A

IV Semester

MINOR PROJECT REPORT

ON

"Machine Learning Based Occupancy Estimation."

BY

Kutikuppala Abhilash
(191000024)
N. Venkata Ramana
(191000026)
V.V. Subash Chandra
(191000055)

UNDER THE GUIDANCE OF
Dr. Venkanna U.

DEPARTMENT of COMPUTER SCIENCE AND ENGINEERING
Dr. SPM INTERNATIONAL INSTITUTE OF INFORMATION TECHNOLOGY NAYA RAIPURATAL NAGAR - 493661, INDIA

May 8, 202

Declaration

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

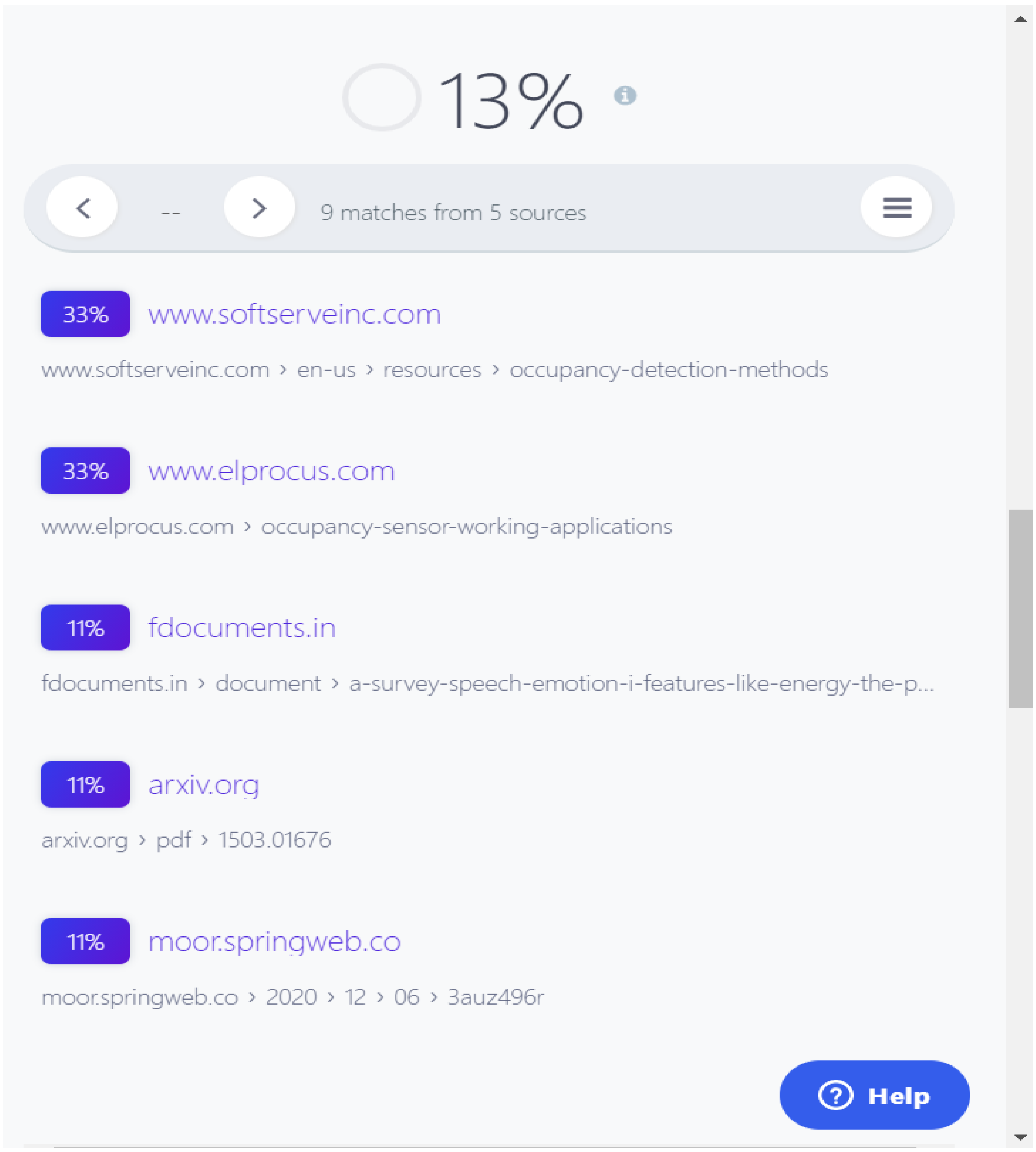
Kutikuppala Abhilash (191000024)

N. Venkata Ramana (191000026)

V.V. Subash Chandra(191000055)

Date :8th May 2021

Plagiarism Report



13%

-- 9 matches from 5 sources

33% www.softserveinc.com

www.softserveinc.com › en-us › resources › occupancy-detection-methods

33% www.elprocus.com

www.elprocus.com › occupancy-sensor-working-applications

11% fdocuments.in

fdocuments.in › document › a-survey-speech-emotion-i-features-like-energy-the-p...

11% arxiv.org

arxiv.org › pdf › 1503.01676

11% moor.springweb.co

moor.springweb.co › 2020 › 12 › 06 › 3auz496r

Certificate

This is to certify that the project titled “Machine Learning Based Occupancy Estimation ” by “Abhilash, subash chandra and Venkata ramana” has been carried out under my supervision and that this work has not been submitted elsewhere for a degree/diploma.

Dated: 8th May, 2021

Dr. (Venkanna U) Assistant
ProfessorDepartment of CSE
DSPM IIIT Naya Raipur-493661

Acknowledgments

At the outset, we would like to express my whole hearted and deep sense of gratitude to my guide Dr.Venkanna U for his guidance, help and encouragement throughout our research work. We greatly admire his attitude towards research, creative thinking, hard work and dedication in work. We are highly grateful to him for patiently checking all my manuscripts and thesis. This thesis would not have been possible without his bounteous efforts. More than guide, he is our mentor for shaping our personal and professional life, without whom we would not have been where we are today. We owe our profound gratitude to Dr.Venkanna U for his supports in all respects.

Kutikuppala Abhilash (191000024)

N. Venkata Ramana (191000026)

V.V. Subash Chandra(191000055)

ABSTRACT

Now-a-days buildings are growing rapidly and uses large amount of energy. Most of the energy gets wasted on HVAC systems. Occupancy (presence and number of occupants) is one of the most important factors impacting energy efficiency of HVAC systems. Hence, knowing occupancy information is vital for demand driven HVAC controls, that directly impacts on energy-related building control systems. So, occupancy information without privacy issues (like using cameras) gained importance. Recent works are done on this problem. Existing solutions proposed are yes or no classification, estimating for low count etc using sensors data. Using yes or no classification, the energy consumption may not be saved to a great extent. In this work, we have trained a model which classifies the state of room based on number of occupants (empty, low, fair, high). For this we are using data collected by different sensors (co2, humidity, temperature). We are using classification algorithms to train the model. This information could be used to solve energy related problems in buildings.

Contents

1 Introduction **1**

1.1 Major Contributions of the project 1

1.2 Problem Statement 2

2 Literature Survey **3**

2.1 Existing Works 3-4

3 Proposed Solution **6**

4 Experiments **9**

4.1 Software and Hardware Requirements 9

4.2 Implementation 10

4.2.1 Phase I implementation 10

4.2.2 Phase II Implementation 11

4.2.3 Phase III Implementation 12

4.2.4 Phase IV Implementation 13

5 Results and Analysis **14**

6 Practical Applications **17**

7 Conclusion and Future Scope **18**

References **19**

List of Figures

1. **Four Phases of proposed solution……………………………………..6**
2. **Block Diagram……………………………………………………….8**
3. **Sensor Setup ………………………………………………………...10**
4. **Data pushed into cloud…………………………………………...11**
   4.1. Co2 vs time data…………………………………………........11
   4.2. Temperature vs time data……………………………………….11
   4.3. Humidity vs time data……………………………………………12
   4.4. Occupancy vs time data………………………………………….12
5. **A Sample Data from Dataset…………………………………….13**
6. **Machine Learning Algorithms Used ……………………………13**
7. **Confusion matrix of KNN………………………………………..15**
8. **Deployment of model……………………………………………..15**
9. **Data readings of sensor…………………………………………..16**

List of Tables

1. **Summary of Existing works……………………………………………..5**
2. **Summary of performance metrics of Algorithms………………………...14**

Chapter 1 Introduction

Real-time occupancy data will cause intelligent heating, ventilation, and air conditioning (HVAC) and lighting systems in buildings, which could not only save energy but also improve occupant comfort. With the introduction of the online of Things (IoT), there are now readily accessible sensors which can quantify environmental parameters, and this data are often analyzed using machine learning (ML) to assess human occupancy without the use of video-based systems. Recent studies have shown that buildings with established occupancy patterns can save 30% electricity.[14]

In the early stages of occupancy detection and estimation, invasive systems like cameras [13], WiFi[, wearables, and RFID were used. With growing privacy concerns, researchers have turned their attention to the utilization of non-intrusive environmental sensors for occupancy detection and estimation, like CO2, temperature, light, motion, and humidity. We used subsequent three readily available low-cost sensors for occupancy estimation because the strain of this project is additionally on the utilization of non-intrusive sensors: CO2, temperature, humidity.

So occupancy estimation is that the quantitative sensing of humans present in spaces. Real- time analysis of occupancy can decrease energy consumption through better allocation of building resources. Real-time occupancy information could also be a serious input for automating heating, ventilation and air conditioning (HVAC) and lighting systems in buildings which could not only conserve energy, but also provide better comfort to the occupants. With the arrival of Internet of Things (IoT), there are readily available sensors which can measure the environmental parameters and this data are often analyzed using machine learning (ML) to figure out human occupancy without video based systems. Past research shows that having smart building management with occupancy, can reduce building energy consumption by 30 – 50%.

Regarding the occupancy estimation, several works have been carried out based on one or more environmental variables [7,8,9,10]. However, few works that did not use non-environmental variables as additional support for the estimation were found. On the one hand, there are works, such as Adeogun et al. [10] used pressure, humidity, and CO2, in addition to a Passive Infrared (PIR) sensor. A second example is the work of Chitu et al. [21], which in addition to using CO2, also registers the state of all the airflow sources, obtaining an accuracy of 0.69. On the other hand, among the works that only use environmental variables, Jiang et al. [22] and Zhou et al. [9] were found. Both used CO2 to estimate occupancy, obtaining an accuracy of 0.77 and 0.82, respectively. Another example is the work of Viani et al [8], which obtained an accuracy of 0.82 using temperature, humidity, and CO2. It is relevant to mention that there are no research works based only on environmental variables that do not use CO2. Furthermore, none of the studies based only on environmental variables presents an accuracy similar to that obtained by works using environmental and non-environmental variables, such as Adeogun et al [10].

1.1 Major Contributions of the project

In this paper, we aim to estimate the occupancy level during a room by using multiple heterogeneous sensor nodes with various ML techniques like K-Nearest Neighbours(KNN), SVM (Linear), random forest (RF), Decision trees(DT). especially, our contributions are as follows:

- There are some features which show a change when occupancy count changes like co2, temperature etc. So, we are using data of those features to coach the model. We are using data collected by co2, temperature, humidity sensors. and that we have classified the state of room into 4 types.
- Data preprocessing techniques to switch the data into well organized manner.
- The performance comparison in terms of estimation accuracy and F1 score is noted for various ML algorithms.

1.2 Problem Statement

To accurately estimate the number of occupants in a room using non-intrusive sensors to design power efficient buildings.

The problem that this project is trying to solve is whether it is possible to design a sensor system that keeps track of occupancy in an room and use this information as input to context-aware systems. This problem is interesting because the occupancy information collected, can in turn be used for a variety of smart services. These services can aid in environmental control, energy management, increasing work efficiency, and security applications. A simple motion sensor with a timing device can detect that a room is populated. However, it requires people to move around to keep the system active once the timer has expired. A context-aware sensor system, on the other hand, will keep any environmental control applications up to date with the correct occupancy information, thus keeping the environmental system running.

Some examples of these can be adapting the ventilation system to the number of people in a room or redirecting phone calls if there is more than one person in a meeting room. Lights and equipment can be turned off as soon as an area is not occupied. Depending on the utilization and habits of the occupants, energy savings between ten to fifty percent can be made

Chapter 2 Literature Survey

A lot of research has been administered within the literature for occupancy detection, i.e., if the space is occupied or not. Although detection can help in improving energy savings, estimating the precise number of occupants can make the system even more energy- efficient. a typical implementation is to use RFID tags, as seen during a paper by Li et al in 2012 [5]. during this paper, the researchers have readers and reference tags placed around a neighborhood a tracking tag placed on any human within the world. A notable paper by Abade et al.[3] places multiple sensor nodes that detect noise to count the quantity of humans present. While these designs work for one room, they're too difficult to implement in large settings like office space buildings. Another pattern of labor is based upon computer vision concepts. Companies like VergeSense [4] install a camera per room to detect occupancy. While such an implementation limits hardware to a minimum of one device per room, it still requires meticulous setup and raises privacy concerns. In [10], three ML techniques namely Hidden Markov model (HMM), artificial neural network (ANN) and support vector machine (SVM) were used on a distributed sensor network. it had been shown that HMM gives the simplest performance with 75% accuracy.
In this paper, we aim to estimate the number of occupants in a room by using heterogeneous sensor nodes with various ML techniques like SVM, random forest(RF), decision trees(DT).

2.1 Existing Solutions

IoT-based occupancy monitoring techniques for energy-efficient smart buildings

With the widespread use of Internet of Things (IoT) devices like smartphones, sensors, cameras, and RFIDs, it is now possible to collect vast amounts of data for people monitoring and localization inside commercial buildings. There are numerous opportunities for improving the energy consumption of buildings through smart HVAC systems, which are enabled by such occupancy monitoring capabilities. The major challenges we see in this regard are achieving occupancy monitoring in a minimally invasive manner, such as utilising existing building infrastructure and not requiring the installation of any software on users' smart devices

Occupancy detection from electricity consumption data

The ability to detect when a home is occupied by its residents is needed for a variety of home automation applications. Dedicated sensors, such as passive infrared sensors, magnetic reed switches, or cameras, are typically required in current occupancy detection systems. They studied the suitability of automated electricity metres – which are already in millions of homes around the world – to be used as occupancy sensors using this method.

A Machine Learning Approach to Indoor Occupancy Detection Using Non-Intrusive Environmental Sensor Data

The performance of a variety of machine learning algorithms applied to non-intrusive environmental sensor data (temperature and humidity) to infer human occupancy in closed office spaces is investigated and discussed in this paper. This work serves as a basis for both researchers and practitioners to gain a better understanding of each algorithm's efficiency and output in terms of accuracy, precision, and other metrics. It also goes into how to handle and prepare time series data in the form of office occupancy detection.

Ensemble-based extreme learning machine model for occupancy detection with ambient attributes

This paper proposes a more effective, accurate, and efficient occupancy detection model based on statistical analysis of sensor-based data. A detailed quantification of the relationship between ambient attributes is provided, and an ensemble model based on machine learning technique extreme learning machine is designed to achieve a substantial level of accuracy and efficiency improvement. Furthermore, the paper proposes an online and adaptive model-based online sequential extreme learning machine for performing occupancy detection on real-time data when full data is unavailable and learning is performed with recent data points in the form of streams.

2.2 Summary Of Existing Works

Solution	**Methodology**	**Drawbacks**
IoT-based occupancy monitoring techniques for energy-efficient smart buildings (2015)	Most building entrances and infrastructure objects have video cameras, these devices and others help foster occupancy detection.	• The quality depends on light conditions. • Easily used for user identification or privacy violation
Occupancy detection from electricity consumption data (2013)	A home's pattern of electricity usage generally changes when occupants are present due to their interaction with electrical loads.	• Insensitive when people do not use electricity or some other goods
A Machine Learning Approach to Indoor Occupancy Detection Using Non-Intrusive Environmental Sensor Data(2019)	Uses temperature and humidity data collected from sensors and applying machine learning algorithms .	• Accuracy is low and will get effected when room space is increased.
Ensemble-based extreme learning machine model for occupancy detection with ambient attributes(2020)	Used temperature, humidity, hum. ratio, CO2, light to detect occupancy.	• detection alone can help in improving energy savings, estimating the precise number of occupants can make the system even more energy-efficient.

Table – 1: Summary of Existing works

Chapter 3

Proposed Solution

To build our system we first decided upon the sensors we might use. A 2013 study by Yang et al. [1] investigated multiple off-the-shelf sensors and mapped the correlation between their measurements and human presence. They demonstrated that, a well-placed light sensor, CO2, temperature, and humidity showed most variance compared to multi-human occupancy. We decided to use the latter three because the light sensor's performance was dependent upon specific placement, which might have limited deployment ease. We are Estimating for comparatively large occupants and dividing it to 4 groups:

Group	-	Range
Empty (0)	-	0
Low (1)	-	1, 2
Fair (2)	-	3, 4
High (3)	-	5, 6

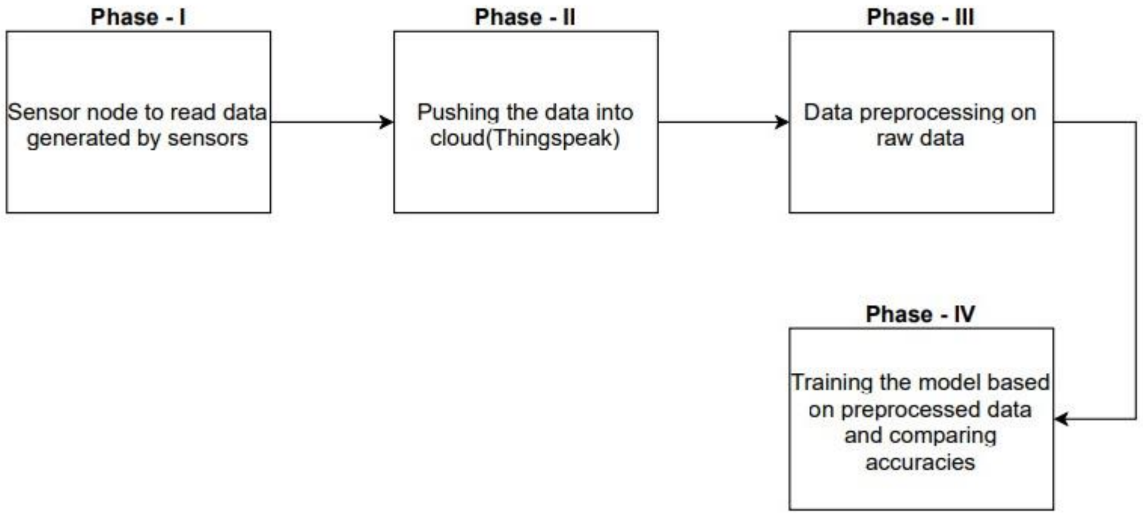
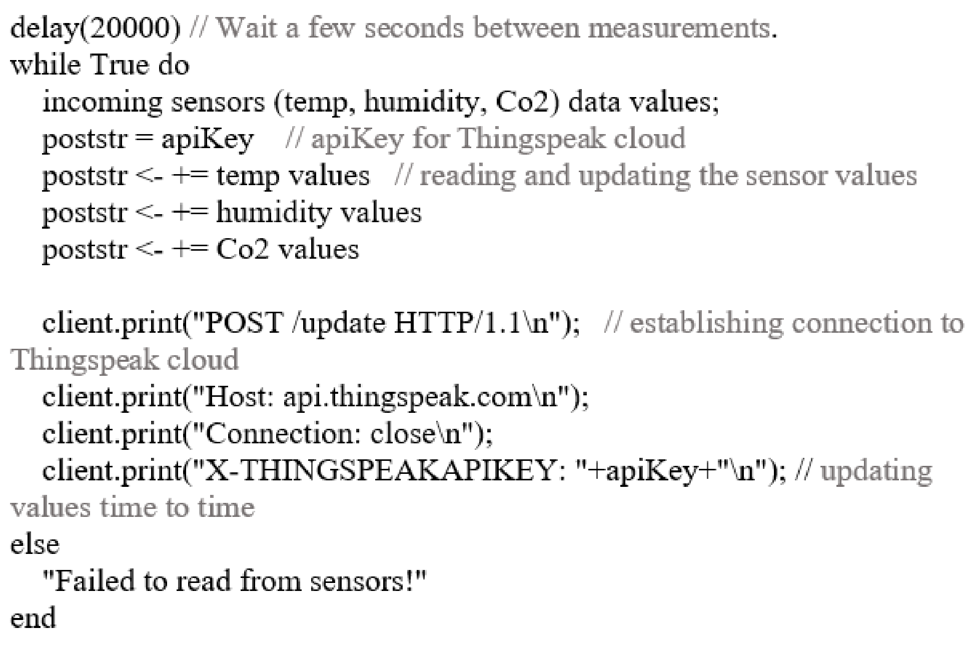

Phase - I
Sensor node to read data generated by sensors
Phase - II
Pushing the data into cloud(Thingspeak)
Phase - III
Data preprocessing on raw data
Phase - IV
Training the model based on preprocessed data and comparing accuracies

Figure – 1: Four Phases of proposed solution

Algorithm for reading sensor data and pushing those to cloud



```
delay(20000) // Wait a few seconds between measurements.
while True do
   incoming sensors (temp, humidity, Co2) data values;
   poststr = apiKey    // apiKey for Thingspeak cloud
   poststr <- += temp values   // reading and updating the sensor values
   poststr <- += humidity values
   poststr <- += Co2 values

   client.print("POST /update HTTP/1.1\n");   // establishing connection to
Thingspeak cloud
   client.print("Host: api.thingspeak.com\n");
   client.print("Connection: close\n");
   client.print("X-THINGSPEAKAPIKEY: "+apiKey+"\n"); // updating
values time to time
else
   "Failed to read from sensors!"
end
```

Algorithm:

- A delay of 200 seconds is maintained to read the sensor data values
- When the condition holds true:
- The data values of various sensors are collected and kept being updated for every 200 seconds.
- And then a connection is established to cloud (api.thingspeak.com)
- when the connection is not interrupted, the sensor values are pushed to cloud with a delay of 15 seconds
- The values are then shown in the form of seperate fields (named as field1, field2...)
- When the condition holds true:
- "Failed to read from sensors" is displayed with aa warning note that somewhere the connection has lost

BLOCK DIAGRAM:

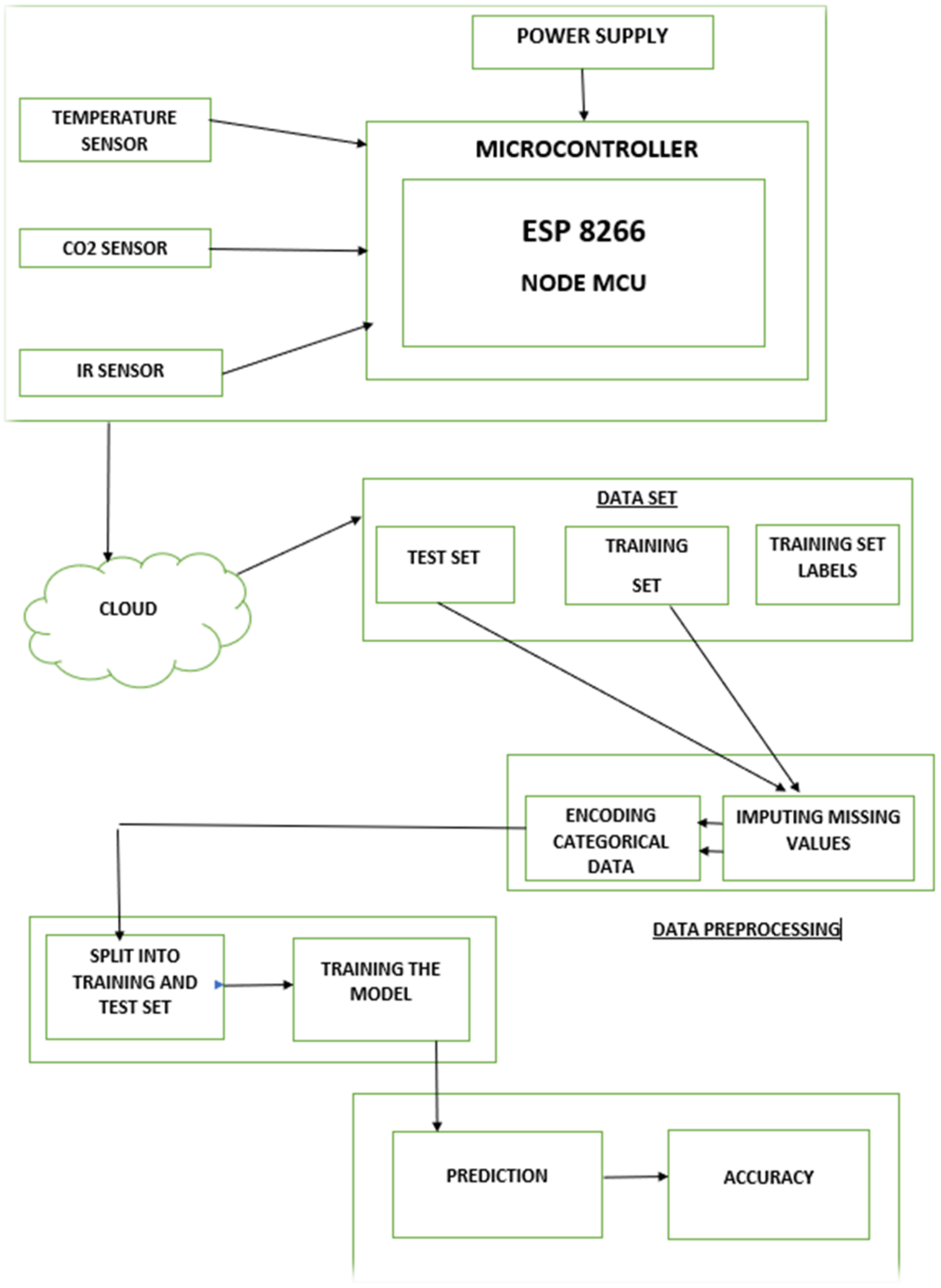


Figure – 2: Block Diagram

Chapter 4

4.1 Software Requirements

1. Machine Learning Algorithms:

Decision Tree: It belongs to supervised learning algorithm. Decision tree can be usedto classification and regression both having a tree like structure. In a decision tree building algorithm first the best attribute of dataset is placed at the root, then training dataset is split into subsets. Splitting of data depends on the features of datasets. This process is done until the whole data is classified and we find leaf node at each branch. Information gain can be calculated to find which feature is giving us the highest information gain. Decision trees are built for making a training model which can be usedto predict class or the value of target variable.

KNN: This approach can be used for classification as well as regression. It is one of the most basic machine learning algorithms. It saves the cases and reviews the majority of the k neighbours with which it resembles the most when it receives new data. KNN renders predictions directly from the training dataset.

SVM: Each n-dimensional feature vector is basically a point in an n-dimensional feature space, where n is the number of features present in the dataset. Each vector belongs toone of the k classes. Unlike LDA and QDA, SVM does not make any assumptions about the data. The algorithm attempts to fit an optimal hyperplane between the two classeswith the help of support vectors. Therefore, for k classes, the number of classifiers learned by the algorithm are k(k−1)/2 which are put to a majority vote. A non-linear separation boundary can be obtained by using a kernel. SVM has a tunable penalty hyperparameter for indicating the tolerable error or misclassifications in fitting the hyperplane.

Random Forest: It is a supervised classification algorithm. Multiple number of decision trees taken together forms a random forest algorithm i.e the collection of many classification tree. It can be used for classification as well as regression. Each decision tree includes some rule based system. For the given training dataset with targets and features,the decision tree algorithm will have set of rules. In random forest unlike decision treesthere is no need to calculate information gain to find root node. It uses the rules of each randomly created decision tree to predict the outcome and stores the predicted outcome. Further it calculates the vote for each predicted target. Thus, high voted prediction is considered as the final prediction from the random forest algorithm.

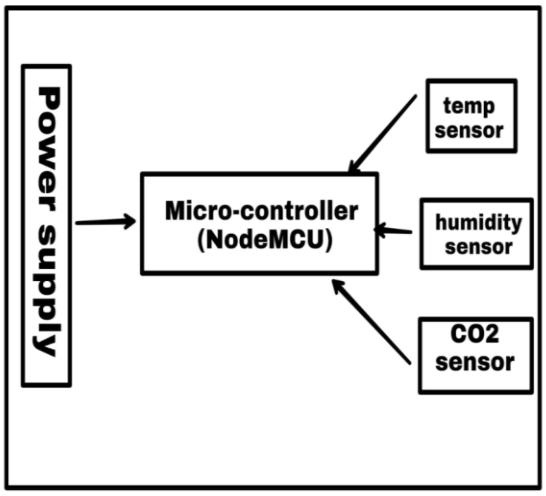
Naive-Bayes: It is a technique for constructing classifiers which is based on Bayes theorem used even for highly sophisticated classification methods. It learns the probability of an object with certain features belonging to a particular group or class. In short, it is a probabilistic classifier. In this method occurrence of each feature is independent of occurrence another feature. It only needs small amount of training datafor classification, and all terms can be precomputed thus classifying becomes easy, quick.

4.2 Implementation

Phase-1(Setting up Sensor node):

To build our system we first decided upon the sensors we would use. A 2013 study by Yang et al. investigated multiple sensors and mapped the correlation between their measurements and human presence. They demonstrated that, in order, a well-placed light sensor, CO_2, temperature, and humidity showed most variance compared to multi-human occupancy. We decided to use the latter three because the light sensor's effectiveness was dependent upon specific placement, which would have limited deployment ease.

Our system prototype consisted of a NodeMCU. We integrated a *SEN0159* CO_2 sensor using a along with an *DHT22 Temp+Humidity* sensor. Given that our raw data would be 4-dimensional (time, CO_2, temperature, humidity). we've deployed our prototype during a medium sized room. the info for our machine learning structure was collected during a room over 3000 minutes. Above mentioned sensors give accurate measurement upto 6 decimals.



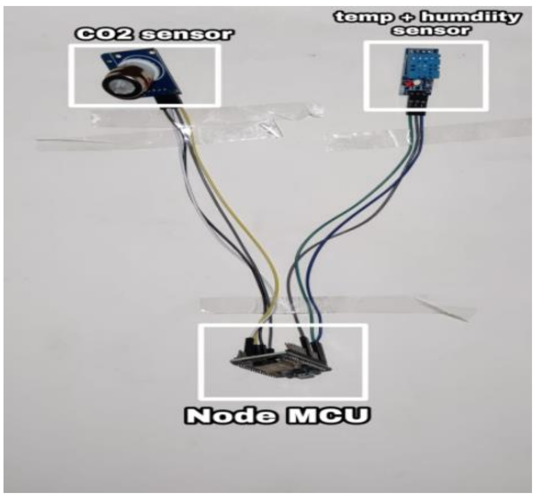


Figure – 3: Sensor Node Setup

Phase-2(Pushing data into cloud):

ThingSpeak is an Open-Source IoT application and API to store and retrieve data from Hardware devices and Sensors. It uses HTTP Protocol over the Internet or LAN for its communication. The MATLAB analytics is included to analyze and visualize the data received from your Hardware or Sensor Devices.

The Read API key allows you to read data from a private channel. You can find the Read

API key for a channel on the API Keys tab of your ThingSpeak channel view. If you are reading data from a public channel, you do not need a Read API key. Save your channel Read API key in a variable for convenience.

We pushed the data collected using the sensor node made in the above phase into cloud. And retrieved data into a csv file.

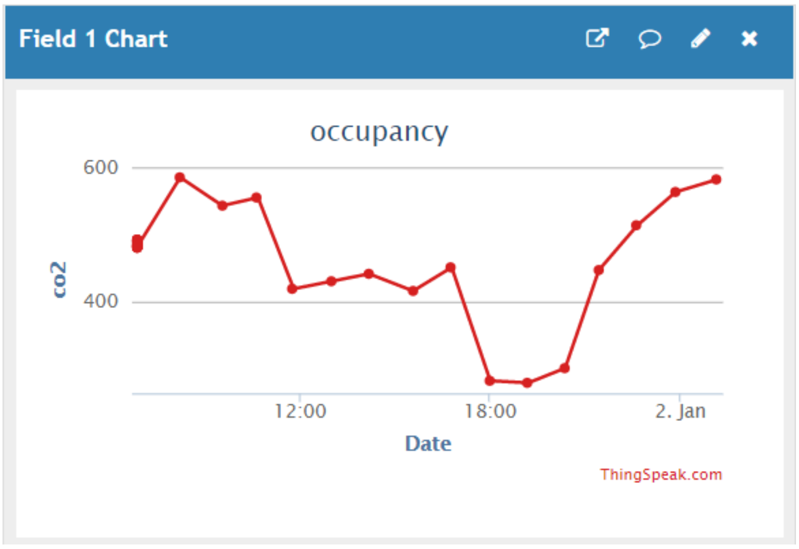


Figure 4.1 Co2 vs time data

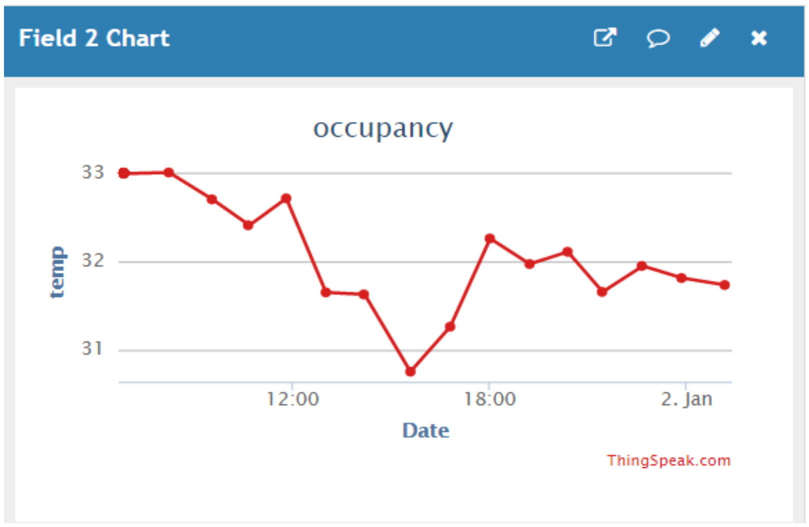


Figure 4.2 temperature vs time data

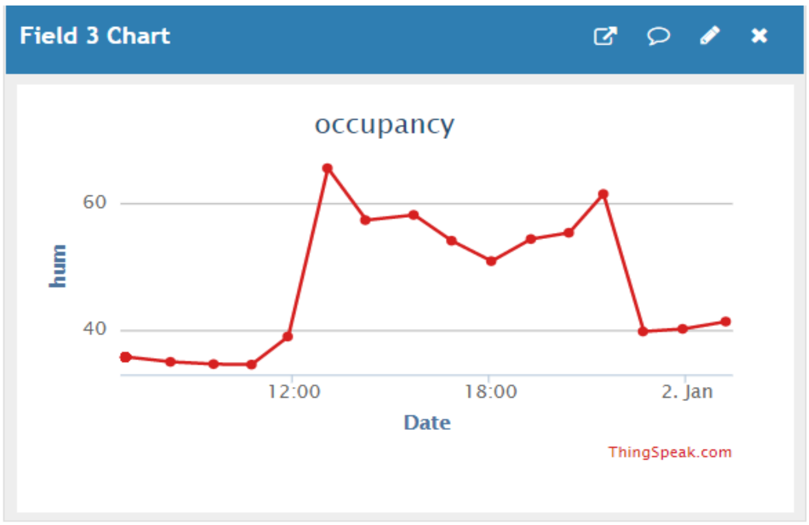


Figure 4.3 humidity vs time data

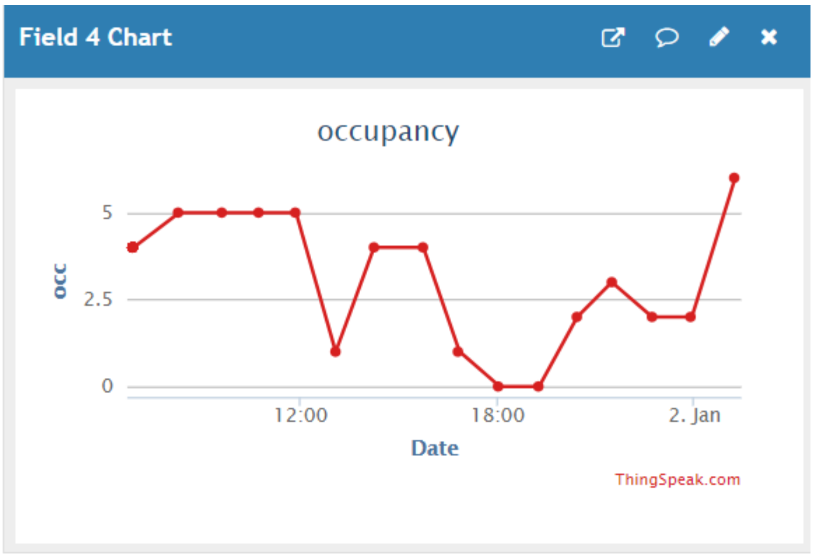
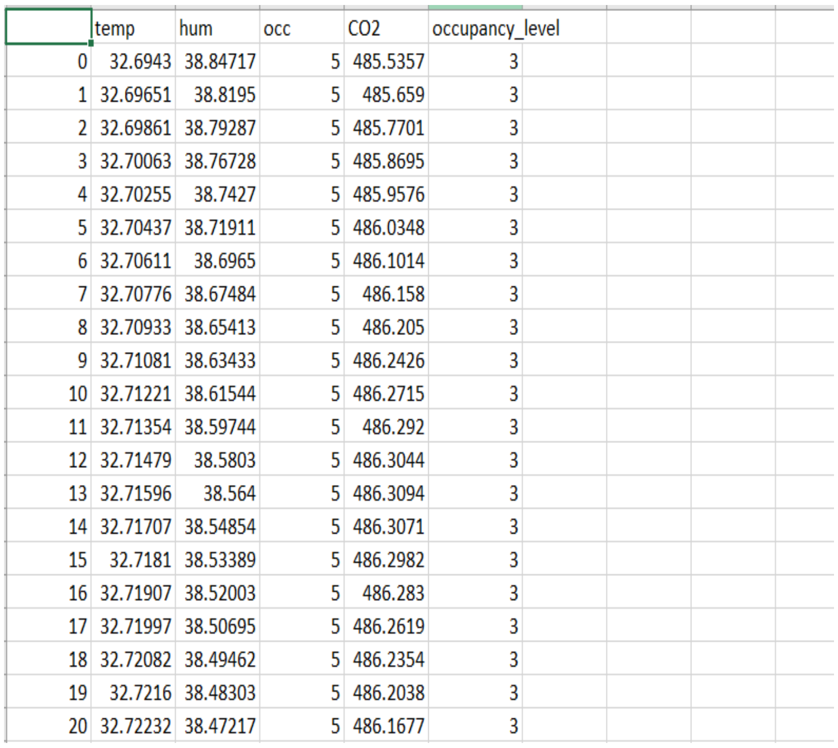


Figure 4.4 number of occupants vs time

Phase-3(Data preprocessing):

All sensor nodes did not send their data at the same time. A few seconds of variation was present between the arrival times of data of sensors.We merged the time-stamps within a given time interval into a common stamp. We have collected data for each 30 seconds and have over 7000 instances. Rows with missing data were deleted. For occupants count, the inhabitants in the home kept a personal track of every time they entered and left the living room. After the training data was collected, these logs and saved images were manually studied to count the number of humans present in the living room for each sensor data entry. The ground truth of occupancy count was appended. We have appended a new occupancy level column based on number of occupants dividing it into empty(0), low(1), fair(2), high(3).



	temp	hum	occ	CO2	occupancy_level
0	32.6943	38.84717	5	485.5357	3
1	32.69651	38.8195	5	485.659	3
2	32.69861	38.79287	5	485.7701	3
3	32.70063	38.76728	5	485.8695	3
4	32.70255	38.7427	5	485.9576	3
5	32.70437	38.71911	5	486.0348	3
6	32.70611	38.6965	5	486.1014	3
7	32.70776	38.67484	5	486.158	3
8	32.70933	38.65413	5	486.205	3
9	32.71081	38.63433	5	486.2426	3
10	32.71221	38.61544	5	486.2715	3
11	32.71354	38.59744	5	486.292	3
12	32.71479	38.5803	5	486.3044	3
13	32.71596	38.564	5	486.3094	3
14	32.71707	38.54854	5	486.3071	3
15	32.7181	38.53389	5	486.2982	3
16	32.71907	38.52003	5	486.283	3
17	32.71997	38.50695	5	486.2619	3
18	32.72082	38.49462	5	486.2354	3
19	32.7216	38.48303	5	486.2038	3
20	32.72232	38.47217	5	486.1677	3

Figure – 5: A Sample Data From Dataset

Phase-4(Applying Machine Learning Algorithms):

As we are classifying the occupancy state into 4 groups, multi-class classification algorithms are best suited to train the data. We are using these algorithms.

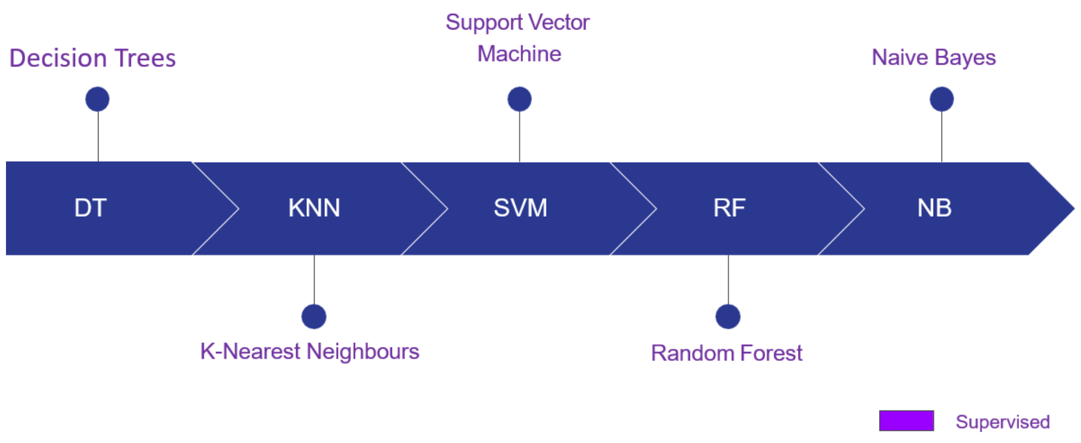


Figure – 6: Machine Learning Algorithms Used

Chapter 5

Results and Analysis

The ML algorithms discussed within the previous section were implemented using Scikit-learn [6]. Metrics like accuracy, F1 score and confusion matrix were calculated. Since the info is of time-series nature, data wasn't Randomised before cross validation to avoid data points almost like test data stepping into the training data. needless to say, the entire dataset with all the sensors, performs the best at estimating the level of occupants accurately. KNN performed well among all. We got accuracy of 87.12% and f1-score (86.8%). Except Naïve Bayes, remaining algorithms got an accuracy above 80%.

Performance parameters

Algorithm	Accuracy	F1-score
KNN	87.12	86.86
SVM	80.7	80.2
Decision Trees	84.55	84.4
Random Forest	85.1	84.9
Naive Bayes	60	59

Table – 2: Summary of performance metrics of Algorithms.

Confusion Matrix of Best Performed Algorithm:

K – Nearest Neighbours performed well against all other algorithms and gave an accuracy over 87%. Below is the confusion matrix of KNN.
It describe the performance of a classification model. It shows relation between true values vs predicted values of test dataset. Overally 226 wrong predictions were made out of 1788.
If we take this for Occupancy detection(yes or no), only 48 wrong assumptions were made which gives us about 99.88% accuracy for occupancy detection.

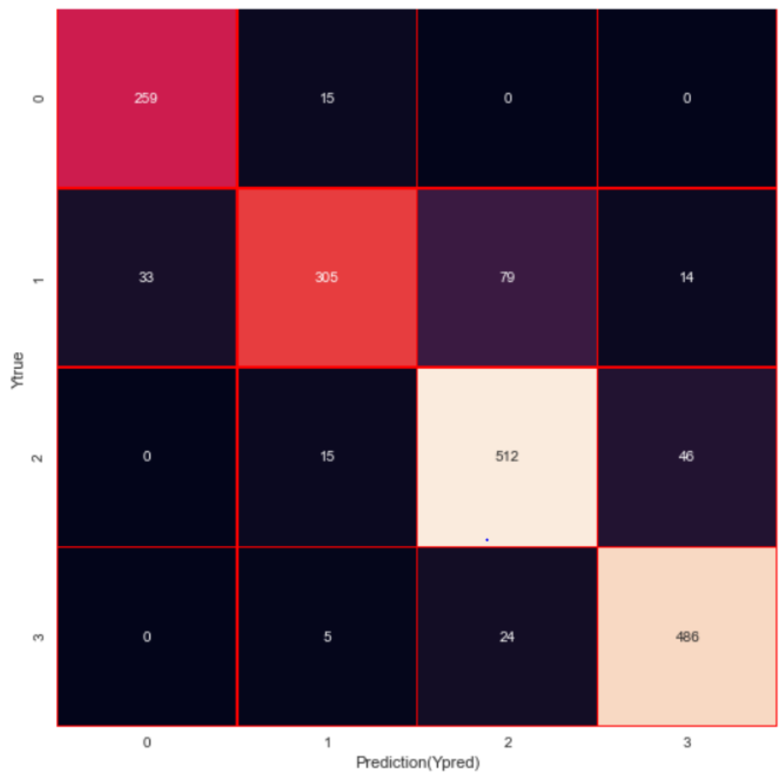


Figure 7-Confusion matrix of knn

The below **figures 8** shows real deployment and collection of data. We have measured temperature, humidity and co2 levels through sensors and Manually noted details of no. of occupants in a room. we have tested both the cases with and without occupants and collected all the readings. At first we started 5 members in a room and started collecting the reading and slowly we have been sending one by one outside and measured reading for 4,3,2,1,0 number of occupants. Main important note was measuring the readings we make sure occupants were there in the room for atleast 5 min so that we can conclude with some average reading for that number of occupants. Our observation in this whole process was that temperature and humidity values will not change that frequently but co2 sensor will show significant change.

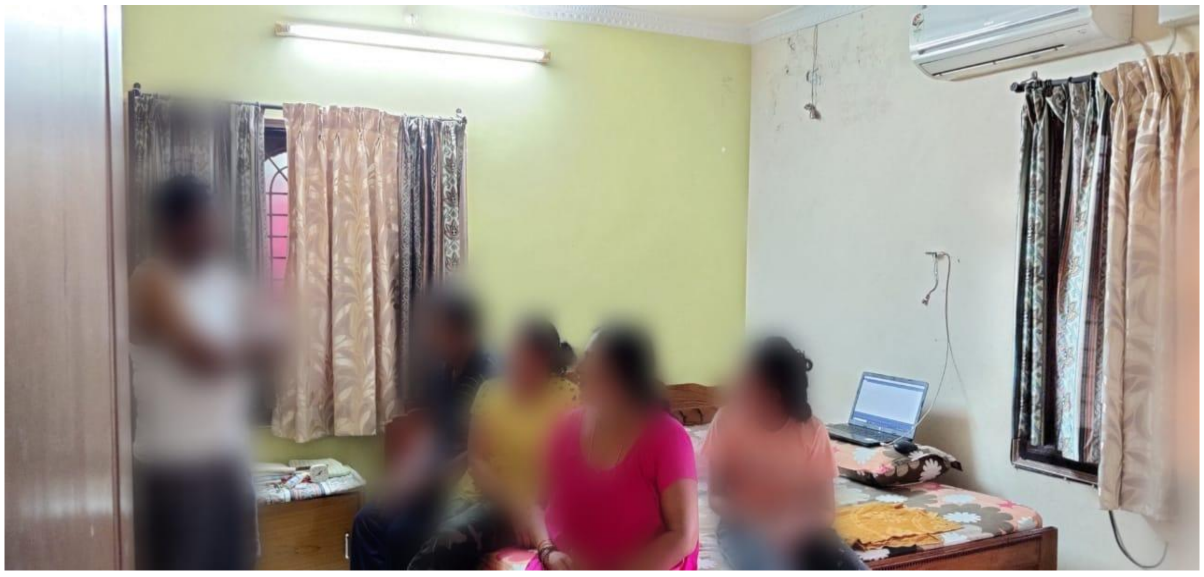
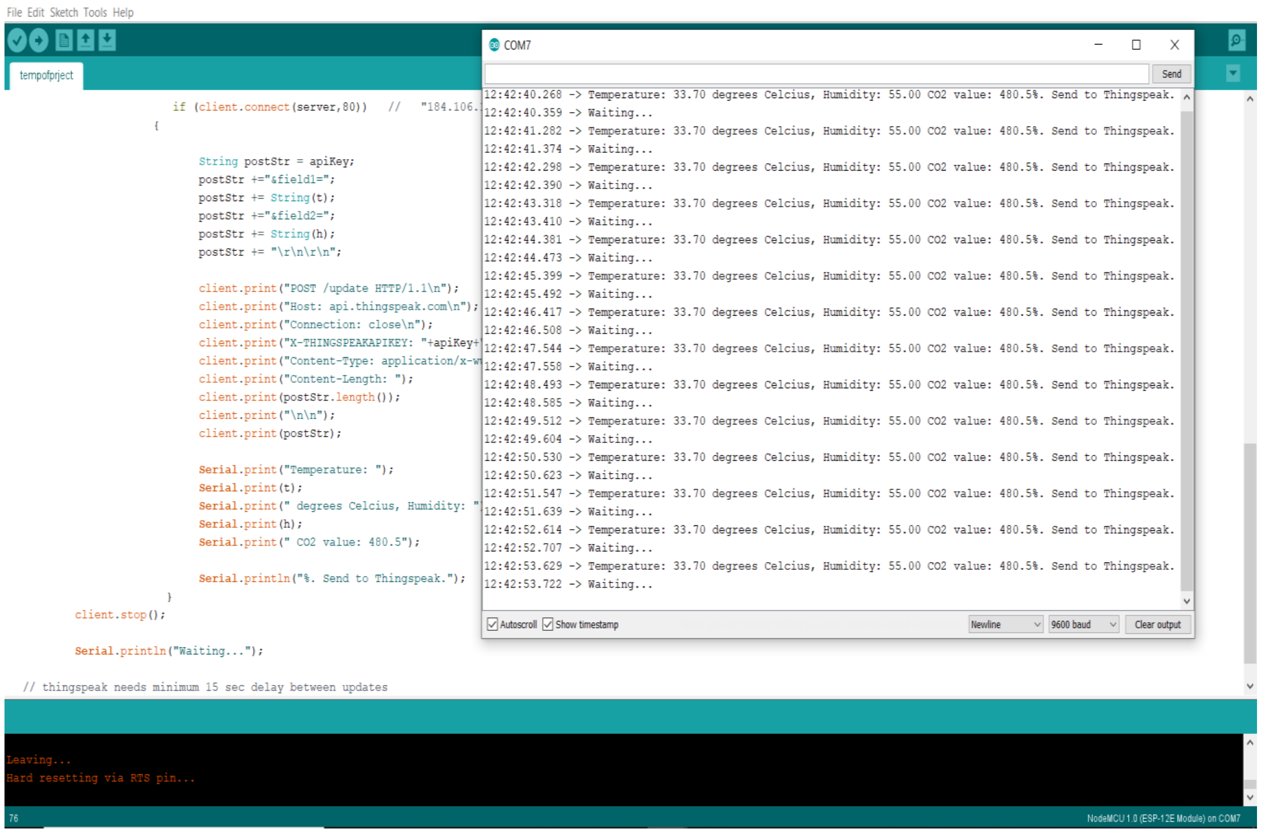


Figure – 8: Deployment of the model

Depending upon the number of occupants, the values are measured from time to time and the same data is reflected in the Thingspeak cloud with a delay of around 10 seconds which is as shown in the below **figure-9**



```
                if (client.connect(server,80))   //   "184.106.
              {

                   String postStr = apiKey;
                   postStr +="&field1=";
                   postStr += String(t);
                   postStr +="&field2=";
                   postStr += String(h);
                   postStr += "\r\n\r\n";

                   client.print("POST /update HTTP/1.1\n");
                   client.print("Host: api.thingspeak.com\n");
                   client.print("Connection: close\n");
                   client.print("X-THINGSPEAKAPIKEY: "+apiKey+
                   client.print("Content-Type: application/x-w
                   client.print("Content-Length: ");
                   client.print(postStr.length());
                   client.print("\n\n");
                   client.print(postStr);

                   Serial.print("Temperature: ");
                   Serial.print(t);
                   Serial.print(" degrees Celcius, Humidity: "
                   Serial.print(h);
                   Serial.print(" CO2 value: 480.5");

                   Serial.println("%. Send to Thingspeak.");
               }
      client.stop();

      Serial.println("Waiting...");

  // thingspeak needs minimum 15 sec delay between updates
```

```
12:42:40.268 -> Temperature: 33.70 degrees Celcius, Humidity: 55.00 CO2 value: 480.5%. Send to Thingspeak.
12:42:40.359 -> Waiting...
12:42:41.282 -> Temperature: 33.70 degrees Celcius, Humidity: 55.00 CO2 value: 480.5%. Send to Thingspeak.
12:42:41.374 -> Waiting...
12:42:42.298 -> Temperature: 33.70 degrees Celcius, Humidity: 55.00 CO2 value: 480.5%. Send to Thingspeak.
12:42:42.390 -> Waiting...
12:42:43.318 -> Temperature: 33.70 degrees Celcius, Humidity: 55.00 CO2 value: 480.5%. Send to Thingspeak.
12:42:43.410 -> Waiting...
12:42:44.381 -> Temperature: 33.70 degrees Celcius, Humidity: 55.00 CO2 value: 480.5%. Send to Thingspeak.
12:42:44.473 -> Waiting...
12:42:45.399 -> Temperature: 33.70 degrees Celcius, Humidity: 55.00 CO2 value: 480.5%. Send to Thingspeak.
12:42:45.492 -> Waiting...
12:42:46.417 -> Temperature: 33.70 degrees Celcius, Humidity: 55.00 CO2 value: 480.5%. Send to Thingspeak.
12:42:46.508 -> Waiting...
12:42:47.544 -> Temperature: 33.70 degrees Celcius, Humidity: 55.00 CO2 value: 480.5%. Send to Thingspeak.
12:42:47.558 -> Waiting...
12:42:48.493 -> Temperature: 33.70 degrees Celcius, Humidity: 55.00 CO2 value: 480.5%. Send to Thingspeak.
12:42:48.585 -> Waiting...
12:42:49.512 -> Temperature: 33.70 degrees Celcius, Humidity: 55.00 CO2 value: 480.5%. Send to Thingspeak.
12:42:49.604 -> Waiting...
12:42:50.530 -> Temperature: 33.70 degrees Celcius, Humidity: 55.00 CO2 value: 480.5%. Send to Thingspeak.
12:42:50.623 -> Waiting...
12:42:51.547 -> Temperature: 33.70 degrees Celcius, Humidity: 55.00 CO2 value: 480.5%. Send to Thingspeak.
12:42:51.639 -> Waiting...
12:42:52.614 -> Temperature: 33.70 degrees Celcius, Humidity: 55.00 CO2 value: 480.5%. Send to Thingspeak.
12:42:52.707 -> Waiting...
12:42:53.629 -> Temperature: 33.70 degrees Celcius, Humidity: 55.00 CO2 value: 480.5%. Send to Thingspeak.
12:42:53.722 -> Waiting...
```

```
Leaving...
Hard resetting via RTS pin...
```

```
76
```

Figure – 9: Data readings of sensor

Dataset construction

The data from all sensor nodes was not sent at the same time. There was a few seconds of variance in the arrival times of sensor data. We created a popular stamp by combining the time stamps within a specified time interval. We have over 7000 instances of data collected every 30 seconds. Rows with missing data were removed from the table. As you can see we noted down the reading with different number of occupants in the room. So according to the readings which we have noted we had written occupants count in the occupants column of the dataset. In this way we had created our dataset.

After constructing the dataset we split the dataset manually into test and training data. Test dataset is used for making the predictions and training data is used for training the Machine Learning model.

Chapter 6

Practical Applications

Since, the proposed solution solves the matter of low-availability, single point of computation, and unnecessary space getting used, it's many applications in world. Following are few real-life applications of the proposed solution:

- These models are frequently used within indoor places for activating or deactivating electric lighting. When space is empty then it can be assumed that there is not moving can be detected. So doesn’t need to turn ON the light. The energy can be conserved by deactivating the lights.
- This solution can also be implemented by large buildings which wastes large amount of energy on HVAC systems. Using this model, helps in HVAC automation and can save energy.
- Several systems are benefited by the occupancy information gathered in buildings, mainly the Heating, Ventilation, and Air Conditioners (HVAC) system.

An example of a safety application can be inside an automated factory environment. The occupancy sensor system can be installed in a factory to monitor unsafe areas, such as places with moving robots unable to detect the presence of humans. If occupancy is detected in such an area, the information can be used to temporarily turn off or modify the behavior of the robots.

Today this is more difficult as key cards are not hung in a key rack in the same fashion. Installing an occupancy sensor system that keeps track of the occupancy in various parts of a hotel and would aid in similar fashion as the key rack. An area where an occupancy sensor system can have greater difficulty working would be in an environment such as a hospital. The difficulty would be due to occupancy changes caused by activities such as the birth of new life.

Chapter 7

Conclusion and Future Scope

While occupancy analytics may be a growing application domain, current sensing solutions are lacking thanks to their inability to scale to realistic environments and encroachment on privacy. This motivated the event of a non-intrusive occupancy estimator that attempts to attenuate scaling costs and maximize data privacy. A prototype was built using an embedded controller which used CO2, temperature and humidity. we got an accuracy of 85%. This paper provided a functional verification of the feasibility of a non-intrusive occupancy estimator, however didn't improve the system and software architecture. This leaves numerous avenues to be taken that might further the event of such a tool . Notably, in terms of the system architecture, more sensors are often carefully examined and added so as to extend the dimensionality of the input data.

The experiments in this work were conducted in a small room. We plan to extend this model to large workspaces in the future. Certain derived features like time and type of day can also be taken into account provided our dataset is large and spans multiple weeks. We also plan to conduct real-time experiments in the near future. In order for the a system to determine the occupancy in a room, it would have to sense the environment, analyze the data (i.e. detecting and counting people), generate an answer, and make the answer available in a suitable format. In other words, it has to detect and count the number of occupants in a room and communicate this to a context broker. In this thesis the occupancy sensor system should send notifications of changes in occupancy indicating that there are zero, one, or many person/persons in a room.

References

[1] Yang, Zheng & Li, Nan & BecerikGerber,Burcin & Orosz, Michael. (2013). Asystematic approach to occupancy modeling in ambient sensor-rich buildings. Simulation. 10.1177/0037549713489918.

[2] King, Jennifer. Smart Buildings: Using Smart Technology to Save Energy in Existing Buildings. American Council for an EnergyEfficient Economy, 2017, SmartBuildings: Using Smart Technology to Save Energy in Existing Buildings.

[3] Abade B, Perez Abreu D, Curado M. A Non Intrusive Approach for Indoor Occupancy Detection in Smart Environments. Sensors (Basel).2018;18(11):3953. Published 2018 Nov 15. doi:10.3390/s18113953

[4] "AI Powered Utilization Analysis" VergeSense, vergesense.com/.

[5] Li N., Calis G., Becerik-Gerber B. Measuring and monitoring occupancy with an RFID based system for demand-driven HVAC operations. Autom. Constr. 2012;24:89–99. doi: 10.1016/j.autcon.2012.02.013.

[6] F. Pedregosa et al., "Scikit-learn: Machine learning in python," J. Mach. Learn. Res., vol. 12, pp. 2825–2830, Nov. 2011.

[7] Yuan, Y.; Li, X.; Liu, Z.; Guan, X. Occupancy estimation in buildings based on infrared array sensors detection. IEEE Sens. J. 2019, 20, 1043–1053.

[8] Viani, F.; Polo, A.; Robol, F.; Oliveri, G.; Rocca, P.; Massa, A. Crowd detection and occupancy estimation through indirect environmental measurements. In Proceedings of the 8th European Conference on Antennas and Propagation (EuCAP 2014), Hague, The Netherlands, 6–11 April 2014

[9] Zhou, Y.; Chen, J.; Yu, Z.J.; Li, J.; Huang, G.; Haghighat, F.; Zhang, G. A novel model based on multi-grained cascade forests with wavelet denoising for indoor occupancy estimation. Build. Environ. 2020, 167, 106461

[10] Adeogun, R.; Rodriguez, I.; Razzaghpour, M.; Berardinelli, G.; Christensen, P.H.; Mogensen, P.E. Indoor occupancy detection and estimation using machine learning and measurements from an IoT LoRa-based monitoring system. In Proceedings of the Global IoT Summit (GIoTS 2019), Aarhus, Denmark, 17–21 June 2019; pp. 1–5

[11] Chitu, C.; Stamatescu, G.; Stamatescu, I.; Sgârciu, V. Assessment of Occupancy Estimators for Smart Buildings. In Proceedings of the 2019 10th IEEE International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications (IDAACS),

[12] Jiang, C.; Chen, Z.; Su, R.; Masood, M.K.; Soh, Y.C. Bayesian filtering for building occupancy estimation from carbon dioxide concentration. Energy Build. 2020, 206, 109566.

[13] V. Erickson, S. Achleitner, and A. Cerpa, "POEM: Power-efficient occupancy-based energy management system," in Proc. 12th Int. Conf. Inform. Process. Sensor Netw. (IPSN). New York, NY, USA: ACM, 2013, pp. 203–216.

[14] V. Garg and N. Bansal, "Smart occupancy sensors to reduce energy consumption," Energy and Buildings, vol. 32, no. 1, pp. 81 – 87, 2000.