# Mini Project - I Report

on

**Data acquisition & IOT enabled weather monitoring/forecasting system**

*Submitted By*


Bathina Anudeep (18BEC006)
Devata Naveen Nischal (18BEC011)
Neelampalle Nikhil Kumar (18BEC031)
Shaik Mastan Shariff (18BEC042)

Under the guidance of
Dr. Prakash Pawar
Asst.Professor, Department of ECE

INDIAN INSTITUTE OF INFORMATION TECHNOLOGY
DHARWAD.

# Acknowledgment

We would like to express our sincere gratitude to Dr. Prakash Pawar, Asst.Professor, Department of ECE-IIIT Dharwad for his guidance and constant support throughout the course of this minor project. We would also like to thank all the faculty and administration of the institute who ensured the needs fulfilled for the completion of this project.

Date : May 2021
Place: Hubballi

Bathina Anudeep(18BEC006)
Devata Naveen Nischal (18BEC011)
Neelampalle Nikhil Kumar (18BEC031)
Shaik Mastan Shariff (18BEC042)

# Preamble

The system proposed for monitoring weather conditions in a particular place like temperature, humidity, CO Level using sensors, sensors detect changes in environment and send it to the users for making statistical analysis, IoT is the technology used for monitoring, collecting, controlling and connecting the system to worldwide, which is the more efficient and advanced solution for accessing the information in the world. A growing demand for integrated security and safety systems leads to an increase in the initiatives taken for smart community projects. This system proposes a real-time weather monitoring system designed for any environment that displays weather parameters such as the intensity of rainfall, temperature, etc. obtained from the sensors to the cloud by implementing message queuing telemetry transport protocol. The proposed system is portable, affordable and the data can be accessed at any instant. The Internet of Things is implemented to make informed decisions and to optimize the experience of the residents by providing them with real time data and sending customized alert notifications via email to the right person. The clients can subscribe to the channel from anywhere in the world and get the updates from the implemented system in their smartphone and PC.

## CONTENTS

# Chapter 1: Introduction

Nowadays, there is an increase in demand for service over the internet. The world is running towards the Internet of Things (IoT) generation and nearly 50 billion devices will be connected to the internet by the end of 2020. IoT enables not only human human interaction but also human device interaction as well as device-device interaction. This advancement will have an impact on various sectors such as transportation, automation, and energy. IoT heightens the degree of automation which creates an essential impact in smart communities. For example, the residents of the smart home might not be aware of the environmental conditions outside such as the temperature, rainfall, etc since the climate is conditioned inside the home with IoT.

When the smart homes around a region are networked it becomes a smart community. It is formulated upon the mutual agreement and participation of the homeowners. Smart communities have broadband connectivity that is readily available and is affordable for their residents. Weather monitoring is one of the major aspects that transforms a community into a smart community. Thus, it is necessary to develop a real-time weather monitoring system that can store and display the weather parameters instantly.

The sensors that measure weather parameters such as temperature, humidity, rainfall, etc are placed on the rooftop. The respective data are extracted, stored and uploaded to the cloud by the implementation of IoT. If the setup in this paper is implemented in a cluster of smart homes, it would become a smart community. The residents can monitor the output of the sensors which are placed in their respective homes when required. The data obtained from the connected sensors and the cup anemometer can help in weather forecasting for wind farms and load dispatch centres.

The aim is to develop a smart, affordable and portable weather station along with the advanced facility of storing and displaying the weather parameters in the cloud. Moreover, an app is developed, which is used by the residents to view the real-time weather parameters and to get customized alerts via email.

## Chapter 2: Review of Literature

This prediction is based on understanding of processes and history over different timescales, spatial resolution, costs and accuracy.To more efficiently use the limited amount of water under the impact of global climate change or to resourcefully provide adequate time for flood and drought warning, there is a need to seek an advanced modeling technique for improving stream-flow forecasting on a short-term basis. In any data mining model, the raw data is the first required input. In this part, we will collect and build a dataset about climate related attributes in Jordan over a history of years. The dataset may include attributes related to waterfall, temperature,etc that will be gathered by climate and water domain experts. In the second stage and in order to build a climate and weather classifier, we will train the model through using actual data.After training and evaluating the model, it will be used for future forecasting.

### 2.1 An Interactive Predictive System for Weather Forecasting

A tool was built to parse all weather related information from different websites that store such information. Data mining and AI algorithms were used for future forecasting based on historical data.

### 2.2 Weather Prediction: A novel approach for measuring and analyzing weather data

The proposed system serves as a tool that takes in the rainfall data from a large amount of data as input and predicts the future rainfall in an efficient manner. Predictive analytic models capture relationships among many factors in a data set to assess risk with particular set of conditions to assign a score or weight. These patterns of score/weight found in historical data can be used to predict future values.

### 2.3 Zhang et al. introduced the problem of mining dynamic inter- dimension association rules for local-scale severe weather prediction .

At the beginning they transformed the original weather record database into a new database expressing the change tendencies of the measurements of the weather. And they proposed a new algorithm DIAL, incorporating the algorithm for quantitative association rules and the process of database transformation. Then they used the algorithm for mining quantitative association rules to generate the result rules in quantitative format. Finally, they introduced some predicates to generate the final rules.

## Chapter 3: Report on the present investigation

Will it rain and the match gets dismissed or will it be bright and sunny? How can it be determined whether the conditions are suitable for conducting a cricket match or not? The temperature is too high this summer, will it rain or not? The answer to these types of questions is only given by the concept of machine learning in conjunction with IoT. To develop an IoT system, a thing is required which should be equipped with necessary sensors, actuators and a communication interface. The communication interface is needed to connect the thing to the internet that allows them to send the data to the cloud or a remote machine for monitoring and also for analytics purposes. The thing can also receive the control information based on some analytics and take some actuating decisions. In the proposed system, the things refer to the NodeMCU that includes firmware, which runs on the ESP8266 Wi-Fi SoC. The cloud refers to the ThingSpeak platform for monitoring the data. The analytics refers to the machine learning algorithm i.e. a logistic regression mathematical model. This model is trained with the pre-recorded data values of temperature, humidity and light intensity. Further, it is used for prediction.

### 3.1 NodeMCU (ESP8266 ESP-12E)

NodeMCU is an open-source Lua based firmware and development board specially targeted for IoT based Applications. It includes firmware that runs on the ESP8266 WiFi SoC from Espressif Systems, and hardware which is based on the ESP-12 module.

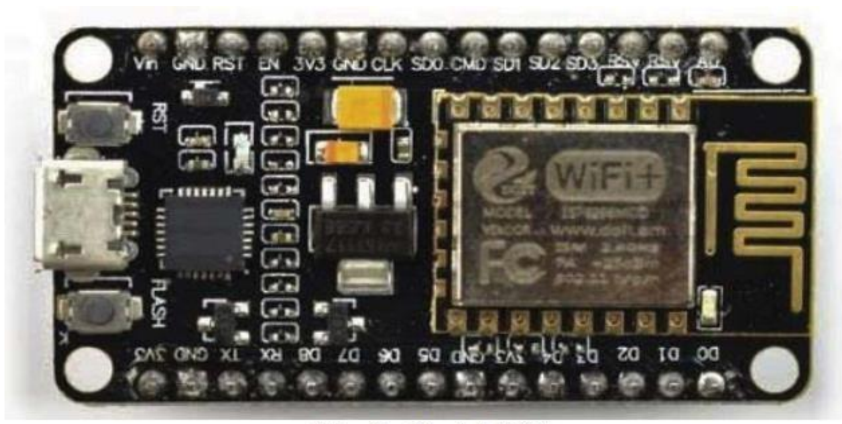
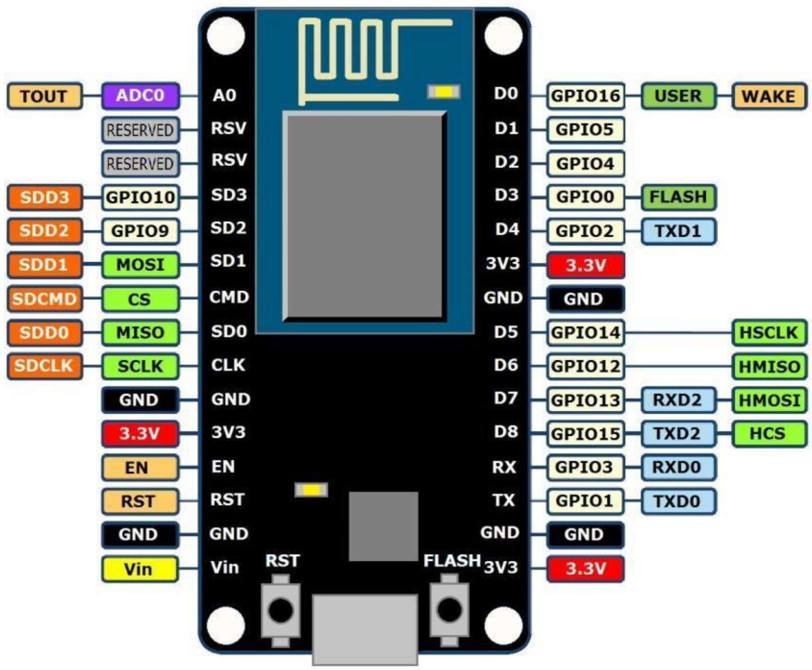



Fig. 2. NodeMCU

Applications of NodeMCU

- Prototyping of IoT devices
- Low power battery operated applications
- Network projects
- Projects requiring multiple I/O interfaces with Wi-Fi and Bluetooth functionalities.

## 3.2 Pin configuration of NodeMCU

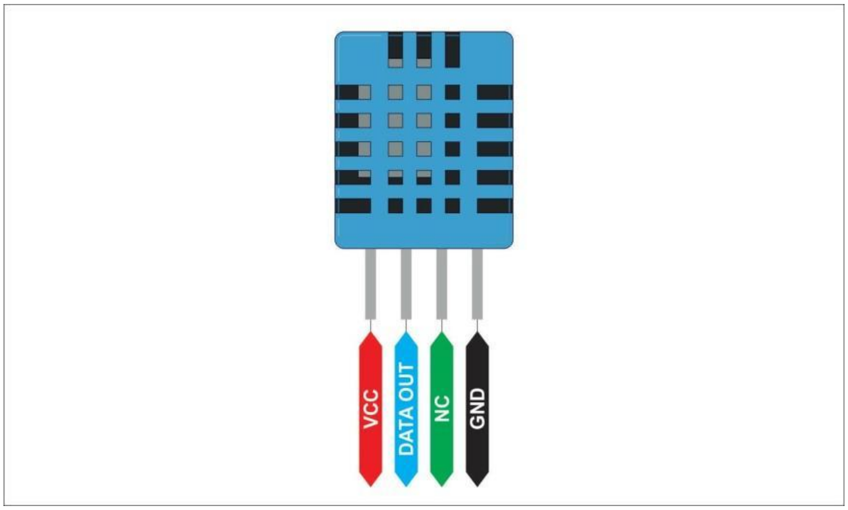



NodeMCU ESP8266 Specifications & Features

- Microcontroller: Tensilica 32-bit RISC CPU Xtensa LX106
- Operating Voltage: 3.3V
- Input Voltage: 7-12V
- Digital I/O Pins (DIO): 16
- Analog Input Pins (ADC): 1
- UARTs: 1
- SPIs: 1
- I2Cs: 1
- Flash Memory: 4 MB
- SRAM: 64 KB
- Clock Speed: 80 MHz
- USB-TTL based on CP2102 is included onboard, Enabling Plug n Play
- PCB Antenna

Small Sized module to fit smartly inside your IoT projects

### 3.3 DHT11 Sensor



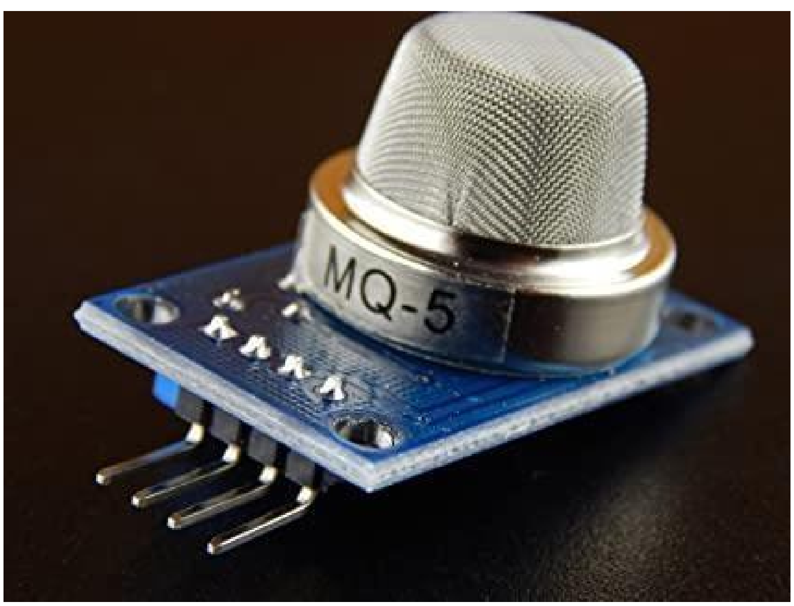

The DHT11 sensor gives humidity & temperature data.

The DHT11 is chosen because it is calibrated, accurate and stable and its single output is digital.

**DHT11 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%

### 3.4 MQ05 Sensor



It is also called a smoke sensor.

They are used in gas leakage detecting equipment in family , industry and forest , are suitable for detecting smoke , LPG, natural gas , and town gas.

FEATURES

* High sensitivity to LPG, natural gas , town gas

* Small sensitivity to alcohol, smoke.

* Fast response .

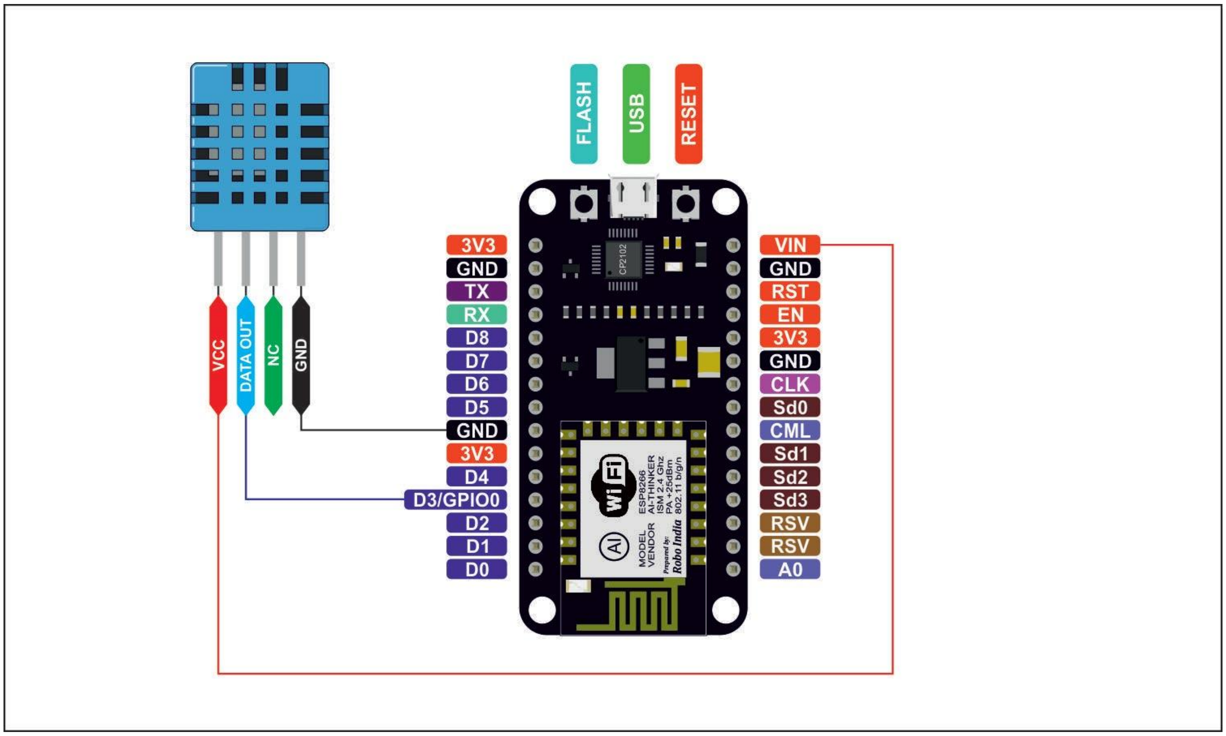
* Stable and long life

* Simple drive circuit

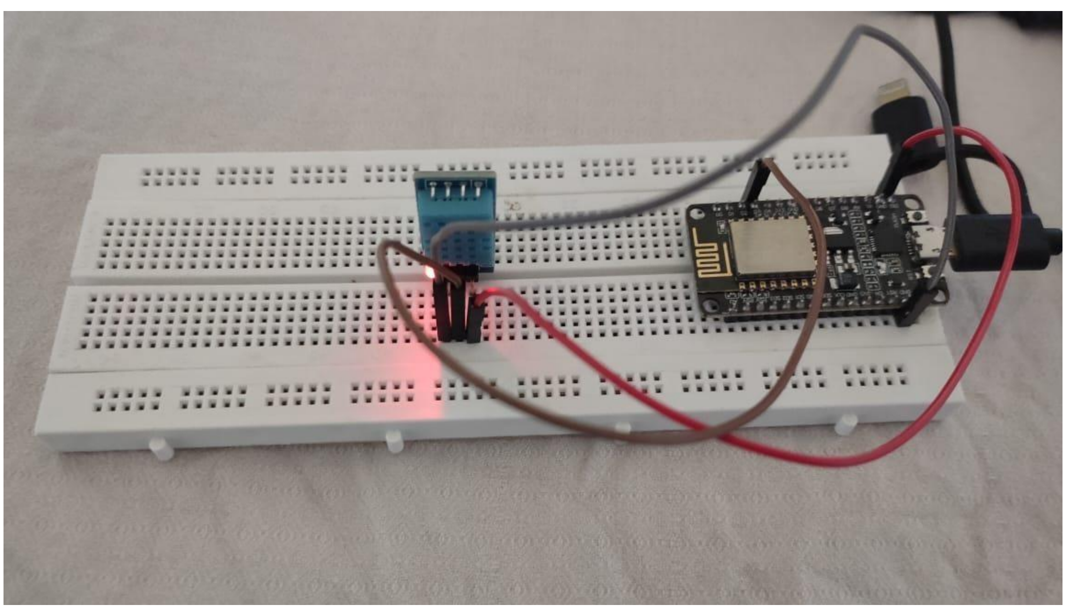
## **3.5 Rain Sensor**


Raindrop Sensor is a tool used for sensing rain. It consists of two modules, a rain board that detects the rain and a control module, which compares the analog value, and converts it to a digital value

## 3.6 Connections

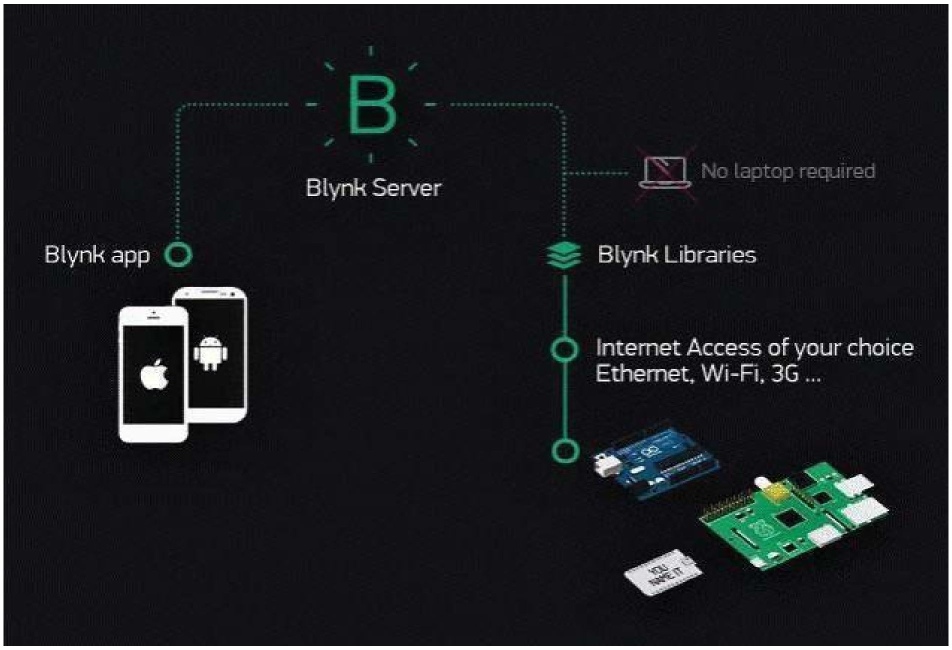



## 3.7 Hardware Setup



### 3.8 Blynk

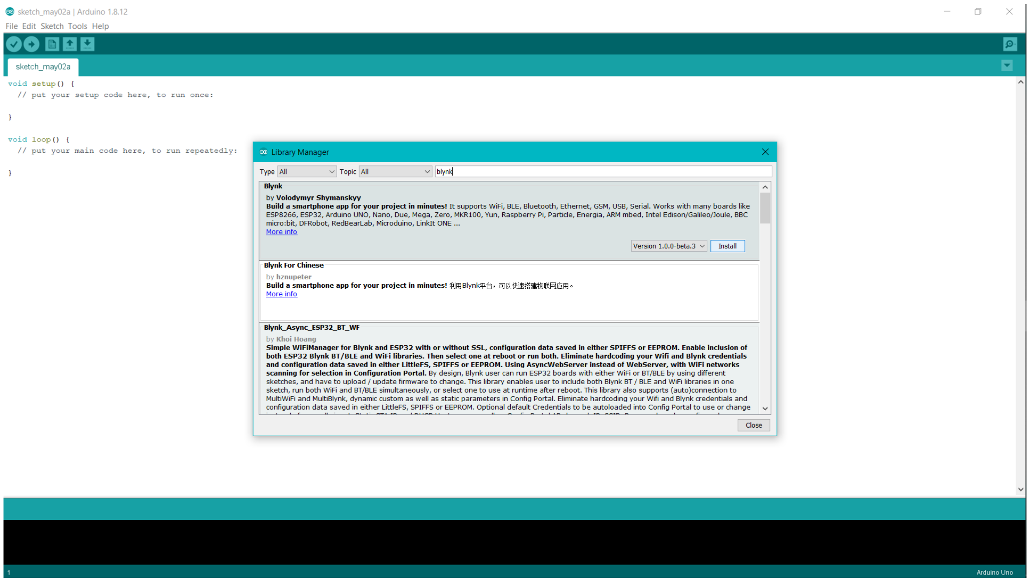
Blynk was designed for the Internet of Things. It can control hardware remotely, it can display sensor data, it can store data, visualize it and do many other cool things. There are three major components in the platform: ...Blynk Server - responsible for all the communications between the smartphone and hardware.




FEATURES

- Similar API & UI for all supported hardware & devices • Connection to the cloud using:
- WiFi
- Bluetooth
- Ethernet
- USB (Serial)
- GSM
- Set of easy-to-use Widgets
- Direct pin manipulation with no code writing
- Easy to integrate and add new functionality using virtual pins
- History data monitoring via Super Chat widget
- Device-to-Device communication using Bridge Widget
- Sending emails, tweets, push notifications, etc.

## 3.9 Installing blynk in arduino ide




.

## 3.10 weather prediction algorithms

### 3.10.1 Linear regression

Linear regression is a **linear model**, e.g., a model that assumes a linear relationship between the input variables (x) and the single output variable (y). More specifically, that y can be calculated from a linear combination of the input variables (x).

Learning a linear regression model means estimating the values of the coefficients used in the representation with the data that we have available.

After applying a linear regression model to our data the predicted results are shown below fig.In this we used one year data for prediction.

### 3.10.2 support vector regression

Support Vector Machines (SVM) are popularly and widely used for classification problems in machine learning. I've often relied on this not just in machine learning projects but when I want a quick result in a hackathon.

But SVM for regression analysis? I hadn't even considered the possibility for a while! And even now when I bring up "Support Vector Regression" in front of machine learning beginners, I often get a bemused expression. I understand – most courses and experts don't even mention Support Vector Regression (SVR) as a machine learning algorithm

### 3.10.3 Random forest

Random forest, like its name implies, consists of a large number of individual decision trees that operate as an ensemble. Each individual tree in the random forest spits out a class prediction and the class with the most votes becomes our model's prediction.

The low correlation between models is the key. Just like how investments with low correlations (like stocks and bonds) come together to form a portfolio that is greater than the sum of its parts, uncorrelated models can produce ensemble predictions that are more accurate than any of the individual predictions. The reason for this wonderful effect is that the trees protect each other from their individual errors (as long as they don't constantly all err in the same direction). While some trees may be wrong, many other trees will be right, so as a group the trees are able to move in the correct direction. So the prerequisites for random forest to perform well are:

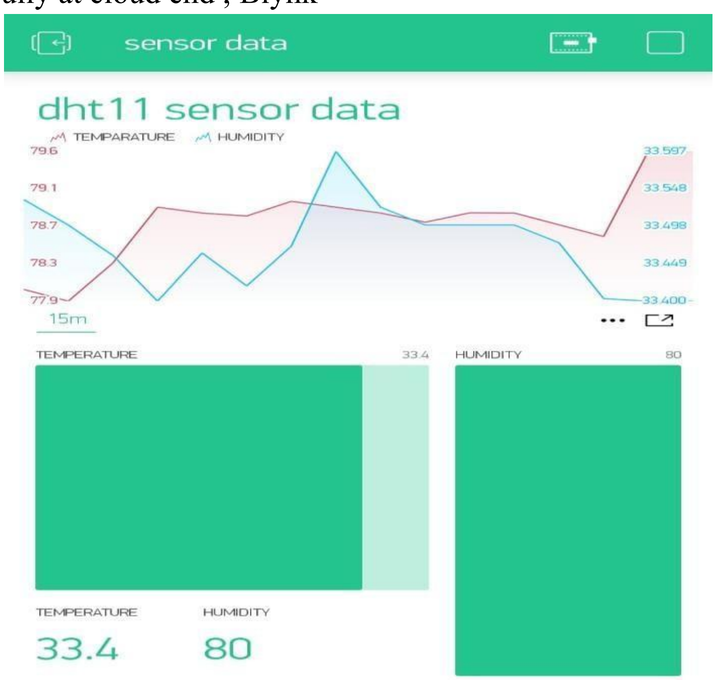
1.There needs to be some actual signal in our features so that models built using those features do better than random guessing.

2.The predictions (and therefore the errors) made by the individual trees need to have low correlations with each other.

## Chapter 4: Results and Discussions

### 4.1 Blynk

The Data is collected successfully at cloud end , Blynk



## 4.2 Predictions

**Comparison of mean square errors for temperature:**

Linear regression:31.5

Random forest:23.31

Support vector regression:39.75

**mean square errors for humidity:**

Linear regression:159.99

**mean square errors for pressure:**

Linear regression:44.841

## 4.3 Observations

The predicted values are almost equal for all the humidity values using Linear regression.Due to the small dataset & the model is taking test sets serialy.
Random forest gives better accuracy compared to linear & support vector regression.

# Chapter 5: Summary and Conclusions

## 5.1 Summary

Sensors are installed to monitor the parameters like temperature, humidity, and CO Value using IDE (Integrated Development Environment) received data and result analysis will be sent to the end user through Wi-Fi. NodeMCU controller used to control all the sensors, and it receives the data from sensors, and sends it to end users through the cloud. This type of model can be used for industrial & domestic purposes (weather monitoring or Humidity monitoring) etc…alert messages can be sent to users from time to time. And predicted the data.

## 5.2 Future Enhancements

- Taking more shots into consideration and build a larger data set
- Efficient algorithm need to be used
- Some more parameters to be considered for assessing the predicted weather performance are:
    - All climates should be considered
    - Data set of 6-7 years should be used
    - Test data should be correlated with the predicting day

**GitHub Link of the project:** https://github.com/thebathina/Mini-project1

## Chapter 6: Appendix

## Codes:

**Arduino codes for Data acquisition using sensors**

### 6.1 DHT11-Temperature & Humidity

```
 #include "DHT.h"       // including the library of DHT11 temperature and humidity sensor
#define DHTTYPE DHT11   // DHT 11

#define dht_dpin 0
DHT dht(dht_dpin, DHTTYPE);
void setup(void)
{
  dht.begin();
  Serial.begin(9600);
  Serial.println("Humidity and temperature\n\n");
  delay(700);

}
void loop() {
    float h = dht.readHumidity();
    float t = dht.readTemperature();
    Serial.print("Current humidity = ");
    Serial.print(h);
    Serial.print("%  ");
    Serial.print("temperature = ");
    Serial.print(t);
    Serial.println("C  ");
  delay(800);
}
```

### 6.2 Rainfall sensor

```
int sensorPin = A0;    // input for LDR and rain sensor
int enable2 = 13;      // enable reading Rain sensor
int sensorValue2 = 0;  // variable to store the value coming from sensor Rain sensor
void setup()
{
// declare the enable and ledPin as an OUTPUT:
pinMode(enable2, OUTPUT);
Serial.begin(115200);
}
```

```
void loop() {
//--------------------------Rain Sensor-------------------------
delay(500);
sensorValue2 = analogRead(sensorPin);
sensorValue2 = constrain(sensorValue2, 150, 440);
sensorValue2 = map(sensorValue2, 150, 440, 1023, 0);
if (sensorValue2>= 20)
  {
  Serial.print("rain is detected");
  digitalWrite(enable2, HIGH);
  }
else
  {
  Serial.print("rain not detected");
  digitalWrite(enable2, LOW);
  }
//Serial.print("Rain value:       ");
//Serial.println(sensorValue2);
Serial.println();
delay(100);
}
```

**6.3 MQ05-Gas sensor**

```
#define MQpin A0                    //MQ module pin is connected to analog pin A0 of NodeMCU
void setup()  //This executes only once in a Program Execution
{
  pinMode(MQpin, INPUT);      //MQ pin has made input in NodeMCU
  Serial.begin(115200);       //Serial Monitor starts with 115200 buad rate
}
void loop()
{
  float MQvoltage = (analogRead(MQpin) * (5.00 / 1023));   //To calculate the voltage read from ADC
  Serial.print("MQ - 05 Sensor = ");
  Serial.print(MQvoltage);            //Printing the voltage value
  Serial.println("Volts");
  delay(1000);                        //Delay of 1 second
}
```

**6.4 DHT11-Temperature & Humidity-Blynk**

```
 #include <ESP8266WiFi.h>
#include <BlynkSimpleEsp8266.h>
#include "DHT.h"        // including the library of DHT11 temperature and humidity sensor
```

```
#define DHTTYPE DHT11   // DHT 11
int enable2 = 13;
#define dht_dpin 0
char auth[] = "TJ35YvwW0I9r7DpGSu5oC3IOpimbJdnh";
char ssid[] = "My Zone";
char pass[] = "sreevani";
DHT dht(dht_dpin, DHTTYPE);
BlynkTimer timer;
void sendSensor()
{
  float h = dht.readHumidity();
  float t = dht.readTemperature(); // or dht.readTemperature(true) for Fahrenheit
  if (isnan(h) || isnan(t))
  {
    Serial.println("Failed to read from DHT sensor!");
    return;
  }
  // You can send any value at any time.
  // Please don't send more that 10 values per second.
  Blynk.virtualWrite(V5, t);
  Blynk.virtualWrite(V6, h);
 }
void setup(void)
{
  dht.begin();
  pinMode(enable2, OUTPUT);
  Serial.begin(9600);
  Blynk.begin(auth, ssid, pass);
  Serial.println("Humidity and temperature\n\n");
  timer.setInterval(1000L, sendSensor);
  delay(700);

}
void loop() {

    float h = dht.readHumidity();
    float t = dht.readTemperature();

    Serial.println();
    Serial.flush();
    Serial.print("Current humidity = ");
    Serial.print(h);
```

```
  Serial.print("%  ");
  Serial.print("temperature = ");
  Serial.print(t);
  Serial.println("C  ");
  Serial.println();
  delay(1000);
  Blynk.run();
  timer.run();
}
```

**Machine learning codes for weather prediction**

**6.5 Linear regression for temperature**

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt

import seaborn
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn import metrics

dataset= pd.read_csv('weather1yeardata.csv')
print(dataset.shape)

dataset.plot(x='day', y='temparature9am' , style='o')
plt.title('temparature9am vs day')
plt.xlabel('days')
plt.ylabel('temparature9am')
plt.show()

plt.figure(figsize=(15,10))
plt.tight_layout()
seaborn.distplot(dataset['temparature9am'])
plt.show()

X= dataset['day'].values.reshape(-1,1)
y= dataset['temparature9am'].values.reshape(-1,1)
X_train,X_test,y_train,y_test = train_test_split(X,y,test_size=0.2, random_state=0)

model =LinearRegression()
model.fit(X_train,y_train)
```

```
print('Intercept is :',model.intercept_)

print('Slope is :' ,model.coef_)

y_pred= model.predict(X_test)

df=  pd.DataFrame({'Actual': y_test.flatten(), 'Predicted': y_pred.flatten()})
print(df)

df1= df.head(20)
df1.plot(kind='bar', figsize=(16,10))
plt.grid(which='major', linestyle='-',linewidth='0.5', color='green')
plt.grid(which='minor', linestyle=':',linewidth='0.5', color='black')
plt.show()

plt.scatter(X_test,y_test,color='gray')
plt.plot(X_test,y_pred,color='red',linewidth=2)
plt.show()

print('Mean absolute error is:', metrics.mean_absolute_error(y_test,y_pred))
print('Mean squared error is:', metrics.mean_squared_error(y_test,y_pred))
print('Root mean squared error is:', np.sqrt(metrics.mean_squared_error(y_test,y_pred)))
```

**6.6 Linear regression for pressure**

```
model2 =LinearRegression()
model2.fit(X2_train,y2_train)

print('Intercept is :',model2.intercept_)
print('Coefficient is :' ,model2.coef_)
y2_pred= model2.predict(X2_test)
df4=  pd.DataFrame({'Actual': y2_test.flatten(), 'Predicted': y2_pred.flatten()})
print(df4)
df5= df4.head(20)
df4.plot(kind='bar', figsize=(16,10))
plt.grid(which='major', linestyle='-',linewidth='0.5', color='green')
plt.grid(which='minor', linestyle=':',linewidth='0.5', color='black')
plt.show()
plt.scatter(X2_test,y2_test,color='gray')
plt.plot(X2_test,y2_pred,color='violet',linewidth=2)
plt.show()
print('Mean abolute error is:', metrics.mean_absolute_error(y2_test,y2_pred))
```

```
print('Mean squared error is:', metrics.mean_squared_error(y2_test,y2_pred))
print('Root mean squared error is:', np.sqrt(metrics.mean_squared_error(y2_test,y2_pred)))
```

**6.7 Linear regression for Humidity**

```
model1 =LinearRegression()
model1.fit(X1_train,y1_train)

print('Intercept is :',model1.intercept_)
print('Coefficient is :' ,model1.coef_)
y1_pred= model1.predict(X1_test)
df2=  pd.DataFrame({'Actual': y1_test.flatten(), 'Predicted': y1_pred.flatten()})
print(df2)
df3= df2.head(20)
df2.plot(kind='bar', figsize=(16,10))
plt.grid(which='major', linestyle='-',linewidth='0.5', color='green')
plt.grid(which='minor', linestyle=':',linewidth='0.5', color='black')
plt.show()
plt.scatter(X1_test,y1_test,color='gray')
plt.plot(X1_test,y1_pred,color='yellow',linewidth=2)
plt.show()
print('Mean abolute error is:', metrics.mean_absolute_error(y1_test,y1_pred))
print('Mean squared error is:', metrics.mean_squared_error(y1_test,y1_pred))
print('Root mean squared error is:', np.sqrt(metrics.mean_squared_error(y1_test,y1_pred)))
```

**6.8 Support vector regression for temperature**

```
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
dataset = pd.read_csv("weather1yeardata.csv")
dataset.head()

dataset.head()
dataset.dtypes
ds  = dataset.drop(columns = 'temparature9am')
X = ds.iloc[::].values

data = pd.read_csv("weather1yeardata.csv")
data.head()
Y = data.iloc[:,1].values
print(Y)
X1=X[:,[2]]
```

```
from sklearn.svm import SVR
regressor = SVR(kernel = 'rbf')
regressor.fit(X1, Y))

# Predicting a new result
y_pred = regressor.predict(X1)
print(y_pred)
from sklearn.metrics import mean_squared_error
mse = mean_squared_error(Y, y_pred)
print(mse)
```

**6.9 Random forest for temperature**

```
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd

dataset = pd.read_csv("weather1yeardata.csv")
dataset.head()
dataset.head()
dataset.dtypes
ds  = dataset.drop(columns = 'temparature9am')
X = ds.iloc[::].values
X1=X[:,[2]]
data = pd.read_csv("weather1yeardata.csv")
data.head()
Y = data.iloc[:,1].values
print(Y)
# Fitting Random Forest Regression to the dataset
from sklearn.ensemble import RandomForestRegressor
regressor = RandomForestRegressor(n_estimators = 500, random_state = 0) #n_estimators : No. of Decision Trees
regressor.fit(X1, Y)

# More number of Decision Tress MAY result into a more accurate prediction

# Predicting a new result
y_pred = regressor.predict(X1)
print(y_pred)

# Visualising the Random Forest Regression results (higher resolution)
#X_grid = np.arange(min(X1), max(X1), 0.01)
#X_grid = X_grid.reshape((len(X_grid), 1))
#plt.scatter(X1, Y, color = 'red')
```

```
#plt.plot(X_grid, regressor.predict(X_grid), color = 'blue')
#plt.title('Random Forest Regression')
#plt.xlabel('storage temparature')
#plt.ylabel('cpu temp')
#plt.show()
from sklearn.metrics import mean_squared_error
mse = mean_squared_error(Y, y_pred)
print(mse)
```

**Chapter 7: References**


[1] Gaurav Verma,Pranjul Mittal,Shaista Farheen "Real Time Weather Prediction System Using IOT and Machine Learning", September 08,2020 at 05:31:31 UTC from IEEE Xplore.

[2] Zi-Qi Huang 1 , Ying-Chih Chen 2 and Chih-Yu Wen 1, "Real-Time Weather Monitoring and Prediction Using City Buses and Machine Learning", 4 August 2020; Accepted: 8 September 2020; Published: 10 September 2020

[3] Palak Kapoor,Ferdous Ahmed Barbhuiya, "Cloud Based Weather Station using IoT Devices", 2019 IEEE Region 10 Conference (TENCON 2019)

[4] Anubha Parashar, "IoT Based Automated Weather Report Generation and Prediction Using Machine Learning ", 2019 2nd International Conference on Intelligent Communication and Computational Techniques (ICCT) Manipal University Jaipur, Sep 28-29, 2019

[5] Mr. Sunil Navadia,Mr. Jobin Thomas,Mr. Pintukumar Yadav,Ms. Shakila Shaikh "Weather Prediction: A novel approach for measuring and analyzing weather data", International conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud) (I-SMAC 2017)

[6] Shubham Madan1 , Praveen Kumar2 , Seema Rawat3 , Tanupriya Choudhury4 Amity University Noida, Uttar Pradesh, India1,2,3, UPES Dehradun4 "Analysis of Weather Prediction using Machine Learning & Big Data ", 2018 International Conference on Advances in Computing and Communication Engineering (ICACCE-2018) Paris, France 22-23 June 2018

[7]Ayham Omary,Ahmad Wedyan,Ahmed Zghoul, Ahmad Banihani, and Izzat Alsmadi "An Interactive Predictive System for Weather Forecasting", 2019 2nd International Conference on Intelligent Communication and Computational Techniques (ICCT) Manipal University Jaipur, Sep 28-29, 2019

[8]Girija C, Andreanna Grace Shires, S "Internet of Things (IOT) based Weather Monitoring System"in International Journal of Engineering Research & Technology (IJERT).

[9]Bulipe Srinivas Rao, Prof. Dr. K. Srinivasa Rao, Mr. N. Ome " Internet of Things (IOT) Based Weather Monitoring system" in International Journal of Advanced Research in Computer and Communication Engineering Vol. 5, Iss 9, 2016.

[10]Prof. S.B. Kamble, P.Ramana P. Rao, Anurag S. Pingalkar, Ganesh S. Chayal "IoT Based Weather Monitoring System" IJARIIE ISSN(O)-2395-4396

[11]R. K. Kodali and K. S. Mahesh, "A low-cost implementation of MQTT using ESP8266, "2nd International Conference on Contemporary Computing and Informatics (IC3I), pp. 404-408, Noida