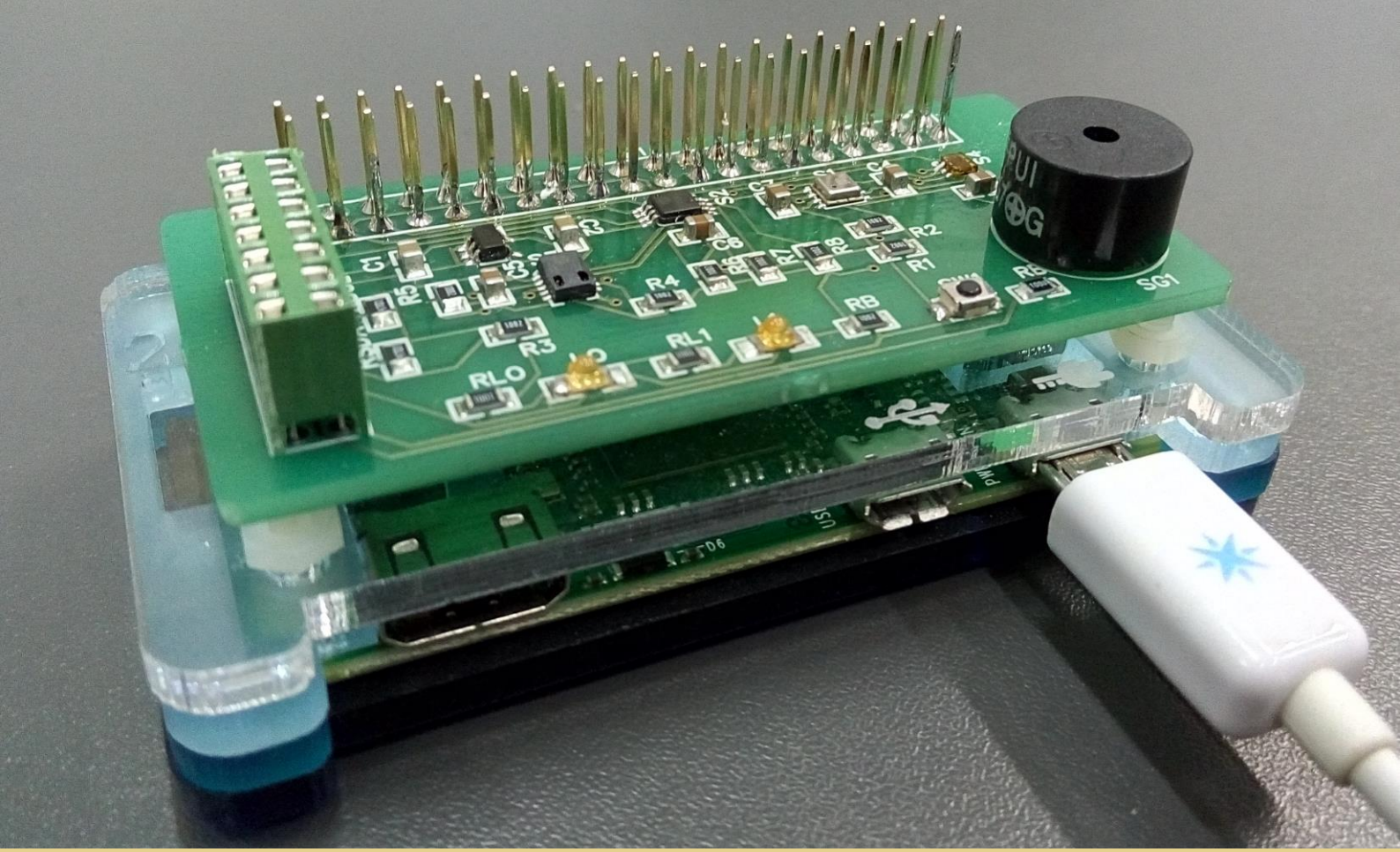


Bruno Miguel Gonçalves Abade

Context-Aware Improved Experiences in Smart Environments

Dissertation submitted in partial fulfilment for the degree of
Master of Science in Electrical and Computer Engineering

Coimbra, April 2018

· U C ·

UNIVERSIDADE DE COIMBRA

Context-Aware Improved Experiences in Smart Environments

Bruno Miguel Gonçalves Abade

Supervisors:

Prof. Dr. Marília Pascoal Curado

Prof. Dr. Hélder de Jesus Araújo

Jury:

Prof. Dr. Nuno Miguel Mendonça da Silva Gonçalves

Prof. Dr. João Paulo da Silva Machado Garcia Vilela

Prof. Dr. Marília Pascoal Curado

Dissertation submitted in partial fulfilment for the degree of Master of Science in Electrical and Computer Engineering

Coimbra, April 2018

The work presented in this document was partially carried out in the scope of the MobiWise project: From mobile sensing to mobility advising (P2020 SAICTPAC/0011/2015), co-financed by COMPETE 2020, Portugal 2020 – Operational Program for Competitiveness and Internationalization (POCI), European Union's ERDF (European Regional Development Fund), and the Portuguese Foundation for Science and Technology (FCT).

Agradecimentos

Ao longo desta jornada várias pessoas fizeram parte da minha vida. Compartilharam alegrias e tristezas, moldaram-me e fizeram-me ser a pessoa que sou hoje. Cada uma dessas pessoas merecia um agradecimento especial, mas sendo este texto limitado, com certeza não irei conseguir agradecer a todos.

Em primeiro, quero agradecer à Prof. Dr. Marília Curado por todo o apoio ao longo deste breve tempo que partilhámos. Sei das minhas limitações, mas com o seu incentivo consegui ultrapassar muitas delas. Uma pessoa que criticava quando tinha de criticar, mas de maneira construtiva. Sempre me mostrou apoio, fazendo com que crescesse enquanto profissional. Embora não tenha tido o prazer de compartilhar aulas, foi uma das melhores professoras que conheci durante estes anos. Muito obrigado pelo seu apoio e da sua grande contribuição na minha formação.

De seguida quero agradecer aos meus colegas do G6.6 por todas as conversas e frases de incentivo, quando por algum motivo me encontrava mais em baixo. Dentre estes há que destacar um em especial, o David Perez Abreu. Embora oficialmente não seja reconhecido como tal, foi quem me coorientou durante todo este percurso. Há que realçar que o meu trabalho em nada está relacionado ao dele, sendo que ele dava do seu tempo para me ajudar e aconselhar. Sem dúvida, juntamente com a Prof. Marília, uma das pessoas que mais contribuiu para o meu trabalho. Um profundo obrigado.

Agradecer também aos amigos que ainda trago da primária, básico e secundário. Embora tenhamos seguido caminhos diferentes continuámos a partilhar bons momentos. Aqui, um agradecimento especial ao Filipe Pato por continuar ao meu lado depois de quase vinte anos. Aos meus amigos LEDZener, aos amigos que fiz no curso e aos amigos que fiz fora do curso e que se tornaram igualmente importantes. Para finalizar, às pessoas que passaram pela minha vida e que embora não partilhemos mais momentos juntos ajudaram-me a aprender e a crescer.

Um agradecimento aos meus avós pela compreensão em dias que respondia menos bem por algo não ter corrido como gostava. Vocês sabem o quanto gosto de vocês. Agradecimento especial ao meu avô Alcindo, foi ele que me ofereceu o meu primeiro computador e me colocou neste 'mundo'. Não é fácil pelo que estás a passar mas peço que tenhas forças para continuar a assistir a mais das minhas conquistas.

Por fim, aos meus pais. Ao longo deste caminho houve altos e baixos mas sempre tiveram uma mensagem de incentivo, nos bons e maus momentos. Se não desisti em alguns ocasiões foi por não os querer desapontar. Obrigado Mãe e Pai.

Por isto tudo e muito mais, muito obrigado!

Bruno Miguel Gonçalves Abade

Resumo

Paradigmas, tais como as cidades inteligentes, ambientes inteligentes e a Internet das Coisas têm sido, nos últimos anos, muito debatidos e usados. Muitos investigadores, em ambientes inteligentes, focam-se na deteção, localização e identificação de pessoas, de modo a adaptarem diferentes ambientes consoante as necessidades e preferências dos cidadãos. Para a sua realização é necessário combinar diferentes sensores, atuadores, modelos matemáticos e técnicas de aprendizagem automática. Apesar de já haverem estudos nesta direção, ainda há espaço para contribuição e este trabalho foca-se nisso.

O objetivo desta dissertação é melhorar a experiência por parte dos utilizadores em ambientes inteligentes baseado na informação obtida em espaços internos, seguindo uma monitorização ocupacional não intrusiva. Assim, o objetivo principal é saber se existe alguém numa determinada divisão e quantos ocupantes se encontram na mesma, informação essa obtida através de sensores.

Numa primeira fase do projeto, o objetivo é desenvolver um dispositivo que adquira informação ambiental usando diferentes sensores, tais como sensores de temperatura, intensidade da luz, ruído e monitorização do dióxido de carbono. Numa segunda fase, aprendizagem automática e mecanismos de reconhecimento de padrões serão usados para a avaliação do desempenho da solução proposta.

Os resultados obtidos durante a investigação demonstram que com sistema criado, através dum Raspberry Pi e da seleção de sensores, é possível obter, processar e armazenar informação ambiente. Adicionalmente, a análise obtida através dos dados adquiridos usando aprendizagem automática e mecanismos de reconhecimento de padrões, descreve que é possível determinar a ocupação em ambientes internos. Portanto, esta informação pode ser tomada em consideração por aplicações de terceiros, de modo, a ajustar o nível de conforto, como por exemplo, numa sala.

Palavras-Chave: Ambientes Inteligentes, Posicionamento Interno, Análise de Dados, Aprendizagem Automática, RaspberryPi.

Abstract

Paradigms such as Smart Cities, Smart Environments, and Internet of Things have been highly debated and used in the last few years. Many researches in smart environments are focused on detection, location, and identification of people, to make different environments adapt to the needs and preferences of citizens. To make this possible, it is necessary to combine different sensors, actuators, mathematical models, and machine learning techniques. Although some works have been performed in this direction, there is still room for contribution, and this research is focused on that.

The objective of this thesis is to improve users' experience in smart environments based on information gathered from indoor spaces following a non-intrusive occupancy monitoring approach. Thus, the main objective is to know if someone is in a room and how many occupants are there, using information gathered from sensors.

In the first stage of the project, the objective is to design and build a device to measure environmental data using different sensors, such as temperature, light intensity, noise, and carbon dioxide monitoring, to estimate the presence of occupants through these environmental features. In the second stage, machine learning and pattern recognition mechanisms should be used to evaluate the performance of the proposed solution.

The results obtained during this research show that the system designed using a Raspberry Pi and a selection of sensors is able to collect, process, and store environmental data. Additionally, the analysis performed over the gathered data using machine learning and pattern recognition mechanisms depicts that it is possible to determine the occupancy of indoor environments. Thus this information could be taken into consideration by a third-party application to adjust the level of comfort, for example, in a room.

Keywords: Data Analysis, Indoor Occupancy, Smart Environments, Machine Learning, Raspberry Pi

"The greatest enemy of knowledge is not ignorance. It is the illusion of knowledge."

Daniel J. Boorstin

Contents

List of Acronyms

ADC	Analogue-to-Digital Converter
AI	Artificial Intelligence
AmI	Ambient Intelligence
API	Application Programming Interface
BFGS	Broyden–Fletcher–Goldfarb-Shanno
BP	Back-Propagation
CISUC	Center for Informatics and Systems of the University of Coimbra
CMOS	Complementary Metal-Oxide-Semiconductor
CO2	Carbon Dioxide
dB	Decibel
FFP	Feed-Forward Propagation
FN	False Negative
FP	False Positive
HVAC	Heating, Ventilation, and Air Conditioning
I2C	Inter-Integrated Circuit
IAQ	Indoor Air Quality
IC	Integrated Circuit
IEU	Integrated Environmental Unit
IoT	Internet of Things

LCT	Laboratory of Communications and Telematics
LED	Light-Emitting Diode
LPF	Low-Pass Filter
LR	Logistic Regression
m	meter
ML	Machine Learning
MOX	Metal Oxide
NIOM	Non-Intrusive Occupancy Monitoring
NN	Neural Network
OS	Operating System
PC	Personal Computer
PCB	Printed Circuit Board
ppm	parts per million
RBF	Radial-Basis Function
RC	Resistor-Capacitor
RPi	Raspberry Pi
SCL	Serial Clock
SDA	Serial Data
SVM	Support Vector Machine
TN	True Negative
TP	True Positive
TVOC	Total Volatile Organic Compounds
V	volt
VOC	Volatile Organic Compounds
USB	Universal Serial Bus

List of Figures

List of Tables

1 Introduction

This document contains the description of the work developed and validated to partial fulfilment for the degree of Master of Science in Electrical and Computer Engineering.

The thesis title is **"Context-Aware Improved Experiences in Smart Environments"**, the research for which is conducted under the guidance of Prof. Dr. Marília Pascoal Curado and Prof. Dr. Hélder de Jesus Araújo, starting in September 2017 on the Laboratory of Communications and Telematics (LCT) - Center for Informatics and Systems of the University of Coimbra (CISUC).

This work is framed in **Mobiwise** project. The main objective of the **Mobiwise** is to connect any sensor, person, and vehicle; and use all possible information to improve the user mobility, through a complete network and services platform for an Internet of Things (IoT) real deployment in a smart city [29]. Within this context, this work aims to research mechanism for occupancy detection.

The first chapter is organised as follows: the first section presents the motivation for this work, followed by the identification of the main objectives to be achieved. The contributions are provided next, and finally, the last section presents the thesis structure and what will be covered in the next chapters.

1.1 Motivation

The amount and diversity of devices connected to the Internet are increasing apace. In the last decades, the use of the Internet has changed from military purposes to domestic purposes. Today, around 40% of the world population has an Internet connection [81]. In 2016, it approached 3.5 billion people worldwide and 7 million in Portugal, which represents around 68% of country's the population [81]. This number only tends to increase.

After connecting people, we are moving now into connecting objects, and estimates indicate that in a not-too-distant period all devices will be connected to the Internet. *"Anything*

that can be connected will be connected." [10]; that is the new rule for the future. Network devices connected to the Internet range from Personal Computer (PC), smartphones, and tablets to pratically almost anything with a sensor on it, such as cars, wearable devices, and houses. The possibilities are tremendous. The IoT is the inter-networking of items embedded with electronic sensors and actuators with network connectivity which enables these 'things' to collect and exchange data between them. The IoT allows objects to be sensed or controlled remotely across existing network infrastructure, creating opportunities for more direct integration of the physical world into computer-based systems and resulting in improved efficiency, accuracy, and economic benefit in addition to reduced human intervention [10, 85]. The projections vary about the evolution of the IoT, but looking at Cisco's calculations it is expected that by 2020 the number of connected devices would reach 50-100 billion [84].

A specific research topic framed in the context of a smart environment is referred to as looking at people, focused on detecting, tracking and identifying to interpret human behaviour. Although some contributions have been performed in this direction, there is still room for contribution, and this research proposal is focused on that [69].

Increasingly, for safety concerns, for health reasons, or only for providing an environment that meets the occupant's preferences, it is necessary to know the presence of people in indoor environments. Although there are already good solutions, such as cameras or wearables devices, issues are often faced, such as privacy (what are you willing to expose for your safety?) and the costs of implementing and deploying these devices.

The main purpose of this thesis is design and built a non-intrusive and low-cost solution to improve user experience in smart environments with ML support through system that monitors temperature, light intensity, noise, and Carbon Dioxide (CO2), to estimate the presence of occupants through this environmental features. Sometimes the sensors to acquire this type of data already exist in these places. First, the data will be collected and analysed, and next it will be applied to ML techniques to conclude if they can be used; in the first stage to detect the presence and in the second stage the number of occupants. As additional work a hardware device was implemented to collect different environmental data.

1.2 Objectives and Contributions

The work presented in this thesis has as main goal the research on occupancy detection using cheap and non-intrusive sensors such as temperature, noise, CO2, and light with

pattern recognition techniques. The first stage is a literature review followed by the study hypothesis and a strategy for data acquisition, building a prototype on the analysis of existing approaches for indoor monitoring. The third stage is applying and analysing ML algorithms to select the most suitable algorithm to identify the occupancy. As additional work, an environmental board was developed to carry out the further work.

The expected outcomes of this work are the following:

- a prototype containing the most appropriate sensors to determine the number of people in indoor environments;
- a dataset of the human density on at least one room at the LCT-CISUC;
- a mechanism to approximate the occupancy in indoor environments based on ML techniques; and
- an environmental board to analyse the room quality and provide some outputs.

The main contribution of this thesis is the research activities using ML techniques with a Non-Intrusive Occupancy Monitoring (NIOM) in an indoor environment. This document supports the reader who is willing to replicate this research and carry out further development.

Last, but not the least, the environmental board produced can provide a complete environmental gathering system that can be used by other LCT researchers. All the sensors are well documented with libraries on the Internet. Some of them were ported to C/C++ language on the course of this work.

1.3 Thesis Structure

This thesis is structured into three parts. The first part presents a brief overview of literature and related work, while the second part describes the hypothesis which supports this research. Regarding the last part, it describes the developments and the results obtained. A brief description of each chapter is presented below.

Chapter 2 (Related Work) presents the main techniques used to detect people and describes the theoretical ML principles of the methods used in this research. These aspects are essential for the understanding of the content of this research, providing the knowledge and tools that form the basis of this work.

Chapter 3 (Methodology for People Density Assessment) discusses the hypothesis for detecting people. Also, it presents the main objectives and the approach to accomplish them.

Chapter 4 (Experience Evaluation) concerns the steps taken for data acquisition and presents the results obtained through the analysis of this data by applying ML techniques of the hypothesis presented in Chapter 3.

Chapter 5 (Additional Contributions) presents some additional contributions that this thesis reached, namely, a sensor board to acquire environmental indoor data.

Chapter 6 (Conclusions and Future Work) concludes this report with a personal opinion of the developed work, the results obtained, and a brief suggestion about possible future work.

2 Related Work

It can be said that computer science is a relatively new branch of science. However, it had a rapid and exponential increment in the last decades. The first human who went to the moon had less computational processing power than the most basic smartphone today. The extraordinary growth of the technology and how the computer science is important can be clearly seen.

Today, our homes were considered smart by the 1960s standards [26]. The computer science is not necessarily only about the PC but also about all devices that have embedded computing power as everyday objects like cell phones, cars, and even washing machines. IoT paradigm thus emerged.

This chapter presents the definition of some concepts and paradigms used during this research. The concepts of Smart Cities and IoT are presented in Section 2.1. Section 2.2 presents the concepts of Smart Environment and Ambient Intelligence (AmI). Section 2.3 addresses the main principles of people detection. Finally, Section 2.4 describes techniques and concepts of ML.

2.1 Internet of Things and Smart Cities

"The 'Thing' in the IoT is the starting point for an IoT solution" [36].

The Internet of Things (IoT) enables physical objects to have smart characteristics such as seeing, hearing, thinking and performing jobs. They communicate together, share information, and coordinate decisions. IoT transforms these objects exploiting their underlying technologies such as ubiquitous and pervasive computing, embedded devices, communication technologies. It can play an important role in improving the quality of our lives. Some applications include home automation, industrial automation, medical aids, mobile healthcare, elderly assistance, intelligent energy management and smart grids, and automotive and traffic management [7, 87].

The emerging paradigm of Smart Cities uses a variety of cheap sensors embedded on traditional things to improve the quality of life. Smart Cities include different service sectors, such as Smart Governance, Smart Mobility, Smart Utilities, Smart Buildings, and Smart Environment where some applications can be structural health of buildings, air quality monitoring, waste management, noise detection, traffic congestion, city energy consumption, smart parking, smart lighting, automation of buildings, and many others [31, 87].

2.2 Smart Environment and Ambient Intelligence

This research focuses on the Smart Environment sector. Designing a Smart Environment is a goal in a variety of disciplines, including pervasive and mobile computing, sensor networks, Artificial Intelligence (AI), robotics, multimedia computing, middleware, and agent-based software. One possible definition of Smart Environment is that *"it is able to acquire and apply knowledge about the environment and its inhabitants to improve their experience in that environment"* [27]. Sensors are used to monitor the environment, sending this information through the communication layer. The databases store the collected information that will be processed into more useful formats. This information is often used by algorithms to make decisions. The decided action is communicated to the application layer, which records the action and enforces it on the physical components [27].

Technically, the Smart Environment can be divided into three categories as follows:

1. Controllable environment: A user can control different devices in more advanced and more efficient ways than it is done in the normal environment. For this, an integrated remote control is used. It can be controlled through physical adjustment, by gesture, or by voice.

2. Programmable environment: A user programmes the environment so it would switch on, switch off or adjust some devices in particular conditions. It can be activated by reacting to time, providing a simple sensor input or assessing and recognizing situations.

3. Intelligent environment: This group is very similar to the previous one, with one exception. The environment would take the decisions and actions without previous user programmes. The system searches for repeated actions. After a pattern has been identified, the environment will programme itself so that, in the next phase, the scenario is recognised.

Using learning mechanisms, to analyse patterns, predict situations, and take decisions and actions, creating new terms such as AmI. AmI brings intelligence to our everyday environments, making it sensitive to us. The main goal is to introduce technology into the environment so that the system embodies users and their surroundings, accumulating data and selecting actions [30]. Large-scale heterogeneous environments encompass human beings (e.g. wearables and physiological sensing environments), real environments (e.g. smart homes/office/cars), and virtual environments (e.g. second life, Google earth) [65].

AmI technology is expected to be: sensitive, adaptive, transparent, ubiquitous, and intelligent. It incorporates contributions from ML, agent-based software, and robotics. It makes use of the AI field, but it should not be considered synonymous with AI [26].

2.3 Approaches to Detect People Density

In Smart Environments and AmI, particular attention has been given to location-based services as a way to offer high-quality intelligent services, while considering human factors such as life patterns, health, and mood of a person [27].

Detecting people is important in many areas such as surveillance, group behaviour modelling, or crowd disaster prevention. For example, we want to provide a safe route for a person to go from place A to place B. For this, we need to know how many people are inside a building and in the safe zones. Because of the impossibility of having users instructing each of the computers in these environments, the computers must know enough about people and their environment so that they can act appropriately with the minimum of explicit instructions (see Figure 2.1) [27].

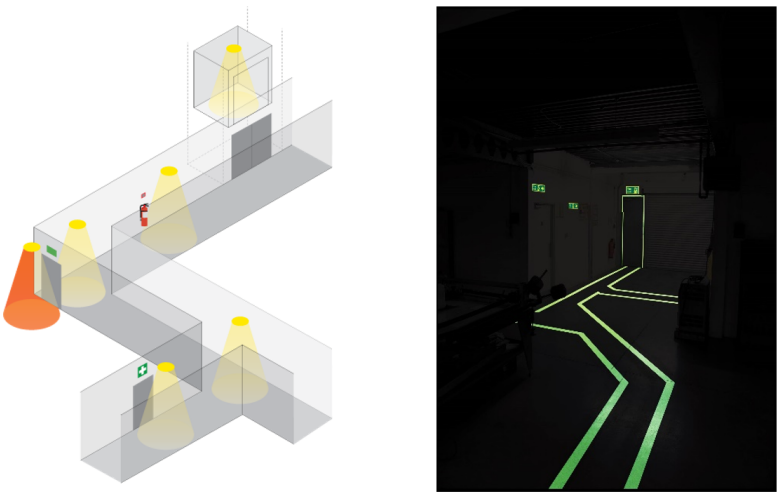


Figure 2.1: Turning on the lights only on safe routes. Adapted from: [45, 52].

Over recent years, the detection of individual objects and persons has improved significantly, but crowd scenes remain particularly challenging for the detection and tracking tasks due to heavy occlusions, high person densities, and significant variation in people's appearance [73].

Various forms of indoor location-aware systems have been developed to recognise people location in smart environments.

Typically, indoor location-aware systems are classified into three types according to the measurement technology: triangulation, proximity, and scene analysis methods (see Figure 2.2) [8, 53].

The triangulation principle uses the geometric properties of triangles to estimate the target location. It has two derivations: lateration and angulation. Lateration estimates the position of a target by measuring its distance from multiple reference points. It is also called range measurement technique. Angulation estimates the location of the desired target by the intersection of several pairs of angle direction lines, each formed by the circular radius from a base station or a beacon station to the mobile target [8, 53].

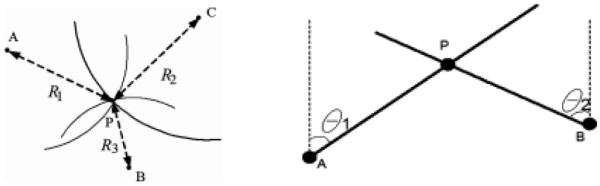
The proximity principle provides symbolic relative location information. Usually, it uses a dense grid of antennas, each having a well-known position. When a mobile target is detected by a single antenna, it is considered close to it [8, 53].

The scene analysis principle collects features of a scene and then estimates the location of a target. The analysis can be of a static scene or a differential scene. The analysis from a static scene deals with the acquired data or datasets to estimate, and the analysis from a differential scene tracks or looks to the differences between successive scenes. It can be implemented with pattern recognition techniques [8, 53].

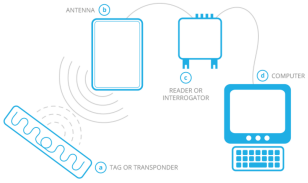
Table 2.1 lists the main advantages and disadvantages of triangulation, proximity, and scene analysis principle. Triangulation methods are very simple to implement but the accuracy depends on the hardware and the propagation model, which will increase the cost and the complexity.

Proximity methods are considered as simple techniques, but their accuracy is limited. For enhancing their accuracy, hardware-based solutions are required. This results in higher cost for the development and maintenance of the positioning system.

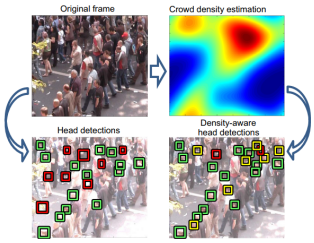
Scene analysis methods do not rely on any theoretical model, or specific hardware, which is an advantage. But it requires a pre-phase for capturing features and it is influenced by environmental changes [66].



(a)



(b)



(c)

Figure 2.2: (a) Triangulation based on lateration (left) and angulation (right) [53]. (b) Proximity method with a RFID technique [83]. (c) Scene analysis method used for crowd density estimation [73].

Table 2.1: Main advantages and disadvantages of indoor location-aware principles. (Source: research and summarisation of the following references: [8, 39, 53, 66])

	Triangulation	Proximity	Scene Analysis
Advantages	- Easy to deploy - Usually implemented by a Wireless technology	- Accuracy of a proximity principle is very high if the person is near an antenna - Simple technique, only needs an antenna and a physical device	- Does not rely on any theoretical model or specific hardware - The location can be inferred using passive observation and features - There is no need for geometric angles or distances
Disadvantages	- Very poor performances if beacon does not have direct sight - A propagation model is required which correlates the corresponding signal properties with distance - Dependency on advanced hardware and transmission techniques, increasing the cost and complexity	- Cost to improve its accuracy - The physical device must stay with the person - Needs many antennas for an accurate positioning system	- Needs to access the features of the environment to compare datasets or scenes - Change of environment has to create a new dataset - Usually, requires more processing time - Large databases

2.4 Machine Learning

"Learning is more like farming, which lets nature do most of the work. Farmers combine seeds with nutrients to grow crops. Learners combine knowledge with data to grow programmes" [32].

Machine Learning (ML) can be defined as the field of study that capacitates computer programmes to learn without being explicitly programmed for that. They can learn from the available data, but no matter how much data they have, the fundamental goal of ML is to generalise beyond the examples in the training set, because it is very unlikely we will see those exact examples again on running time [32].

In ML books and papers, two definitions are widely used. The first is from Arthur Samuel that described it as *"the field of study that gives computers the ability to learn without being explicitly programmed"* [13, 55]. The second is from Tom Mitchell who provided a more modern definition that: *"A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P, if its performance at tasks in T, as measured by P, improves with experience E"* [13, 55]. For example, in a game like checkers E is the experience of playing many games as possible, T is the task of playing the game, and P is the probability that the programme will win the next game [70, 74].

ML problems can be classified in two broad categories [20, 57]:

1. Supervised learning: a series of examples are provided and then generalised to develop an algorithm that applies to new cases (e.g. learning to recognise a person handwriting or voice).

2. Unsupervised learning: the correct responses are not provided, but instead it tries to identify similarities between the inputs that have something in common that will be categorised together (e.g. recommendation of some products from last searches).

The most common cases are supervised and unsupervised learning but in recent years, new categories have appeared. Some are derived from the previous ones, like reinforcement learning and evolutionary learning [48, 3]. Others use hybrid techniques, combining supervised techniques with unsupervised techniques [70].

This work will use supervised learning techniques. This requires that the data used to train the algorithm are already labelled with correct answers. The unsupervised learning is prohibitively complex for some simpler enterprise use cases.

2.4.1 Supervised Learning

As it was said in Section 2.4, in supervised learning the agent gives an input for which it already knows which is the correct output. Relative to the output the problem can be categorised into two main groups [20, 57]:

1. Regression problem: it tries to predict a continuous function from some continuous input (e.g. predict a house price as a function of the number of rooms).
2. Classification problem: it tries to predict a discrete function from some discrete input (e.g. predict which is the number of an image).

To attain the objective of this work, i.e. to identify the people occupancy, a classification problem was used because the dataset has a discrete data and the output can only assume integer values.

In this work, Logistic Regression (LR), Support Vector Machine (SVM) and Neural Network (NN) classifiers were used. These classifiers are largely used in some classification problems [33, 40]. The objective of using these three classifiers is to test the performance of each one and conclude if with this dataset a simple classifier like LR can provide an accuracy similar to a more powerful methods like SVM and NN. *"A more powerful learner is not necessarily better than a less powerful one"* [32].

The next subsections describe the mentioned classifiers.

2.4.2 Logistic Regression

Logistic Regression (LR) is a direct probabilistic interpretation that provides the user with explicit probabilities of classification apart from the class label information. It can be easily extended to the multi-class classification problem [77].

Next subsections present some theoretical concepts about LR. Some of these concepts are also being used in the other classifiers.

Hypothesis function

The hypothesis function, $h_\theta(x)$, is trying to predict the output from the input x with the more accurate fit (Equation 2.1):

$$\widehat{y} = h_\theta(x) = \theta_0 x_0 + \theta_1 x_1 + ... + \theta_n x_n \tag{2.1}$$

where $\widehat{y}$ is the predicted output y and θ is the function parameters [14, 58].

x_0 is set to 1 to deal with the intercept term θ_0, called the 'bias term'. This makes that x and θ have the same number of elements, i.e. $n+1$ terms, so x_0 can be treated as another feature [59].

Equation 2.1 can be represented on a matrix format as shown in Equation 2.2:

$$h_\theta(x) = [\theta_0 \ \theta_1 \ ... \ \theta_n] * \begin{bmatrix} x_0 \\ x_1 \\ ... \\ x_n \end{bmatrix} = \theta^T x \tag{2.2}$$

where θ^T is the transpose of the matrix θ [14, 60].

In a classification problem, the output vector does not have a continuous range of values, so $y \in \{0, 1, ..., n\}$, i.e. discrete values, where the values of y represent the number of classes. The Logistic Function, also called the Sigmoid Function [59], is used in Equation 2.3:

$$g(z) = \frac{1}{1 + e^{-z}} \tag{2.3}$$

where function $g(z)$ maps any real number to a discrete value. It is useful for transforming an arbitrary-valued function into a function better adapted for classification.

With Equation 2.3, the hypothesis function can be rewritten as Equation 2.4 [14, 59]:

$$h_\theta(x) = g(\theta^T x) \tag{2.4}$$

This equation gives the probability that output is 1 if it is a binary classification problem (this type of problem has two classes, i.e. $y = \{0, 1\}$) [14, 59]. For it, the output of the hypothesis must be processed as Equation 2.5:

$$\begin{aligned} h_\theta(x) \geq threshold \rightarrow y = 1 \\ h_\theta(x) \leq threshold \rightarrow y = 0 \end{aligned} \tag{2.5}$$

Typically, the threshold is 0.5. It indicates when $y = 1$ and represents a confidence factor. In practice, it is quite useful when we want to predict $y = 1$ only when is absolutely sure (e.g. health cases) [42].

Beyond binary classification problems, there are also multiclass classification problems (where $y \in \{0, 1, ..., n\}$). In this problem, it was predicted which class that y is member:

$$\begin{gathered} h_\theta^{(0)}(x) = P(y = 0|x; \theta), \ h_\theta^{(1)}(x) = P(y = 1|x; \theta), \ ..., h_\theta^{(k)} = P(y = k|x; \theta) \\ prediction = max_k(h_\theta^{(k)}(x)) \end{gathered} \tag{2.6}$$

Equation 2.6 is used to pick the class for which the corresponding LR classifier outputs the highest probability and returns the class label $(1, 2, ..., k)$ as the prediction for the input example. It is called the One-vs-All classification method [14, 59].

Cost Function

The accuracy of the hypothesis function can be measured by using a cost function (see Equation 2.7). The cost function takes a fancier version of an average of all the results of the hypothesis with inputs compared to the actual outputs:

$$J(\theta) = \frac{1}{m}\sum_{i=1}^{m} Cost(\widehat{y^{(i)}}, y^{(i)}) = \frac{1}{m}\sum_{i=1}^{m} Cost(h_\theta(x^{(i)}), y^{(i)}) \tag{2.7}$$

where the $Cost(h_\theta(x^{(i)}), y^{(i)}) = -y^{(i)}log(h_\theta(x^{(i)})) - (1 - y^{(i)})log(1 - h_\theta(x^{(i)}))$. When $y = 1$ then the second term of the $Cost(h_\theta(x^{(i)}), y^{(i)})$ will be zero and when $y = 0$ then the first term will be zero. The farther the hypothesis from y, the larger the cost function's output [14, 59].

In Equation 2.7, $x^{(i)}$ is the column vector of all feature inputs of the i^{th} training example. A pair $(x^{(i)}, y^{(i)})$ is called a training example and the dataset that it will be used to learn is called a training set. $x_j^{(i)}$ is the value of feature j in the i^{th} training example. The variable m is the number of training examples on the training set and n is the number of features [60].

To summarise: (i) if the correct output is 0, then the cost function will be 0 if the hypothesis function also outputs 0; (ii) if the correct output is 1 then the cost function will be 0 if the hypothesis function also outputs 1; (iii) if the hypothesis is equal to y, then the cost is 0.

Theta (θ) Parameters

With the previous equations, it is possible to calculate the hypothesis function and to measure how well it fits into the data. The next step is to estimate the parameters θ of the hypothesis function.

The way for doing this is to calculate the derivative (the tangential line to a function) of cost function. The slope of the tangent is the derivative at that point and it will give the correct direction to moving. With this, it makes steps down the cost function in the direction of the minimum value. This is where the cost value is the minimum and where the parameters of the hypothesis function will fit better [14, 58].

The algorithm described above is the Gradient Descent algorithm [15]. However, there are more advanced techniques for calculating these parameters like Broyden–Fletcher–Goldfarb-Shanno (BFGS) algorithm, Conjugate gradient, or L-BFGS. This algorithm is from the family of quasi-Newton methods and has faster ways to optimise the parameters θ [14].

Overfitting, Underfitting, and Regularisation

In some cases, the hypothesis function can overfit or underfit the data. High bias or underfitting happens when the hypothesis function maps the data poorly. It can be caused by a function too simple or with few features. On the other hand, it can suffer from high variance or overfitting when the hypothesis function fits the available data but can not predict well the new incoming data [32, 59].

Underfitting can be resolved with the addition of new features. Two possible techniques are applying polynomial regression and combining previous features into a new feature, i.e. $x_3 = x_1 * x_2$. Polynomial regression changes the line behaviour of hypothesis. The solution does not need to be a straight line if that did not fit the data well. It is possible to create additional features from a previous feature to get quadratic, cubic, or other function types. For example, if the hypothesis function is like $h_\theta(x) = \theta_0 + \theta_1 x_1$ it is possible to elevate to an exponent power like $h_\theta(x) = \theta_0 + \theta_1 x_1 + \theta_2 x_1^2$ to have better results [14, 60].

In cases like Polynomial regression, where input values have different ranges, it is useful to perform feature standardisation (see Equation 2.8). Feature standardisation ensures that the dataset has zero-mean and unit variance. This helps to make the gradient descent converging faster [28]:

$$x_j := \frac{x_j - \mu_j}{\sigma_j} \tag{2.8}$$

where μ_j is the average of the values for feature j and σ_j is the standard deviation.

The overfitting problem can be solved reducing the number of features or using regularisation. The regularisation reduces the heaviness of parameters θ_j. For this, a new term must be added at the end of Equation 2.7.

$$J(\theta) = \frac{1}{m} \sum_{i=1}^{m} Cost(h_\theta(x^{(i)}), y^{(i)}) + \frac{\lambda}{2m} \sum_{i=1}^{n} \theta_j^2 \tag{2.9}$$

where λ is the regularisation parameter. The bias term θ_0 must be excluded to avoid penalizing θ_0. Adding this extra summation, it can smooth the output of the hypothesis function to reduce overfitting [14, 60].

2.4.3 Support Vector Machine

Support Vector Machine (SVM) is based on quadratic optimisation of convex function proposed by Boser in 1992 and modified by Cortes and Vapnik in 1995 [18]. The goal of SVM is to map data into a high dimensional space and find a separating hyperplane with the maximal margin [18, 61].

Equation 2.10 defines the decision surface separating the classes [61].

$$\Theta^T x + b = 0 \tag{2.10}$$

where Θ is a weight vector, x is an input vector and b is bias.

The separation between the hyperplane and the closest data point for a given weight vector and bias is called margin of separation, d. If $\Theta^T x + b \geq 0$ then $d_i = +1$. If $\Theta^T x + b < 0$ then $d_i = -1$. The particular hyperplane for which the margin of separation d is maximised is called optimal hyperplane [12].

The cost function has to suffer some modifications for SVM (see Equation 2.11) [61].

$$J(\theta) = \sum_{i=1}^{m} y^{(i)} cost_1(\Theta^T x^{(i)}) + (1 - y^{(i)}) cost_0(\Theta^T x^{(i)}) + \frac{\lambda}{2} \sum_{j=1}^{n} \Theta_j^2 \tag{2.11}$$

where $cost_0(z) = max(0, k(1 + \Theta^T x))$ is the classification cost when $y = 0$ and $cost_1(z) = max(0, k(1 - \Theta^T x))$ is the classification cost when $y = 1$. k is an arbitrary constant defining the magnitude of the slope of the line.

The convention dictates that the C parameter is used to regularise instead of λ. So Equation 2.11 can be rearranged into Equation 2.12. This is equivalent to $C = \frac{1}{\lambda}$. If C decreases, the regularisation is greater and if C increases the regularisation is less [61].

$$J(\theta) = C \sum_{i=1}^{m} y^{(i)} cost_1(\Theta^T x^{(i)}) + (1 - y^{(i)}) cost_0(\Theta^T x^{(i)}) + \frac{1}{2} \sum_{j=1}^{n} \Theta_j^2 \tag{2.12}$$

This model can be easily switched from linear to nonlinear functions through nonlinear mapping called kernel functions [61, 80].

2.4.4 Neural Network

The term Neural Network (NN) in computer science was proposed by Warren McCulloch and Walter Pitts in 1943 to describe a computational model inspired in how the human brain works [22]. In this classifier, the electrical inputs (that a neuron processes in a human brain) are represented as input features, X, that must be processed to obtain a result known as hypothesis function (i.e. axons in a human brain) [14, 62].

The NN logistic function uses the sigmoid activation function to process the input features via a set of activation units (see Equation 2.3 and Equation 2.4). These activation units are grouped in hidden layers between the input features or first layer and the final hypothesis

or the output layer. The NN hypothesis function is depicted in Equation 2.13 [14, 62]:

$$\begin{bmatrix} x_0 \\ ... \\ x_n \end{bmatrix} \rightarrow \begin{bmatrix} a_0^{(2)} \\ ... \\ a_i^{(2)} \end{bmatrix} \rightarrow \begin{bmatrix} a_0^{(3)} \\ ... \\ a_i^{(3)} \end{bmatrix} \rightarrow ... \rightarrow \begin{bmatrix} h_\theta(x)_1 \\ ... \\ h_\theta(x)_k \end{bmatrix} \tag{2.13}$$

In the NN, the θ parameters from LR are instead called weights, Θ. $h_\theta(x)_k$ is the resulting hypothesis for one set of inputs where the k parameter is the number of classes (e.g. for $k = 1$, the result is $h_\theta(x) = [1\ 0\ ...\ 0]^T$). The hidden layer nodes a_j^l are called activation units and represent the activation of node h in layer l. Each row of the parameters will be applied in the inputs to obtain the value for one activation node.

Each layer has its own matrix of weights, $\Theta^{(l)}$. The dimension of these matrices of weights is determined as: if the network has s_j units in layer l and s_{j+1} units in layer $l+1$, then the dimension of $\Theta^{(l)}$ will be $s_{j+1} * (s_j + 1)$. The $+1$ comes from the addition of the bias nodes, x_0 and $\Theta_0^{(l)}$ (E.g. layer 1 has three input nodes and layer 2 has five activation nodes, so the dimension of $\Theta^{(1)}$ will be $5 * 4$ where $s_j = 3, s_{j+1} = 5$ therefore $s_{j+1} * (s_j + 1) = 5 * 4$).

In NN classifier, the Feed-Forward Propagation (FFP) algorithm computes all the activations throughout the network, including the hypothesis $h_\Theta(x)$ and the back-propagation algorithm computes an error term $\delta_j^{(l)}$ that measures how much a node is 'responsible' for any errors in the output [14, 63].

Feed-Forward Propagation

The Feed-Forward Propagation (FFP) algorithm starts to set $x = a^{(1)}$ and defines a new variable $z_j^{(l)}$ that encompasses the parameters inside the sigmoid function (see Equation 2.14) [14, 62].

$$z_j^{(l)} = \Theta_{j,0}^{(l-1)} x_0 + ... + \Theta_{j,n}^{(l-1)} x_j \tag{2.14}$$

Equation 2.15 presents the activation node vector for layer l [14, 62].

$$a^{(l)} = g(z^{(l)}) \tag{2.15}$$

It is necessary to add the bias unit to layer l after computing $a^{(l)}$. This element is $a_0^{(l)}$ and it is equal to 1.

The last step is to determine the hypothesis function (see Equation 2.16) [14, 62].

$$h_\theta(x) = a^{(l+1)} = g(z^{(l+1)}) \tag{2.16}$$

In NN, the prediction is like the one-vs-all classification strategy (see Equation 2.6).

Back-Propagation Algorithm

Back-Propagation (BP) is the terminology to minimise the cost function in NN. It is like gradient descent to LR [14, 63].

In NN, the cost function can be defined as Equation 2.17 [14, 63].

$$J(\Theta) = -\frac{1}{m}\sum_{t=1}^{m}\sum_{k=1}^{K}[y_k^{(t)}log(h_\Theta(x^{(t)})_k) + (1-y_k^{(t)})log(1-(h_\Theta(x^{(t)})_k)] + \frac{\lambda}{2m}\sum_{l=1}^{L-1}\sum_{i=1}^{s_l}\sum_{j=1}^{s_{(l+1)}}(\Theta_{j,i}^{(l)})^2 \tag{2.17}$$

where L is the total number of layers in the network, s_l is the number of units in layer l (without counting the bias unit) and K is the number of output classes. The i , in triple sum, does not refer to training example i. The variable t represents the index of a training example.

To minimise $J(\Theta)$, the partial derivate must be computed. BP computes it for every node. So, the error of node j in layer l has to be calculated. In the last layer, it can be calculated directly by measuring the difference between the network activation and the true output (see Equation 2.18) [14, 63].

$$\delta^{(L)} = a^{(L)} - y \tag{2.18}$$

In the hidden layers, the $\delta^{(L)}$ can be computed based on a weighted average of the error terms of the nodes in layer $(l+1)$ (see Equation 2.19) [14, 63].

$$\delta^{(L)} = ((\Theta^{(l)})^T\delta^{(l+1)}).*g'(z^{(l)}) \tag{2.19}$$

The δ values of layer l are obtained by multiplying the δ values in the next layer with the Θ from this layer. The '.*' represent element-wise, if working with vectors and matrices, with this it is possible to multiply element by element with the derivate $g'(z^{(l)})$.

Equation 2.20 presents the full BP equation:

$$\delta^{(L)} = ((\Theta^{(l)})^T\delta^{(l+1)}).*a^{(l)}.*(1-a^{(l)}) \tag{2.20}$$

The partial derivate terms can be computed by multiplying the activation values and the error values for each training example t (see Equation 2.21) [14, 63]:

$$\frac{\partial J(\Theta)}{\partial \Theta_{i,j}^{(l)}} = \frac{1}{m}\sum_{t=1}^{m} a_j^{(t)(l)}\delta^{(t)(l+1)} \tag{2.21}$$

In Algorithm 1, in Appendix D, the pseudo-code of BP algorithm is presented. The $D_{i,j}^{(l)}$ is used as an accumulator of the partial derivates, i.e. $\frac{\partial J(\Theta)}{\partial \Theta_{i,j}^{(l)}} = D_{i,j}^{(l)}$ [14, 63]. Contrary to

LR classifier, in the NN classifier, the values of Θ is not initialised with zeros. These values must random initialise because when applies back-propagate, all nodes will update to the same value repeatedly (symmetry-breaking) [14, 62].

Let us summarise all the steps in NN classifier. First, choose the layout of NN, i.e. choose the number of hidden layers, the number of outputs units, and the number of hidden units per layer. With this, it can train NN with the following steps:

- Random initialise the weights;
- Use FFP to obtain $h_\Theta(x)$;
- Use BP to compute partial derivatives;
- Use gradient descent or a built-in optimisation function to minimise the cost function with the weights.

2.5 Summary of Chapter 2

This chapter presented the basic concepts to understand this work.

IoT enables physical objects to observe, monitor, and interact with the physical world. In Smart Cities, these 'intelligent' devices, deployed over an area, will generate different types of data that will be transferred to a database, stored and processed.

The Smart Environment is able to probabilistically predict the occupant's location and activities with the best accuracy using pattern recognition techniques. Detecting people density has a larger area of interest such as tracking people.

Indoor location-aware systems can be classified into three types. Triangulation uses geometric properties of the triangle to compute object locations. Proximity determines if an object is near a known location. Scene analysis uses features of a scene observed from a certain reference point.

The most common techniques in ML are Supervised and Unsupervised. LR is a direct probabilistic interpretation. SVM finds a hyperplane with the maximal margin to separate the data with similarities. NN is a classifier inspired by how the human brain works and the hypothesis function is obtained processing the input features via a set of activation units.

In this work, a scene analysis architecture is used to gathering environmental data to apply and analyse the best ML classifier. This architecture, methodology, and the environmental features analyzed are presented in next chapter.

3 Methodology for People Density Assessment

This chapter presents the main architecture and methodology of this work. The Core-Features analyse how the humans influence the environment, more specifically in the indoor environments. It is presented in Section 3.1. Section 3.2 presents IoT architecture model followed in this work with an explanation of the major layers and descriptions of equipment used and the communications between them. Lastly, Section 3.3 presents the methodology with the characterisation of the main the input functions to detect and measuring people occupancy, the local where the sensors were placed, the description how the data were collected, and the metric used to evaluate the ML classifiers.

3.1 Core-Features

This work analysed features such as temperature, CO2, noise, and light. These characteristics are also related to some of today's health concerns caused by noise pollution, gas pollution, and light pollution.

3.1.1 Temperature

The first law of thermodynamics states that energy can be transformed from one form to another but it is impossible to create energy out of nowhere. The second law states that a hot body transfers its energy to a cold one. Also, the human body is subject to these laws [68].

A human body constantly generates heat. It is a side effect of metabolism. This heat is dissipated into the ambient as heat flow and infrared radiation; the rest is rejected in a form of water vapour [50]. At ambient temperature (about 25°C), the skin temperature approaches closely to the in-body temperature (about 36.8°C) [49].

There are seven fundamental factors that can change the heat generated by a human. They are air temperature, radiant temperature, relative humidity, air movement, metabolic rate, and clothing insulation level and moisture permeability. The characteristics of the human body (e.g. weight) and psychological state may also influence the generation of heat [68].

A healthy adult human releases approximately between 100 Watts (resting situation) and 1000 Watts or more (effort situation) [67]. This energy produces heat equivalent to the heat dissipated by a few laptops [50].

The heat transfers can take place by conduction, convection, and radiation. The conduction is driven by temperature difference and the rate of flow. Dependings on the thermal resistance of material, generally, it is ignored between the body and the environment. The convection involves conduction of heat from a solid surface to a moving fluid. A major part of heat transfer is by it. The radiation is a part of electromagnetic spectrum. All bodies above an absolute zero temperature emit and absorb radiation. These three forms of heat transfer are called sensible heat transfer. The heat can also be lost or gained by a change of states such as evaporation and condensation. They are called latent heat [68].

The environment temperature influences the body heat and the body heat influences the environment. *"In a 'cold' environment there will be a layer of 'warmer' air surrounding the body"* [68]. This heat exchanges between bodies is a continuous process. Depending on the size of a room and the number of persons in it, these exchanges can be more or less accentuated.

3.1.2 Carbon Dioxide

Nowadays people spend a large amount of time in an indoor environment. Some studies which show how the air quality affects an occupant's health and comfort are being done [54].

Indoor Air Quality (IAQ) is a term related to the air quality within and around buildings related with the health and comfort of building occupants. IAQ can be affected by gases (including carbon monoxide, radon, carbon dioxide and volatile organic compounds), particulates, microbial contaminants (mould, bacteria), or any mass or energy stressor that can induce adverse health conditions [54].

Carbon Dioxide (CO2) is one of the gases present in the atmosphere. This is one of the final products of cell metabolism. CO2 is also a gas produced from the burning complete combustion of organic material. It is essential for the photosynthesis by plants and other

organisms, which will be consumed by animals, making possible the existence of life.

CO2 is an important gas for the existence of life but at very high concentrations (e.g. greater than 5000 parts per million (ppm)) it can pose a health risk [6]. CO2 concentrations commonly found in buildings are not a direct health risk, but this concentration can be used as an indicator of occupants [6]. Typically, in outdoor environments this concentration is between 300 and 500 ppm. In fact, occupants are the principal indoor source of CO2, increasing the levels compared with outdoor environments [86].

Equation 3.1 gives the rate of change in the concentration of the monitored gas in a room [86]:

$$V\frac{dC}{dt} = Q_{in}C_{in} - Q_{out}C_{out} + S - kC \tag{3.1}$$

where V is room volume in cubic meter $(\mathrm{m})^3$, C_{in}, C_{out}, and C are concentrarions of the monitored gas in the inflow, indoor air, and outflow, respectively, in mg/m^3, Q_{in} and Q_{out} ws into/out of the space in m^3/h, S is the indoor emission source of the monitored gas in mg/h and k is the first-order degradation constant in m^3/h.

3.1.3 Noise

Noise pollution is a big problem in urban environments. In outdoor environments, the persons are exposed to many noise sources such as vehicle noise. According to the green EU paper, *"Environmental noise, caused by traffic, industrial and recreational activities is one of the main local environmental problems in Europe and the source of an increasing number of complaints from the public."* [21].

Indoor environments have background noise produced by loud crowds or household appliances. Excessive noise affects the human behaviour, well-being, productivity, and health. The audio range by a human ear is between 20 hertz (Hz) and 20 kilohertz (kHz) [25]. Sounds outside this range are considered infrasound (below 20 Hz) or ultrasound (above 20 kHz) [25].

One of the sources of noise is the human being. The human being is a communicative being by nature. The human being uses the language to maintain relationships with another human being. From one person to two, from two persons to three, from three persons to a crowd also the noise level increases. Sound levels in offices are relatively low but it is probably the most annoying source in the offices and can lead to increased stress for occupants [46].

3.1.4 Light

Light is an electromagnetic radiation within a certain portion of the electromagnetic spectrum. The visible spectrum is a part of the radiation that is visible to the human eye and is responsible for the sense of sight. The light can be from natural (e.g. sun) or artificial (e.g. lamps) sources [23].

Illuminance is the amount of luminous flux per unit area. It is measured in lux and it is related to brightness as perceived by human vision. In a full sunlight day, the levels of illuminance are 103000 lux, in a cloudy day between 1000 and 10000 lux, in a full moon night is between 0.1 and 0.3 lux and in an overcast night sky is between 3E5 and 1E4 lux. In indoor situations, such as inside a bright office building, the levels of illuminance are between 400 and 600 lux and between 100 and 300 lux in most homes [37].

The era of electric light began in 1879 when Thomas Edison has illuminated the first light bulbs in a New York street [19]. In the next years, light pollution has rapidly increased by the introduction of artificial light in alteration of natural light levels [35]. However, the artificial light has benefited the society, extending the length of the productive day.

"Overillumination refers to the use of artificial light well beyond what is required for a specific activity, such as keeping the lights on all night in an empty office building" [19]. However, by day, it is also normal for artificial lights to be on because it is impossible to place all occupants near natural sources of light.

3.2 Architecture

Several concepts of IoT were introduced in Section 2.1. The IoT architecture model followed in this work has four major layers [10]:

1. Objects Layer: This layer deals with the physical sensors that aim to collect raw data information. It includes sensors and actuators.

2. Communication Layer: This layer deals with the data coming from the sensors to the next layers. It includes an embedded Operating System (OS), signal processors, microcontrollers, and gateway nodes.

3. Analysis Layer: This layer provides data management that is required to extract the necessary information from the amount of raw data collected. It includes data mining, analytics services, and device management.

4. Application Layer: This layer deals with the utilisation of the processed data. It includes services and applications.

Figure 3.1 represents the architecture used in this work. The next subsections exhibit each layer in more detail.

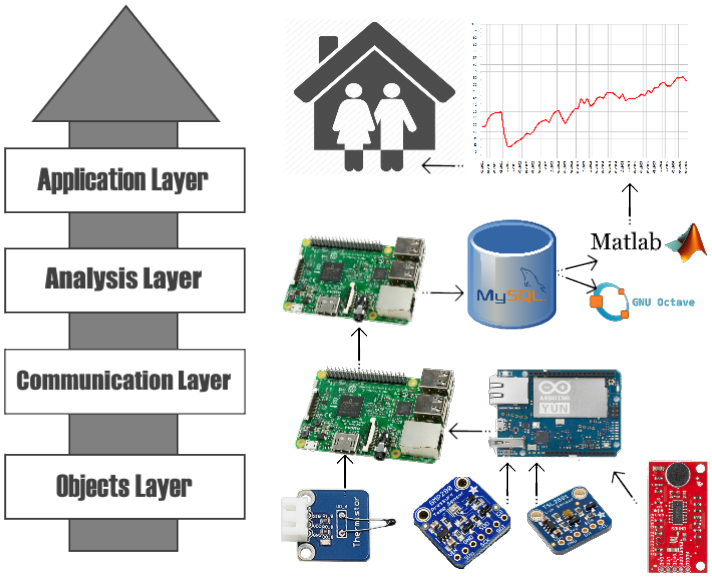


Figure 3.1: IoT Architecture used in this work.

3.2.1 Objects Layer

In this work, four features were analyzed, namely temperature, CO2, noise, and light. The sensors used are described in the next subsections.

Temperature

A thermistor module (Figure A.3 in Appendix A) was used to record the ambient temperature. A thermistor is a resistor whose resistance varies with temperature. This variation, when measured, can be used to calculate the temperature around the resistance. This sensor is used as inrush current limiters, temperature sensors, self-resetting overcurrent protectors, and self-regulating heating of elements.

There are two types of thermistors depending if the resistance increases or decreases with temperature. In this case a thermistor NTC was used. With NTC thermistors, resistance decreases as temperature rises.

A voltage divider with a known resistance is used to measure the thermistor resistance.

The value of it is given by Equation 3.2.

$$R_2 = R_1 \left(\frac{V_{out}}{V_{in} - V_{out}} \right) \tag{3.2}$$

where R_2 $[Ohm(\Omega)]$ is the thermistor resitance, R_1 $[\Omega]$ is a known resistance, V_{out} $[Volt(V)]$ is the output voltage and V_{in} $[V]$ is the input voltage.

Steinhart-Hart equation is a model of the resistance of a semiconductor at different temperatures [82]. The Beta (β) parameter equation, which is essentially the Steinhart–Hart equation, is used when Steinhart–Hart coefficients are not published by the thermistor manufacturers (see Equation 3.3).

$$\frac{1}{T} = \frac{1}{\beta} ln \left(\frac{R}{R_{T0}} \right) + \frac{1}{T_0} \tag{3.3}$$

where T is the temperature in Kelvin, $R[\Omega]$ is the resistance of the thermistor, β is a coefficient of the thermistor (usually between 3000 and 4000), R_{T_0} is the thermistor resistance at 25ºC and T_0 is the temperature for nominal resistance (always 25ºC) [1].

Carbon Dioxide

The CCS811 Sensor Breakout by Adafruit (Figure A.5 in Appendix A) was used to collect CO2 raw data. This sensor is produced by AMS. It has a metal oxide (MOX) sensor and a microcontroller that controls power to the plate, reads the analogue voltage, and provides an Inter-Integrated Circuit (I2C) interface to read from (Subsection 3.2.2). It will measure eCO2 (equivalent calculated carbon-dioxide) concentration between 400 and 8192 ppm, and Total Volatile Organic Compounds (TVOC) concentration between 0 and 1187 ppm [4].

Noise

The SparkFun Sound Detector by SparkFun (Figure A.7 in Appendix A) was used to collect the noise raw data. This board is a small audio sensing board with three different outputs. It provides an audio output, a binary indication of the presence of sound and an analogue representation of its amplitude [78].

In this work, only the amplitude is collected to avoid privacy issues. The amplitude was given in Volts (V) but with Equation 3.4 can easily be converted to Decibel (dB). In that equation, V_{dB} is the voltage in dB, V_{out} is the output voltage in volts, and V_{ref} is the reference voltage in volts.

$$V_{dB} = 20 log_{10} \left(\frac{V_{out}}{V_{ref}} \right) \tag{3.4}$$

Light

The TSL2591 Breakout by Adafruit (Figure A.6 in Appendix A) was used to collect light intensity raw data. This sensor is produced by AMS.

The TSL2591 sensor transforms light intensity into a digital signal output. This sensor combines one broadband photodiode and one infrared-responding photodiode on a CMOS integrated circuit. It has two integrating ADC to convert the photodiode currents into a digital output that represents the irradiance measured on each channel. That means the infrared, full-spectrum, or human-visible-light can be measured. These values are provided by I2C where illuminance (ambient light level) is given in lux [5]. This value was derived using an empirical formula to approximate the human eye response (for more details see [38]).

3.2.2 Communication Layer

In the thermistor sensor case, the communication is stabilised by 8 bits Analogue-to-Digital Converter (ADC) (it is a system that converts an analogue signal to digital signal) between the 10kΩ resistor and the thermistor. Once the ADC value is obtained, Equation 3.5 gives the output voltage V_{out} [1].

$$V_{out} = \frac{ADC_{reading}}{ADC_{resolution}}.V_{in} \tag{3.5}$$

where V_{out} is the output voltage in volt (V), $ADC_{reading}$ is the value obtained in the ADC in bits, $ADC_{resolution}$ is the maximum value that can be obtained by ADC in bits (in this case 255), and V_{in} is the input voltage in V.

The ADC (Figure A.4 in Appendix A) and the other sensors described in Subsection 3.2.1 communicate with Arduino Yun (a microcontroller board based on the ATmega32u4 and the Atheros AR9331) through the I2C and the Arduino Yun (Figure A.2 in Appendix A) communicates with RPi (Figure A.1 in Appendix A) by Serial Communication.

The data collected were stored in a database (in this case another RPi) connected through an Ethernet cable.

Inter-Integrated Circuit

The Inter-Integrated Circuit (I2C) protocol was developed by Philips in 1982 and was intended to allow multiple slaves to communicate with one or more masters. This interface only requires two signal wires to exchange information, namely Serial Data (SDA), and Serial Clock (SCL). Those two wires can support up to 1024 slave devices and both signals are

bidirectional and require a positive supply voltage via pull-up resistors. It supports devices communications from 0kHz to 5MHz (for more information see [79]).

RPi and Arduino Yun

The Raspberry Pi (RPi) is a small single-board computer developed by the RPi Foundation. This board allows the execution of a wide variety of tasks and it has a large popularity because of its low cost and large community support.

RPi is used when one computer is necessary but the processing power required is not high. It can deal with applications that involve hardware logic but its potential is better explored in applications that need an optimised software solution because it runs a Linux OS and it allows several programming languages. The main difference in a microprocessor, like RPi and microcontroller, like Arduino, is that in the microcontroller, the code is written to control the hardware directly and in a microprocessor, programmes have been written to run in OS. The main rule for using microprocessor instead of the microcontroller is *"If you can describe it with less than two and's, get a microcontroller. If you need more than two and's, get a microprocessor"* [47].

RPis support the I2C communication protocol to access sensors; however the libraries to enable this communication are not available on the Internet. For this reason and taking into consideration the time constraints, the available libraries for these modules to Arduino, in this case by Adafruit, were used.

Serial Communication

The communication between the RPi and Arduino is carried out by Serial Communication performed by the Universal Serial Bus (USB) port.

To do this communication, the core-library 'Serial' of Arduino and the library 'wiringPi' were used [9, 71].

3.2.3 Analysis Layer

The data collected by the RPis were stored in a MySQL database server (an open-source relational database management system). The quantity of data does not justify another type of database but in big data cases it is recommended to use a non-relational database [51].

Raw data processing was done using Matlab Software (a multi-paradigm numerical computing environment). The Matlab Software is a powerful programme when it is necessary to

deal with matrix manipulations. It is possible to optimise the operations involving matrices and vectors by vectorisation. Also, the Matlab software is great for plotting data.

3.2.4 Application Layer

This last layer uses the previous layers to acquire raw data by sensors, storing and processing them to apply ML techniques to perform the main goal of this work: to detect people through non-intrusive sensors. The ML algorithms were developed in Matlab Software with the equations in Section 2.4. To SVM model was used the libsvm (an open source library that implements a sequential minimal optimisation algorithm for kernelised SVM supporting classification and regression problems developed by National Taiwan University [17]).

3.3 Methodology and Implementation

The main purpose is to detect occupants in a room with cheap and non-intrusive sensors.

The methodology is based on three stages: (i) feature selection and pre-processing, i.e. the data that is used to build the model; (ii) the nature of the model-building algorithm; (iii) the type of model that is trained. The type of collected data was presented in Section 3.1. The model-building algorithm and the type of models were presented in Section 2.4. In this work, supervised learning was used as model-building technique and LR, SVM and NN as the type of models.

The experiments were conducted in a LCT-CISUC room which is located in the middle of the G Tower of the Department of Informatics Engineering. It has a floor area of 8.5x5.5 m^2 and 4.15 m of height (see Appendix B and Figure 3.3). This room has a small occupancy change (the maximum number of occupants is five and the minimum number of occupants is zero) and a very low ventilation. The only ventilation in the room is the door and windows cracks. Most of the time, the door is closed and the windows are always closed. The Heating, Ventilation, and Air Conditioning (HVAC) equipment was off during the time of tests to prevent any influence on the data collected.

The following subsections present the input functions that measuring the people density with the presented non-intrusive sensors and how the data were collected.

3.3.1 Input Functions

The inputs to detect people occupancy is presented in this subsection. The sensor data were studied separately, with the correlation between them outside the scope. A binary classification problem and a multi-class classification problem were tested. The first pretends to detect the presence of occupants and the second the number of occupants.

Temperature Input

There is a sensor in the hallway and a sensor inside the room close to the door. The difference between the two temperatures at these two places is small considering the proximity between them. This difference is caused by the presence or absence of occupants. This difference in value will be greater when there are occupants and less, zero, or negative, when the room is empty. It was verified whether it is possible with this temperature to determine if occupants were present or not. Afterwards, two more sensors were placed in the room to reduce the fluctuations. Equation 3.6 gives the input function tested for this type of data.

$$num[n] = \frac{t_1[n] + t_2[n] + t_3[n]}{3} - t_{out}[n] \tag{3.6}$$

where num is the number of occupants, $t_{1,2,3}$ is the indoor temperature in Celsius of the three sensors, t_{out} is the outdoor temperature in Celsius, and n is the instant of time.

Carbon Dioxide Input

How the occupants can change the CO2 levels in a room was described in Subsection 3.1.2. Two CO2 sensors were placed in the room to collect this data. Equation 3.7 gives the input function tested for this type of data.

$$num[n] = \frac{eco_1[n] + eco_2[n]}{2} \tag{3.7}$$

where num is the number of occupants, $eco_{1,2}$ is the indoor Carbon Dioxide (CO2) in ppm of the two sensors, and n is the instant of time.

Noise Input

Three sensors were placed in the room, close to the occupants, to collect this data. In working environments, where occupants are working for the most of time, it was expected that this feature would not be able to detect the presence of the occupants. Equation 3.8

gives the input function tested for this type of data.

$$num[n] = sd_1[n] + sd_2[n] + sd_3[n] \tag{3.8}$$

where num is 1 if an occupant is detected and 0 if not, $sd_{1,2,3}$ is the sound detector in dB of the three sensors, and n is the instant of time.

Light Input

One sensor was placed in the room to collect this data. The sensor was placed as far from the window as possible so that the main light source incident on it was one of the lamps. Because not every desk in the room receives direct natural light, the room lights are always on when someone is there. Only the binary case was tested, i.e. whether the presence of occupants was detected or not. Equation 3.9 gives the input function tested for this type of data.

$$num[n] = lg[n] \tag{3.9}$$

where num is 1 if an occupant is detected and 0 if not, lg is the light intensity in lux of the light sensor, and n is the instant of time.

3.3.2 Nodes Placement

Three Nodes (RPi+Arduino) were placed in the room (see Figure 3.2 and Figure 3.3).

Node1 has two temperature sensors and one sound detector, and it works as the database. All the nodes communicate with it to exchange data and storing. This node is also responsible to control the number of occupants (explained in the next subsection). Node2 has one temperature sensor, one CO2 sensor, and one sound detector. Node3 has one temperature sensor, one CO2 sensor, one light sensor, and one sound detector. With this disposition every occupant in this room is covered.

In Figure 3.4, the yellow icon is the outdoor temperature sensor, the red icons are the indoor temperature sensor, the blue icons are the CO2 sensor, the green icon is the sound detector sensor, and the pink icon is the light sensor.

3.3.3 Ground Truth

In supervised learning techniques, the agent must give the output set to train the classifier (see Section 2.4). The Ground Truth expresses the notion of data that is known to be correct. In this work, a simple mechanism with two buttons (blue to enter and red to leave) was

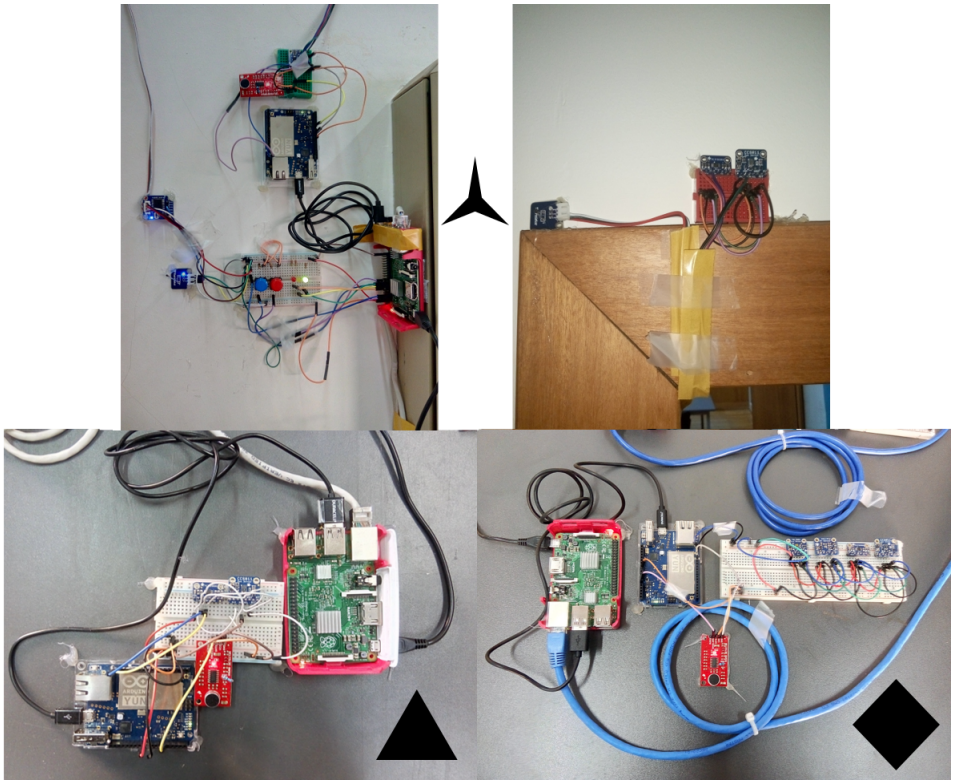


Figure 3.2: Figure of the actual placement of nodes and sensors. Above is Node1 (left indoor sensors and right outdoor temperature sensor). Below are Node2 (on the left) and Node3 (on the right).

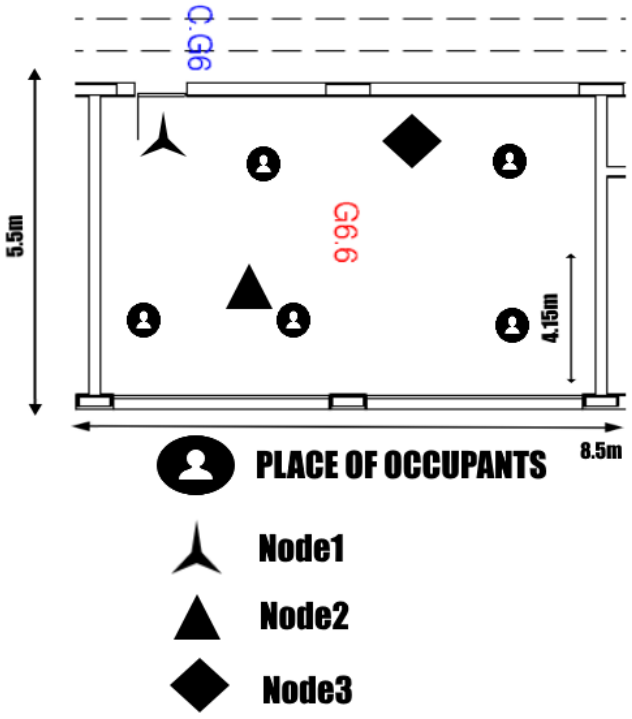


Figure 3.3: Room of proof concept.

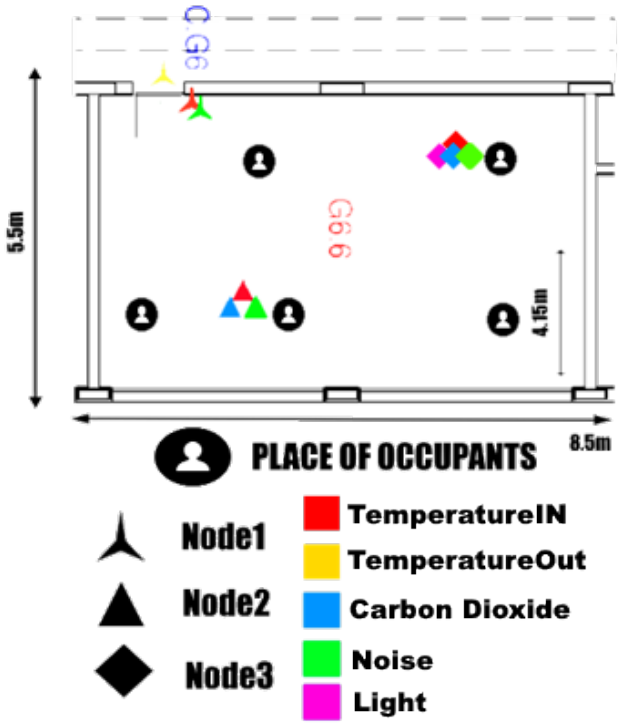


Figure 3.4: Sensors placement.

developed to create the ground truth (see Figure 3.5). Every time that an occupant presses one of these buttons, the counter is increased or decreased, respectively. To visualise if the number of occupants is correct, three Light-Emitting Diodes (LEDs) were introduced as a binary counter ($2^n - 1$ occupants in the room). The left LED is the most significant and the right LED is the least significant. The number of total occupant's by minute is the average of samples acquired every 10 seconds.

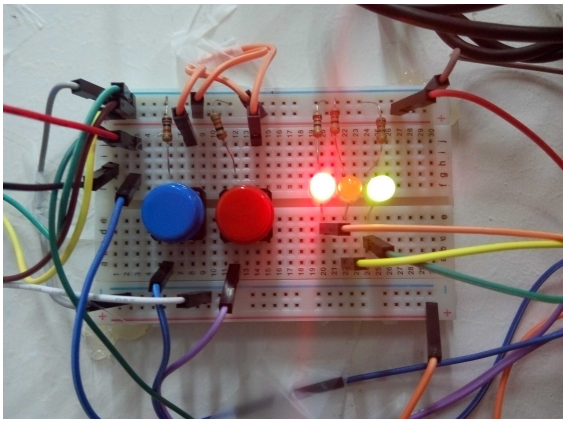


Figure 3.5: Mechanism developed to obtain the ground truth.

3.3.4 Data Collection and Storage Strategy

The sensors used in this work are presented in Section 3.2.1 and the equipment to acquire the raw data is presented in Section 3.2.2.

Every time that Arduino Yun received a signal by the RPi, it gets ten samples with a 100 microseconds delay to eliminate some noise in the data. Then an average is done and this averaged data is sent to RPi.

The RPi make this request six times (60 seconds). Finally, a new average is performed of these six values and the new value is stored in a MySQL database (a MySQL Application Programming Interface (API) was used [56]). With this average, it is possible to decrease the fluctuations in the data. The previous method was used in all sensors to have a timeline between the heterogeneous sensors. The MySQL database stores this average data with the time and data acquisition. In Algorithm 2, in Appendix D, it is possible to see the pseudo-code related to collecting.

3.3.5 Classifier Performance Evaluation

To analyse the performance of a classifier, a judging criterion is necessary. True Positives (TPs) and True Negatives (TNs) represent the correct classification/prediction if the entry belongs to the positive class or negative class, respectively. False Negatives (FNs) and False Positives (FPs) represent the incorrect classification/prediction if the entry does not belong to the negative and positive classes, respectively [70, 72].

Accuracy measures the percentage of entries that were correctly classified (see Equation 3.10), and the miss rate measures the percentage of entries that were incorrectly classified (see Equation 3.11) [70, 72].

$$Accuracy = \frac{TP + TN}{N} * 100 \tag{3.10}$$

where N is the total size of training data set.

$$Missrate = (100 - Accuracy) \tag{3.11}$$

To evaluate a classifier, it is necessary to verify the accuracy to new entries. The classifier can have a high accuracy when tested with the training dataset but can have a low accuracy with a new dataset. So, it is recommended to split the data into training dataset and testing data set [14, 64]. The training data are suitable to train the classifier and the testing data are suitable to measure their performance to new entries. Typically, the data set is divided into

three datasets: training data set to train the classifier, cross-validation data set to adjust the parameters and testing data set to verify the performance of the classifier [14, 64]. In this work, the dataset was split into training data set and testing data set.

In certain cases, the data set can have skewed classes, i.e. one class has a low set of data. For example, assuming that the training data set contains 0 positive and 100 negative entries, and if all instances are predicted correctly, the accuracy will be 100% but the classifier had no chance of learning the hidden patterns. With the previous example, it can be said that the accuracy does not work well when the dataset is unbalanced, i.e. it has more data in one class than in the other.

In this work, the room was occupied on average eight hours per working day, so the positive classes are skewed classes. The F-Score was used to predict the performance of classifiers. It is a technique that measures the discrimination of classes, through a harmonic mean of two metrics, recall and precision (see Equation 3.14) [14, 64]. Recall measures the percentage of entries that belong to the positive class and was classified/predicted correctly (see Equation 3.12) [70, 72]. Precision measures the percentage of proportion over the entries of the predicted positive class that really belong to positive class (see Equation 3.13) [70, 72]. To have a high F-Score, both precision and recall must be high.

$$Recall = \frac{TP}{TP + FN} * 100 \tag{3.12}$$

$$Precision = \frac{TP}{TP + FP} * 100 \tag{3.13}$$

$$FScore = 2 * \frac{Precision * Recall}{Precision + Recall} \tag{3.14}$$

Equation 3.14 can only be applied to binary classification problem, but it can be extrapolated to a multi-class classification problem. Micro-Average Method and the Macro-average method are defined to measure the performance.

The Micro-average method sums up the individual TP, FN and FP contribution of each class label (see Equation 3.15 and 3.16) [11, 75].

$$Recall = \frac{TP_1 + TP_2 + ... + TP_k}{TP_1 + TP_2 + ...TP_k + FN_1 + FN_2 + ...FN_k} * 100 \tag{3.15}$$

where k is the class label.

$$Precision = \frac{TP_1 + TP_2 + ... + TP_k}{TP_1 + TP_2 + ...TP_k + FP_1 + FP_2 + ...FP_k} * 100 \tag{3.16}$$

The Macro-average method takes the average of precision and recall of each class label (see Equation 3.17 and 3.18) [11, 75].

$$Recall = \frac{Recall_1 + Recall_2 + ... + Recall_k}{k} * 100 \tag{3.17}$$

where k is the class label.

$$Precision = \frac{Precision_1 + Precision_2 + ... + Precision_k}{k} * 100 \tag{3.18}$$

Micro and macro average behaving are quite different and may give different results. It is hard to decide between one of these two methods to choose the classifier. *"There is no complete agreement among authors on which is better. Some believe that "microaveraged performance is somewhat misleading (...) because more frequent topics are weighted heavier in the average" [Wiener et al. 1995, page 327] and thus favour macroaveraging. Others (actually, the majority of researchers) believe that topics should indeed count proportionally to their frequence, and thus lean towards microaveraging."* [75]. Finally, the F-Score is calculated, similar to Equation 3.14, for the two methods.

3.4 Summary of Chapter 3

This chapter presented the architecture and methodology used in this work. The features used were temperature, CO2, noise, and light. The environment temperature influences the human body heat, and the human body heat influences the environment. Depending on the room size and the number of occupants, the environment temperature can change slowly or quickly. In indoor environments, another thing that changes with the presence of occupants is the level of CO2. Depending on the airflow exchanges this change can also be slow or quick. Human noise can be a feature to detect the presence of occupants in a room. In working environments, it is impossible to deliver natural light source to all occupants so the artificial light is on, most of the time.

The IoT architecture implemented has four major layers: objects layer, communication layer, analysis layer, and application layer.

Three nodes were placed in the room to test the hypothesis to detect people density presented in this chapter. Each node has some sensors connected. The ground truth is given by a mechanism of two buttons that the occupant presses when arriving or leaving. Every 60 seconds, the data of all nodes are saved in a MySQL Database server, leading to a short system latency. This data will be pre-processed and the ML models will be applied.

The F-Score will be used to judge the classifier performance. This technique measures the discrimination of classes using recall and precision. In the multiclass problem, the F-Scores are used with micro and macro averaging.

Next chapter presents the data analysis and the results and discussion of ML algorithms application.

4 Experience Evaluation

In the previous chapter, the equipment and the strategy adopted to acquire data were described. This chapter presents the analysis of data in Section 4.1, the results of applying the ML Algorithms in Section 4.2 and a discussion of the previous two sections in Section 4.3

First, the data were analysed and it was thought the best strategy to use this data. It was assumed that everyday occupants always kept the doors and windows closed, pressed the button when arriving and leaving. Every time an occupant was in the room, the light was on otherwise the light was off. The data was changed manually in cases that these assumptions didn't occur and was verified.

4.1 Data Analysis

Data acquisition did not occur on the same period because some sensors were already in the laboratory but the others had to be ordered. So, a consensus was reached that it was preferable to start acquiring data and test with it the ML algorithms. When the other sensor arrive, a new acquisition system would be implemented to acquire the new data.

Figure 4.1 represents 2880 samples of temperature raw data. The blue data were acquired on November 11, 2017, and represent the difference between indoor temperature and outdoor temperature without occupants. The red data were acquired on November 10, 2017, and represent the difference between indoor temperature and outdoor temperature with occupants. The graph is in Celsius degrees by hours. In this figure, it is possible to conclude that the difference is higher with occupants than without occupants. The first occupant arrived around 9 AM and the last occupant left around 6 PM. The exceptions are around 10 AM and around 12 AM. The first exception happens because the incidence of the sun in the room which on this period of test occurs at this hour, increasing the indoor temperature. The second happens because the occupants left the room to have lunch.

Figure 4.2 represents 4320 samples of CO2 raw data. The blue data were acquired on

December 17, 2017, the red data were acquired on December 26, 2017, and the yellow data were acquired on December 18, 2017. They represent the indoor CO2 levels during a day without any occupant, with one occupant and with more than one occupant in the room, respectively. The graph is in ppm by hours. Through the analysis of the red line, it is possible to see that when an occupant arrived the CO2 levels increased to around 500 ppm. This increase was more noticeable in the yellow line when more than one occupant was in the room, increasing to around 2000 ppm. In days without occupants, as shown by the blue line, the levels were between 400 and 450 ppm.

Figure 4.3 represents 2880 samples of sound amplitude raw data. The blue data were acquired on December 17, 2017, and the red data were acquired on December 18, 2017. They represent the sound amplitude during a day without and with occupants in the room, respectively. The data are in dB by the hour. In this figure, it is possible to see that the differences are not very large, having the same maximum with and without people.

Figure 4.4 represents 2880 samples of light intensity raw data. The blue data were acquired on December 17, 2017, and the red data were acquired on December 26, 2017. They represent the light intensity during a day without and with occupants in the room, respectively. The data are in lux by the hour. In this figure, it is possible to see that when an occupant arrived, close to 10 AM, the lux increased to around 110 and when he left, close to 6 PM, the lux decreased to zero on the red line. In the blue line, the same problem caused by solar incidence around 10 AM occurs like in the temperature graph. This proves the increase of temperature when there are no occupants around this hour.

Analysing the data from other days, it was possible to see that some data have outliers (an observation point that is distant from other observations) and some noise (data with a large amount of additional meaningless information).

4.1.1 Outlier Filtering

The approach to deal with outliers is in Algorithm 3, in Appendix D [41]. Given an array, x of length m, creates a new array y with m length, and sets the first element equal to the first element of array x. Then, each element of the array x is verified and if the division of the new element by the previous is in a threshold range defined by the user. If the answer is positive, the element in position i of the new array is equal to the element in position i of the old array. If the answer is negative, the new element in position i of the new array is equal to the previous element in the new array.

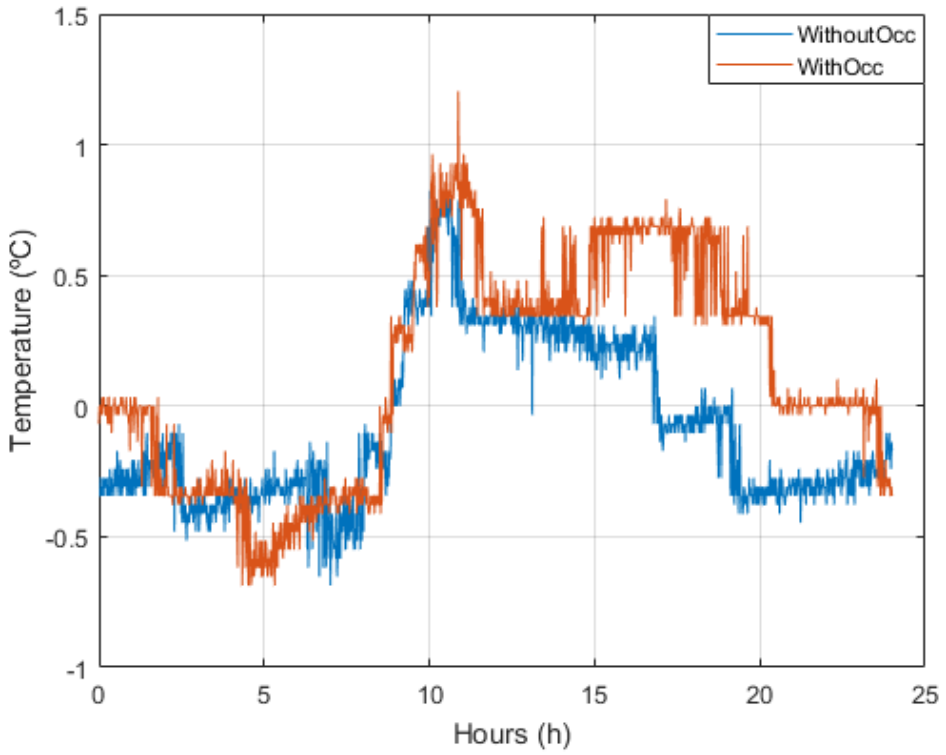


Figure 4.1: Raw data of temperature sensor by Node1 acquired on November 10 and 11, 2017.

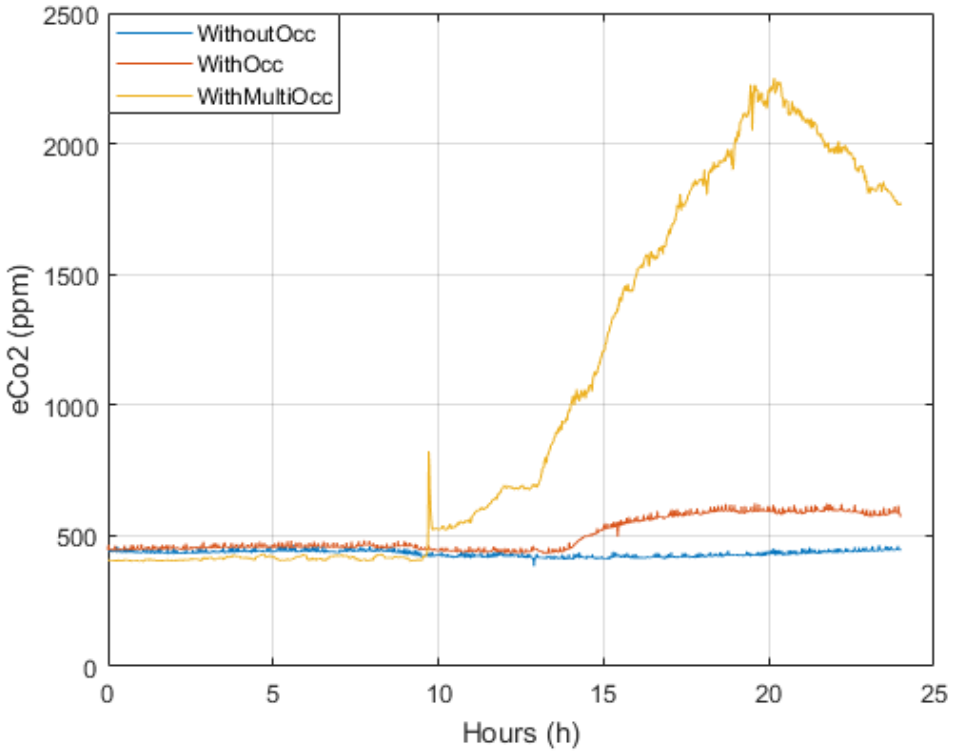


Figure 4.2: Raw data of CO2 sensor by Node3 acquired on December 17, 18, and 26, 2017.

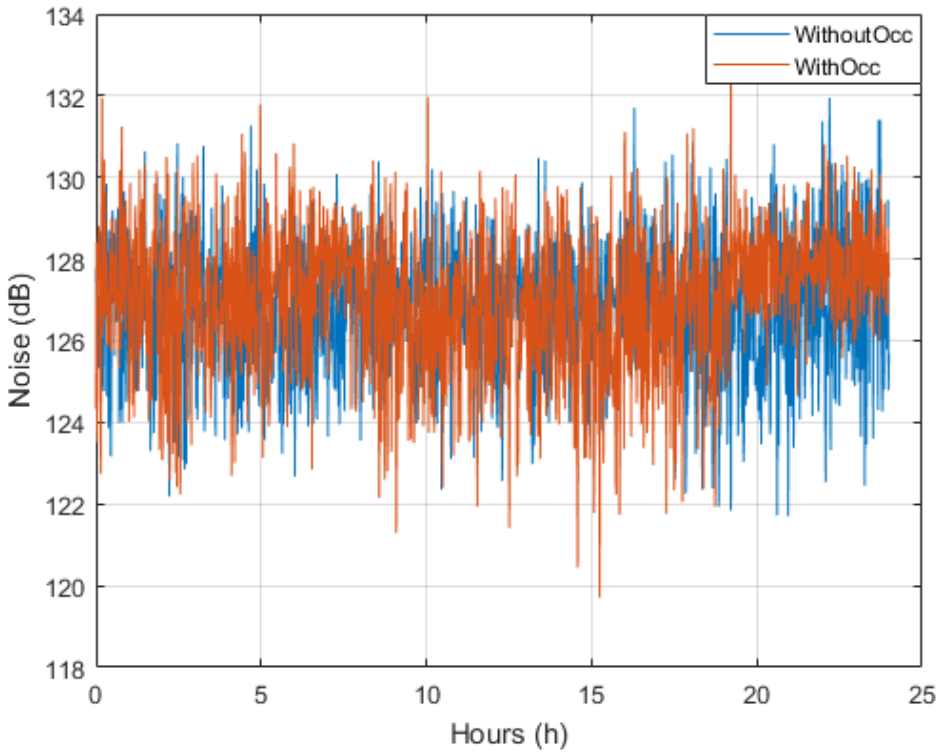


Figure 4.3: Raw data of amplitude collected from sound sensor of Node2 acquired on December 17 and 18, 2017.

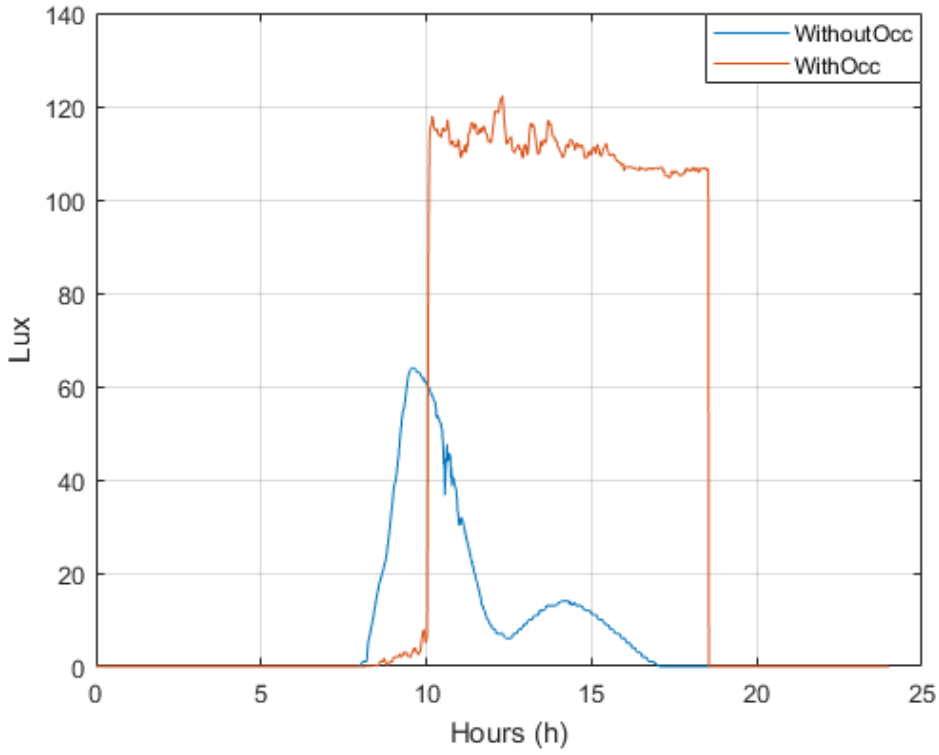


Figure 4.4: Raw data of light sensor of Node3 acquired on November 17 and 26, 2017.

4.1.2 Noise Filtering

A Low-Pass Filter (LPF) was applied to deal with noisy data. By definition, a LPF is a circuit that offers easy passage to low-frequency signals and difficult passage to high-frequency signals. This type of filters is commonly used for removing noise from a signal using simple Resistor-Capacitor (RC) architecture, i.e. the circuit consists of a resistor in series with a load and a capacitor in parallel with the load. This type of filter is known commonly as a first-order filter because it has only one reactive component, the capacitor, in the circuit. Equation 4.1 gives the discrete implementation of the first order LPF [24].

$$y[n] = \alpha x[n] + (1 - \alpha)y[n - 1] \tag{4.1}$$

where $\alpha = \frac{\Delta_T}{RC+\Delta_T}$ is the smoothing factor, y is the filtered output, x is the input, n is the sample index and Δ_T is the sampling period. R and C are the value of resistor and capacitor, respectively. Because the solution sought was via software, the RC had to be changed. In a LPF, signals with a frequency lower than a certain cutoff frequency pass and signals with frequencies higher than the cutoff frequency are attenuated. With a cutoff frequency (f_c) it is possible to calculate the smoothing factor, α. Equation 4.2 gives this relation between them.

$$f_c = \frac{1}{2\pi RC} = \frac{\alpha}{(1 - \alpha)2\pi\Delta_T} \tag{4.2}$$

where $RC = \Delta_T\frac{(1-\alpha)}{\alpha}$. Calculating the next value through this smoothing factor and the previous value, it was possible to reduce the data noise, making the transitions between samples slower and pleasant.

4.1.3 Results of Filtering

Figure 4.5 illustrates the filtering (blue line) and the raw data of temperature (red line). This solution was applied with the other data too. The smoothing factor used was 0.15. This value and the outlier threshold was obtained experimentally verifying what value gave the best filtering. With this, the data are cleaner with less noise and without outliers.

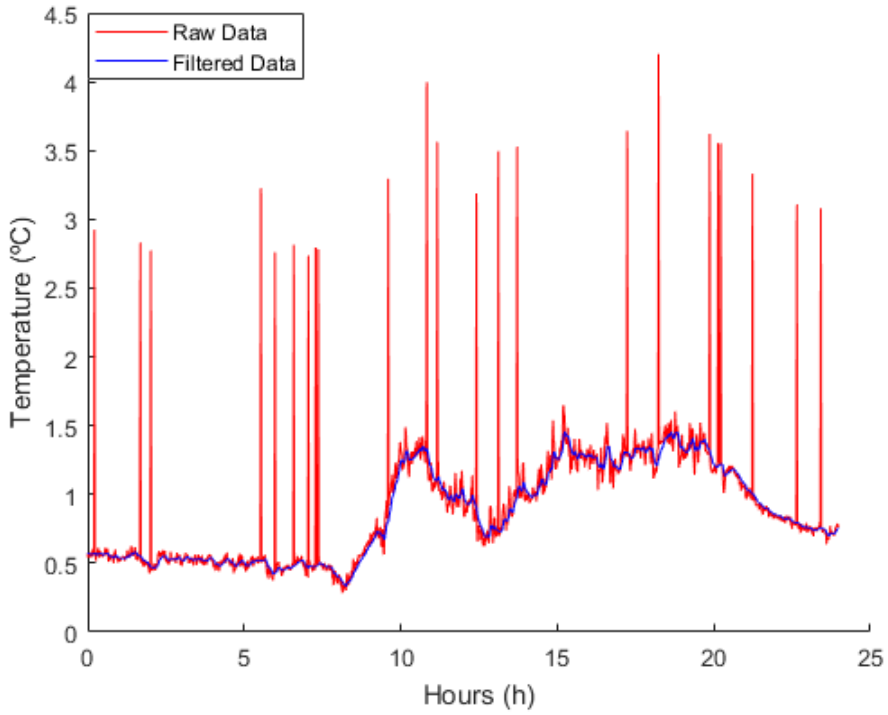


Figure 4.5: Outlier filter and Low-Pass Filter in the temperature data.

4.2 Machine Learning Algorithms Application

In the first stage, data are pre-processed, the outliers were removed, and a filter was applied to remove the noisy data. Next, the ML techniques were applied to this new data. The main purpose of using ML is to find patterns in the data to arrive at a classifier that can perform a correct prediction of the room occupants. First, binary case was analysed and, if applicable, the multi-class case was analysed.

4.2.1 Dataset

The dataset used was acquired in 10 consecutive days. As previously said, the temperature data and data from the other sensors had different collected days. The temperature data started on November 10, 2017, at 00h00min and end on November 19, 2017, 23h59min. The other data used started on December 14, and end on December 23, 2017. The datasets have 14400 training examples. This dataset was divided, 60% to train the classifier and 40% to test. Tables 4.1, 4.2, and 4.3 present the number of samples in each class. It is possible to see that the dataset has skewed classes, having more examples in one classe than anothers. In Appendix C, it is possible to see the dataset plot.

Table 4.1: Number of samples in the temperature dataset for the binary problem.

	Training set	Testing set	Total examples
Positive examples	2444	1108	3552 (≃25%)
Negative examples	6196	4652	10848 (≃75%)
Total dataset	8640 (60%)	5760 (40%)	14400 (100%)

Table 4.2: Number of samples in other datasets for the binary problem.

	Training set	Testing set	Total examples
Positive examples	2273	1089	3362 (≃23%)
Negative examples	6367	4671	11038 (≃77%)
Total dataset	8640 (60%)	5760 (40%)	14400 (100%)

Table 4.3: Number of samples in temperature dataset for the multi-class problem.

	Training set	Testing set	Total examples
Class 1 (y=1)	279	176	455 (≃3%)
Class 2 (y=2)	476	289	765 (≃5%)
Class 3 (y=3)	928	341	1269 (≃10%)
Class 4 (y=4)	446	302	748 (≃5%)
Class 5 (y=5)	315	0	315 (≃2%)
Class 6 (y=0)	6196	4652	10848 (≃75%)
Total dataset	8640 (60%)	5760 (40%)	14400 (100%)

4.2.2 Approach to Classification

The LR classifier was trained with $\lambda = 0$ and the $threshold = 0.5$. The SVM classifier was used with Radial-Basis Function (RBF) Kernel. This Kernel (Equation 4.3) only needs two parameters, C is a penalty cost parameter and γ is the inverse of the standard deviation [18].

$$k(x, y) = e^{(-\gamma||x-y||^2)} \tag{4.3}$$

This classifier was trained with $C = 1$ and $\gamma = 0$. The NN classifier was trained with $\lambda = 0$, $hiddenunits = 1$ (number of hidden layers units), and three layers (input layer, hidden layer, and output layer).

To obtain the highest F-Score some parameters could be changed. It is possible to

add polynomial features in LR and NN to increase the complexity of the classifier and in some cases to improve the performance. Algorithm 4, in Appendix D, presents how these parameters were calculated. For LR, a represents the λ parameter (λ_{vec} = [0 0.001 0.003 0.01 0.03 0.1 0.3 1 3 10 30 100]), b represents the polynomial degree ($poly_d = 1:10$), and c represents the $threshold = 0.5$. For the SVM, a represents the γ parameter (γ equal to previous λ), b represents the polynomial degree $poly_d = 1$, and c represents the C parameter (C = [0.1 0.3 1 3 10]). For the NN, a represents the λ parameter (λ = [0 0.01 0.1 1 10]), b represents the polynomial degree ($poly_d = 1:5$), and c represents the number of hidden layer units ($hiddenunits$ = [1 2 3 5]).

Algorithm 4 calculates the classifiers with all the possible combinations with this parameters and returns the parameters with the best F-Score.

4.2.3 Results of Classifiers

Binary Problem Results

The binary problem aims to determine whether an occupant was in room ($y = 1$) or not ($y = 0$). Table 4.4 presents the results by applying the classifiers with the dataset without changing the parameters.

Table 4.4: F-Score results of applying ML algorithms into data to a binary problem.

	LR	**SVM**	**NN**
Temperature	85.35%	87.05%	86.60%
Carbon Dioxide	12.40%	31.62%	0%
Noise	1.81%	2.48%	0%
Light	95.40%	93.59%	93.11%

Analysing the results of the binary problem without changing the parameters the classifier that give the best F-Score was the SVM classifier for all the features except light sensor. The NN classifier had the lowest f-score in almost the cases. With one feature and with this dataset, the NN classifiers have the lowest F-Score. In some cases the result was 0%.

In the PC on which the tests occurred, on average, training with a NN takes 4.79 seconds, compared to LR that was 0.16 seconds and SVM that was 1.80 seconds.

It was expected that the F-Score result of noise data will be low to detect people. The occupants passed most of the time in silence. In future works, it is advisable to change the

acquired sample interval to 1 second to detect effectively the noise of occupants in cases that it applies.

The expected F-Score of the light data was high. But it can only detect whether the occupants are present or not.

As a larger number of occupants in the room usually results in higher CO2 concentrations, this data can detect the number of occupants, as can be seen in Figure C.7. However, because the room does not have a good air flow rate, this concentration reduces slowly and can take hours to stabilise. So the approach did not perform well. One possible approach is to calculate the derivative and then check whether if it has a certain slope so as to determine if an occupant arrived or left. To do this, another type of data would have to be collected. The problem is that there are not sufficient and suitable days with exactly one occupant, two, and so on in a room, at the same time and the same amount of time to have a large dataset to train the classifier.

The temperature data suffers from the same problem that the CO2 data. It is impossible to have a fixed number of occupants in the room. So it was important to have a dataset with more data for calculating the time taken for temperature to stabilise to improve the results. However, even without this knowledge, the results were satisfactory, around 87%, to detect the presence of occupants.

Tables 4.5, 4.6 and 4.7 present the parameters and results that give the highest F-Score by applying a LR, SVM, and NN classifier with the dataset, respectively.

When performing a new F-Score and changing the parameters and the polynomial degree, some features show a significant improvement like CO2. Light reached 99%. The temperature and noise did not show a significant growth.

Table 4.5: F-Score results of parameters that perform the highest score for LR binary classifiers.

	λ	$Poly_d$	**F-Score**
Temperature	3	2	87.42% (↑2.07%)
Carbon Dioxide	0.01	8	33.42% (↑21.02%)
Noise	0.003	3	2.72% (↑0.91%)
Light	0.001	5	99.22% (↑3.82%)

Table 4.6: F-Score results of parameters that perform the highest score for SVM binary classifiers.

	γ	C	F-Score
Temperature	0.3	3	87.43% (↑0.38%)
Carbon Dioxide	10	1	49.36% (↑17.74%)
Noise	30	10	4.87% (↑2.39%)
Light	100	0.3	99.22% (↑5.63%)

Table 4.7: F-Score results of parameters that perform the highest score for NN binary classifiers.

	λ	$Poly_d$	h_{units}	F-Score
Temperature	0	1	5	87.43% (↑0.83%)
CO2	0.1	4	5	46.03% (↑46.03%)
Noise	0.01	5	5	4.65% (↑4.65%)
Light	0	4	1	99.22% (↑6.11%)

Despite the fact that the CO2 levels can tell the number of occupants, the data analyse has to suffer changes before applying a ML technique. The noise had a low F-Score and the light indicated only the presence or absence of occupants. For these reasons, only the temperature was analysed in a multi-class problem.

Multi-class Problem Results

The multi-class problem aims to determine the number of occupants in a room. During this work, there were five occupants in the room mostly but sometimes there were more. In these cases, it was assumed five occupants because this occurs in one-off situations. Analysing Table 4.4, it is possible to verify that only the temperature data can have good results in a multi-class problem. The F-Score (micro) gives the accuracy, so it was decided to use the F-Score (macro) to evaluate the classifier. Table 4.8 presents the results. From Tables 4.9, 4.10 and 4.11, it is possible to verify the best parameters and the F-Score (macro) results for the LR, SVM, and NN classifiers, respectively.

Unfortunately, it was not possible to count the number of occupants using the temperature data (around 25% without changing the parameters). In this case, all the classifiers give

almost the same value. Changing the parameters, LR gives the best results for this dataset, around 34%. It was also assumed that the human body surface had a uniform temperature and a uniform heat production, but it is not true. The human body has a distinct physical shape and also has complex thermo-physiological properties. However, it is difficult to include those factors into a numerical constant in an indoor climate.

Table 4.8: F-Score (macro) results by applying ML Algorithms with the dataset to a multi-class problem.

	Temperature
LR	24.39%
SVM	24.90%
NN	24.80%

Table 4.9: F-Score (macro) results of parameters that perform the highest score for LR multi-class classifiers.

	λ	$Poly_d$	**F-Score (macro)**
Temperature	0.01	8	34.34% (↑9.95%)

Table 4.10: F-Score (macro) results of parameters that perform the highest score for SVM multi-class classifiers.

	γ	**C**	**F-Score (macro)**
Temperature	100	1	28.65% (↑3.75%)

Table 4.11: F-Score (macro) results of parameters that perform the highest score for NN multi-class classifiers.

	λ	$Poly_d$	h_{units}	**F-Score (macro)**
Temperature	0.1	5	5	28.65% (↑3.85%)

4.3 Summary of Chapter 4

The data acquisition occurred in some weeks between November and December. First, the data were analysed and the best strategy to use this data was studied. It was verified that some data had outliers and noise. Two filters were applied to reduce this problem, an outlier filter, and a LPF.

Next, ten consecutive days were selected to a total of 14400 data to each dataset and the ML techniques were applied. The dataset was divided into two, a training set and a testing set. The first was to train the classifier and the second was to test its performance with a ratio of 0.6/0.4, respectively.

Despite the fact that the LR was training more quickly, the choice goes to SVM. The time difference between them is not significant with this amount of data, and the results were slightly better. The NN does not become a choice because this classifier has a large training time compared to the others because it needs a huge computing load, which increase the calculation time. With more number of hidden layer units and polynomial features it has, more time it will need.

As features, light and temperature presented good results to detect the presence of occupants. CO2 has to undergo another type of analyses to be used with ML techniques. In this case of test, the noise did not present good results because the occupants passed most of the time in silence.

Temperature was the only feature that was applied a multi-class classifier. The results was not satisfied in order to detect the number of occupants.

The F-Score can be improved by changing the parameters or the polynomial degree, making more complex functions. In some cases, this improvement was notorious, while in other cases it was not.

5 Additional Contributions

This chapter presents additional work resulting from this thesis. The LCT-Envboard is presented in Section 5.1 with an overview, a pinout description, the built-in sensors, and the design. Section 5.2 describes some tests that were done.

5.1 LCT - EnvBoard

This thesis has identified the need to develop a prototype board to collect environmental data in the LCT-CISUC.

The main requirements for this board were:

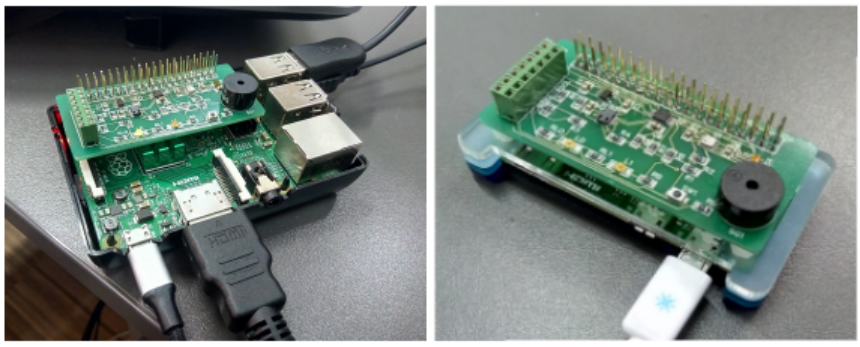
- Raspberry compatibility: The RPi and the Arduino boards are widely used in academic and home environments. There is a large community support and the cost is low. The RPi has been selected because of the fact that it contains a microprocessor that can be used as if it was a computer (see Section 3.2.2).
- Use of digital sensors: Digital sensors have been selected instead of analogue sensors because they have a controller that makes the calculations and gives the final user the correct measurement value. Analogue devices require this processing and conversion on the part of the developer. Analogue devices also need ADC converters that RPi does not contain. These sensors are also more prone to noise.
- Types of data: The types of data that sensors must collect are temperature, humidity, light, and air quality monitoring.
- Output alert: There is the need to have a device to emit an alert in certain cases. The choice was a led and a buzzer.
- Low power consumption: A board with low power consumption to continually gather data.

The LCT-EnvBoard emerged to address these requirements.

5.1.1 Overview

The LCT-EnvBoard, in Figure 5.1, allows to acquire environmental data such as temperature, humidity, barometric pressure, Volatile Organic Compounds (VOC), TVOC, equivalent CO2, and the infrared and full spectrum light. Some of the main pollutants of today, described in Section 3.1, can be monitored by this board. It only allows a maximum consumption of 150mA and it has two outputs to emit alerts. The board can be used like a RPi Hat (an add-on board for the RPi GPIO pins) in every model B or as a breakout board (a simple Printed Circuit Board (PCB) containing one or more Integrated Circuit (IC) required to use this device with other devices) via I2C communication. The board head pins on top allow the use of the GPIO pins of RPi.

Figure 5.1: LCT-EnvBoard in RPi2 and RPiZero



5.1.2 Pinouts

In Figure 5.2, it is possible to see the sensors and the pinouts of the header board:

- VCC - It can be used to output (RPi Hat) or input (Breakout board). If the board was used like a hat, it does not connect other voltage sources on it. The voltage input and voltage output are 5V.
- 3.3V - This is the 3.3V output from the voltage regulator. Maximum current output from this is 150mA.
- GND - Common ground for power and logic.
- SCL - I2C clock pin.
- SDA - I2C data pin.
- MCP9808Interrupt - This is the interrupt/alert pin from the MCP9808.

- CCS811Wake - This is the wake-up pin for the sensor.
- CCS811Reset - This is the reset pin. When it is pulled to ground the sensor resets itself.
- CCS811Interrupt - This is the interrupt-output pin from CCS811. It is 3V logic and can be used to detect when a new reading is ready or when a reading gets too high or too low.
- TSL2591Interrupt - This is the interrupt pin from the TSL2591.
- LED, Buzzer, and Button - Connect to any RPi GPIO pin.

Sensor	I2C Address
BME680	0x77
CCS811	0x5A
MCP9808	0x18
TSL2591	0x29

Pin 1	Vcc	Pin 2	GND
Pin 3	3.3V	Pin 4	GND
Pin 5	SDA	Pin 6	SCL
Pin 7	CCS811Reset	Pin 8	TSL2591Interrupt
Pin 9	CCS811Wake	Pin 10	LED
Pin 11	CCSInterrupt	Pin 12	Buzzer
Pin 13	MCP9808Interrupt	Pin 14	Button

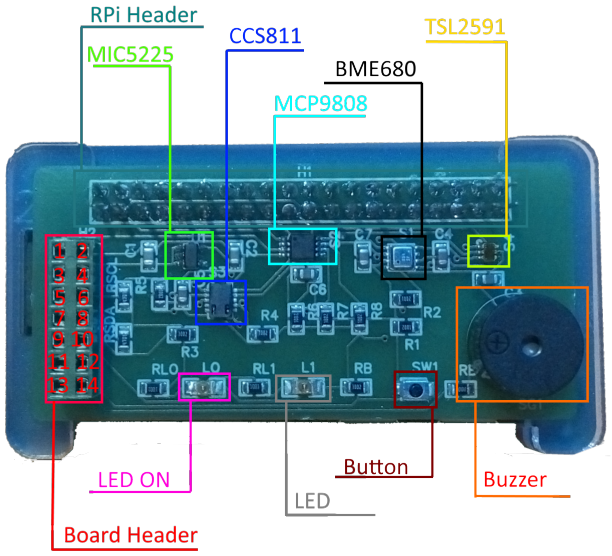


Figure 5.2: Pinout of LCT-EnvBoard.

5.1.3 Hardware

The sensors used in this thesis provided precise measurements. During pre-research, they presented highly accurate data and low power consumption. So, some of them were incorporated on this board. Another reason for this choice is that sensors are well documented and have APIs for Python and AVCr language that can be ported to another software. In this work, some of these libraries were ported to C/C++. The choice of sensors fell on BME680 by Bosch, TSL2591 by AMS, CCS811 by AMS, and MCP9808 by Microchip.

The TSL2591 and the CCS811 sensor have already been presented in Section 3.2.1. The BME680 is an Integrated Environmental Unit (IEU) that allows measuring temperature,

humidity, barometric pressure, and VOC gas. It contains a small MOX sensor that heats and changes resistance based on the VOC in the air [76]. The MCP9808 is an I2C digital temperature sensor. It is highly accurate with a typical accuracy of ±0.25°C over the sensors -40°C to +125°C range and precision of +0.0625°C [44]. The choice of two sensors that can measure the temperature was because the MCP9808 is more accurate and can be used to calibrate the CCS811 sensor with more precise results. Another reason for the choice of these sensors choice was that all allow clock-stretching [16]. Basically, the clock stretching occurs when a slave device forces the clock to slow down at times to delay the master sending more data. The CCS811 on the RPi only works if the I2C speed is decreased.

The MIC5225 is a linear regulator by Microchip. All sensors picked work with 3.3V, so this regulator is suitable to give a clean and stable 3.3V source [43].

As a input, the board has a button switch. As output a LED and an active buzzer (audio signalling device with an internal oscillator) are there.

5.1.4 Board Design

The board layout was produced in EasyEDA. EasyEDA is a web platform that enables hardware engineers to design schematics, simulate, and print circuit boards [34]. The circuit schematics and the PCB are shown in Appendix E.

5.2 Preliminary Tests and Troubleshooting

To test if all components work as expected, the following tests were performed:

Test 1 – Power on, voltage measuring, and I2C detect. The first test was power on the board on RPi. On the board, there is a LED that turns on if it receives the correct voltage. Next, the outputs of VCC and 3.3V were tested. The results were 4.70 and 3.27V, respectively. Finally, in Linux Console was typed 'i2cdetect -y 1' to verify if every sensor was detected. This test was successful.

Test 2 – Button, LED and buzzer. Next, a wire was connected to the buzzer and LED and another in the switch to the RPi. A script was run to test to check whether the buzzer makes a sound and the LED turns on when the button is pressed. This test was successful.

Test 3 – MCP9808 Sensor. The third test was to use the MCP9808 sensor. A room was heated and cooled to test if the measurement was following its variation. This test was successful.

Test 4 – CCS811 Sensor. The fourth test was to use the CCS811 sensor. It was breathed up in the sensor to verify if the CO2 concentration is increasing. This test was successful.

Test 5 – TSL2591 Sensor. The fifth test was to use the TSL2591 sensor. A lamp was directed to the sensor and the light was turned on and off. This test was successful.

Test 6 – BME680. The final test was to use the BME680 sensor. The sensor gets values but this value was not true. The temperature returned was -138^{o}C, which is not correct. This test failed.

It was verified that the heat in board increased because of the RPi2. This increase was because the sensor was on top of the heat sink of the RPi2 tested. The RPizero tested had an acrylic box, so this was not verified. It is possible to use the board like a breakboard and to resolve the problem in RPi2.

The CCS811 sensor uses I2C clock stretching. The RPi cannot do this without drastically reducing the I2C speed. (for more information see Ref. [2]).

The sensors with a Metal Oxide (MOX), CCS811, and BME680 need to be run for 48 hours during the first time use to 'burn it in', and then for 30 minutes in the desired mode every time the sensor is in use. The reason is that the sensitivity levels of the sensor will change during early use and the resistance will slowly rise over time as the MOX warms up to its baseline reading.

Temperature and humidity in BME680 sensor did not give the correct results. The problem can arise from the library or from the sensor. The other values, like a gas resistor, vary but it is difficult to say that the values are correct when the temperature is not correct.

This board only permits direct connection with 3.3V. However, with a bi-directional logic level converter (a small device that safely steps down 5V signals to 3.3V and steps up 3.3 to 5V at the same time) can be used by 5V devices.

5.3 Summary of Chapter 5

This chapter presented the LCT-EnvBoard. It provides a complete environmental gathering device, which is composed of a LED, a buzzer, a switch, and four sensors that can acquire the temperature, humidity, barometric pressure, altitude, eCO2, TVOC, VOC, and the infrared and full spectrum light. It has a low power consumption limit of less than 150mA. This board can be connected to any device with the I2C protocol. The purpose of this board is to fill a gap that exists in LCT-CISUC.

6 Conclusions and Future Work

Nowadays, companies and researchers are working on enhancing the quality of life of citizens, using the IoT paradigm to reach the idea of building smart environments. In this context, it would be beneficial to have mechanisms to predict or estimate the occupancy of indoor environments to make smart decisions about how to self-adapt to the environmental conditions. Some researches have been done in this area; however, there are some open issues, such as privacy, cost, and data storage. In this work, research was performed to accomplish occupancy detection with non-intrusive devices using sensors such as temperature, noise, CO2, and light.

A functional system, made up of a device to gather and process environmental data and to analyse the data patterns over the collected data regarding people occupancy in indoor environments using ML techniques, was tested in this research. The analysis allows to affirm that with features like noise data in working environments the performance of the recognition system might be degraded. However, with features like temperature, CO2, and light data, it will be possible to improve the detection of occupants. Thus, the objectives defined at the beginning of this research were fulfilled.

The main contributions of this research are: (i) a low-cost prototype to collect and process environmental data for indoor scenarios, which could be used by the upcoming researchers to easily acquire data; (ii) an analysis of occupancy for indoor environments using a ML approach; and (iii) a study of the most suitable sensors to determine occupancy in indoor environments.

For future works, it is necessary to further study about the full correlation between the environmental data used in this research. A starting point could be the analysis of features and their impact on the model using principal component analysis (PCA). Additionally, an analysis of the performance and accuracy of other classifiers, such as unsupervised learning, in ML mechanism could be performed. Regarding the prototype, it is recommended to test the sensor SGP30 over the CCS811, because for a theoretical perspective the former is more

stable and does not need clock stretching.

On a personal level, this work has given me the capacity to carry out research work with autonomy, integrating knowledge acquired along the course besides to gain new knowledge (such as IoT, prototyping, ML, and pattern recognition techniques) and promoting the development of my capacity for critical analysis, synthesis, and creativity.

> *"An interesting direction that researchers in the future may consider is not only the ability to adjust an environment to fit an individual's preferences, but to use the environment as a mechanism for influencing change in the individual."*
>
> — Cook, Diane J., and Sajal K. Das.

7 Bibliography

[1] Lady Ada and Tony DiCola. Thermistor, 2017. `https://learn.adafruit.com/thermistor` [Online: accessed November 16, 2018].

[2] Dean Miller (Adafruit). Raspberry pi wiring & test, January 2018. `https://learn.adafruit.com/adafruit-ccs811-air-quality-sensor/raspberry-pi-wiring-test` [Online; accessed February 1, 2018].

[3] Chris Adami, C Titus Brown, and W Kellogg. Evolutionary learning in the 2d artificial life system 'avida'. In *Artificial life IV*, volume 1194, pages 377–381. The MIT Press Cambridge, MA, 1994.

[4] AMS AG. Ccs811 – ultra-low power digital voc sensor for monitoring indoor air quality (iaq). `http://ams.com/eng/Products/Environmental-Sensors/Air-Quality-Sensors/CCS811` [Online; accessed February 1, 2018].

[5] AMS AG. Tsl25911 ambient light sensor. `http://ams.com/eng/Products/Light-Sensors/Ambient-Light-Sensors/TSL25911` [Online; accessed February 1, 2018].

[6] AINSI/ASHRAE. Ainsi/ashrae standard 62.1- 2013 ventilation for acceptable indoor air quality. *American Society of Heating, Ventilating, and Air Conditioning Engineers*, 2013.

[7] Ala Al-Fuqaha, Mohsen Guizani, Mehdi Mohammadi, Mohammed Aledhari, and Moussa Ayyash. Internet of things: A survey on enabling technologies, protocols, and applications. *IEEE Communications Surveys & Tutorials*, 17(4):2347–2376, 2015.

[8] Klaithem Al Nuaimi and Hesham Kamel. A survey of indoor positioning systems and algorithms. In *Innovations in information technology (IIT), 2011 international conference on*, pages 185–190. IEEE, 2011.

[9] Arduino. serial. https://www.arduino.cc/reference/en/language/functions/communication/serial [Online; accessed February 3, 2018].

[10] Mohsen Attaran. The internet of things: Limitless opportunities for business and society. *Journal of Strategic Innovation and Sustainability Vol*, 12(1):10–29, 2017.

[11] Pierre Baldi, Søren Brunak, Yves Chauvin, Claus AF Andersen, and Henrik Nielsen. Assessing the accuracy of prediction algorithms for classification: an overview. *Bioinformatics*, 16(5):412–424, 2000.

[12] Robert Berwick. An idiot's guide to support vector machines (svms), 2003. http://web.mit.edu/6.034/wwwbob/svm-notes-long-08.pdf [Online; accessed November 22 2017].

[13] Harsh Bhasin and Surbhi Bhatia. Application of genetic algorithms in machine learning. *IJCSIT*, 2(5):2412–2415, 2011.

[14] C Bishop. Pattern recognition and machine learning (information science and statistics), 1st edn. 2006. corr. 2nd printing edn. *Springer, New York*, 2007.

[15] Léon Bottou. Large-scale machine learning with stochastic gradient descent. In *Proceedings of COMPSTAT'2010*, pages 177–186. Springer, 2010.

[16] I2C Bus. Clock stretching. https://www.i2c-bus.org/clock-stretching [Online; accessed February 5, 2018].

[17] Chih-Chung Chang and Chih-Jen Lin. Libsvm – a library for support vector machines. https://www.csie.ntu.edu.tw/~cjlin/libsvm [Online; accessed February 3, 2018].

[18] Yi-Wei Chen and Chih-Jen Lin. Combining svms with various feature selection strategies. In *Feature extraction*, pages 315–324. Springer, 2006.

[19] Ron Chepesiuk. Missing the dark: health effects of light pollution. *Environmental Health Perspectives*, 117(1):A20, 2009.

[20] Bertrand Clarke, Ernest Fokoue, and Hao Helen Zhang. *Principles and theory for data mining and machine learning*. Springer Science & Business Media, 2009.

[21] European Commission. Green paper on future noise policy (com(96) 540), 1996.

[22] Wikipedia contributors. Artificial neural network — Wikipedia, the free encyclopedia, 2017. https://en.wikipedia.org/wiki/Artificial_neural_network [Online; accessed December 20 2017].

[23] Wikipedia contributors. Electromagnetic spectrum — Wikipedia, the free encyclopedia, 2018. https://en.wikipedia.org/wiki/Electromagnetic_spectrum [Online; accessed February 1 2018].

[24] Wikipedia contributors. Low-pass filter — Wikipedia, the free encyclopedia, 2018. https://en.wikipedia.org/wiki/Low-pass_filter#Simple_infinite_impulse_response_filter [Online; accessed February 6 2018].

[25] Wikipedia contributors. Sound — Wikipedia, the free encyclopedia, 2018. https://en.wikipedia.org/wiki/Sound/ [Online; accessed February 1 2018].

[26] Diane J Cook, Juan C Augusto, and Vikramaditya R Jakkula. Ambient intelligence: Technologies, applications, and opportunities. *Pervasive and Mobile Computing*, 5(4):277–298, 2009.

[27] Diane J Cook and Sajal K Das. How smart are our environments? an updated look at the state of the art. *Pervasive and mobile computing*, 3(2):53–73, 2007.

[28] Christos Davatzikos, Kosha Ruparel, Yong Fan, DG Shen, M Acharyya, JW Loughead, RC Gur, and Daniel D Langleben. Classifying spatial patterns of brain activity with machine learning methods: application to lie detection. *Neuroimage*, 28(3):663–668, 2005.

[29] Instituto de Telecomuicações. Project: Mobiwise: from mobile sensing to mobility advising, 2017. https://www.it.pt/Projects/Index/4493 [Online: accessed January 16, 2018].

[30] Peter J Denning. *The Invisible future: the seamless integration of technology into everyday life.* McGraw-Hill, Inc., 2001.

[31] Mischa Dohler, Ignasi Vilajosana, Xavi Vilajosana, and Jordi Llosa. Smart cities: An action plan. In *Proc. Barcelona Smart Cities Congress, Barcelona, Spain*, pages 1–6, 2011.

[32] Pedro Domingos. A few useful things to know about machine learning. *Communications of the ACM*, 55(10):78–87, 2012.

[33] Stephan Dreiseitl and Lucila Ohno-Machado. Logistic regression and artificial neural network classification models: a methodology review. *Journal of biomedical informatics*, 35(5):352–359, 2002.

[34] EasyEDA. Easyeda. `https://easyeda.com` [Online; accessed February 3, 2018].

[35] Fabio Falchi, Pierantonio Cinzano, Christopher D Elvidge, David M Keith, and Abraham Haim. Limiting the impact of light pollution on human health, environment and stellar visibility. *Journal of environmental management*, 92(10):2714–2722, 2011.

[36] The Eclipse Foundation. The three software stacks required for iot architectures, 2016. `https://insights.ubuntu.com/wp-content/uploads/4a5b/Eclipse-IoT-White-Paper-The-Three-Software-Stacks-Required-for-IoT-Architectures.pdf` [Online: accessed January 20, 2018].

[37] Kevin J Gaston, Jonathan Bennie, Thomas W Davies, and John Hopkins. The ecological impacts of nighttime light pollution: a mechanistic appraisal. *Biological reviews*, 88(4):912–927, 2013.

[38] Kerry Glover. Developing a custom lux equation, August 2011. `http://ams.com/eng/content/view/download/145096` [Online; accessed February 1, 2018].

[39] Yanying Gu, Anthony Lo, and Ignas Niemegeers. A survey of indoor positioning systems for wireless personal networks. *IEEE Communications surveys & tutorials*, 11(1):13–32, 2009.

[40] Peter Harrington. *Machine learning in action*, volume 5. Greenwich, CT: Manning, 2012.

[41] Victoria Hodge and Jim Austin. A survey of outlier detection methodologies. *Artificial intelligence review*, 22(2):85–126, 2004.

[42] Guang-Bin Huang, Qin-Yu Zhu, KZ Mao, Chee-Kheong Siew, Paramasivan Saratchandran, and Narasimhan Sundararajan. Can threshold networks be trained directly? *IEEE Transactions on Circuits and Systems II: Express Briefs*, 53(3):187–191, 2006.

[43] Micrel Inc. Mic5225, July 2008. `http://ww1.microchip.com/downloads/en/DeviceDoc/mic5225.pdf` [Online; accessed February 1, 2018].

[44] Microchip Technology Inc. Mcp9808, August 2011. http://ww1.microchip.com/downloads/en/DeviceDoc/25095A.pdf [Online; accessed February 1, 2018].

[45] Jalite. Escape route lighting. http://www.photoluminescent-signs.com/en/jalite/photoluminescent-safety-signs/fire-safety-communication/escape-route-lighting [Online; accessed January 22, 2018].

[46] K Jensen and Edward Arens. Acoustical quality in office workstations, as assessed by occupant surveys. *University of California Berkeley, USA*, 2005.

[47] Patrick Di Justo. Raspberry pi or arduino uno? one simple rule to choose the right board, December 2015. https://makezine.com/2015/12/04/admittedly-simplistic-guide-raspberry-pi-vs-arduino [Online; accessed February 3, 2018].

[48] Leslie Pack Kaelbling, Michael L Littman, and Andrew W Moore. Reinforcement learning: A survey. *Journal of artificial intelligence research*, 4:237–285, 1996.

[49] Vladimir Leonov. Thermoelectric energy harvesting of human body heat for wearable sensors. *IEEE Sensors Journal*, 13(6):2284–2291, 2013.

[50] Vladimir Leonov and Ruud JM Vullers. Wearable electronics self-powered by using human body heat: The state of the art and the perspective. *Journal of Renewable and Sustainable Energy*, 1(6):062701, 2009.

[51] Tingli Li, Yang Liu, Ye Tian, Shuo Shen, and Wei Mao. A storage solution for massive iot data based on nosql. In *Green Computing and Communications (GreenCom), 2012 IEEE International Conference on*, pages 50–57. IEEE, 2012.

[52] Thorlux Lighting. Emergency systems. http://www.thorlux.com/emergency-systems [Online; accessed January 22, 2018].

[53] Hui Liu, Houshang Darabi, Pat Banerjee, and Jing Liu. Survey of wireless indoor positioning techniques and systems. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, 37(6):1067–1080, 2007.

[54] L Mølhave, Geo Clausen, B Berglund, J de Ceaurriz, A Kettrup, T Lindvall, M Maroni, AC Pickering, U Risse, H Rothweiler, et al. Total volatile organic compounds (tvoc) in indoor air quality investigations. *Indoor Air*, 7(4):225–240, 1997.

[55] Andres Munoz. Machine learning and optimization, 2014. https://www.cims.nyu.edu/~munoz/files/ml_optimization.pdf [Online: accessed January 25, 2018].

[56] MySQL. 27.8 mysql c api. https://dev.mysql.com/doc/refman/5.7/en/c-api.html [Online; accessed February 5, 2018].

[57] Nasser M Nasrabadi. Pattern recognition and machine learning. *Journal of electronic imaging*, 16(4):049901, 2007.

[58] Andrew Ng. Machine learning course. *Coursera.org: Resources: Week 1 Lecture Notes*, 2017. https://www.coursera.org/learn/machine-learning/home/info [Online: accessed November 20, 2017].

[59] Andrew Ng. Machine learning course. *Coursera.org: Resources: Week 3 Lecture Notes*, 2017. https://www.coursera.org/learn/machine-learning/home/info [Online: accessed November 20, 2017].

[60] Andrew Ng. Machine learning course. *Coursera.org: Resources: Week 2 Lecture Notes*, 2017. https://www.coursera.org/learn/machine-learning/home/info [Online: accessed November 20, 2017].

[61] Andrew Ng. Machine learning course. *Coursera.org: Resources: Week 7 Lecture Notes*, 2017. https://www.coursera.org/learn/machine-learning/home/info [Online: accessed November 20, 2017].

[62] Andrew Ng. Machine learning course. *Coursera.org: Resources: Week 4 Lecture Notes*, 2017. https://www.coursera.org/learn/machine-learning/home/info [Online: accessed November 20, 2017].

[63] Andrew Ng. Machine learning course. *Coursera.org: Resources: Week 5 Lecture Notes*, 2017. https://www.coursera.org/learn/machine-learning/home/info [Online: accessed November 20, 2017].

[64] Andrew Ng. Machine learning course. *Coursera.org: Resources: Week 6 Lecture Notes*, 2017. https://www.coursera.org/learn/machine-learning/home/info [Online: accessed November 20, 2017].

[65] Yoosoo Oh, Jonghyun Han, and Woontack Woo. A context management architecture for large-scale smart environments. *IEEE Communications Magazine*, 48(3), 2010.

[66] Apostolia Papapostolou and Hakima Chaouchi. Scene analysis indoor positioning enhancements. *annals of telecommunications-annales des télécommunications*, 66(9-10):519–533, 2011.

[67] Joseph A Paradiso. Systems for human-powered mobile computing. In *Design Automation Conference, 2006 43rd ACM/IEEE*, pages 645–650. IEEE, 2006.

[68] Ken Parsons. *Human thermal environments: the effects of hot, moderate, and cold environments on human health, comfort, and performance*. CRC press, 2014.

[69] Alex Pentland. Looking at people: Sensing for ubiquitous and wearable computing. *IEEE Transactions on Pattern analysis and machine intelligence*, 22(1):107–119, 2000.

[70] Deyban Perez, Miguel A Astor, David Perez Abreu, and Eugenio Scalise. Intrusion detection in computer networks using hybrid machine learning techniques. In *Computer Conference (CLEI), 2017 XLIII Latin American*, pages 1–10. IEEE, 2017.

[71] Wiring Pi. Serial library. `http://wiringpi.com/reference/serial-library` [Online; accessed February 3, 2018].

[72] David Martin Powers. Evaluation: from precision, recall and f-measure to roc, informedness, markedness and correlation. *Journal of Machine Learning Technologies*, 2011.

[73] Mikel Rodriguez, Ivan Laptev, Josef Sivic, and Jean-Yves Audibert. Density-aware person detection and tracking in crowds. In *Computer Vision (ICCV), 2011 IEEE International Conference on*, pages 2423–2430. IEEE, 2011.

[74] Maria Schuld, Ilya Sinayskiy, and Francesco Petruccione. An introduction to quantum machine learning. *Contemporary Physics*, 56(2):172–185, 2015.

[75] Fabrizio Sebastiani. Machine learning in automated text categorization. *ACM computing surveys (CSUR)*, 34(1):1–47, 2002.

[76] Bosch Sensortec. Bme680 - datashet, August 2017. `https://ae-bst.resource.bosch.com/media/_tech/media/datasheets/BST-BME680-DS001-00.pdf` [Online; accessed February 1, 2018].

[77] Shirish Krishnaj Shevade and S Sathiya Keerthi. A simple and efficient algorithm for gene selection using sparse logistic regression. *Bioinformatics*, 19(17):2246–2253, 2003.

[78] SparkFun. Sound detector hookup guide. https://learn.sparkfun.com/tutorials/sound-detector-hookup-guide [Online; accessed February 1, 2018].

[79] SFUPTOWNMAKER (SparkFun). I2c. https://learn.sparkfun.com/tutorials/i2c [Online; accessed February 3, 2018].

[80] Nikolay Stanevski and Dimiter Tsvetkov. Using support vector machine as a binary classifier. In *International Conference on Computer Systems and Technologies–CompSys Tech*, 2005.

[81] Internet Live Stats. Internet users, 2017. http://www.internetlivestats.com/internet-users [Online: accessed January 16, 2018].

[82] John S Steinhart and Stanley R Hart. Calibration curves for thermistors. In *Deep Sea Research and Oceanographic Abstracts*, volume 15, pages 497–503. Elsevier, 1968.

[83] Textile. How rfid works. https://www.textile-id.com/how-rfid-works-in-textile [Online: accessed January 23, 2018].

[84] Karen Tillman. How many internet connections are in the world? right now, 2013. https://blogs.cisco.com/news/cisco-connections-counter [Online: accessed January 16, 2018].

[85] Ovidiu Vermesan and Peter Friess. *Internet of things: converging technologies for smart environments and integrated ecosystems*. River Publishers, 2013.

[86] Yan You, Can Niu, Jian Zhou, Yating Liu, Zhipeng Bai, Jiefeng Zhang, Fei He, and Nan Zhang. Measurement of air exchange rates in different indoor environments using continuous co2 sensors. *Journal of Environmental Sciences*, 24(4):657–664, 2012.

[87] Andrea Zanella, Nicola Bui, Angelo Castellani, Lorenzo Vangelista, and Michele Zorzi. Internet of things for smart cities. *IEEE Internet of Things journal*, 1(1):22–32, 2014.

Appendix A

Pictures of Equipment Used

Figure A.1: Raspberry Pi 2.

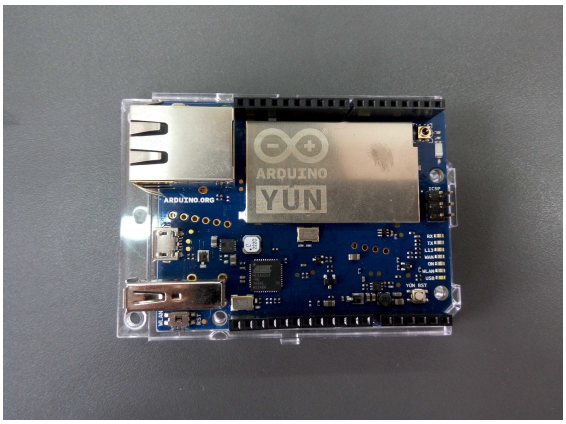


Figure A.2: Arduino Yun.

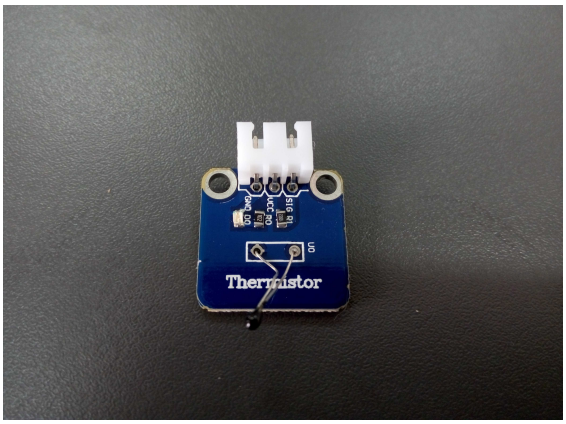


Figure A.3: Thermistor module.

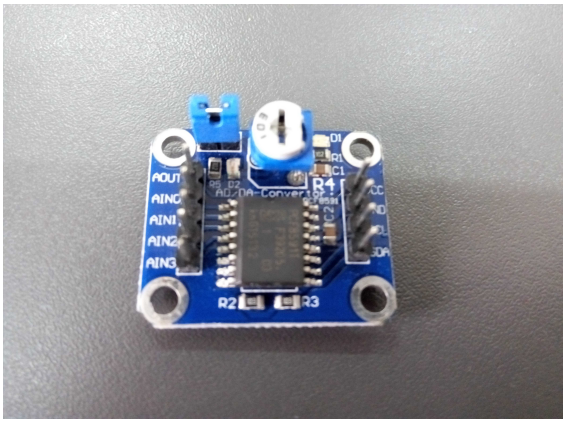


Figure A.4: ADC PCF8591 module.

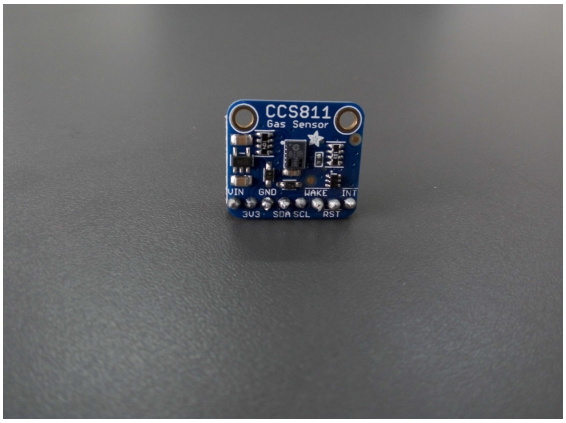


Figure A.5: AMS CCS811 Breakoutboard by Adafruit.

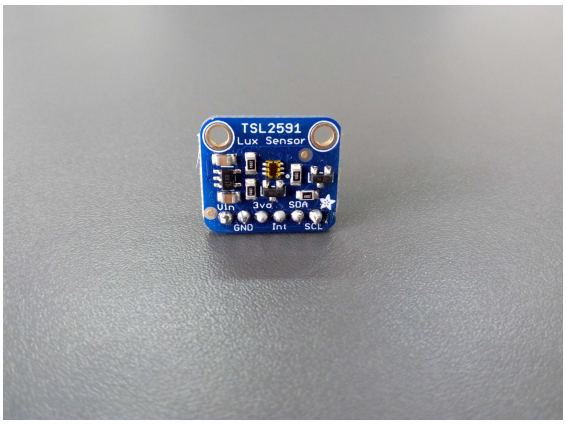


Figure A.6: AMS TSL2591 Breakoutboard by Adafruit.

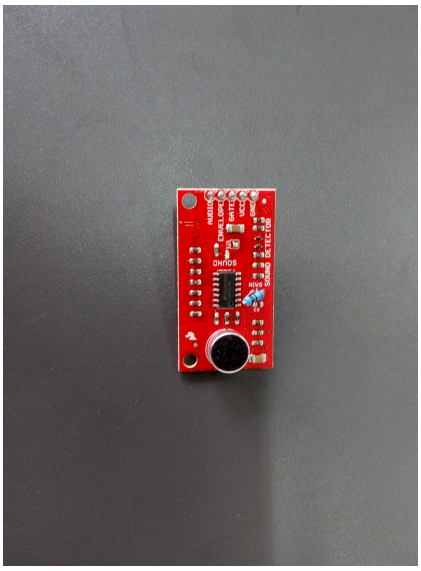
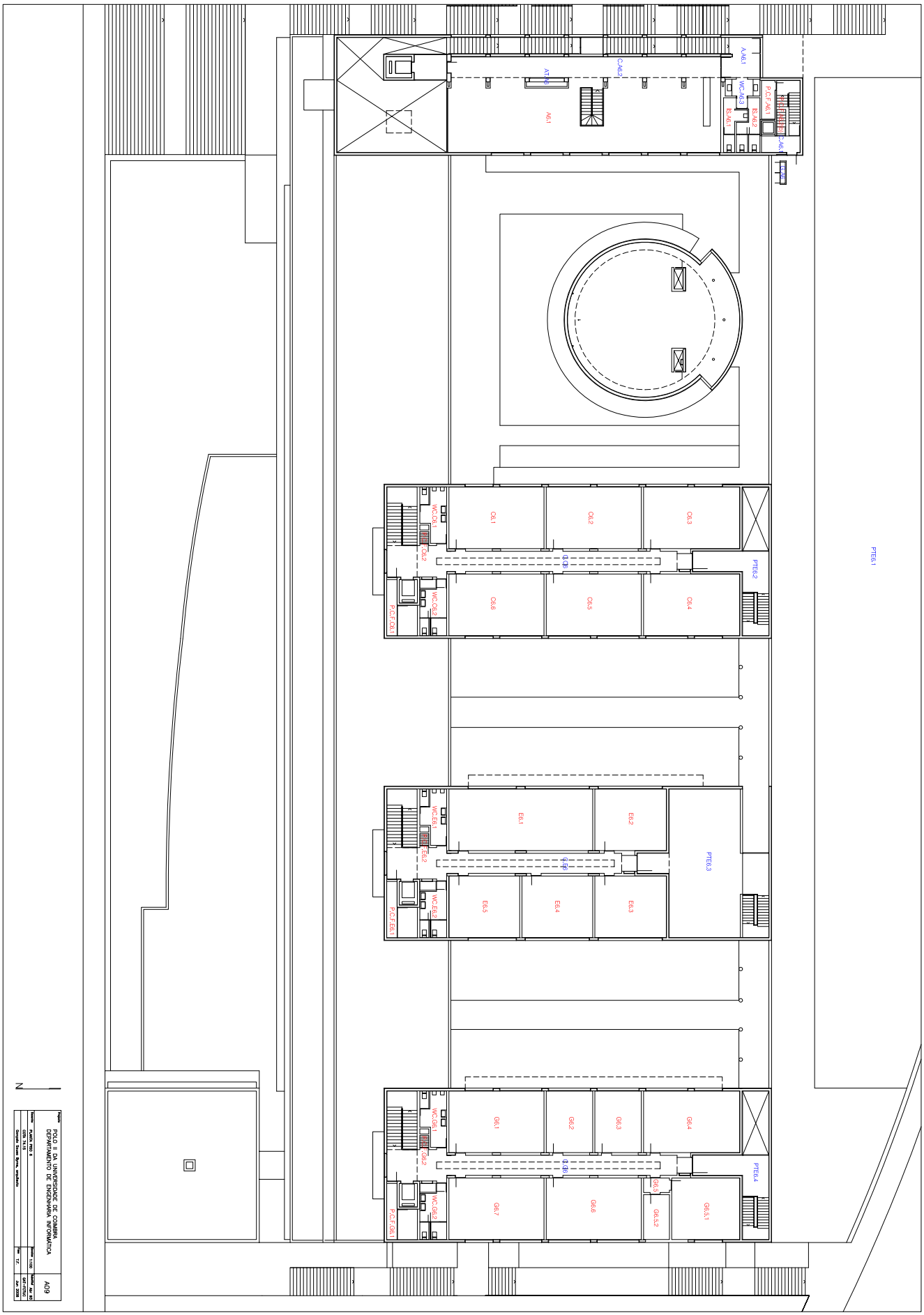


Figure A.7: Sparkfun Sound Detector.

Appendix B

Plant of 6º floor of Department of Informatics Engineering


PÓLO II DA UNIVERSIDADE DE COIMBRA
DEPARTAMENTO DE ENGENHARIA INFORMÁTICA
A09
A6.1
WC.A6.3
P.C.F.A6.1
A6.2
WC.G6.1
G6.1
G6.2
G6.3
G6.4
G6.7
G6.6
G6.5.2
G6.5.1
PTE6.4
WC.G6.2
P.C.F.G6.1
WC.E6.1
E6.1
E6.2
E6.5
E6.4
E6.3
PTE6.3
WC.E6.2
P.C.F.E6.1
WC.C6.1
C6.1
C6.2
C6.3
C6.6
C6.5
C6.4
PTE6.2
WC.C6.2
P.C.F.C6.1
PTE6.1

Appendix C

Dataset

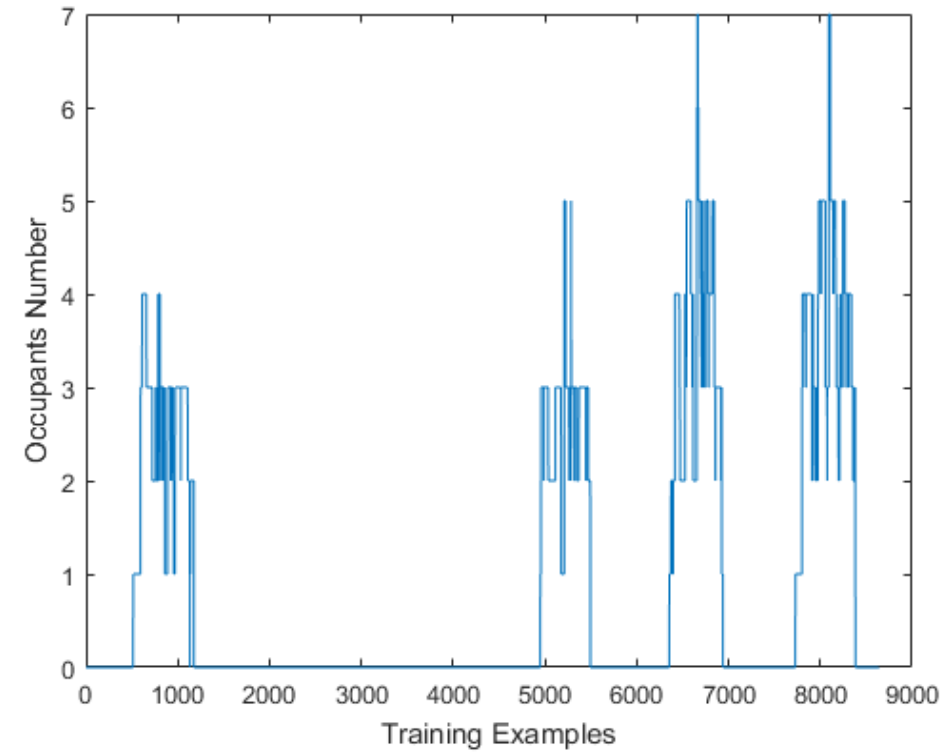


Figure C.1: Training Dataset of Temperature Ground Truth.

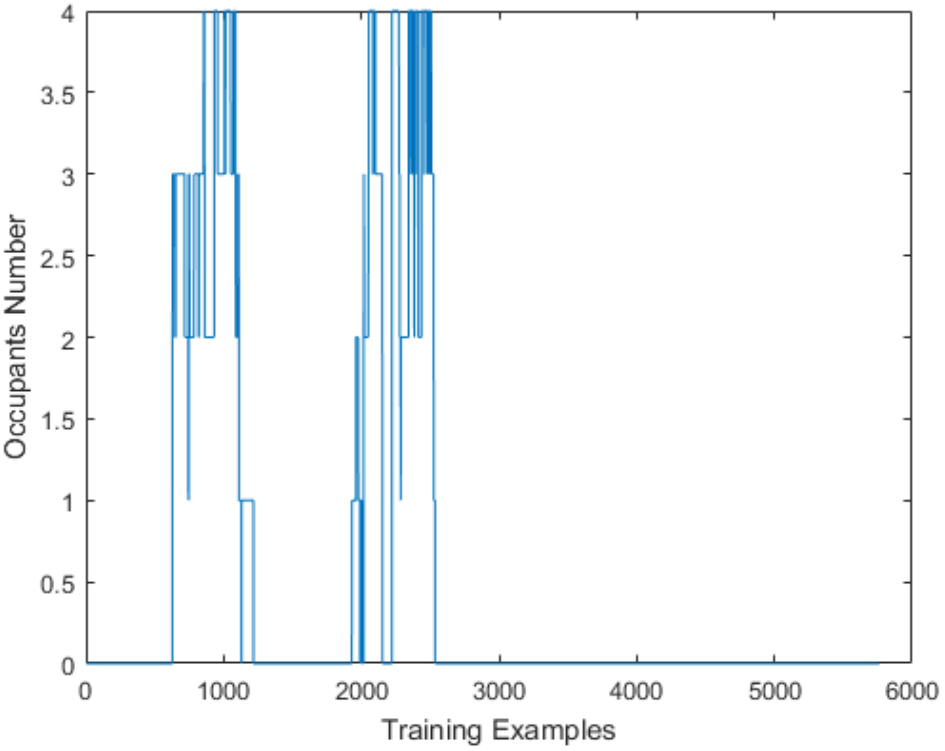


Figure C.2: Testing Dataset of Temperature Ground Truth.

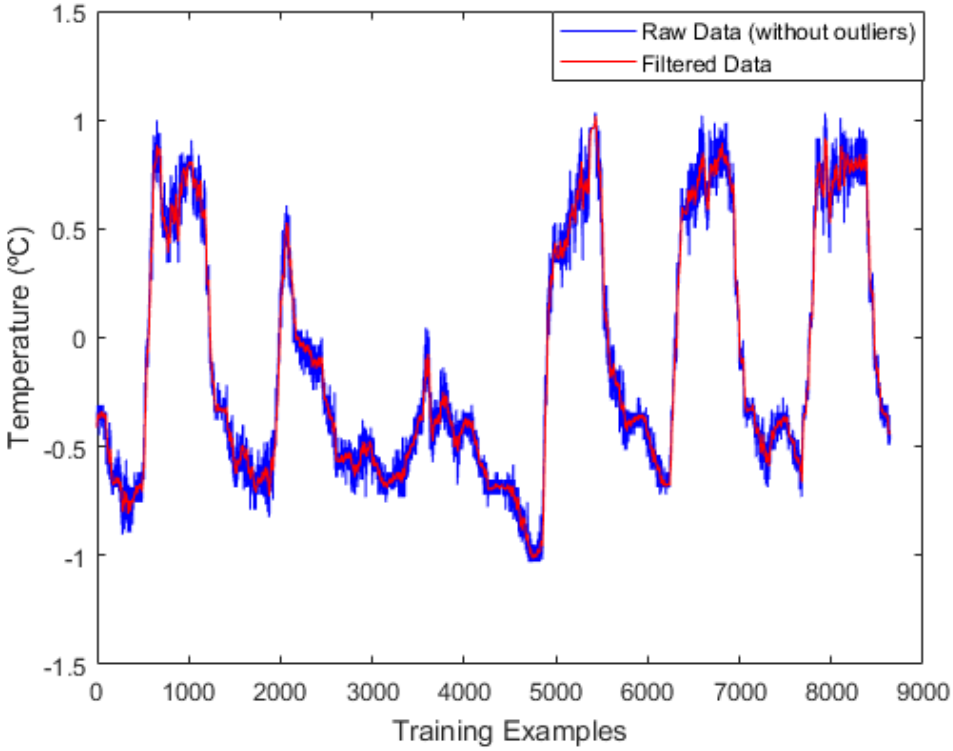


Figure C.3: Training Dataset of Temperature in Celsius per sample.

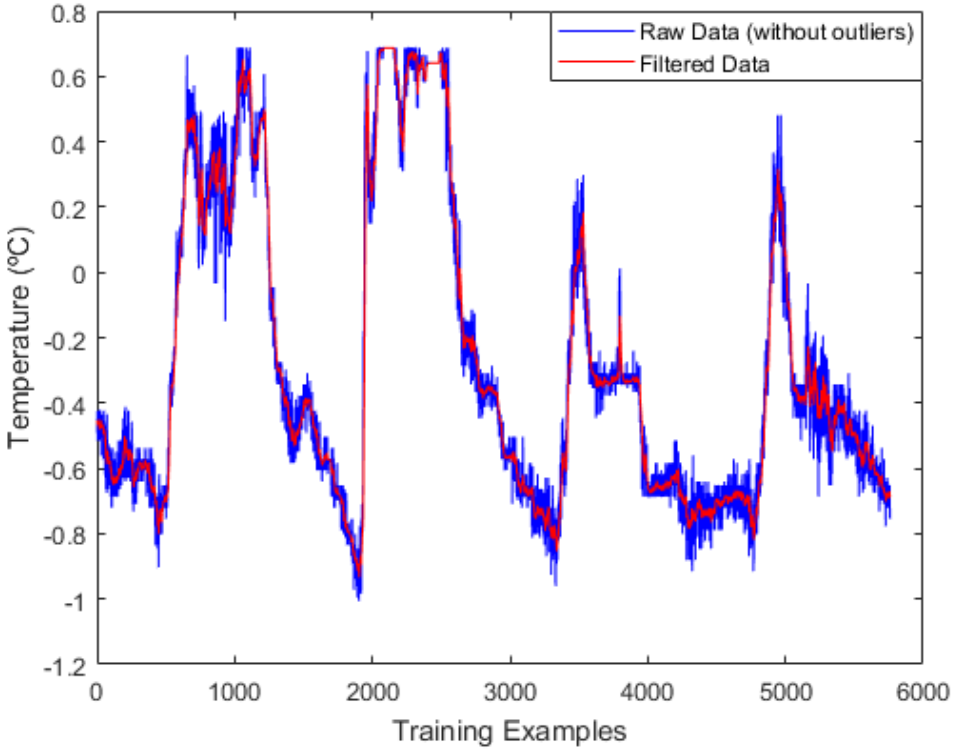


Figure C.4: Testing Dataset of Temperature in Celsius per sample.

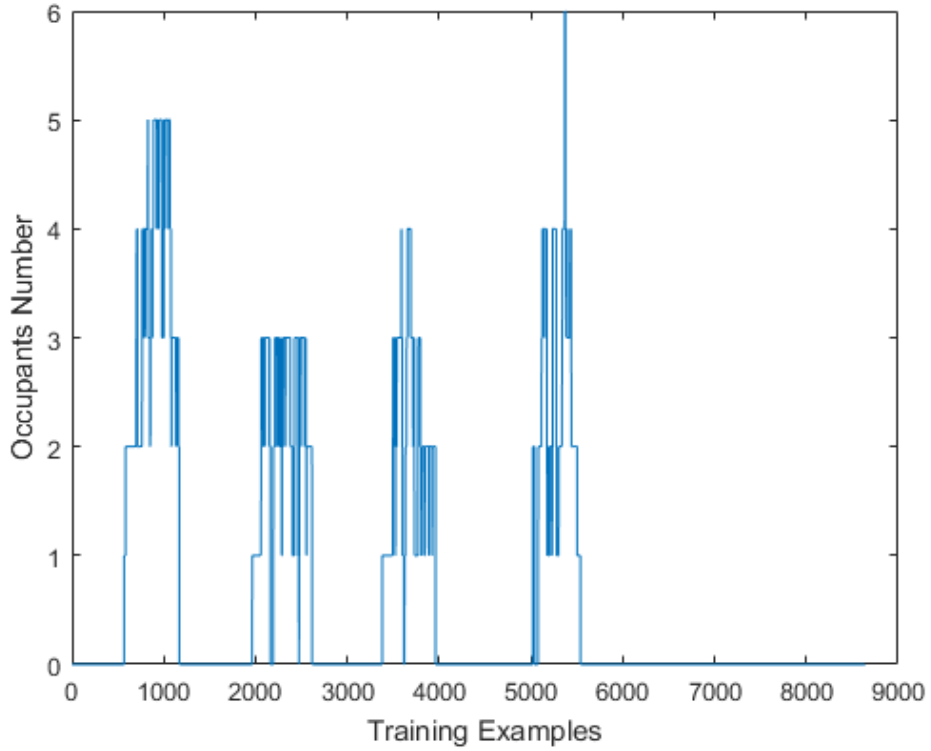


Figure C.5: Training Dataset of Ground Truth except for Temperature Data.

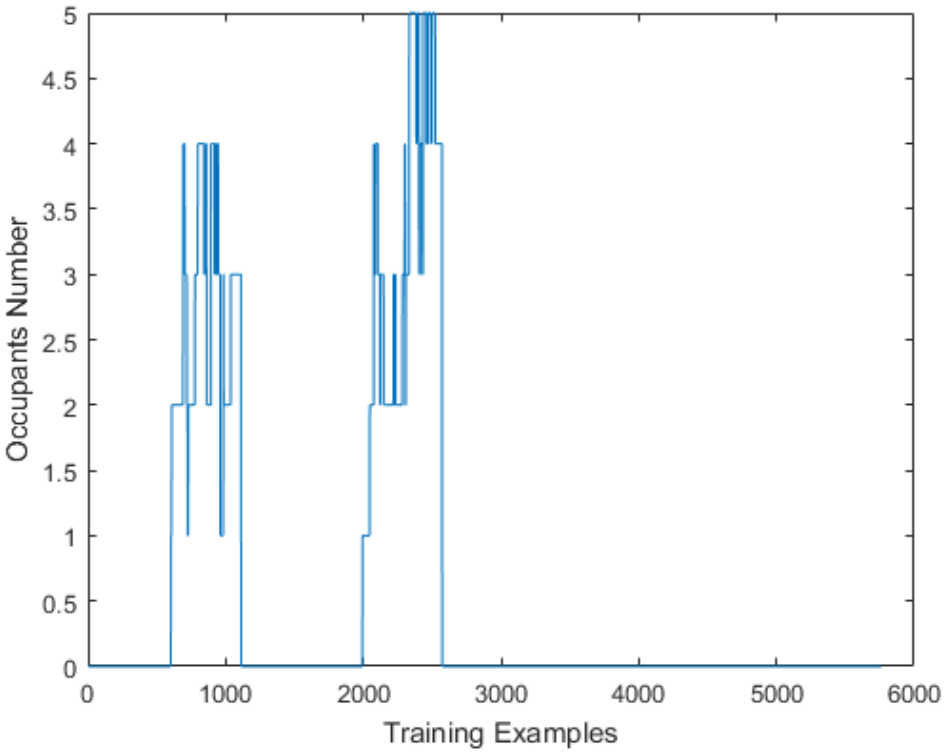


Figure C.6: Testing Dataset of Ground Truth except for Temperature Data.

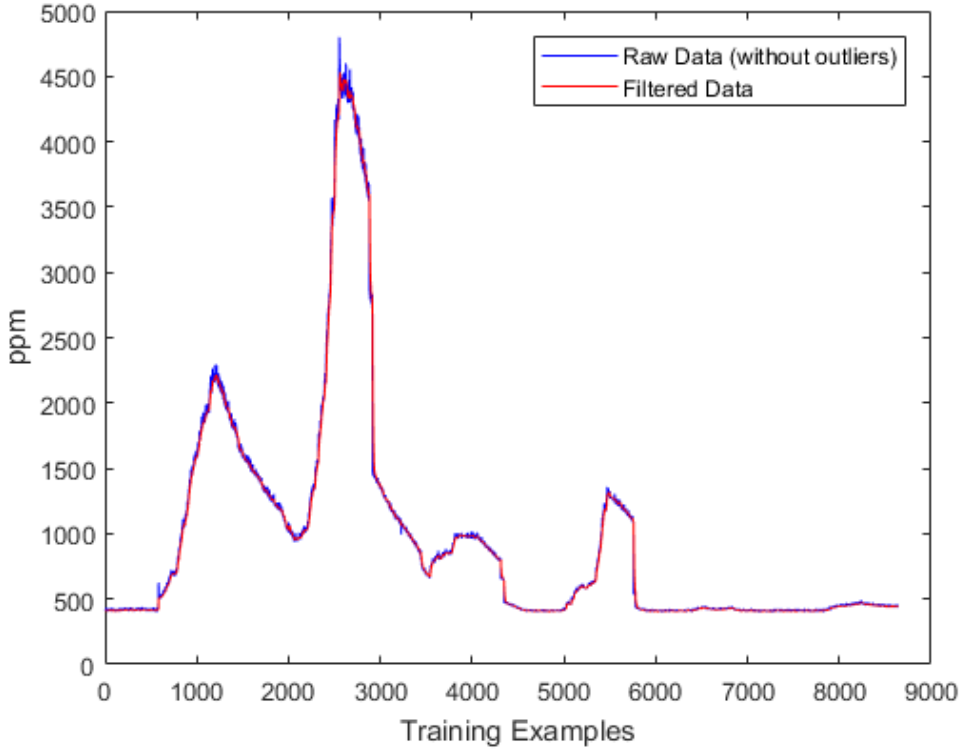


Figure C.7: Training Dataset of CO2 in ppm per sample.

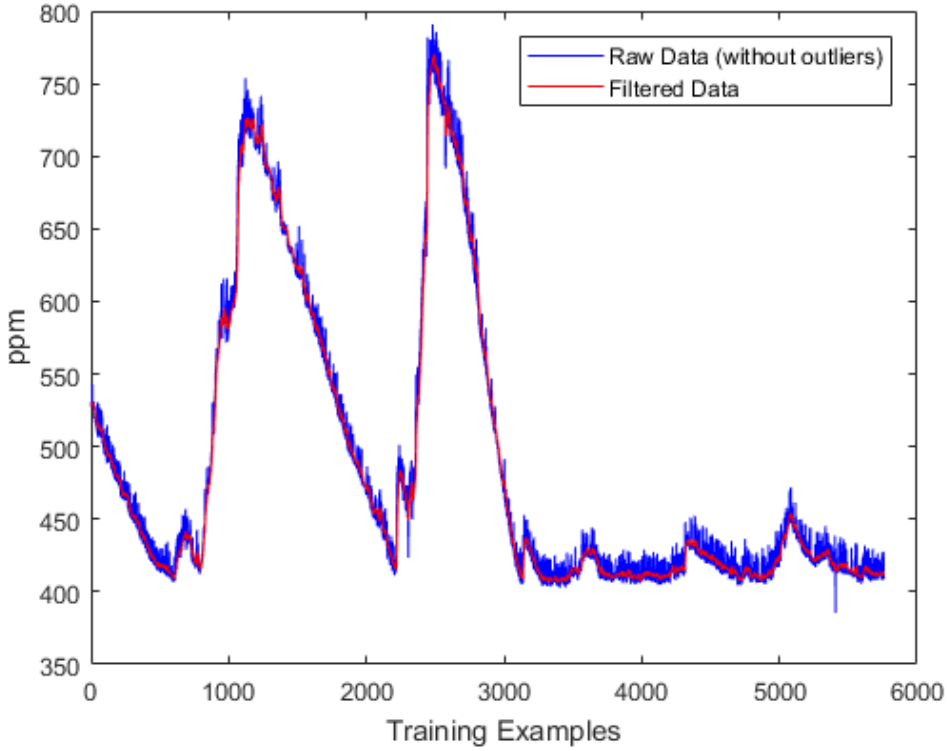


Figure C.8: Testing Dataset of CO2 in ppm per sample.

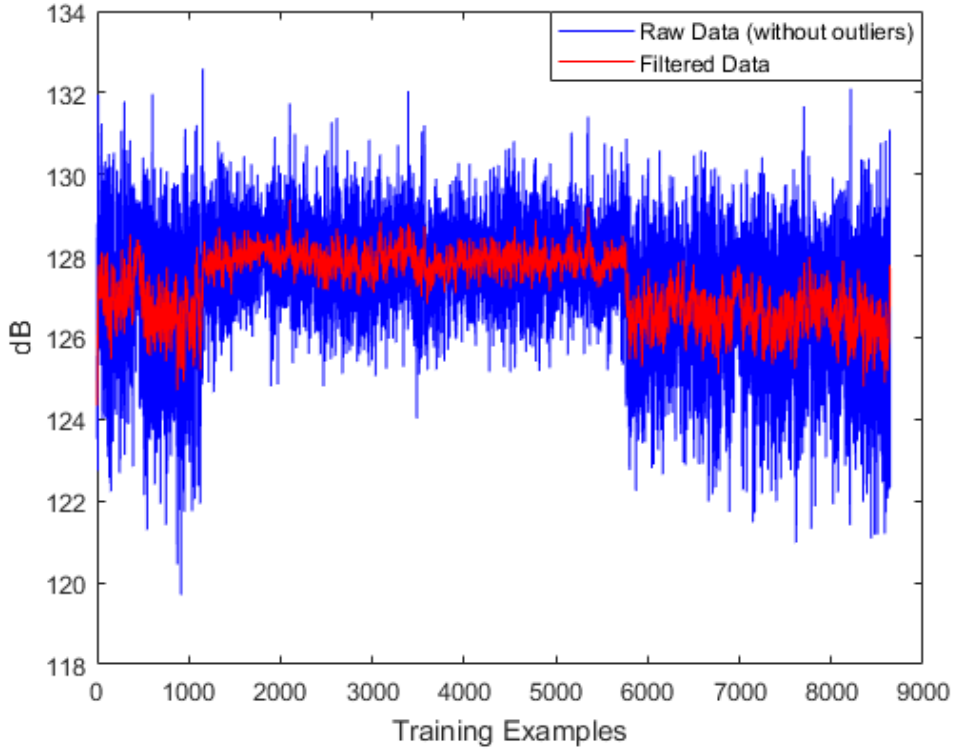


Figure C.9: Training Dataset of Noise in dB per sample.

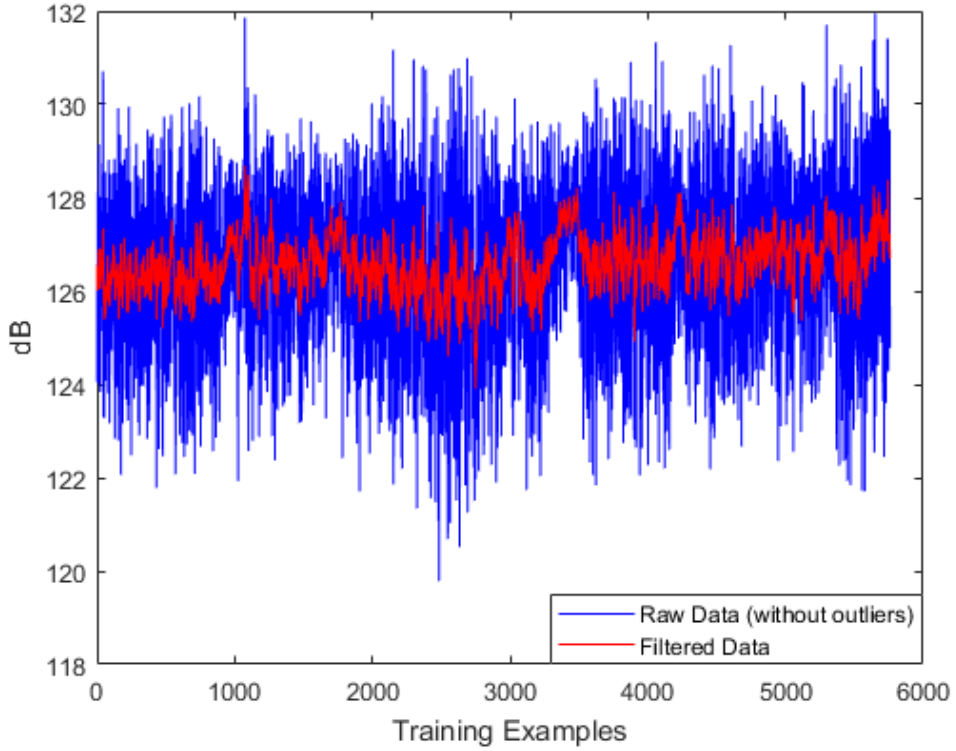


Figure C.10: Testing Dataset of Noise in dB per sample.

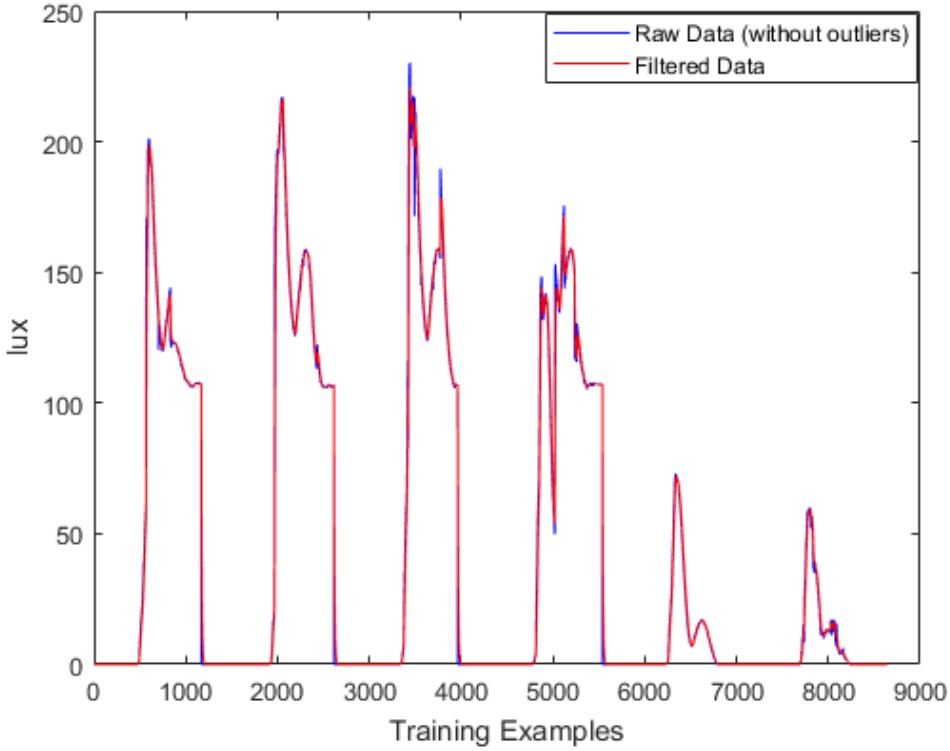


Figure C.11: Training Dataset of Light in lux per sample.

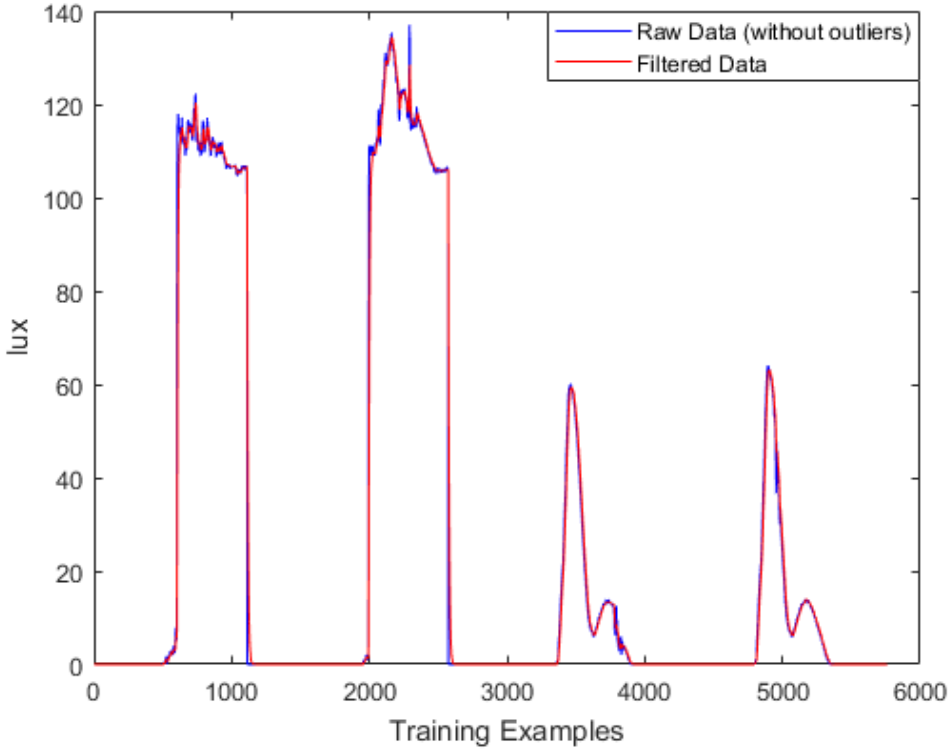


Figure C.12: Testing Dataset of Light in lux per sample.

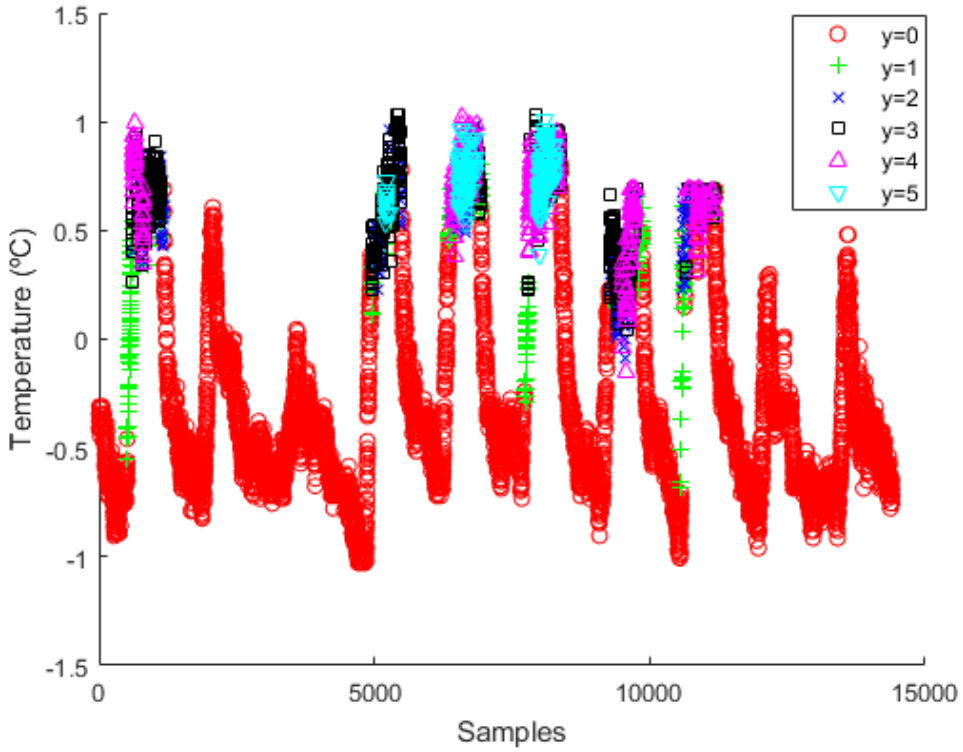


Figure C.13: Dataset of Temperature in Celsius per sample with the classes/labels.

Appendix D

Algorithms described in the document

Algorithm 1 Back-propagation algorithm

procedure

Given training set $(x^{(1)}, y^{(1)}), ..., (x^{(l)}, y^{(l)})$*:*

Set $\Delta_{i,j}^{(l)} := 0$

For training example $t = 1$ *to* m*:*

Set $a^{(1)} := x^{(t)}$

Perform forward propagation to compute $a^{(l)}$ for $l = 2, 3, ..., L$

Using $y^{(t)}$ compute $\delta^{(L)} - y^{(t)}$

Compute $\delta^{(L-1)}, \delta^{(L-2)}, ..., \delta^{(2)}$ using equation 2.20

$\Delta_{i,j} := \Delta_{i,j} + a_j^{(l)} \delta^{(l+1)}$

if $j \neq 0$ **then**

$D_{i,j}^{(l)} := \frac{1}{m}(\Delta_{i,j}^{(l)} + \lambda \Theta_{i,j}^{(l)})$

else

$D := \frac{1}{m} \Delta_{i,j}^{(l)}$

Algorithm 2 Collecting and storage the data

procedure ARDUINO

Start the Serial Communication with 9600 bps

Initialize the sensors

loop:

if Received any message of RPi **then**

Collect and average ten samples with a 100 microseconds delay from sensors

Send the averaged samples to the RPi

procedure RASPBERRYPI

Initialize the GPIO pins

Activate the GPIO to the buttons

Start the Serial Communication with 9600 bps

Initialize the communication with MySQL Database

loop:

Update with the actual number of occupants

Every 10 seconds, ask and receive samples from Arduino

Every 1 minute, make a average to the samples received and send to the database

Algorithm 3 Outliers filtering algorithm.

procedure

Given an array $(x(1), ..., x(m))$ *and a threshold:*

Create a new array y with the length of x

Set $y(1) := x(1)$

For $i = 2$ *to* m:

if $(1 - threshold < \frac{x(i)}{y(i-1)})$ and $(\frac{x(i)}{y(i-1)} < 1 + threshold)$ **then**

$y(i) = x(i)$

else

$y(i) = y(i-1)$

Return the new array y

Algorithm 4 Parameters to Highest F-score Algorithm.

procedure

Given training set $(x_0^{(1)}, y^{(1)}), ..., (x_0^{(l)}, y^{(l)})$ *testing set* $(x_{cv0}^{(1)}, y_{cv}^{(1)}), ..., (x_{cv0}^{(l)}, y_{cv}^{(l)})$ *and* a, b *and* c *parameters:*

for `i=1:length(b)` **do**:

$x = polyFeatures(x_0, b(i))$

$x_{cv} = polyFeatures(x_{cv0}, b(i))$

$x = featureNormalize(x)$

$x_{cv} = featureNormalize(x)$

for `j=1:length(a)` **do**:

for `k=1:length(c)` **do**:

Training the classifier with $b(i)$, $a(j)$, $c(k)$, x and y

Predict the classifier with x_{cv}, y_{cv} and the parameters calculated earlier

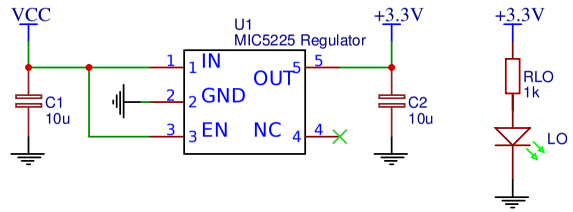
Calculate the f-score value of classifier and save them to a array(j,i,k)

Calculate the maximum f-score classifier

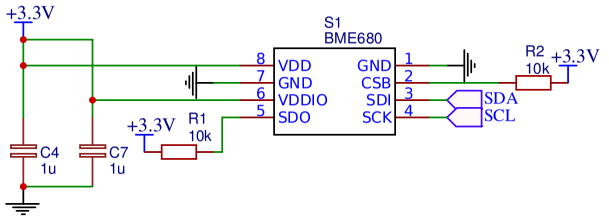
Returns the parameters with the highest value

Appendix E

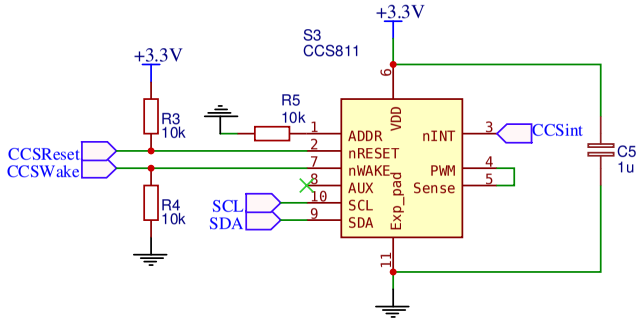
Circuit Schematics and Printed Circuit Board Design



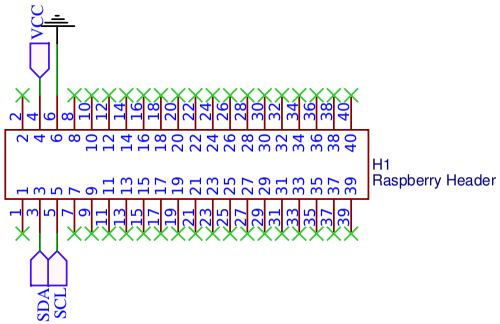
IC REG LINEAR 5V 150MA

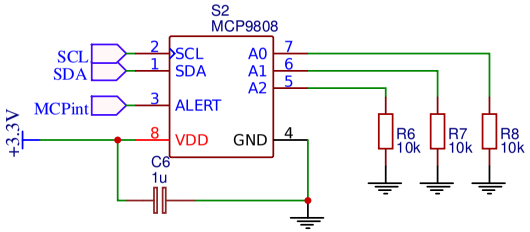


Integrated Environmental Sensor I2C (0x77)

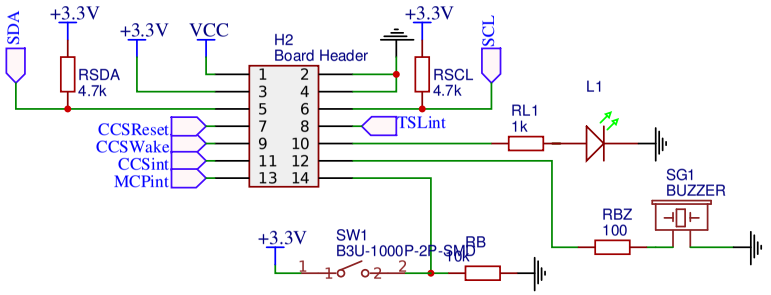


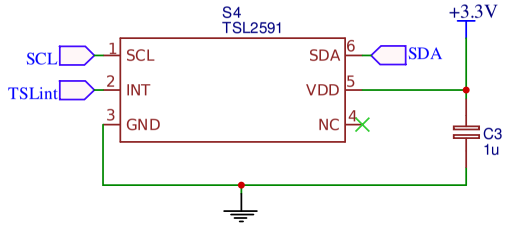
Air Quality Sensor I2C (0x5A)





Temperature Sensor I2C (0x5A)





Ambient Light Sensor I2C (0x29)

TITLE: LCT-EnvBoard		REV: 1.0
Date: 2017-12-27	Sheet: 1/1	
EasyEDA V4.11.9	Drawn By: Bruno Abade