to a ThingSpeak Channel. An applet in IFTTT named PushSafer helps in transmitting a priority alert notification to the connected devices directly triggered by NodeMCU itself which helps the person using the app to take the precautionary measures. A buzzer is also fixed to the nodes deployed in the forest to alert if there is forest fire so that people or animals approaching towards that location will be made known of the danger.

3.2 PREDICTION USING FUZZY LOGIC

3.2.1 BASIC CONCEPTS

3.2.1.1 What is fuzzy logic?

The term fuzzy refers to things which are not clear or are vague. In the real world many times we encounter a situation when we can't determine whether the state is true or false, their fuzzy logic provides a very valuable flexibility for reasoning. In this way, we can consider the inaccuracies and uncertainties of any situation.

In Boolean system truth value, 1.0 represents absolute truth value and0.0represents absolute false value. But in the fuzzy system, there is no logic for absolute truth and absolute false value. But in fuzzy logic, there is intermediate value too present which is partially true and partially false.

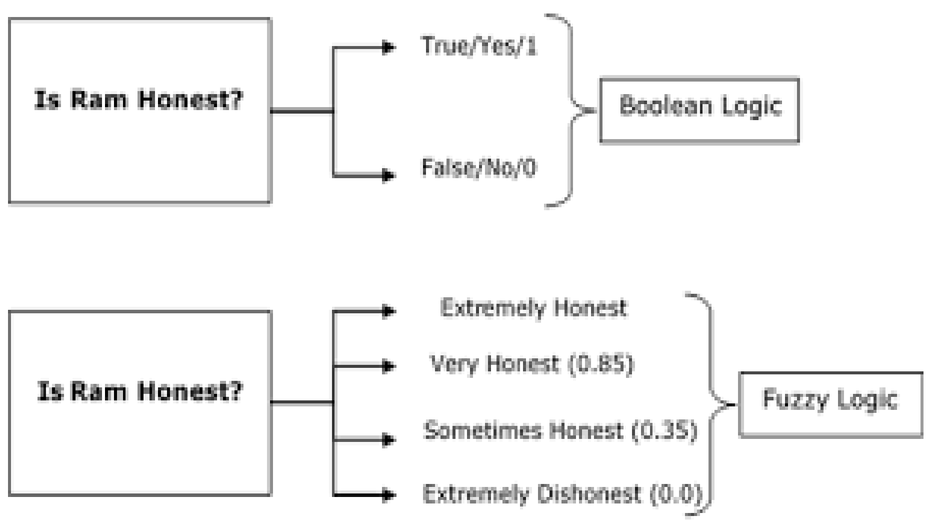


Figure 7- Fuzzy logic example

3.2.1.2 Fuzzy logic architecture

Its Architecture contains four parts:

Rule base

It contains the set of rules and the IF-THEN conditions provided by the experts to govern the decision-making system, on the basis of linguistic information. Recent developments in fuzzy theory offer several effective methods for the design and tuning of fuzzy controllers. Most of these developments reduce the number of fuzzy rules.

Fuzzification

It is used to convert inputs i.e. crisp numbers into fuzzy sets. Crisp inputs are basically the exact inputs measured by sensors and passed into the control system for processing, such as temperature, pressure, rpm's, etc.

Inference engine

It determines the matching degree of the current fuzzy input with respect to each rule and decides which rules are to be fired according to the input field. Next, the fired rules are combined to form the control actions.

Defuzzification

It is used to convert the fuzzy sets obtained by inference engine into a crisp value. There are several defuzzification methods available and the best suited one is used with a specific expert system to reduce the error.

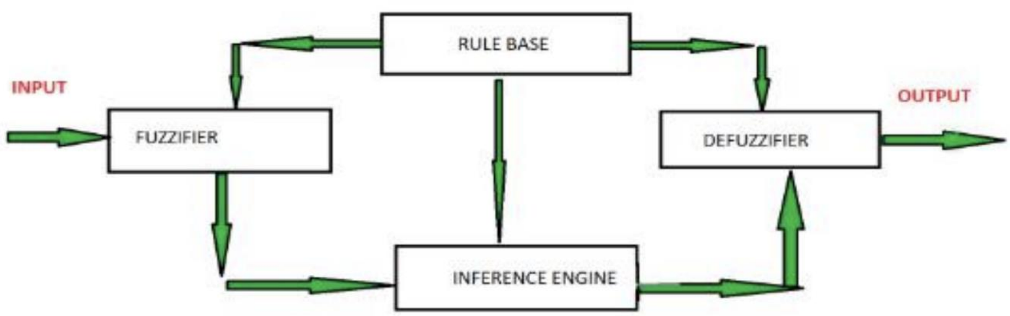


Figure 8- Fuzzy logic architecture

3.2.1.3 Membership function

A graph that defines how each point in the input space is mapped to membership value between 0 and 1. Input space is often referred as the universe of discourse or universal set (u), which contain all the possible elements of concern in each particular application.

There are largely three types of fusiliers:

- singleton fuzzifier,
- Gaussian fuzzifier, and
- trapezoidal or triangular fuzzifier

3.2.1.4 Advantages of Fuzzy Logic System

- This system can work with any type of inputs whether it is imprecise, distorted or noisy input information.

- The construction of Fuzzy Logic Systems is easy and understandable.
- Fuzzy logic comes with mathematical concepts of set theory and the reasoning of that is quite simple.
- It provides a very efficient solution to complex problems in all fields of life as it resembles human reasoning and decision making.
- The algorithms can be described with little data, so little memory is required.

3.2.2 FUZZY CONTROL SYSTEMS

3.2.2.1 What is Fuzzy Control?

- It is a technique to embody human-like thinking’s into a control system.
- It may not be designed to give accurate reasoning but it is designed to give acceptable reasoning.
- It can emulate human deductive thinking, that is, the process people use to infer conclusions from what they know.
- Any uncertainties can be easily dealt with the help of fuzzy logic.

3.2.2.2 The Tipping problem example

The ‘tipping problem’ is commonly used to illustrate the power of fuzzy logic principles to generate complex behavior from a compact, intuitive set of expert rules. Let’s create a fuzzy control system which models how you might choose to tip at a restaurant. When tipping, you consider the service and food quality, rated between 0 and 10. You use this to leave a tip of between 0 and 25%.

We would formulate this problem as:

Antecedents (Inputs)

Service

- Universe (ie, crisp value range): How good was the service of the wait staff, on a scale of 0 to 10?
- Fuzzy set (ie, fuzzy value range): poor, acceptable, amazing

Food quality

- Universe: How tasty was the food, on a scale of 0 to 10?
- Fuzzy set: bad, decent, great

Consequents (Outputs)

- Universe: How much should we tip, on a scale of 0%to 25%
- Fuzzy set: low, medium, high

Rules

- IF the service was good or the food quality was good, THEN the tip will be high.
- IF the service was average, THEN the tip will be medium.
- IF the service was poor and the food quality was poor THEN the tip will be low.

Usage

If we tell this controller that we rated:

- the service as 9.8, and
- the quality as 6.5,

it would recommend we leave:

- a 20.2% tip.

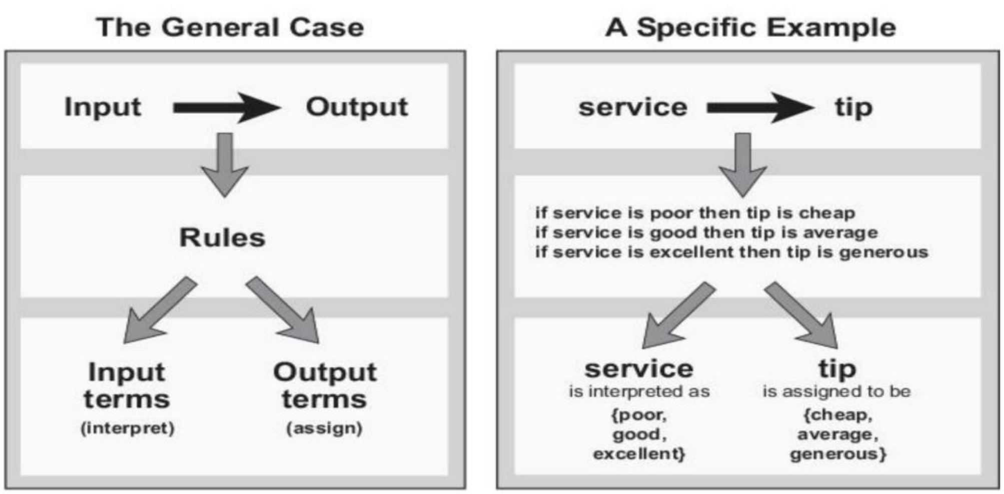


Figure 9- Fuzzy control system example

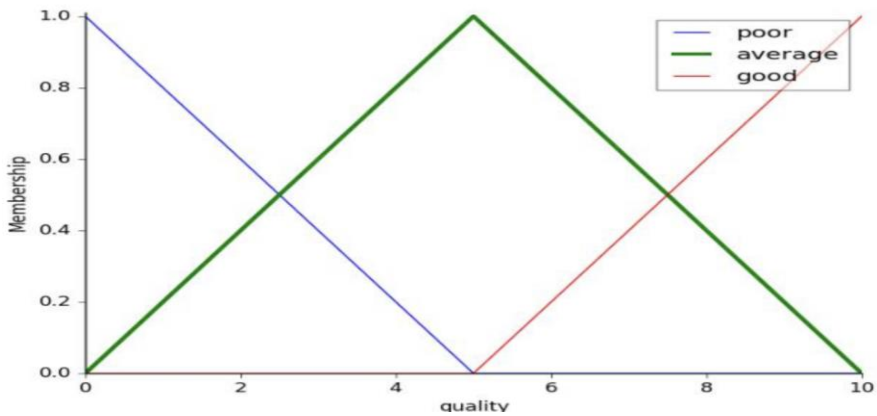


Figure 10- Membership function for quality

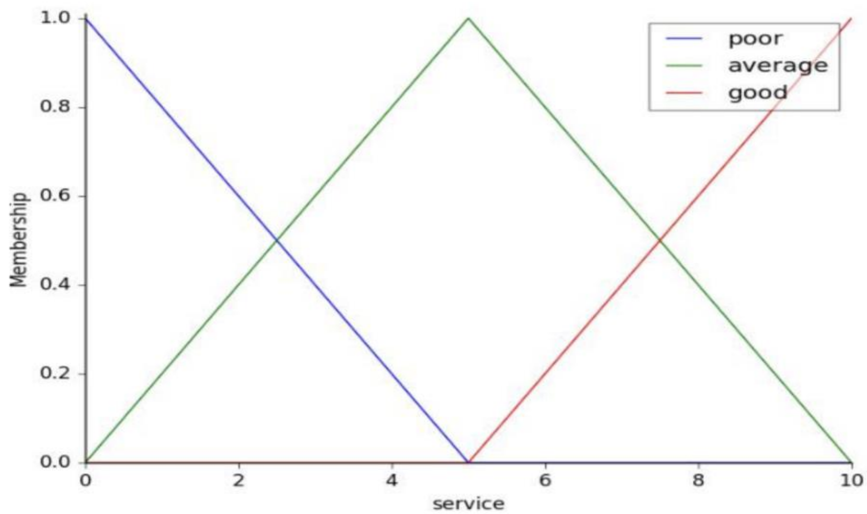


Figure 11- Membership function for service

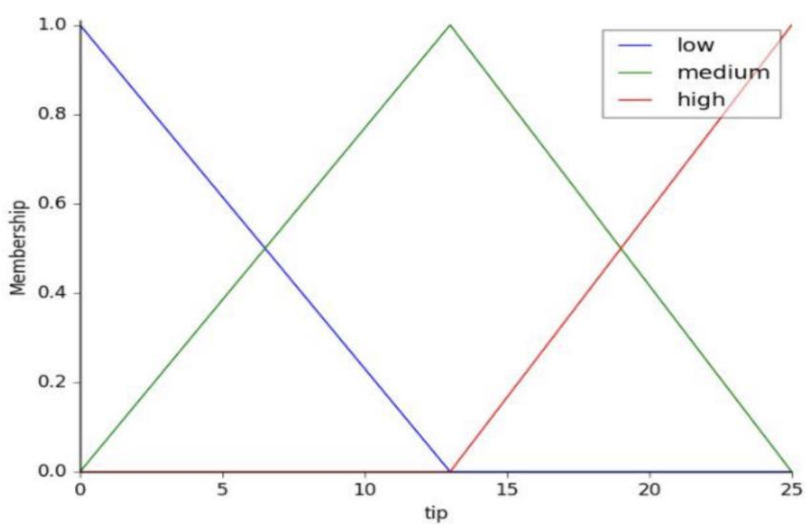


Figure 12- Membership function for tip 20

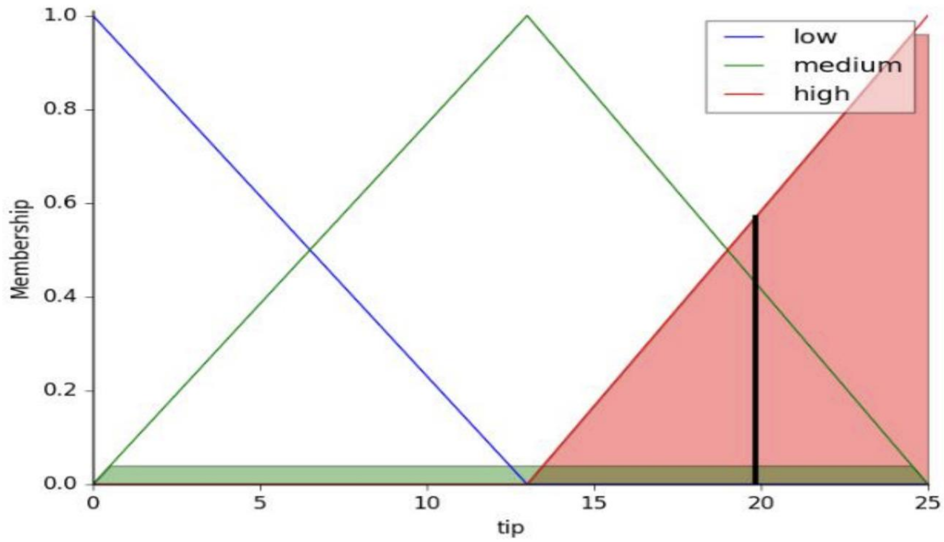


Figure 13- Recommended tip

The resulting suggested tip is **20.24%.**

4 REQUIREMENTS SPECIFICATION

4.1 HARDWARE REQUIREMENTS

Node MCU- NodeMCU is an open source IoT platform. It includes firmware which runs on the ESP8266 Wi-Fi SoC from Espress if Systems, and hardware which is based on the ESP-12 module. Specifications:

- **Type:** Single-board microcontroller
- **Operating system:** XTOS
- **CPU:** ESP8266
- **Memory:** 128k bytes
- **Storage:** 4Mbytes
- **Power:** USB

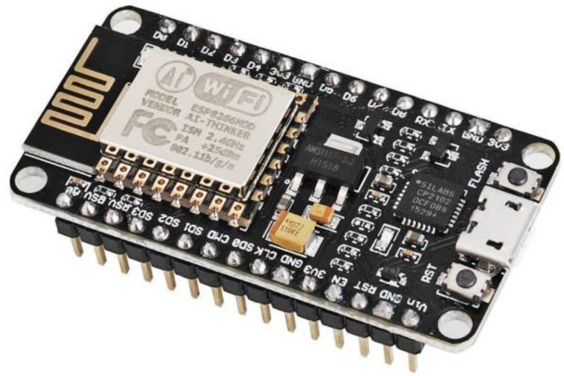


Figure 14- NodeMCU

DHT11- Temperature & Humidity Sensor features a temperature & humidity sensor complex with a calibrated digital signal output. By using the exclusive digital-signal-acquisition technique and temperature & humidity sensing technology, it ensures high reliability and excellent long-term stability. This sensor includes a resistive-type humidity measurement component and an NTC temperature measurement component, and connects to a high performance 8-bit microcontroller, offering excellent quality, fast response, anti-interference ability and cost-effectiveness.

Specifications:

- **Operating Voltage:** 3.5V to 5.5V
- **Operating current:** 0.3mA (measuring) 60uA (standby)
- **Output:** Serial data
- **Temperature Range:** 0°C to 50°C
- **Humidity Range:** 20% to 90%
- **Resolution:** Temperature and Humidity both are 16-bit
- **Accuracy:** ±1°C and ±1%

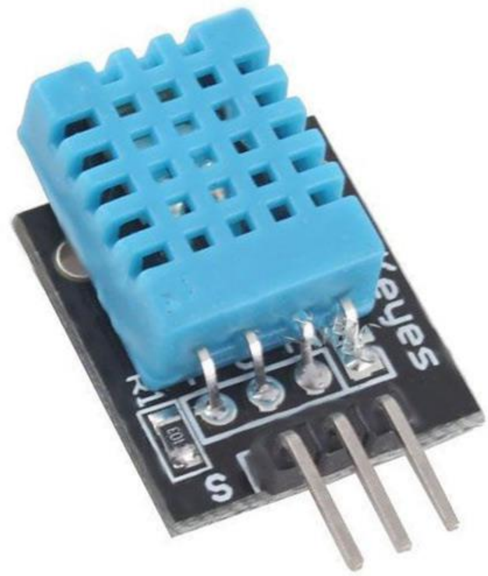


Figure 15- DHT11 Sensor

Smoke Sensor(MQ2)- The MQ-2 is a flammable gas and smoke sensor detects the concentrations of combustible gas in the air and outputs its reading as an analog voltage. The sensor can measure concentrations of flammable gas of 300 to 10,000 ppm

Features:

- Operating Voltage is +5V
- Can be used to Measure or detect LPG, Alcohol, Propane, Hydrogen, CO and even methane
- Analog output voltage: 0V to 5V
- Digital Output Voltage: 0V or 5V (TTL Logic)
- Preheat duration 20 seconds
- Can be used as a Digital or analog sensor
- The Sensitivity of Digital pin can be varied using the potentiometer

Figure 16- Smoke Sensor (MQ2)

Flame sensor- A flame sensor module consists of a flame sensor (IR receiver), resistor, capacitor, potentiometer, and comparator LM393 in an integrated circuit. Features: λ Can detect infrared light with a wavelength ranging from700nmto1000nm. λ The far-infrared flame probe converts the light detected in the form of infrared light into current changes. 28 λ Sensitivity is adjusted through the onboard variable resistor with a detection angle of 60 degrees. λ Working voltage is between 3.3v and 5.2v DC, with a digital output to indicate the presence of a signal. λ Sensing is conditioned by an LM393 comparator.

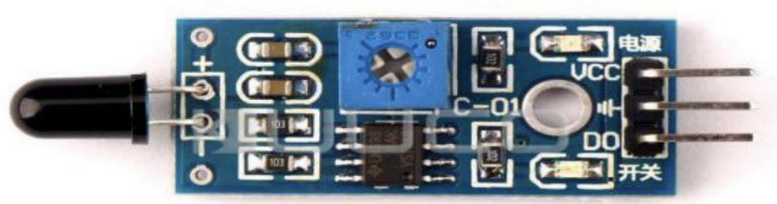


Figure 17- Flame Sensor

Buzzer (Active)- An active buzzer will generate a tone using an internal oscillator, so all that is needed is a DC voltage. Specifications:

- Module using S8550 transistor drive
- Operating voltage 3.3V-5V
- With a fixed bolt hole, easy to install
- Small board PCB size: 3.2cm * 1.3cm approx
- When the I/O port input low, the buzzer Sound

Figure 18- Buzzer (Active)

4.2 SOFTWARE REQUIREMENTS

4.2.1 Arduino IDE

The Arduino Integrated Development Environment - or Arduino Software(IDE) - contains a text editor for writing code, a message area, a text console, a toolbar with buttons for common functions and a series of menus. It connects to the Arduino and Genuino hardware to upload programs and communicate with them. Programs written using Arduino Software (IDE) are called sketches. These sketches are written in the text editor and are saved with the file extension .ino. The editor has features for cutting/pasting and for searching/replacing text. The message area gives feedback while saving and exporting and also displays errors. The console displays text output by the Arduino Software (IDE), including complete error messages and other information. The bottom right-hand corner of the window displays the configured board and serial port. The toolbar buttons allows you to verify and upload programs, create, open, and save sketches, and open the serial monitor.

5 SYSTEM DESIGN

5.1 Setting up Arduino IDE

- Download Arduino IDE. λ Open your IDE and click on "File -> Preferences".
- In "Additional Boards Manager URLs" add this line and click on"OK": http://arduino.esp8266.com/stable/package_esp8266com_index.json
- Go to "Tools -> Board -> Boards Manager", type "ESP8266"andinstall it.
- Go again to "Tools -> Board" and select "Generic ESP8266 Module".
- Install necessary libraries like Adafruit MQTT Library, ArduinoUno WiFi Dev Ed Library, PubSubClient etc.

5.2 CIRCUIT DIAGRAM

The temperature and humidity sensor (DHT11), the smoke sensor (MQ-2) and the flame sensor are connected to the node MCU. Correct mapping has to be done between NodeMCU and ESP8266 pins. The numbers of the pins in the board don’t map to the numbers of the pins on the ESP8266. So, for example, pin D1 of the board doesn’t map to GPIO1 of the ESP8266 (it actually maps to GPIO5).

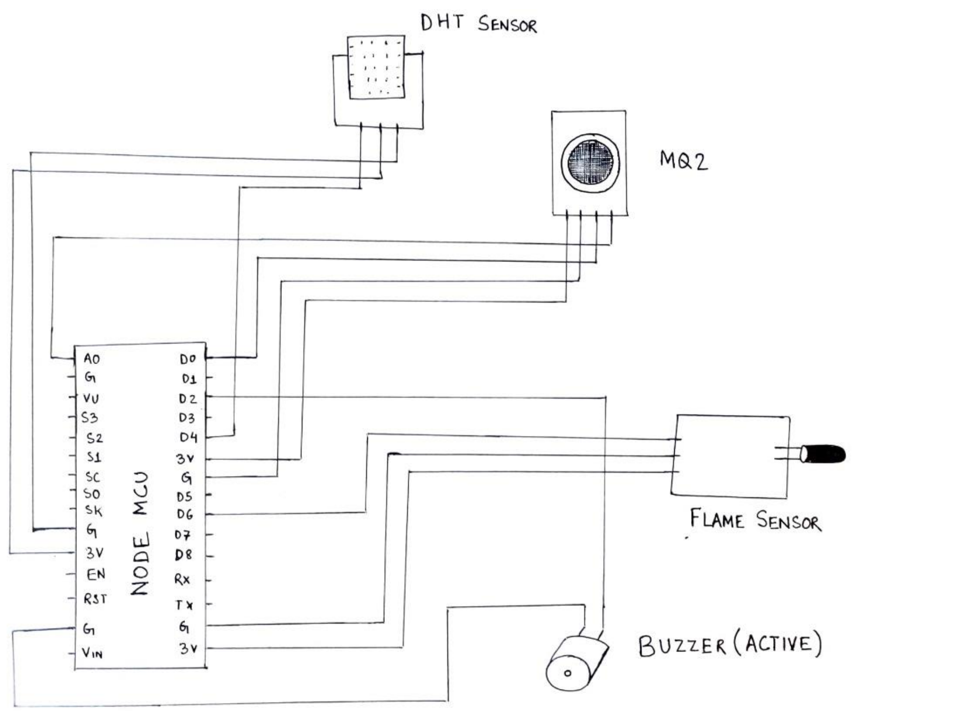


Figure 19- Circuit Diagram

5.3 SYSTEM FLOW DIAGRAM FOR FOREST FIRE DETECTION

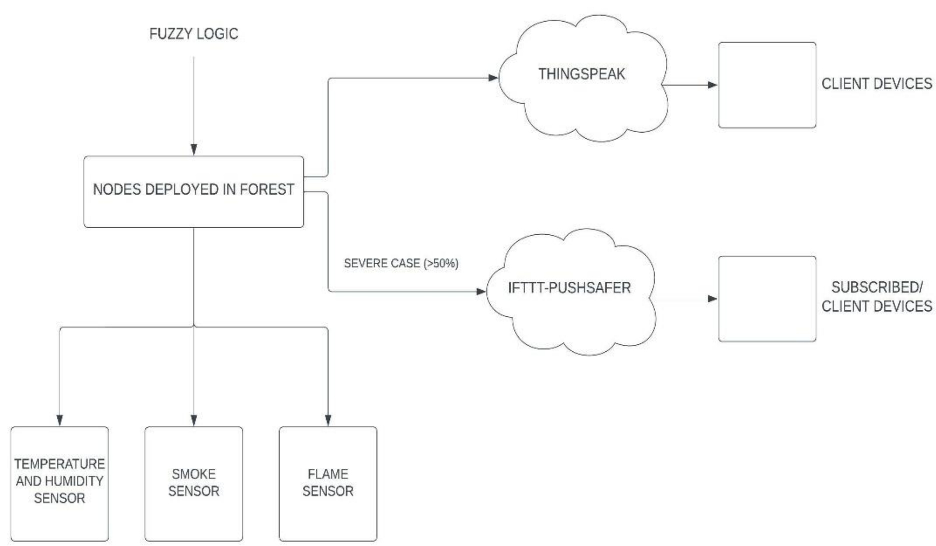


Figure 20- System Flow

5.4 PREDICTION OF OCCURRENCE OF FOREST FIRE

Based on the threshold values of the sensors the nodes are classified in to one of the following: no forest fire, mild, moderate and severe.

Code Snippet-

```
float C1 = 1; //severe
float C2 = 0.66;//moderate
float C3 = 0.33; //mild
  if(T>t_severe) T=t_severe; // if temperature is above 35°C, set
temperature to maximum in our range
if(flame==0) flame=1;
else flame=0;
  // Fuzzy rules
float w1 = trimf(T,t_moderate,t_severe,t_severe);//severe
float w2 = trimf(T,t_mild,t_moderate,t_severe);//moderate
float w3 = trimf(T,t_mild,t_mild,t_moderate);//mild
float w4 = trimf1(T,t_curr,t_mild,t_moderate);//mild
   // Defuzzyfication
float z = (w1*C1 + w2*C2 + w3*C3+ w4*C3*w4)/(w1+w2+w3+w4);
if(Status1== "Severe" && z<=0.80)
{
  z=0.99;
  return z;
}
else if(Status1== "Moderate" && z<=0.66)
{
  z=0.6789;
  return z;
}
else if(Status1== "Mild" && z<0.33)
{
  z=0.3769;
  return z;
}
return z;
```

Fuzzy logic is applied such that it gives the percentage of occurrence of forest fire. Detailed explanation of fuzzy logic is given in section 3.2.

5.5 PUBLISHING DATA TO THINGSPEAK

- Once you've connected the materials, visit ThingSpeak and create an account there. And then, create a new channel.
- After the creation of new channel, set the field names and make widgets to get a proper visualization of the data getting transmitted by the NodeMCU.
- After that, you can upload the code and make sure to fill the data as yours -start from API key and Field name, and also Access Point setting which will be connected to ESP-8266 Wireless Module (AP and Password).
- Also set the field for the data received to get a well-organized data.
- After you upload it, make sure that the WiFi has internet connection. Then, open serial monitoring to see data getting read by the sensors.

Screenshots-

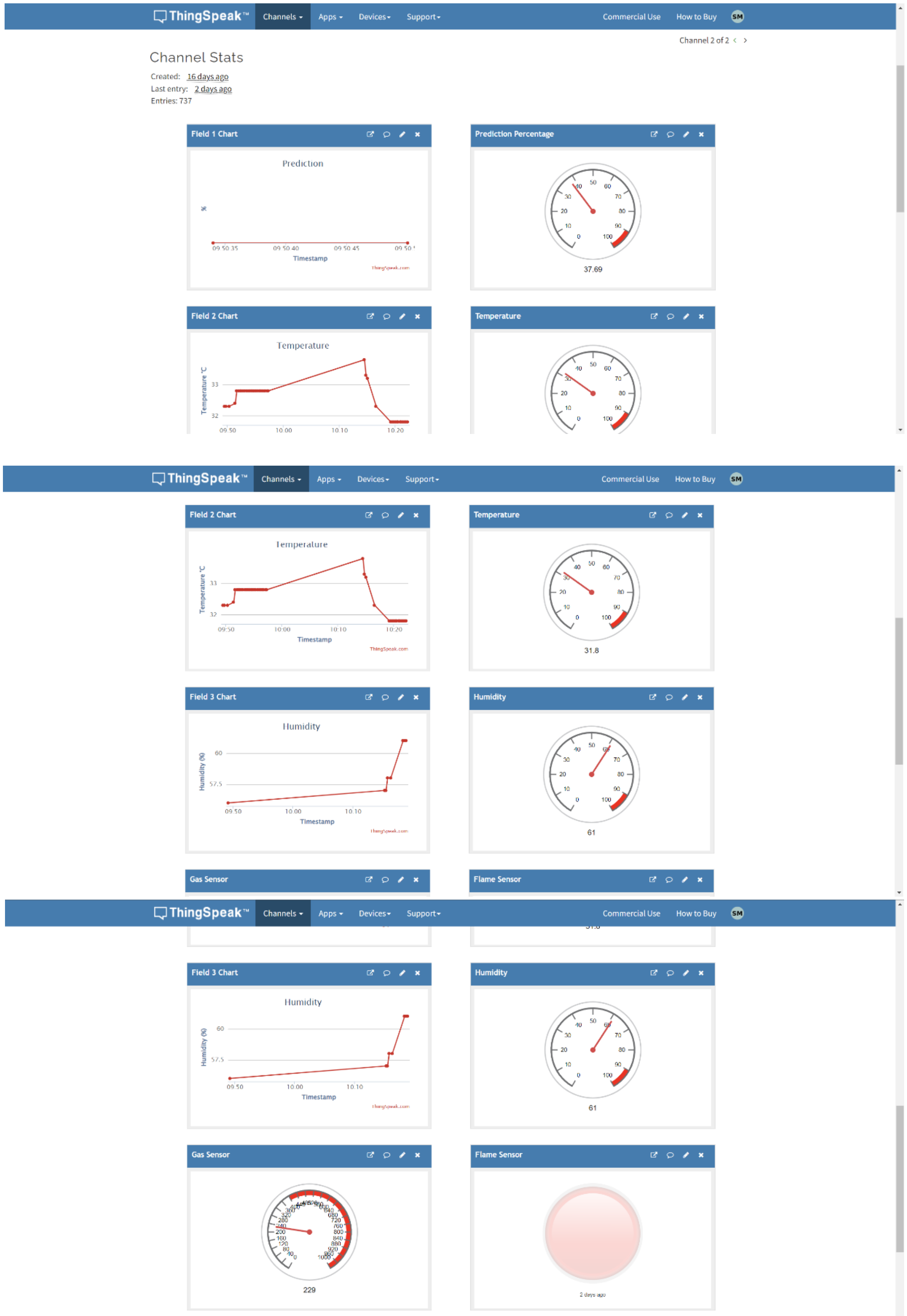


Figure 21- Data visualization in Thingspeak

5.6 SENDING ALERT

When the chances/probability of the forest fire goes greater than 50%, NodeMCU triggers the URL to activate the PushSafer app which sends a Priority alert notification with an Siren sound and three vibrations, set by us, to smartphone.

Setting up the notification- First open the IFTTT home page if you do not have an account then first create a free account on the IFTTT and sign in to the account in order to make the IFTTT applet.

When we will click on the create button a new window will open in which we will have two options; in the first option we will create the if statement and in the second we will perform the action. It works like, if this “some action” then that “some activity to perform”.

Then click on “if this” and click on the add button and in choose a service search for the web hook. When we click on the webhooks another window will open in which we will select receive a web request. Then we will give a name to the event “forest fire” and click on the create trigger button. When this event is triggered, it will send notification to the IFTTT applet.

Now the request is triggered so when this event is triggered what we will do, we will click on the “then that”. When we will click on the “then that” we will search for notifications in choose a service tab then click on the PushSafer notification. Now after the selection of the notifications a window will open in which we will select send a notification from the IFTTT App. Then we will write the notification in which we will send the alert notification through the IFTTT applet when the predicted probability is greater than 50%. We will set the priority, sound, vibrations, title and message inside the PushSafer configurations.

Then click on the web hook in order to get the key we will click on the documentation. Now by clicking on the documentation we will get the key and also we can test our application.

In order to receive the notification in the mobile we will need to install an PushSafer application in the mobile by simply downloading it from the Play store and login with the same account which we have used in the designing of IFTTT applet.

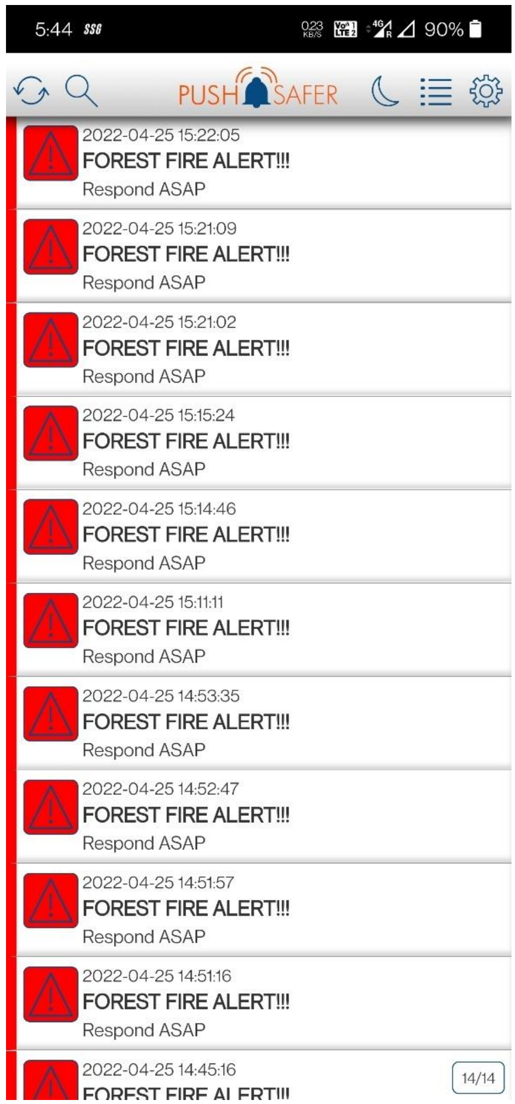


Figure 22- Alert on Smartphone

Additional enhancement- An active buzzer is connected to the NodeMCU which produces a warning alarm sound which will be useful to alert the public or even wildlife to refrain from going into the forest when there is a chance of occurrence of forest fire.

6 RESULTS AND DISCUSSION

The results are a number from the set [0, 100] which is the scale of fire probability. Decision making could be done on the basis of output obtained by making few conditions. If the output is between 0 and 50, then the probability of fire is from very low to medium and we can keep track of all the sensors on ThingSpeak. If the output is greater than 50 then the probability of fire is high and very high. A priority alert is constantly sent to all the devices connected including the smartphone, till someone responds.

7 CONCLUSION

The proposed forest fire detection handles the uncertainty present in the data effectively and gives the best results with very low false alarm rate. The decision based on this approach is more accurate. The membership functions and the parameters can be changed and modified as required. Rules also could be altered and adjusted according to parameters for further extending the work on this model.

8 REFERENCES

[1] Jun Zheng. 2009. Wireless Sensor Networks-A Networking Perspective. A John Wiley & Sons Inc. Publication.

[2] A. S. Tenenbaum, C. Gamage, B. Crispo, “Taking Sensor Networks from the Lab to the Jungle,” IEEE Computer, vol.39, Issue 8, IEEE Computer Society, 2006.

[3] P. Manjunatha, A.K. Verma and A. Srividya, “MultiSensor Data Fusion in Cluster based Wireless Sensor Networks Using Fuzzy Logic Method,” 2008 IEEE Region 10 Colloquium and the Third ICIIS, Kharagpur, INDIA December 8-10.

[4] Hamdy Soliman, Komal Sudan, Ashish Mishra, “A Smart Forest-Fire Early Detection Sensory System: Another Approach of Utilizing Wireless Sensor

and Neural Networks," IEEE SENSORS 2010 Conference, 2010.

[5] Giovanni Laneve, "Continuous Monitoring of Forest Fires in the Mediterranean Area Using MSG," IEEE TRANSACTIONS ON GEOSCIENCE AND REMOTE SENSING, VOL. 44, NO. 10, OCTOBER 2006.

[6] Jerome Vicente, "An image processing technique for automatically detecting forest fire," International Journal of Thermal Sciences, 2002.

[7] B. Ugur Toreyin, "Wildfire Detection Using LMS Based Active Learning," IEEE, 2009.

[8] A.K. Singh, Harshit Singh, "Forest Fire Detection through Wireless Sensor Network using Type-2 Fuzzy System," 2012 International Journal of Computer Applications (0975 – 8887).

[9] Dongrui U, Maowen Nie "Comparison and Practical Implementation of Type-Reduction Algorithms for Type2 Fuzzy Sets and Systems," 2011 IEEE International Conference on Fuzzy Systems June 27-30, 2011, Taipei, Taiwan.

[10] Timothy J. Ross. 2004. Fuzzy Logic with Engineering Applications. A John Wiley & Sons Inc. Publication.

APPENDIX

<Sample code, snapshot etc.>

NODEMCU CODE-

```
#include <ThingSpeak.h>
#include "DHT.h"
#include <ESP8266WiFi.h>
#include <WiFiClientSecure.h>

unsigned long myChannelNumber = 1701474;
const char * myWriteAPIKey = "U6A9SEZOZX78K9YN";

const char* resource =
"https://maker.ifttt.com/trigger/forest_fire/json/with/key/o3UmGdPco8QoD871FmUf95Fq0EpjyP40ZtJWYXmrt5x";
const char* server = "maker.ifttt.com";

#define DHTPIN 2     // what digital pin the DHT22 is conected to
#define DHTTYPE DHT11   // there are multiple kinds of DHT sensors
#define         MQ_PIN                     (0)     //define which analog input channel you are going to use

DHT dht(DHTPIN, DHTTYPE);

float Prob=0.0;
float t=0.0;
float h=0.0;
int sensorValue;
float sensorValue1 = 0.0;
int flame = 0;
int flame_sensor = 12; //Connected to D6 pin of NodeMCU
int buzzer = 4; //Connected to D2 pin of NodeMCU
int temp = 0;

float t_severe=37.5;
float t_moderate=33.5;
float t_mild=31.0;
float t_curr=0.0;
/********************************************************************************/
const char* ssid     = "Sam1"; // Your WiFi ssid
const char* password = "12345677"; // Your Wifi password;
/**********************************************************************************/


// Initialise the WiFi and MQTT Client objects
WiFiClient wifiClient;

/**************************************************************/

const char* Status1="No Forest fire";

void setup()
{
  pinMode(flame_sensor, INPUT);
  pinMode(buzzer, OUTPUT);

  digitalWrite(buzzer, LOW);

  Serial.begin(9600);
  Serial.setTimeout(2000);
  Serial.print("Attempting to connect to SSID: ");
  Serial.print(ssid);
```

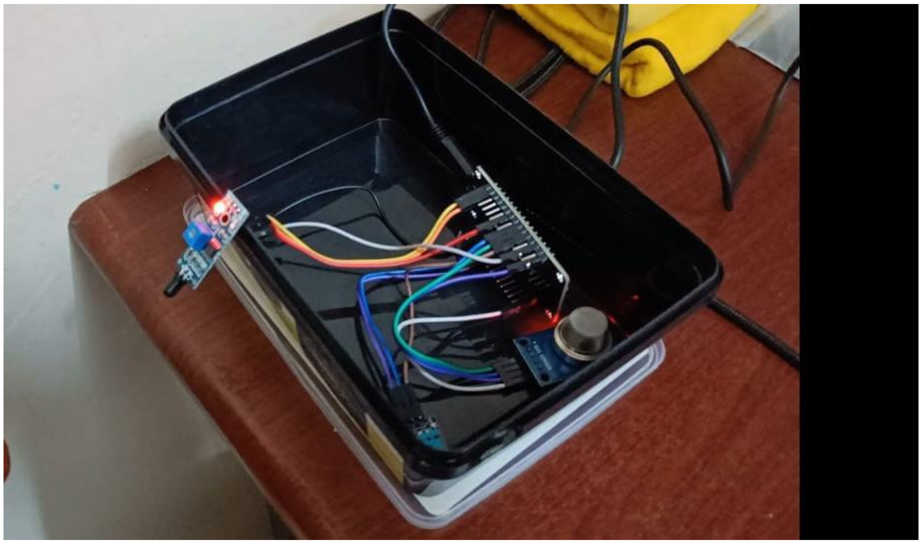
```
  WiFi.begin(ssid, password);
  ThingSpeak.begin(wifiClient);  // Initialize ThingSpeak
  // Attempt to connect to WiFi network:
  while (WiFi.status() != WL_CONNECTED) {
    Serial.print(".");
    // Connect to WPA/WPA2 network. Change this line if using open or WEP  network:
    // Wait 5 seconds for connection:
    delay(1000);
  }
  Serial.print("\n");
  dht.begin();
}

void loop()
{
  delay(2000);
  ReadDHT11();
  gasSensor();
  flameSensor();
  prediction();
}

/*TEMPERATIRE AND HUMIDITY CALCULATION*/
void ReadDHT11()
{
  h = dht.readHumidity();

  t = dht.readTemperature();

  float f = dht.readTemperature(true);

  // Check if any reads failed and exit early (to try again).
  if (isnan(h) || isnan(t) || isnan(f)) {
    Serial.println(F("Failed to read from DHT sensor!"));
    return;
  }

  // Compute heat index in Fahrenheit (the default)
  float hif = dht.computeHeatIndex(f, h);
  // Compute heat index in Celsius (isFahreheit = false)
  float hic = dht.computeHeatIndex(t, h, false);


  Serial.print(F("Humidity: "));
  Serial.print(h);
  Serial.print(F("%  Temperature: "));
  Serial.print(t);
  Serial.print(F("°C "));
  Serial.print(f);
  Serial.print(F("°F  Heat index: "));
  Serial.print(hic);
  Serial.print(F("°C "));
  Serial.print(hif);
  Serial.println(F("°F"));
  ThingSpeak.writeField(myChannelNumber, 2, t, myWriteAPIKey);
  ThingSpeak.writeField(myChannelNumber, 3, h, myWriteAPIKey);

  if(temp==0)
  {
    t_curr= dht.readTemperature();
    t_mild=t_curr+ 2.0 ;
    t_moderate=t_mild+3.0;
```

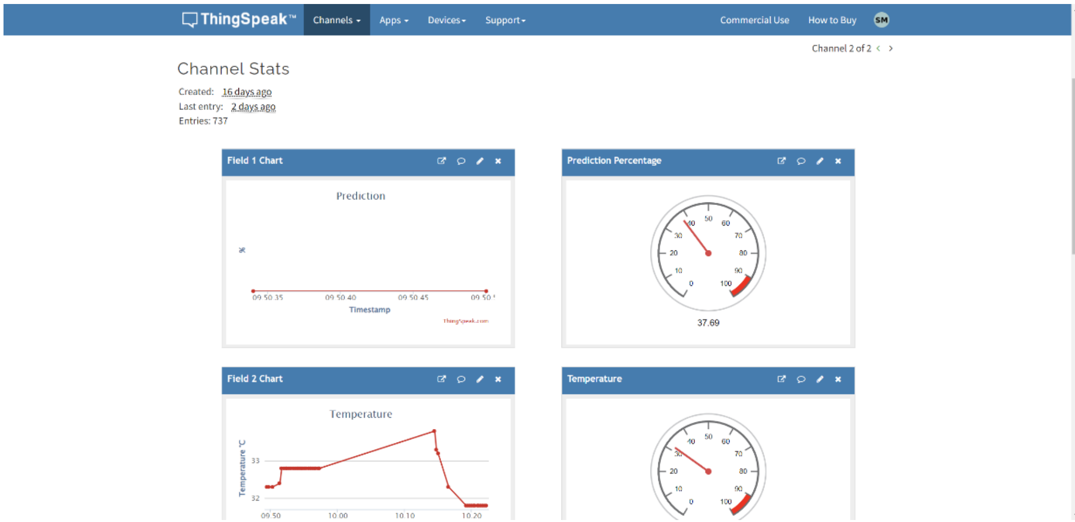
```
    t_severe=t_moderate+2.0;
    temp=1;
  }
}
/*****************************************************************************/
/*MQ2 SENSOR*/
void gasSensor()
{
  sensorValue1 = analogRead(MQ_PIN); // read analog input pin 0
  ThingSpeak.writeField(myChannelNumber, 4, sensorValue1, myWriteAPIKey);

  sensorValue = digitalRead(D0); // read analog input pin 0
  Serial.print("Sensor Value: ");
  Serial.print(sensorValue1);

  if (sensorValue1 > 400)
  {
    Serial.print(" | Smoke detected!");
  }
}

/*****************************************************************************/
/*FLAME SENSOR*/

void flameSensor() {

  flame = digitalRead(flame_sensor); //reading the sensor data on A0 pin
  ThingSpeak.writeField(myChannelNumber, 5, flame, myWriteAPIKey);
  if (flame == 0)
  {
    Serial.print("\nALERT FIRE DETECTED\n");
  }
  else
  {
    Serial.print("\nNO FIRE DETECTED\n");
  }
}

void Buzzerfn()
{
  pinMode(D2,OUTPUT);
  digitalWrite(D2, HIGH);
  delay(200);
  digitalWrite(D2, LOW);
  delay(200);
}

void prediction()
{
  if(flame==0 || t>t_severe || (t>t_moderate && sensorValue==1))
  {
    Status1="Severe";
    Serial.print("Severe");
    digitalWrite(buzzer, HIGH);
  }
  else if(t>=t_moderate || (t>t_mild && sensorValue==1))
  {
    Status1="Moderate";
    Serial.print("Moderate");
    Buzzerfn();
  }
  else if(t>=t_mild || sensorValue==1)
```

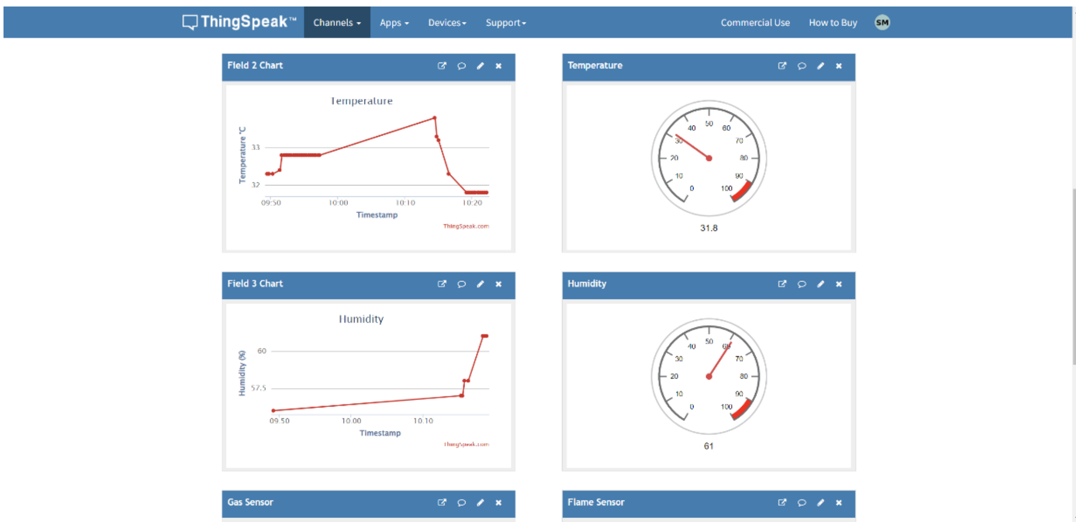
```
 {
   Status1="Mild";
   Serial.print("Mild");
   Buzzerfn();
 }
 else
 {
   Status1="No Forest fire";
   Serial.print("No Forest fire");
   pinMode(D2,INPUT);
 }

 float conf=classify(t,sensorValue,flame);
 Serial.print(" chance of forest fire in %=");
 Prob=conf*100;
 ThingSpeak.writeField(myChannelNumber, 1, Prob, myWriteAPIKey);
 Serial.println(Prob);
 Serial.print("\n");
 if(Prob>50)
 {
   WiFiClient client;

     client.print(String("GET ") + resource +
                  " HTTP/1.1\r\n" +
                  "Host: " + server + "\r\n" +
                  "Connection: close\r\n\r\n");

     int timeout = 5 * 10; // 5 seconds
     while(!!!client.available() && (timeout-- > 0))
     {
       delay(100);
     }
     while(client.available())
     {
       Serial.write(client.read());
     }
     client.stop();
 }
 else
 {
     return;
 }
}


float trimf(float x,float a, float b, float c){
  float f;
  if(x<=a)
     f=0;
  else if((a<=x)&&(x<=b))
     f=(x-a)/(b-a);
  else if((b<=x)&&(x<=c))
     f=(c-x)/(c-b);
  else if(c<=x)
     f=0;
  return f;
}
float trimf1(float x,float a, float b, float c){
  float f;
  if(x>=b || x>=c) return 0;
  else if(x>a) return (x-a)/(b-a);
```

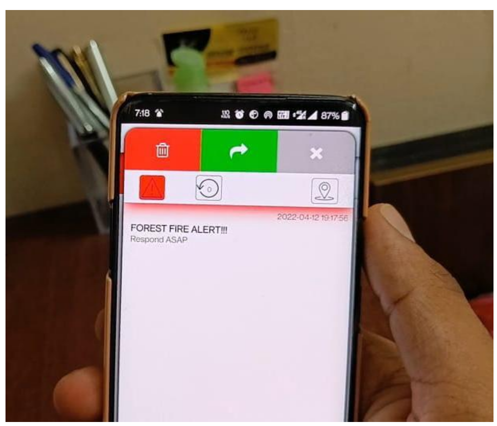
```
  if(x==a) return 0.001;

}
// Function for predicting atmospheric comfort from temperature and relative humidity
float classify(float T,float smoke,float flame ){
  float C1 = 1; //severe
  float C2 = 0.66;//moderate
  float C3 = 0.33; //mild

  if(T>t_severe) T=t_severe; // if temperature is above 35°C, set temperature to maximum in our range
  if(flame==0) flame=1;
  else flame=0;

  // Fuzzy rules
  float w1 = trimf(T,t_moderate,t_severe,t_severe);//severe
  float w2 = trimf(T,t_mild,t_moderate,t_severe);//moderate
  float w3 = trimf(T,t_mild,t_mild,t_moderate);//mild
  float w4 = trimf1(T,t_curr,t_mild,t_moderate);//mild

    // Defuzzyfication
  float z = (w1*C1 + w2*C2 + w3*C3+ w4*C3*w4)/(w1+w2+w3+w4);
  if(Status1== "Severe" && z<=0.80)
  {
    z=0.99;
    return z;
  }
  else if(Status1== "Moderate" && z<=0.66)
  {
    z=0.6789;
    return z;
  }

  else if(Status1== "Mild" && z<0.33)
  {
    z=0.3769;
    return z;
  }
  return z;
}
```

SNAPSHOTS-









For entire code and videos – **https://drive.google.com/drive/folders/17wsuO3QMp2uFqSr91OQuplmZk42h3JqD?usp=sharing**