UNIVERSIDADE FEDERAL DE MINAS GERAIS
Instituto de Ciências Exatas
Programa de Pós-Graduação em Ciência da Computação

Guillermo Ponce Contreras

A smart IoT office employing machine learning for personalized user comfort

Belo Horizonte
2019

Guillermo Ponce Contreras

A smart IoT office employing machine learning for personalized user comfort

Final Version

Thesis presented to the Graduate Program in Computer Science of the Federal University of Minas Gerais in partial fulfillment of the requirements for the degree of Master in Computer Science.

Advisor: Daniel Fernandes Macedo
Co-Advisor: Adriano Alonso Veloso

Belo Horizonte
2019

Ficha catalográfica elaborada pela bibliotecária Belkiz Inez Rezende Costa
CRB 6ª Região nº 1510

Ponce Contreras, Guillermo.

P792s A smart IoT office employing machine learning for personalized user comfort [manuscrito] / Guillermo Ponce Contreras. — Belo Horizonte, 2019.
xxiii, 90 f. il.; 29cm.

Orientador: Daniel Fernandes Macedo.
Corrientador: Adriano Alonso Veloso.
Dissertação (mestrado) - Universidade Federal de Minas Gerais, Instituto de Ciências Exatas, Departamento de Ciência da Computação.
Referências: f.81-89

1. Computação – Teses. 2. Aprendizado de máquina – Teses. 3. Algoritmos de predição – Teses 4.Ergonomia – Teses .5. Conforto do usuário – Teses. 6. Automação de sala– Teses. I. Macedo, Daniel Fernandes. II. Veloso, Adriano Alonso. III.Título.

C DU 519.6*73(043)

UNIVERSIDADE FEDERAL DE MINAS GERAIS
INSTITUTO DE CIÊNCIAS EXATAS
PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

FOLHA DE APROVAÇÃO

A Smart IoT Office Employing Machine Learning for Personalised User Comfort

GUILLERMO PONCE CONTRERAS

Dissertação defendida e aprovada pela banca examinadora constituída pelos Senhores:

[illegible]

PROF. DANIEL FERNANDES MACEDO - Orientador
Departamento de Ciência da Computação - UFMG

[illegible]

PROF. ADRIANO ALONSO VELOSO - Coorientador
Departamento de Ciência da Computação - UFMG

[illegible]

PROF. JOSÉ MARCOS SILVA NOGUEIRA
Departamento de Ciência da Computação - UFMG

[illegible]

PROF. GILBERTO MEDEIROS RIBEIRO
Departamento de Ciência da Computação - UFMG

[illegible]

PROF. WLADMIR CARDOSO BRANDÃO
Departamento de Ciência da Computação - PUC/MG

Belo Horizonte, 29 de Abril de 2019.

A Dios, mis padres y amigos.

Acknowledgments

Tenho muito para agradecer. Começarei com um agradecimento especial a meu orientador, Daniel Macedo. Apesar da diferença em nossos perfis, ele conseguiu canalizar meu potencial da melhor maneira possível. Esse trabalho nos permitiu crescer como pessoas e como profissionais.

Devo continuar agradecendo a todos os colaboradores da equipe de IoT do Laboratório Winet (DCC-UFMG). Por causa deles, esse trabalho é uma realidade. Aprendi muito.

Não me esqueço de meus professores, parceiros de laboratório e colegas de mestrado. Eles também foram muito importantes. A troca de conhecimentos, idéias e cultura foi muito enriquecedora.

Tenho uma grande dívida com este país maravilhoso e com a universidade pública. Hoje, mais do que nunca, me sinto muito abençoado com as oportunidades acadêmicas que me ofereceram. Espero poder pagar essa dívida de alguma forma.

Indo para um lado mais pessoal... Devo agradecer a Deus por me dar forças para não desistir. Aos meus pais e irmãos por acreditarem em mim e me apoiarem.

Agradeço também a meus amigos mais próximos, pois permitiram que esta luta fosse mais agradável. Para eles, posso dizer que não fiz apenas um mestrado em computação, também fiz um mestrado na vida. Eu amo todos eles.

Finalmente, devo mencionar uma pessoa muito especial, Vicky. Chegou perto do fim, mas me deu a última força de que precisava para terminar este trabalho. Ela me ensinou que ... “Quando a inspiração vem, ela deve encontrar você trabalhando.”

"*Ser más para servir mejor.*"
(San Ignacio de Loyola)

Resumo

A Internet das Coisas (IoT) é uma rede que permite aos objetos coletar e transmitir dados entre si. Desde o surgimento da IoT, muitas aplicações foram geradas impactando muitas industrias, mas também a vida cotidiana das pessoas comuns. Uma das aplicações mais estudadas e desenvolvidas da IoT é a automação de ambientes, por exemplo, hoje em dia existem muitos produtos que melhoram as condições ambientais das casas. No local de trabalho, seu uso está relacionado à melhoria da ergonomia e, portanto, a proporcionar maior conforto aos trabalhadores de escritório, o que afeta diretamente a produtividade.

À medida que surgiam tecnologias mais inovadoras, a automação tornou-se mais sofisticada, oferecendo melhores soluções. Inicialmente, fornecia interfaces que permitiam uma gama maior de opções, incluindo controle remoto de dispositivos. Mais tarde, os usuários podem até agendar a execução de determinadas tarefas, por exemplo, que o ar condicionado seja ligado meia hora antes da chegada ao escritório. No entanto, há mais de uma década, o foco tem sido a incorporação de maior inteligência à automação, o que significa que o sistema pode aprender sobre as preferências do usuário e realizar as configurações que mais lhes agradam. Atualmente, existem muitas tecnologias e produtos de automação que prevêem ações do usuário; no entanto, muitas delas requerem muita intervenção do usuário ou são imprecisas. Aqui está o desafio para a automação do ambiente: fornecer uma boa experiência ao usuário, com menos intervenção e mais conforto.

Este trabalho apresenta um sistema de automação de escritório que inclui um agente inteligente para a orquestração de dispositivos. O sistema é composto por atuadores (ar condicionado, persianas, interruptor de luz) e sensores que obtêm dados do ambiente interno. Ele também possui um aplicativo móvel para que o usuário interaja com o sistema enviando solicitações de ação ou indicando seu nível de conforto. O agente inteligente, denominado orquestrador, utiliza o algoritmo XGBoost para aprender sobre o usuário e prever suas avaliações em uma determinada condição ambiental. A avaliação é definida de forma a indicar o motivo do desconforto e sua intensidade. Usamos o nível de desconforto para escolher a ação mais apropriada em uma lista de ações predefinidas.

Para demonstrar que nossa proposta melhora a experiência do usuário, definimos duas métricas, uma com base no desconforto do usuário em um cenário específico e outra com base no número de interações. Nossos resultados mostram que no sistema de automação com a proposta o usuário se sente mais confortável em relação ao sistema sem a proposta. Além disso, o uso de nossa proposta diminui o número de solicitações de mudança no estado dos atuadores.

Palavras-chave: Aprendizado de máquina, Algoritmos de previsão, Personalização, Conforto do usuário, Ergonomia, Produtividade, Automação de ambientes

Abstract

The Internet of Things (IoT) is a network that allows objects to collect and transmit data to each other. Since the appearance of IoT, many applications have been generated impacting many industries, but also the everyday life of ordinary people. One of the most studied and developed applications of IoT is the automation of environments, for instance, nowadays there are many products improving the environmental conditions of houses. In the workplace, its usage is related to improving ergonomics and therefore providing greater comfort to office workers, which has a direct impact on productivity.

As more innovative technologies appeared, automation became more sophisticated offering better solutions. Initially, it provided interfaces that allowed a greater range of options, including remote control of devices. Later, users could even schedule the execution of certain tasks, for example, that the air conditioner is turned on half an hour before the arrival at the office. Nevertheless, for more than a decade, the focus has been on incorporating greater intelligence to automation, which means that the system can learn about user preferences and perform the configurations that suit them. Currently, there are many technologies and automation products that predict user actions, however, many of these require a lot of user intervention or are inaccurate. Here lies the challenge for environment automation: to provide a good experience to the user, with less intervention and more comfort.

This work presents an office automation system that includes a smart agent that customizes the environment through orchestration of devices. The system is composed of actuators (air conditioner, shades, light switch) and sensors that obtain data from the internal environment. It also has a mobile application so that the user interacts with the system by sending requests for action or indicating his level of comfort. The smart agent, denominated orchestrator, utilizes the algorithm XGBoost to learn about the user and predict his evaluations in a certain environmental condition. The evaluation is defined in such a way that it indicates the reason for the discomfort and its intensity. We use the level of discomfort to choose the most appropriate action from a list of predefined actions.

To demonstrate that our proposal improves the user experience we defined two metrics, one based on the discomfort of the user in a specific scenario and another based on the number of interactions. Our results show that in the automation system with the proposal the user feels more comfortable in relation to the system without the proposal. In addition, the use of our proposal decreases the number of change requests in the state of the actuators.

Keywords: Machine Learning, Prediction Algorithms, Customization, User comfort, Ergonomics, Productivity, Office Automation

List of Figures

List of Tables

Contents

Chapter 1

Introduction

In the last two decades, the number of devices connected to the Internet has grown gradually, but in 2010 this growth was explosive, reaching 12.5 billion devices distributed among just 6.8 billion people, i.e. almost two devices per person. As reported by the Internet Business Solutions Group, by 2020 the number of devices connected to the Internet will reach between 50 and 100 billion, hence each person will have around six devices [26]. Aside from their proliferation, devices also increased their capabilities inducing a novel paradigm: the *Internet of Things (IoT)*.

According to [69], IoT is a dynamic global information network composed of smart devices connected to the Internet. [14] indicated that all these objects should be able to interact with each other and to cooperate to reach common goals. For instance, the purpose of a sensor-based system that monitors anomalies may be to detect and alert about emergencies in a fragile elderly person's room ([79]). Like this one, there are many IoT applications, which can be classified into different domains. [63] organized them in terms of market: i) environmental monitoring, ii) health-care; iii) security and surveillance; iv) smart business/inventory and product management; v) smart cities and; vi) smart homes/smart building management. The latest one is related to environment automation systems, which is the focus of this work.

Nowadays, there is a concept related to health care that gains importance: *Ergonomics*. The broad concept of ergonomics is more than using a comfortable chair; it is a multidisciplinary field that also involves physiological biomechanical conditions, humidity, lighting, seating, equipment, working hours, stress and also legal considerations ([87]). According to [47], ergonomics is one of the major factors shaping the performance of an individual, in addition to process and personal factors. The Institute for Work & Health confirmed it after the study about the effect of ergonomics on employee productivity, showing that by using adjustable chairs and by receiving ergonomics training, workers reported less soft-tissue injury and pain, thus increasing their productivity by 18 percent. This percentage represented an annual profit of more than USD$ 25.000 for a total investment of just USD$ 1.000 per worker. On the other hand, [38] details how indoor temperature affects several human responses such as thermal comfort, perceived air quality, sick building syndrome symptoms and performance at work. This idea is reinforced in the work of [43], who mentions that the exposal to elevated levels of temperature and humidity provokes fatigue, headache and difficulty for thinking, leading to a low productivity in workers.

Some of the parameters that involve ergonomics can be controlled with automation systems. For example, air conditioning can regulate the temperature and humidity of the environment, as a consequence, many companies are giving relevance to office automation deployments ([42]). Nowadays, there are automation systems that allow the user to modify remotely the status of their devices. This is very useful, since users can gain time to perform their tasks. However, having a system that can automate the room intelligently, adapting the devices to the preferences of the user, would make this time gain greater and the user experience more satisfactory.

Understanding that worker's productivity improves with the incorporation of office automation systems, we propose an IoT platform that customizes a working room based on the user evaluation. Initially, the actuators involved in the improvement of comfort (e.g. air conditioning unit, luminaries and sun blinds) have a default configuration. Given these initial conditions of the office, the user will be asked to evaluate the system with a certain frequency. In this way it will be possible to know what you like and what you don't. The room customization is the result of the application of actuation rules that uses a predicted user evaluation in certain environment conditions as decision reference.

1.1 Objectives

As mentioned previously, it is possible to improve the worker productivity by adapting the office conditions to the user preferences. Understanding this premise, we propose an intelligence orchestration of a smart IoT office. This broad objective requires several specific objectives:

- Build an smart IoT office system that allows office automation.
- Investigate how to collect environment data related to comfort using IoT devices.
- Automate intelligently the office to improve the user experience via customization, i.e., adjusting the office for the comfort of the individual user. This automation should be supported by machine learning algorithms to predict the actions of the user.
- Improve the user experience via actuations in the IoT devices.

1.2 Contributions

The main contributions of this proposal are:

- Build an IoT infrastructure with diverse smart devices, sensors and actuators.
- Propose and evaluate a machine learning model that predicts user evaluations for each user.
- Demonstrate that our proposal improves the comfort of the user when compared with an automation system without proposal.

1.3 Document organization

We organize this work into seven chapters. Chapter 1 presents an introduction, motivation and the objectives. Chapter 2 discusses some important concepts such as ergonomics and machine learning algorithms. Chapter 3 presents products and patents related to our proposal, but also the prediction algorithms used in automation systems. Chapter 4 introduces our office automation system details the hardware and software components. Chapter 5 presents our proposal to predict the user actions. Chapter 6 shows our experiments and their results. Finally, chapter 7, concludes this dissertation.

Chapter 2

Concepts

This chapter presents the main concepts related to the proposal. Section 2.1 presents a definition of the IoT and the ITU architecture model. Section 2.3 gives a brief introduction to the terms of *home automation* and *smart home*, which helps us to explain better what office automation is. Section 2.3.1 lists the most relevant advantages of *office automation*. Section 2.4 presents the concept of *ergonomics*, its importance for worker productivity, and the ergonomic factors (noise, CO_2, temperature, humidity) considered in this work. In section 2.5, we present ManIoT, a platform for IoT device management, which is used as the basis of our smart office system. Section 2.6 presents the machine learning algorithms that are used in this work. Section 2.7 shows SHAP, a tool used for explaining the output of any machine learning model. Finally, in section 2.8, we conclude this chapter.

2.1 Internet of Things

There are many definitions of IoT, but in this dissertation, we take as reference the ITU-T report titled Series Y: Global Information Infrastructure, Internet Protocol Aspects and Next-Generation Network ([52]). Here, the authors defined IoT from the point of view of technical standardization. They indicated that IoT is a global structure that provides advanced services of interconnection of objects (things), whether physical or virtual. In this context, physical things are capable of being sensed, actuated and connected such as robots or electrical equipment; and virtual things are capable of being stored, processed and accessed. Furthermore, ITU-T presents a list of fundamental characteristics and another of high-level requirements for IoT implementation.

The fundamental IoT characteristics are: a) inter-connectivity of anything with the global information and communication infrastructure; b) things-related services that ensure privacy protection and semantic consistency between physical and virtual things; c) heterogeneity i.e. enable to work with different hardware platforms and networks; d) dynamic changes, which is the possibility to change the devices' status (sleep, wake up, connected, etc) dynamically; e) enormous scale, which predicts that the number of devices will be at least an order of magnitude larger than the devices connected to the current Internet.

The high-level requirements for the implementation of IoT systems defined by ITU-T are: a) the IoT should be capable of recognizing things through a unique identifier, which allows IoT to include heterogeneous things in a unified way; b) interoperability among heterogeneous and distributed systems; c) automatic networking to adapt to different application domains, communication environments and devices; d) automatic service provisioning such as sensing, communication and processing data of things; e) location-based capabilities; f) security between things connection considering confidentiality, authenticity, and integrity of both data and services; g) privacy protection that ensures the private information of users of things; h) human body related services, i.e. it must be able to capture, communicate and process the data of the static and dynamic functions of the human body without human intervention. This service must be of high quality and high security; i) plug and play capability to reduce the complexity of use; and j) manageability, i.e, only high-level configurations should be managed by persons, but day-to-day operations should work automatically without the user participation.

2.2 IoT Reference Model

The ITU architecture reference model is composed of four layers: application layer, service support and application support layer, network layer, and device layer. Figure 2.1 shows the reference model.

- **Application layer:** contains all the IoT applications.
- **Service support and application support layer:** has generic support capabilities such as data processing or data storage; and specific support capabilities for particular applications.
- **Network layer:** network capabilities such as access and transport resource control functions, mobility management, authentication, authorization and accounting; and transport capabilities that focus on providing the control and management of data.
- **Device layer:** it has devices and gateway capabilities. The device capabilities ensure the correct communication between devices and the IoT network that includes their direct and indirect interaction (gathering and uploading of data), the construction of ad-hoc networks and easy changing of state (sleeping and waking

up). The gateway capabilities are related to the support of different kinds of wired and wireless connection technologies.

Complementing the four layers, there are management and security capabilities that act over the layers.

- **Management capabilities:** IoT management capabilities cover the traditional fault, configuration, accounting, performance and security (FCAPS) classes, i.e., fault management, configuration management, accounting management, performance management, and security management.

  They can be separated into two classes: generic and specific. Some generic management capabilities are device management, local network topology management, and traffic and congestion management.

- **Security capabilities:** Similar to management capabilities they are two classes: generic and specific. Generic security capabilities include: at the application layer: authorization, authentication, application data confidentiality and integrity protection, privacy protection, security audit and antivirus; at the network layer: authorization, authentication, use data and signalling data confidentiality, and signalling integrity protection; and at the device layer: authentication, authorization, device integrity validation, access control, data confidentiality and integrity protection. On the other hand, specific security capabilities are related to specific application requirements, for instance, security policies in a bank mobile application.

2.3 Office Automation

Before presenting the concept of office automation and its advantages, it is relevant to clear up two related concepts: *smart home* and *home automation*. Generally, these terms are used interchangeably but they have different meanings. *Home automation* is just one of the many things that a *smart home* can do. Actually, *smart home* technology is comprised of connected (by networking technology) smart devices. In other words, the whole of smart devices that are connected in the same network and has the capability for communication are part of a *smart home*. However, the automatic operation of devices, either by explicit programming or prediction mechanisms, allows for *home automation*. [106].

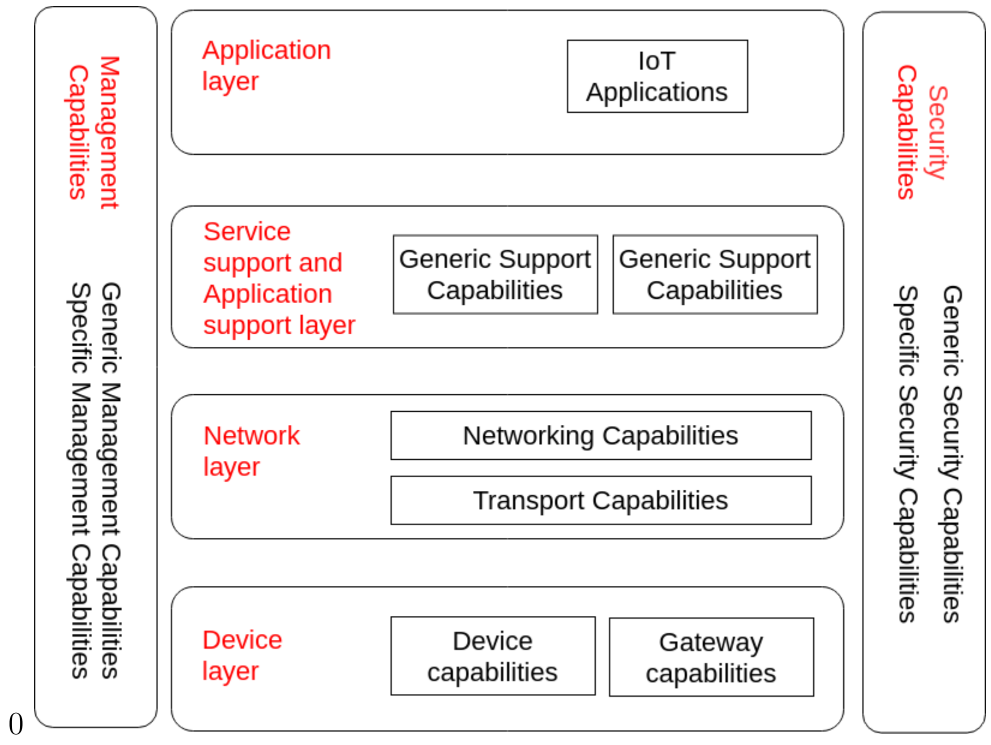

Management Capabilities
Generic Management Capabilities
Specific Management Capabilities
Application layer
IoT Applications
Service support and Application support layer
Generic Support Capabilities
Generic Support Capabilities
Network layer
Networking Capabilities
Transport Capabilities
Device layer
Device capabilities
Gateway capabilities
Security Capabilities
Generic Security Capabilities
Specific Security Capabilities
0

Figure 2.1: ITU IoT architecture model.

On the other hand, most *smart home systems* can be used in *smart offices*. Therefore, the same home automation methods can be used for office automation. For instance, access control, air conditioning and lighting systems are useful in both kinds of environments [60]. Perhaps the biggest difference between these two concepts is the purpose of their implementation. While the first one is used to provide security and comfort, the second also involves user productivity.

2.3.1 Benefits of office automation

- **Reduce operating costs:** In the United States, offices represent around ten percent of the energy consumption, while in developing economies they account for eight percent of the consumption. Most of this energy is wasted in heating, cooling, and lighting of unoccupied rooms. By using intelligent energy management systems, it is possible to reduce energy consumption by 20 percent, which is an economy of \$11.7 billion to \$20.5 billion per year in 2025 [60].

- **Improve security and safety:** IoT can implement security systems to detect intrusions with the help of cameras and pattern recognition software, sensors in doors and windows, and infrared motion sensors. Also, an IoT platform can provide more safety, as it can implement an intelligent fire alarm with sensors that detect smoke and carbon monoxide, activating different kinds of actuators like sending an emergency message [97].

- **Monitor employees:** Companies use IoT solutions to supervise their employees' activity. For instance, fitness monitors can track activity and interactions among employees. Companies can analyze this data to find possible improvements [60].

- **Improve comfort/productivity:** Office automation also helps office workers to improve their comfort and hence their productivity. Some simple tasks such as turning on the light can take some seconds but they can disrupt some very important task and provoke distractions, which require invaluable time to reconnect our mind with the purpose of the task. Currently, smart offices support indoor climate control to maintain an acceptable level of temperature, lighting and fresh air. Intelligent thermostats learn about the user schedule and program the moment when heating/cooling will activate, and the thermostat could also allow a smart phone interaction and control the consumed energy [97].

2.4 Ergonomics

According to the U.S. Department of Labor Occupational Safety and Health Administration [66], *ergonomics* can be defined as the study of work. More specifically, the science of designing the job to fit the worker, rather than physically forcing the worker's body to fit the job by adapting tasks, work stations, tools, and equipment. The application of these considerations can reduce physical stress and eliminate some musculoskeletal disorders related to work activity. Ergonomics is based on a number of scientific disciplines, including physiology, biomechanics, psychology, anthropometry, industrial hygiene, and kinesiology.

There are some workplace settings that influence negatively the worker productivity in case of sedentary jobs, for example, an incorrect keyboard inclination and the frequent use of the mouse, however [29] considers that the most important aspect is the worker's seat, which should be comfortable. This section focuses in environmental factors that affect productivity, such as noise and air quality (CO2, temperature and humidity).

Many studies about ergonomics reached interesting conclusions that are related to this research. For instance, there is a relationship between noise and worker performance. [22] indicates that a moderate level of noise (85 dB) can affect performance negatively. On the other hand, [48] concludes that besides noise affect the accuracy of the worker, it decreases the rate of work. A more recent study, [44] remarks that noise just affects women. Beyond it, [45] say that the impact of noise on performance depends on the nature of the task (reading, arithmetic problems), the nature of the distractor (noise or music) and the personality of the participant (extrovert or introvert). The conclusion of this study was that an environment with noise affects extremely introvert people when they do cognitive tasks, while there is no effect on extrovert people.

Quality of air is relevant for the occupant's health, productivity and comfort. [102] reports that the concentration of contaminants such as gases and particles affects the quality of air. Although the mixture of gases is difficult to measure, CO2 is easy to measure and generally is used as an air quality metric. [16] support the same idea affirming that high levels of CO2 concentrations (above 3000 ppm) indicate improper ventilation and poor air quality. The American Society of Heating Refrigerating and Air Conditioning Engineers (ASHRAE) recommends that indoor CO2 levels should not exceed the local outdoor concentration by more than about 650 ppm (parts per million). Given that an appropriate outdoor level of CO2 generally is between 380 and 500 ppm, then indoor CO2 levels may not exceed 1,035 ppm.

Temperature is also related to performance. Working in a hot environment (over 32 °C) has a negative impact as much as working in a cold one (below 10 °C) [70]. [43] complement the idea indicating that there is not a bad effect between 21 and 25 °C. On the other hand, the productivity decreases 2% per degree centigrade as the temperature increases above 25 °C. [101] affirm that although there is not a significant effect on cognitive performance, individuals feel better with variations in air velocity compared to a constant velocity. In his work, [38] show the importance of another factor: *humidity*. For that purpose, they performed an experiment exposing 30 females to four different scenarios with varying levels of temperature and humidity. The conclusions were that by decreasing temperature and humidity: i) perception of air freshness improved positively; ii) fatigue, headache, and difficulty in thinking decreased; and iii) the best scenario was 20 °C and 40% of relative humidity.

All these studies support our work, as they confirm that there is a strong relationship between office automation and ergonomics. This is because office automation provides functionalities that improve the parameters that are considered critical in ergonomics, such as temperature, humidity and lighting. We identified some ranges of parameters that ensures high levels of comfort, but we also find indications of many parameters depend on each person. Confirmation of this hypothesis will be demonstrated in the chapter 6, where the preferences of different users will be observed.

2.5 ManIoT

For implementing our proposal, our automation system required different technologies of sensors for keeping environmental data, RFID readers and tags for identifying users and a web application for handling the air conditioner unit. Further, the system should be able to work with more components such as smart bulbs, web services, and new sensors. That is the reason why we need to use a platform that supports a wide variety of devices regardless of the technology or the manufacturer.

Nowadays, there are many IoT platforms such as RestThing [74] and BEIOT [19]. Nevertheless, for implementing our proposal we will use the ManIoT platform [13]. The advantage over the other proposals in the literature is its capacity of handling any type of device. This flexibility is important to our work because during the implementation of the proposal we can decide to add or remove some physical magnitudes and actuators. Further, this platform was implemented in our laboratory and many of the components that it had already implemented are useful for our purpose.

ManIoT is a platform for managing IoT devices developed in the Wireless Network (WINET) [1] [13]. Its objectives are to integrate and to manage the individual functionalities of each device at an IoT network, allowing the creation of context-aware services. For example, the use of a light sensor to control a bulb's brightness. ManIoT manages the devices locally and remotely but also supports the addition of new kinds of devices. Moreover, it supports basic IoT services such as node discovery, data storage, and authentication.

The advantage of ManIoT over other platforms is that it can establish two management scopes: local and global. Hence, ManIoT uses a *local manager* for handling any device that composes a specific scenario. For instance, a *local manager X* regulates the light intensity and turns light bulbs on or off in a house. However, a *local manager Y* is in charge of a scenario for managing the energy in a factory. Then, it is clear that a *local manager* acts in his own scope and cannot influence another manager. In the case where the electric company wants to program power cuts in a city, the *global manager* acts enforcing his policies over the *local manager*. Then, the task of the *global manager* is to standardize the actions performed in different scenarios using high-level rules. The topology of the ManIoT platform is represented in Figure 2.2, where it is shown that the global manager sends high-level rules to the local managers.

As a consequence of using two scopes, it is possible to: a) manage the scenario

[1]WINET is a research group of the Computer Science Department (DCC) of the Federal University of Minas Gerais. The group participates in several research projects in the topic of wireless networks, developing new networking protocols as well as contributing to network management and security services for wireless communication. WINET has several ongoing national and international partnerships, and our researchers publish in the main symposia and journals of the area.

policies independently from other scenarios; b) inject more relevant and standard policies over the existing ones using the global manager; c) support new scenarios without altering other scenario features.

The platform defines a data model and an information model with the objective of standardizing the data format used in communication between applications, services and devices. In that way, ManIoT stores device id, device status (on/off) among other features and it can add new ones according to the needs of the scenario. In addition, it employs open protocols and standards of data modeling such as XML (eXtensible Markup Language) and REST.

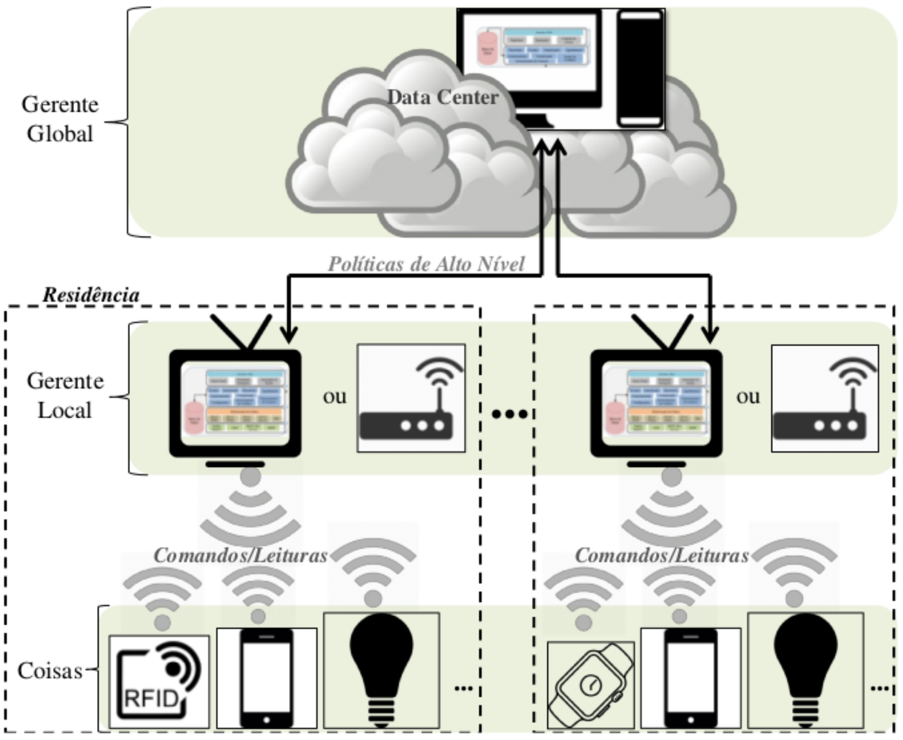


Figure 2.2: Topology of the ManIoT Platform.

2.6 Machine Learning Algorithms

The term machine learning refers to the automated detection of meaningful patterns in data. There are many applications such as search engines, clinical diagnostics, market analysis, robotics, etc. In the book *Understanding Machine Learning From Theory*, the authors mention four parameters to classify the learning algorithms [88].

- **Nature of the interaction:** The learning could be *supervised* or *unsupervised*, in the first case, the experience, also known as training example, contains significant information that is missing in the unseen test examples. It is expected that the acquired expertise predict the missing information for the test sample. In other words, the environment could be consider as a teacher that supervises the learner by providing extra information (labels). In the case of *unsupervised learning*, do no exists the concepts of training and test data. The model is trained with the purpose of generating a compressed version of the data. The best example of unsupervised learning is clustering the data into subsets of similar characteristics.

In addition, there is an intermediate learning setting, called *reinforcement learning*. In this case, an agent explores the space of possible strategies and receives feedback on the outcome of the choices made. Hence, using this information a good policy must be deduced [54].

- **Learner role:** the learning can involve *active* or *passive* learners. The first kind of user interacts with the environment at training time by posing queries or performing experiments and, the second one just observes the information provided by the environment without causing any influence. For best comprehension, considering the use case of spam labeling, a *passive learner* just waits for users to mark the e-mails coming to them, but an *active learner* could select some e-mails and ask users to label them, in this way the learner can understand what spam is.

- **Helpfulness of the Teacher:** learning can be based on a process that involves a helpful teacher, who is trying to give the learner some useful information for achieving the goal, but another way to learn is by just observing the data nature (environment) without having any interaction, hence, the learning scenarios can be model by postulating that the training data is generated by some random process. This idea is the basic building block in the concept of *statistical learning*. Learning also occurs when the learner's input is generated by an adversarial teacher. For instance, in the spam filtering use case, the spammer makes an effort to mislead the spam filtering designer.

- **Online versus Batch Learning Protocol:** A batch learning algorithm uses a training set to generate an output hypothesis, which is a function that maps instances of an input set X into a label set Y. The batch learning algorithm is expected to do statistical generalization, in the sense that its output hypothesis predicts the labels Y of previously unseen examples X sampled from the distribution. The online learning is based on a learner that receives a sequence of data entries and processes them one by one. In each round the algorithm uses an internal hypothesis to predict the label of an instance. This label is stored in memory, since it will be compared with the true value of the label to update the hypothesis ([36]).

Four this work we choose four predictions algorithms: *K-Nearest Neighbor (KNN)*, *Multilayer perceptron (MLP)*, *Random Forest (RF)*, *eXtreme Gradient Boosting (XG-Boost)*. All of these are the most representatives algorithms of their style, for example, there several boosting algorithms but XGBoost is the most popular. The comparison of these algorithms helps us to determine which algorithm we have to use as part of our predictor.

2.6.1 K-Nearest Neighbor

KNN is one of the most popular methods for classification and regression problems, it is based on the idea that instances generally exists in close proximity to other instances that have similar properties. Hence, the label or value of an instance can be determined by observing their k neighbors ([30]).

The classification algorithm consists of determining the label of the class of an unclassified sample point from the labels of the neighboring elements. To execute this algorithm, two parameters must be defined first: the number of classes and the number of k neighbors to be consulted. Hence, it starts by calculating the distance of the new point with respect to all the elements of our dataset. The next step is to select the k neighbors with the least distance, followed by the calculation of the mode of these. The mode determines the label of the sample point.

K-Nearest Neighbor for regression problems follows the same mechanism as the classifier, however, instead of choosing by voting the most repeated class in its neighborhood (mode), a new number is generated, which represents the mean of the neighbors ([11]). The simplest KNN regressor assigns uniform weights to all neighborhoods of the query point. There is also a variation that uses a distribution based on distance, that is, each point in the neighborhood is assigned a weight proportional to the inverse distance of the point, or even another type of distance can be used ([53]).

2.6.2 Multilayer perceptron

The multilayer perceptron, as the name implies, is a neural network formed by several layers of perceptrons, which are the simplest neural networks. [81] introduced the perceptron as a binary classifier that transforms an input value to an output value through a function. The perceptron has many limitations, but the main ones are: (a) it is cannot deal with problems that are not linearly separable, and (b) it can only generate a binary output. However, MLP is capable of solving them, this due to the use of the back-propagation learning algorithm that helps to define the weights ([72]).

MLP consists of a layer of input neurons, a layer of output neurons and several layers of intermediate neurons, also called hidden layers. The input values are introduced into the network through links, which have random weights that modify their values, in the same way all the neurons of each of the layers are connected to the neurons of the anterior and posterior layers by similar links. The prediction process consists of introducing the data in the network, which is modified by the weights of the input layer links and each of the intermediate layers until the end. The final layer will return the output of the network which will be evaluated through the error it generates.

The power of the MLP is to learn about that error and the best-known way is through the technique called backpropagation. This technique consists of going back from layer to layer until the beginning correcting the values of the weights of the links. The correction of errors is given by the derivation of the function used in each neuron considering also the value of the error. When the input layer is reached, the prediction process must be repeated and the final error evaluated again, which must be less in each new iteration. MLP stops when it is detected the error is no longer considered, returning the final prediction ([100]).

2.6.3 Random Forest

RF is a method to solve classification and regression problems. The first general idea of RF was introduced by [50], who indicated that the accuracy of the predictions could be improved without suffering overfitting from the idea of Random subspace selection. This means that decision trees are constructed from a reduced version of feature dimensions. This is to prevent a single characteristic from influencing the entire learning and prediction process. Other works that contributed to the final definition of random forest were [12] and [37].

[21] defined properly RF, applying the random subspace method together with bagging. The latter is a method that predicts a class or final value after consulting the mode or the mean of a set of decision trees. Then, RF creates a set of datasets from a data set with dimension d. The elements of each dataset are chosen randomly with replacement, so there may be repeated elements. Then, RF uses these datasets to create trees leaving aside some variables or dimensions. The number of variables that must be

left out can be $\sqrt{d}$ and must be chosen randomly. To classify a new entry data, it must pass through all the trees and generate the outputs. The final result will be the most voted class or in case of regression a mean of all the predictions.

2.6.4 eXtreme Gradient Boosting

Also known as XGBoost, it is a scalable end-to-end tree boosting system that became famous for achieving the best results in machine learning competitions ([25]). Similarly to Random Forest, XGBoost obtains a final result with the help of a set of weaker predictors. Nevertheless, it is an implementation of the gradient boosted trees algorithms, which means that it uses the *boosting* method with decision trees to ensure the best final solution.

Boosting is a meta-supervised learning algorithm that assembles a set of weak tree classifiers to obtain a robust global ranking. However, this method does not consult the classifiers at the same time, but in a sequential manner. Supposing a set of weak binary classifiers, the training instances in the space X from the distribution D, and the space X is composed of three parts X_1, X_2, and X_3. Boosting tests a first classifier H_1 and just obtain correct classifications in spaces X_1 and X_2, meaning that H1 has problems to classify in X_3. In the next step, boosting derives a new distribution D' from D (with the intention of making the mistakes of H_1 more evident). Hence, a classifier H_2 is used to try to classified the entities in the space X_3. Nevertheless, H_2 also has problems and cannot give correct classification for a specific space, in this case, X_2. The combination of H_1 and H_2 classifies perfectly in X_1 but presents some mistakes with X_2 and X_3. A new distribution D'' is derived from D', and a classifier H_3 is used to classify the problematic data in X_2 and X_3. Finally, the combination of H_1, H_2, and H_3 forms a perfect classifier ([109]). In the case of regression, the weak learners are regression trees and each tree maps an input data point to its leafs that contains a continuous number. XGBoost minimizes a regularized objective function that combines a convex loss function with a penalty term for model complexity ([25]).

2.7 SHAP

Today, many methods aim to understand machine learning models. Nevertheless, most of them are based on heuristic and not individualized for each prediction. The authors of SHAP, Scott et. al, highlight that traditional feature attribution methods are inconsistent because they can assign lower importance to some features in cases when they present increased importance. Hence, false interpretations can be created [59].

SHAP (SHapley Additive exPlanations) is an approach that unifies many methods of machine learning model interpretation to explain the output of any machine learning model, helping to understand data with better intuition, improving running time, clustering performance and presenting a reduced feature importance detection [59]. To achieve it, the author connects some methods such as [78], [98], [89], [32], [15], [58] and [86] with game theory.

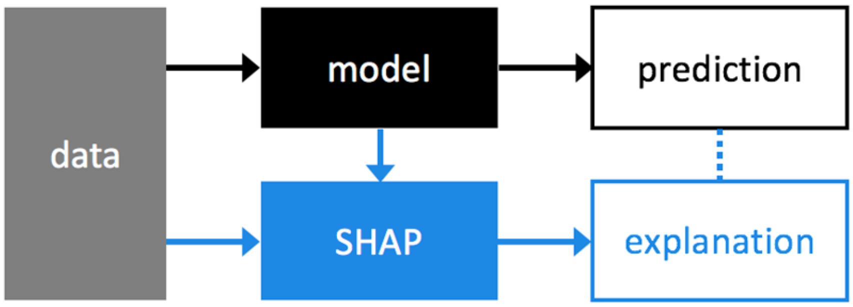


Figure 2.3: SHAP explanation model.

Figure 2.3 gives a notion of tool intervention over the prediction model. It has a method to create a model explainer which is the structure in charge of *shap_values* generation. The *shap_values* are a set of numbers that represent the features relevance. SHAP makes available five explainers depending on the kind of predictor model:

- **TreeExplainer:** a fast and exact algorithm to compute SHAP values for trees and ensembles of trees.
- **DeepExplainer:** a faster (but only approximate) algorithm to compute SHAP values for deep learning models that are based on connections between SHAP and the DeepLIFT algorithm.
- **GradientExplainer:** implementation of expected gradients to approximate SHAP values for deep learning models. It is based on connections between SHAP and the Integrated Gradients algorithm. This explainer is slower than DeepExplainer and makes different approximation assumptions.

- **LinearExplainer:** implementation for linear models with independent features. It also allows to show feature correlation through the calculation of the co-variance matrix.

- **KernelExplainer:** implementation used for any model. It makes no assumptions about the model type, for that reason is slower in comparison to any models for specific algorithms, like the four previous ones.

In SHAP, the author proposes a rich visualization of individualized feature attributions that improves over classic attribution summaries and partial dependence plots, and a unique supervised clustering (clustering based on feature attributions). SHAP have four plots:

- `summary_plot`: This method generates a colored graphic where is possible to identify the level of relevance of each feature over every sample. In Figure 2.4, the vertical bar on the right of the graphic represents this relevance, where color red indicates a high relevance. The plot also shows the impact distribution of each feature over all samples. In addition, it is possible to set this method to use the mean absolute value of the SHAP values for each feature, the result is a standard bar plot (produces stacked bars for multi-class outputs) as shown in Figure 2.5.

- `dependence_plot`: Different to `summary_plot`, this method is useful to reveal dependency of the model over a given feature. It compares the SHAP value of the selected feature versus the value of the feature for all the examples in the dataset. Considering that SHAP values represent a feature's responsibility for a change in

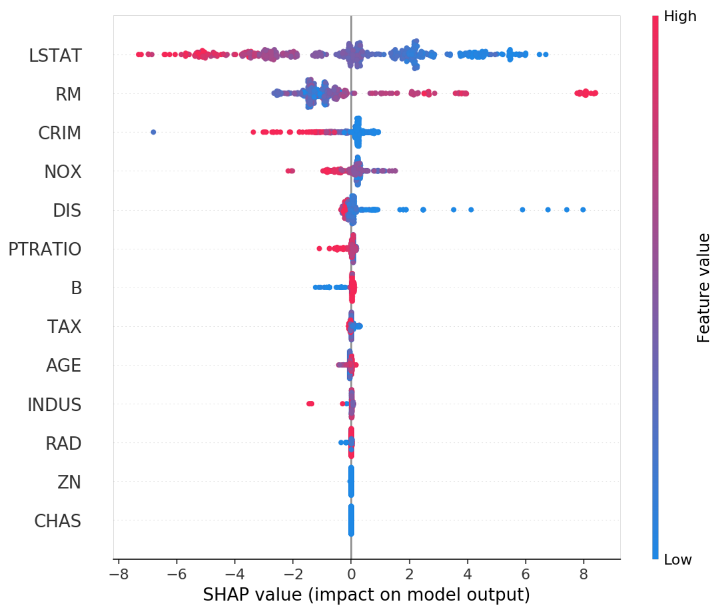


Figure 2.4: SHAP `summary_plot`. It shows the effects of all the features for dataset Boston.

the model output and the boston dataset, the plot in Figure 2.6 shows the change in predicted house price in relation to the number of rooms (RM). Vertical dispersion at each value of RM represents interaction effects with the feature RAD(index of accessibility to radial highways). This feature was selected automatically by SHAP. From the plot, it is induced that the average number of rooms per house has less impact on home price for areas with a high RAD value.

- `force_plot`: This method allows to explain which feature is contributing more to the prediction. It can compare the SHAP values with the values of a specific input instance as shown in Figure 2.7. But also, it can show a graph with the interaction with all between all the input instances as shown in Figure 2.8. In the Figures, features pushing the prediction higher are shown in red, and those pushing the prediction lower are in blue.

- `image_plot`: This plot is used for an explanation of deep learning models as shown in figure 2.9, which is related to the well-known dataset MNIST. The plot explains ten outputs for four different images that represent digits between 0 and 9. Red regions are more relevant for the model's output while blue regions are less relevant. From the image, it is possible to conclude that for the 'zero' image the blank middle is important, while for the 'four' image the connection on the top determines if is four our nine.

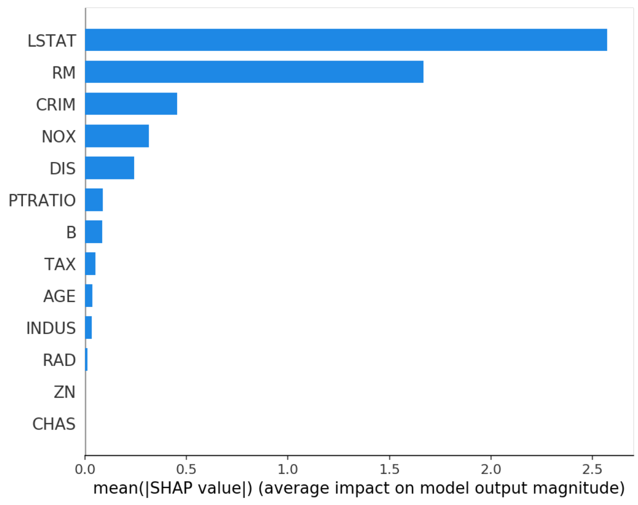


Figure 2.5: SHAP `summary_bar_plot` that shows the effects of all the features for dataset Boston.

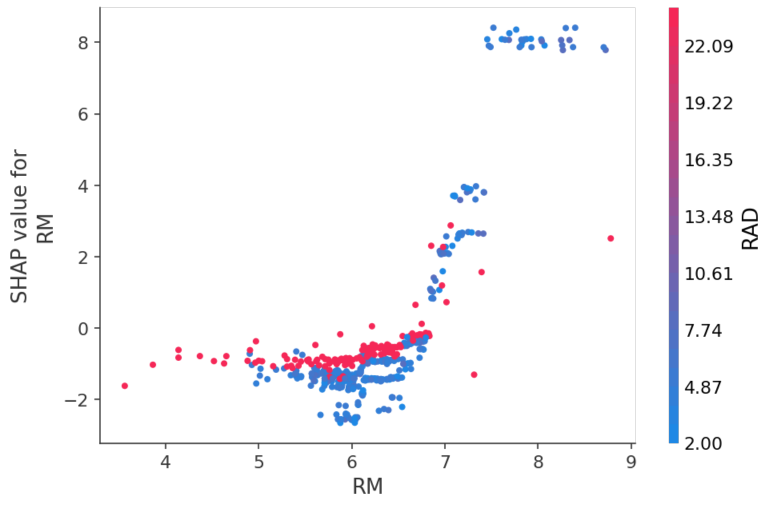


Figure 2.6: SHAP `dependence_plot` to show the effect of a single feature across the whole dataset Boston.

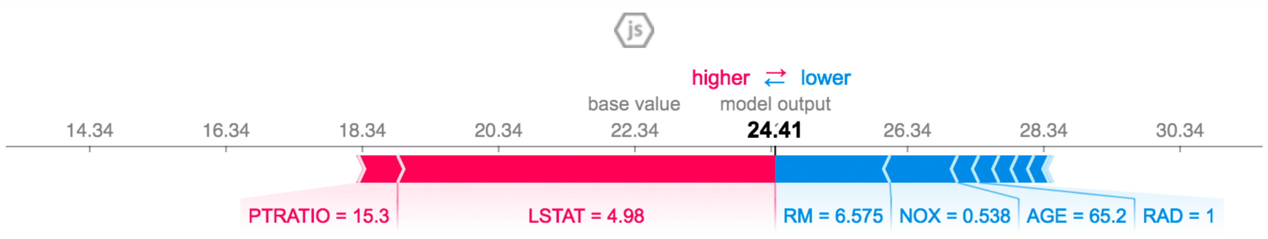


Figure 2.7: SHAP `force_plot` to visualize the prediction's explanation of one element of the for dataset Boston.

2.8 Conclusion

This chapter shows the concepts use in our proposal. We presented the concept of the *Internet of Things*, *office automation*, *ergonomics*, and its importance. Furthermore, we present the ManIoT because we use this platform to implement the part of office automation. We compare some prediction algorithms to choose the best and use it as part of our solution. Finally, we present SHAP, a tool that shows us the correlation between variables, which is useful to understand that every user has different preferences.

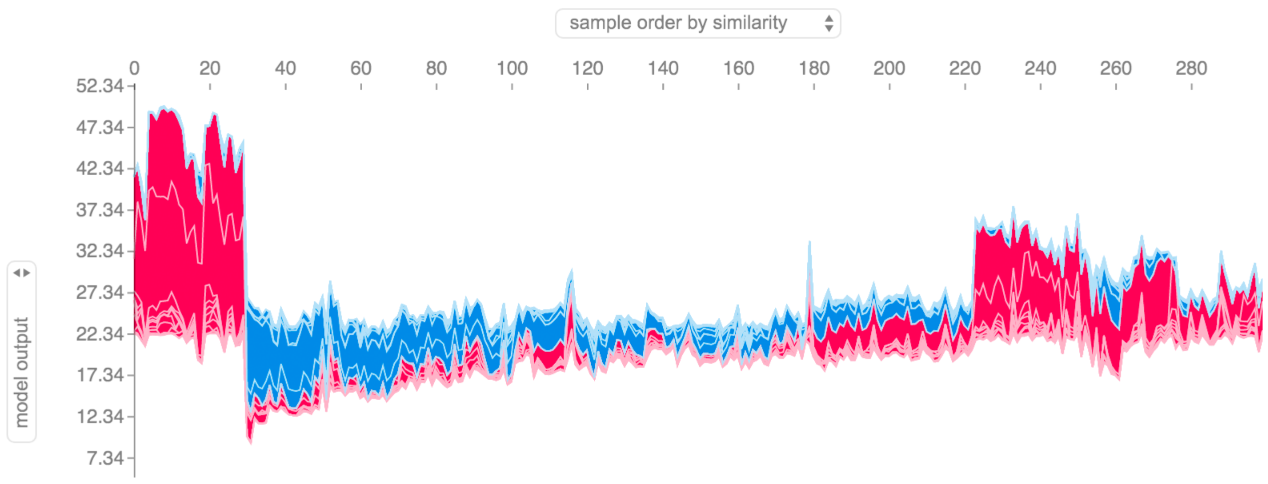


Figure 2.8: SHAP `force_plot` to visualize the prediction's explanation of all the predictions for dataset Boston.

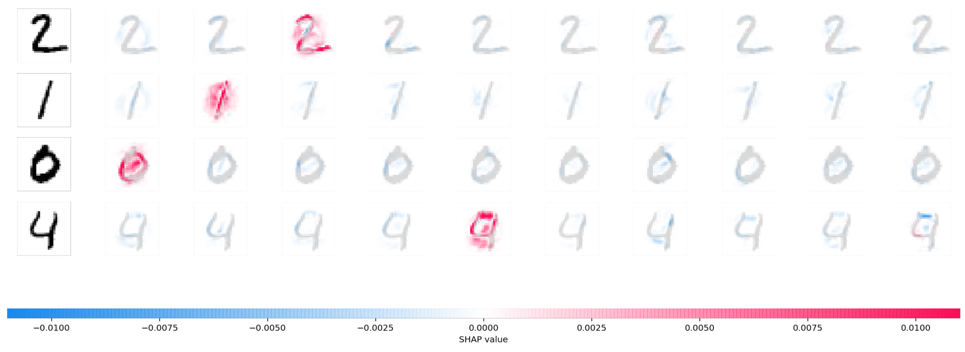


Figure 2.9: SHAP plot of the feature attributions for dataset MNIST.

Chapter 3

Related Works

The development of this dissertation is not only intended to provide a contribution to the scientific community on the subject of predictions in intelligent environments but also to create a prototype for a future marketable product. For these reasons, this chapter presents three sections: a list of similar products found in the market (3.1.1), a list of patents related to our work (3.1.2), and a revision of the literature about prediction algorithms for intelligent environments (3.2).

3.1 Environment Automation

3.1.1 Products

The different devices in this list generally work with air conditioning units, lights, blind shades, and switches connected to an electronic device. It is important to say that most of the products that are developed to improve the office environment are useful for home automation and vice-versa.

- *Kapsul* is a window air conditioner device that measures considerable changes in the room environment to change its configuration. For instance: if the room is very hot, i.e. it surpasses 25 degrees Celsius, then the air conditioner turns on automatically. Furthermore, *Kapsul* has an application that allows to control and schedule certain configurations [4].

- *Ambi Climate* is a device that controls air conditioning units of different brands as long as they have infrared communication. It regulates the temperature to avoid extreme climates (too hot or too cold) guaranteeing the user comfort and saving energy. The user can change the air conditioning unit parameters through a mobile application. *Ambi Climate* learns about the user based on his preferences, and other data such as the temperature, metabolism, humidity, air flow and other environmental conditions in the room [1].

- *Evapolar Personal Air Control* is a cooler device that humidifies, purifies and chills the air using evaporative technology. It is appropriate for cooling small spaces in fronts of it, such as a bed, a couch or a desk. *Evapolar* is made based on an exclusive nano-material that avoids bacteria proliferation. One of the most innovative characteristics is its patented nano-material, *EvaBreeze*, which avoids bacteria proliferation. Through a mobile application, the user is able to adjust the humidity, temperature, and the interface color. *Evapolar* is compatible with most smart home systems [3].

- *Nature Remo* is a device that helps to control the air conditions inside a room. It has integrated some sensors for detecting movement, temperature, humidity, luminosity, and noise. *Nature Remo* has a mobile application that can be used to control the air conditioning units (AC) as long as they have an infrared interface. Moreover, it is possible to execute some changes remotely, disable the AC when the user is not present and program some functions according to rules defined by the user. *Nature Remo* is compatible with *Amazon Echo, Google Home, Alexa, and IFTTT* [7].

- *Sensibo Sky* is compatible with *IFTTT, Google Home, SmartThings, HomeBridge*, and *Amazon Echo*. It uses geolocation to recognize if the user is inside the room. Moreover, it allows a mobile application to control the device remotely [9].

- *Mommit Cool* is a kit for controlling the AC, it is composed of a) a mobile application used to change the AC parameters and program some preferences in certain schedules, b) the *'Mommit cool'* which is a device that adheres to the air conditioner turning it intelligent; and c) the *'Mommit gateway'* that serves as a gateway to link the AC and the mobile application. *Mommit Cool* has a *smart mode* used to learn about user preferences. Additionally, it uses geolocation to identify when the user is near or far from home and then deactivate the AC for saving energy [6].

- *Distech Controls' Smart Room Control Solution* is a system for controlling the AC, lights, and sun-blinds inside a room such as offices, waiting rooms, bedrooms among others. This solution considers some parameters defined by energy saving studies for adjusting the devices. In comparison with the other solutions, it does not have a mobile application, but it has a specific interface panel for each actuator. Its biggest advantage is that can control several devices and sensors, however, it does not learn about the user [2].

- *Klevernes* is a smart lighting system of switches, dimmers, and outlets compatible with the most famous home automation systems such as *Amazon Alexa, Google*

Home, SmartThings. The installation of these devices is intuitive and fast. *Kleverness* controls its devices with gestures and voice. It uses artificial intelligence to learn about the user routine to automate some tasks. It also mimics the user to make the home appear occupied when the user is on vacation. The user can use the *Kleverness* mobile application to change or schedule the devices [5].

- *Nest Learning Thermostat* allows making a schedule with specific configurations. It saves energy by applying some rules that help to turn off the HVAC when it is not necessary. Some rules are based on the user presence, which is detected with presence sensors and geolocation. Nest also is oriented to give security to the user, since it sends alerts when anomalous conditions occur and mimics the user behavior when he/she is out of the home. Nest has a mobile application which configures the schedules, but also controls the diversity of devices that are compatible [8].

3.1.2 Patents

There are many patents related to this work, many of them offer a wide variety of functions such automation of specific devices, remote control through mobile phones, inclusively programming actions through rules. The objectives can vary from the point of view of the purpose to identify the guidelines for safety, energy saving and comfort. This section presents the patents studied in organized according the status. We consider four patent status: (a) *application*, it is an initial status that prevents others from practicing the claimed invention; (b) *granted*, when the invent is already protected; c) *expired*, when the protection time finished; (d) *abandoned*, when the applicant let to pay the fees.

We search this list of patents in using the tool Google Patents, where a search was made based on keywords such as *smart office*, *smart home*, *home automation* and related actuators, such as *air conditioning*, *blinds*, *switches* and *lights*. Each of the patents found had a list of related patents, which were also analyzed. Of all the patents found, it was filtered by those most current, but also those that had the characteristics closest to our solution or to more specific parts of it.

- Application:
  - [$CN104914899A$] *Multi-functional intelligent office system.* Multi-functional system for intelligent room. Monitors the environment, has a security module and another for access control. It makes use of several sensors to detect luminosity, temperature, humidity, and smoke, and these values are transmitted

using the Zigbee protocol. The system can establish some safety rules such as sending an alert if smoke is detected. For access control, it uses a magnetic card system. Additionally, the system relies on remote monitoring via mobile devices [99].

- [US20030149576A1] *Personalized smart room.* Automation of hotel rooms. At the time of booking the hotel user must fill out a form with his preferences, so you can pre-configure some functions inside the room such as turning the television on or off, opening curtains, turning off lights at certain times, etc. These functions may or may not require direct user interaction, but the idea is that they are automatic [94].
- [US20130099011A1] *Energy efficiency promoting schedule learning algorithms for intelligent thermostat.* Programmable thermostat changes the temperature at a specific time or at a specific schedule. Moreover, it learns the user habits and generates an automatic schedule depending on the interactions. The claims are: a) interface with ring shape for interactions, sensors, communication with HVAC systems, (b) use of predefined rules; (c) method for generating automatically a programmed schedule [61].
- [WO2013058820A1] *User-friendly, network connected learning thermostat and related systems and methods.* Intelligent thermostat with an internet connection. Composed by a) a wall-mountable back plate containing a microcontroller used for capturing sensor data and managing the air conditioners or heating apparatus; b) mobile central unit containing a microprocessor for processing machine learning algorithms, LCD screen, buttons and modules for wireless communication, also contains a rechargeable battery and power connector. The claim is a) A programmable device for controlling an HVAC (heating, ventilation, and air conditioning) system, which is the active state can perform high energy consumption, processing and storage activities, such as user interaction and customs learning; and that in the inactive state can perform scheduled tasks of low consumption [40].
- [WO2015025315A1] *Method and apparatus for controlling an HVAC system.* The patent references a method and controller of the HVAC system. The claims are a) a controller with a processing and storage unit, b) set of mapping rules between devices and possible actions, c) component to choose which action to apply, d) communication component to transmit data to the HVAC unit [82].

• Granted:

- [US6964370B1] *RFID smart office chair.* A chair that fits according to the preferences of the user who is sitting. To recognize the user the chair uses an

RFID reader, considering that the user must use an RFID device. The system has a storage unit that is accessed t using a wireless network. The storage unit is used to save or retrieve the preferences of each user, so each time the user is recognized the chair adjusts automatically. The chair allows you to raise or lower the back of the chair, adjust the height of the base of the seat and move the arm to the seat [49].

- [CN105105510A] *Intelligent adjusting desk and control method thereof.* Intelligent desk that lets you adjust some features like height or board slope. It can also detect which user is using the system. All settings are automatically performed to improve user posture and sedentary behavior [105].
- [CN204393704U] *Intelligent office desk.* A Desk that has mechanical structures such as lifting rods, screw rods, and motors along with a display module that allows user interaction. The aim of this invention is to improve user comfort and reduce customizing the conditions of the desk according to certain parameters that are standard or defined by the user [35].
- [US20160260019A1] *Smart office desk interactive with the user.* A desk that contains various sensors, mechanical components (mechanical feet, tray, and footrest) to adjust and a user interface that shows some information such as postures, user habits and interactions with the system. It also has other services such as calendar and event information. This desk learns about user preferences using machine learning algorithms so it can propose certain physical actions. The outputs of the system are recommendations and statistics about the daily activity of the user [80].
- [CN207306428U] *An intelligent control office furniture combination of learning health.* This patent is oriented to take care of the user health, it refers to some furniture inside an office, such as desk, control panels, speakers, infrared sensors for presence detection, lamps, lights sensible to the presence and other mechanical parts that can massage the user [104].
- [US20140266669A1] *Devices, methods, and associated information processing for security in a smart-sensored home* The patent is related to apparatus, systems, methods, and related computer program products for providing home security objectives. This patent claims tree general specifications: (a) a method of calculating and reporting a security score for a home; (b) a method of creating and using a neighborhood security network to detect security-related conditions and to distribute notifications; and (c) a method of pre-alarm condition trend detection and notification.
- [CN201220438536] *Office and learning health care furniture combination capable of being intelligently controlled.* A system that learns to take care of

health. It is composed of a set of furniture such as a chair, desk, curtain, lamp, keyboard, speakers, pedal and infrared sensors for the presence. The system adjusts the different devices for optimum health configuration. For example, the chair is in a way that helps the person to improve the posture or a suitable luminosity to avoid vision problems [68].

– [$KR101212778B1$] *Cloud computing based smart office system and server for managing the same and method for managing the same.* This patent is related to a network of intelligent devices or multi-sensors that are connected to a server or a cloud computing system, that has as purpose to provide home security. For instance, some rules can be pre-defined or the system can provide machine learning models to mimic the user costumes during the user absence. For example, if a dog barks at midnight, the system can turn on the light as a human does. The patent claims are (a) a method based on the system logs that can calculate and report a score to define the security level at home, (b) a method to use a security network in a neighborhood and detect some conditions to distribute notifications, (c) a method to pre-define and send alarms [31].

– [$CN202025224U$] *Intelligent office energy-saving control system.* This system based on Zigbee technology is oriented to save energy within working environments. The purpose of the system is to use pre-defined rules to control all the devices inside a room such as the air conditioner, lamps, computers and other electrical systems in general. It consists of a central unit, several sensors, and controlled devices. Communication between the devices and the central module is via Ethernet or via a wireless network. The central unit is also connected to a GSM/3G module to save power during wireless communication [107].

- Expired:

  – [$US3640455A$] *Air temperature control system.* Is an HVAC system that regulates the temperature in a set of rooms. The patent is related to the device that uses a thermostat and some pre-defined rules to produce hot or cold air to regulate the air temperature inside the room [67].

  – [$US5544036A$] *Energy management and home automation system.* System oriented for power management and home automation consisting of one or more power controllers and their respective consumer devices. Each controller is programmed to act according to a list of pre-defined rules, but the user can modify it. The system is programmed to consume less energy when environments are idle, but the energy consumed may also be reduced in the event of a request from the utility. The claims are: a) system for controlling the energy consumed by the device, b) energy management system of large areas, c) system for power management based on a set of rules defined by an occupant [23].

- Abandoned:
  - [US20140098247A1] *Home Automation And Smart Home Control Using Mobile Devices And Wireless Enabled Electrical Switches.* A platform that allows to control and automate houses. The system operates on smart plugs and switches, allowing the user to turn them on and off. These devices are on a wireless local area network and are handled through mobile applications. Additionally, each switch contains some sensors such as smoke, temperature, light, pressure among others [75].

The table 3.2 shows the different aspects considered by each of the patents considered in this bibliographic review. In order to understand the comparative table, it is necessary to consult the table 3.1, which shows the relationship between the abbreviated name of the column and its description. In the patent comparison table we can observe that the first two letters of the patent code indicate the country of origin. In the list we can see eight patents from the United States (US), six from China (CN), and one from Korea (KR). In addition, there are two with the prefix WO, which refers to WIPO ("World Intellectual Property Organization").

From the table we can infer that most patents related to internal environment automation (homes and offices) are referred to hardware components and generally the devices that are most patented are related to modify the air conditions, so we have intelligent thermostats and controllers of air conditioning units. On the other hand, all patents allow the user to interact directly with the system and a few do not have a mobile application, instead they offer a computer application or a panel. Another observation is that almost all patents are focused on solving an isolated problem such as offering a comfortable seat or lighting management and are not flexible to work with devices from different manufacturers. Furthermore, there is no patent that refers to an integrated automation system. In relation to the detection of presence, only some half of the patents consider it within their solutions. Finally, only four patents refer to the use of machine learning algorithms to predict the user's actions.

Column name	**Description**
Serial number	Patent number
Date	Patent start date
Pre	Recognition of user presence
Swi	Management of switches
ACU	Management of air Conditioning Units
Hea	Management of heating Units
Lig	Management of lights
Sha	Management of shades
Fur	Related to furniture such as chair and desks
Har	Creation of hardware components
Aut	Environmental automation
Fle	Flexibility to incorporate devices of different manufacturers
Use	Possibility of direct user interaction through an interface
Mob	Mobile interface
ML	Learns about the user through machine learning

Table 3.1: Description of the columns of the patent table

3.2 Prediction Algorithms oriented to Smart Environments

Before the first integrated systems of automation of environments appeared, some precursor technologies had been developed such as Jini, Bluetooth, SIP standards, Xerox PARC's Zombie Board, Microsoft's Home project, the Cisco Internet Home, and the Verizon Connected Family project. Two concrete solutions were the MIT Intelligent Room [96] and Georgia Tech Aware Home [56]. They introduced houses with plenty of sensors to track the user activities.

Although there were many advances in the automation industry, there was great resistance to implement automation systems. [65] identified the main reason: *the difficulty of use*. Users had to manually configure the devices to obtain optimal results and many

Serial number	**Date**	**Pre**	**Swi**	**Acu**	**Hea**	**Lig**	**Sha**	**Fur**	**Har**	**Aut**	**Fle**	**Use**	**Mob**	**ML**
CN104914899A	2015-09-16	X		X	X	X			X	X		X	X	
US20030149576A1	2003-08-07		X	X	X	X	X		X	X		X	X	
US20130099011A1	2013-04-25			X	X				X	X		X	X	X
WO2013058820A1	2013-04-25	X		X	X				X	X		X	X	X
WO2015025315A1	2015-02-26			X	X				X	X		X	X	
US6964370B1	2015-11-15	X						X	X	X		X	X	
CN105105510A	2015-09-19	X						X	X	X		X		
CN204393704U	2015-06-17	X						X	X	X		X		
US20160260019A1	2016-09-08	X						X	X	X		X	X	X
CN207306428U	2018-05-04	X				X		X	X	X		X	X	
US20140266669A1	2014-09-18	X	X	X	X	X				X	X	X	X	X
CN201220438536	2012-08-31					X	X	X	X	X		X		
KR101212778B1	2012-04-25	X	X	X		X			X			X		X
CN202025224U	2011-11-02		X						X	X		X	X	
US3640455A	1972-02-08			X	X				X	X		X		
US5544036A	1996-08-06		X	X	X	X			X	X		X		
US20140098247A1	2014-04-10		X	X	X	X			X	X	X	X	X	

Table 3.2: List of patents related to office automation

times preferred to hire the services of technicians, which implied additional expenses. [65] proposed a solution to this: the Neural Network House (ACHE), which analyzes (using RC thermal model and reinforcement learning) the lifestyle and desires of the inhabitants to learn about the users and then anticipate their needs. ACHE had two objectives: (a) to anticipate inhabitants needs (heating, lighting, ventilation and inclusively water temperature) by learning about them; (b) to save energy.

MavHome, an integrated smart home with an intelligent agent, was developed at the University of Texas at Arlington and included artificial intelligence, machine learning, databases, mobile computing, robotics, and multimedia computing [28]. Its architecture (similar to most smart home systems), has four layers: the physical layer, communication, information and decision. The physical layer consists of all the hardware used as the actuators, and sensors. The communication layer allows the flow of data. The information layer is responsible for collecting and storing the data. Finally, the decision layer uses the information acquired to choose the most appropriate actions.

Many automation solutions focus their efforts to become as autonomous as possible. To achieve it, they base their decision making on predictive algorithms that use historical information of the user's actions. Most of these algorithms were based on Markov models or Bayesian networks. This section focuses on the prediction algorithms used to make decisions in smart environments [27]. We based this review on the survey of prediction algorithms for smart homes presented by [103].

3.2.1 Active LeZi (ALZ)

The prediction algorithm used by MavHome was Active LeZi. It is an improvement of a compression algorithm called LZ78, which allows understanding files by calculating the probability of occurrence of a sequence of characters [27].

[110] explain that LZ78 generates a dictionary (tree structure) for the character sequences, which can be used to apply a Markov model. In the beginning, the tree is null, but it is filled as a new string is read. The algorithm has a sequential iteration of each input string. Each time a sequence of characters is not found, it is inserted to the dictionary. If the sequences are found, they are replaced by their code in the dictionary. This is a compression model, then it has an encoder and a decoder. However, to be used as a predictor, only the encoder is necessary.

LZ78 could not be used for predictions because it has problems with the bounds of the sequences but also with its efficiency. Hence, some improvements were proposed, *LeZi update* [18] and *Active LeZi* [46] solve the issue of convergence, but only the second one gives a solution for the problem of bounds of sequences. It uses windows that change their size k as the algorithm is running. Initially, the window size is defined by the longest string observed in LZ78 parsing. Lezi update contains more nodes then LZ78, hence, it provides a more complete Markov model and more storage capacity. As consequence, it allows an improvement of performance. *Active LeZi* uses the LZ78 tree to build a probability distribution, which is used for the prediction. The symbol with more probability is the next symbol to appear.

[41] developed the *time-varingLeZi(TALZ)* which improves *Active LeZi*. They observed patterns in human activity, meaning that people tend to execute the same actions at the same hours. Hence, they took advantage of this knowledge and created subtrees from the context of a specific hour of the day.

3.2.2 Flocking

Flocking is a concept associated with groups of some species of animals, specifically birds [77]. In computer science, it is used to cluster elements according to their features. Flocking algorithms are used to model problems in robotics, artificial intelligence, and are also relevant in automated environments [64].

The clustering of elements is useful to analyze the activity of daily living (ADL) of users, especially those with anomalous behavior. A group of ADLs is tried as a cluster which is processed quickly and efficiently. The advantage of flocking over other clustering methods as k-means is its adaptability. It can work perfectly when a new element is inserted or removed from a cluster.

[77] defined three essential rules for flocking implementation: (a) to avoid collisions among other members of the group, (b) to remain at an appropriated distance from other members and (c) to move with a similar speed. Nevertheless, it has some shortcomings, as there are cases when a unique cluster is formed. [55] incorporates two new rules: (a) to maintain similar elements together and (b) to maintain dissimilar elements separated. These rules ensure more defined clusters and as a consequence, a more precise form to identify ADLs that are relevant for environmental adaptation.

3.2.3 SPEED

[10] created the algorithm called Sequence Prediction via Enhanced Episode Discovery (SPEED) to predict the following actions of smart home users.

Its creators defined the concept of an episode, which is a set of sequential events that the user periodically executes. SPEED is oriented to work with binary events, to identify if a device was turned off or on. The recorded episodes are processed and accommodated in a Markov model of finite order, which will be used to calculate the next action. The Markov model consists of generating a set of decision trees that are formed from the human activity data. This data is represented by letters: uppercase letters representing active status and lowercase letters inactive status. In Table 3.3, the states of the devices are represented, for example for device 2, A is turned on and a is turned off. An episode would be a string such as $ABCDbcda$ that starts with the off state and ends with the on the state of a device. As events are processed, longer episodes are recorded. However, the idea is to prune unnecessary branches to achieve greater efficiency.

Device	**1**		**2**		**3**		**4**	
Event	ON	OFF	ON	OFF	ON	OFF	ON	OFF
	A	a	B	b	C	c	D	d

Table 3.3: Home devices and their states

Once the tree is built, the partial match prediction (PPM) method is used to predict the next activity. This method will choose the event most likely to be in the tree. The formula for calculating the probability of each event is shown in equation 3.1, where ϕ is the symbol to predict, $P_k(\phi)$ is the final probability of symbol ϕ, k is the length of the phase, $p_k(\phi)$ is the probability of seeing ϕ after the length phase and $e_k(\phi)$ is the escape probability of symbol ϕ.

$$P_k(\phi) = p_k(\phi) + e_k(\phi).P_{k-1}(\phi) \tag{3.1}$$

3.2.4 Nash H-Learning

Nash H-Learning was a proposal for environments that aimed to improve user comfort and reduce energy consumption. However, it is about improving the environment with multiple users. The problem they tried to solve was challenging due to the fact that the relevant contexts of multiple inhabitants in the same environment are often inherently correlated and inter independent with each other. [85] show that this is an NP-hard problem, however, they obtained interesting results.

The authors have two preliminary works. In the first, they designed an optimal algorithm for tracking the location of users in an internal environment. That work was based on the management of the dictionary and learning along the lines of the mobility profile of an inhabitant [83]. In the second work, they proposed resource management techniques for the usual users, as well as a cooperative entropy learning policy. They analyzed the results of the uncertainty of several users and their most likely routes [84]. Nevertheless, the complexity of the problem was not characterized as they do in their next work, which is Nash H-Learning.

Nash H-Learning has its bases in Nash equilibrium theory to minimize the uncertainty of the joint location, but also not to sacrifice the preferences of a user in relation to the others.

3.2.5 Apriori Algorithm

The Apriori Algorithm is a well-known algorithm in data mining. Initially, it was used to detect recurrent patterns in database transactions. The output of this algorithm is the association rules of form $X \rightarrow Y$, where X and Y are subsets of the set of transactions. The rule indicates that the occurrence of the elements of X implies the occurrence of the elements in Y. The algorithm has two steps. First, the elements grouped, and the most frequent elements are identified. Then, the elements are pruned to generate the association rules [95].

In the first step, the elements are grouped together in an agglomerative way. This means that at the beginning each element represents a unit set and merges with another depending on whether it satisfies the predetermined support threshold. In the second step, the association rules are created, which must meet a predetermined confidence threshold. Confidence is defined as the percentage of transactions that contain X but also Y.

[108] adapted the Apriori Algorithm and proposed optimizations to be used in intelligent environments. They found that they had to access the database many times, which made it slow, instead, they proposed to load data into memory. The second idea is based on the fact that a set of elements cannot be obtained by adding new elements to the set of non-frequent items. This reduces the number of interactions that can be supported to calibrate a set of elements. In their work, they observed a great improvement in relation to the conventional Apriori algorithm.

[76] reported that the Apriori algorithm had difficulties in identifying sets of periodic elements. For that reason, they proposed a new model called Frequent and Periodic Activity Miner (FPAM), which defines granules of one-hour and one-day to recognize periodic activities. In this way the classification is done in a more intelligent way, identifying the frequency and periodicity of activities.

Algorithm	**Data Structure**	**Model**	**Data**	**Category**
Active LeZi	Tree	Markov	Location and Time	LeZi
Flocking	Tree	Markov	Event Sequence	Episode Discovery
SPEED	Cluster	Rules	ADL	Clustering
Nash H-Learning	Matrix	Rules	Temporal Relation	IA
Apriori Algorithm	Users	Markov	Location	Q-Learning

Table 3.4: Prediction Algorithms

3.3 Discussion

Our work has a scientific contribution to show a new way of predicting user actions under certain environmental circumstances using their feedback (evaluations) and a list of rules of action. But also, it explores the possibility of obtaining a patent, since it takes advantage of the flexibility of ManIoT (base platform) to increase the components involved in the intelligent automation of an internal environment such as a house or an office. Finally, our interdisciplinary work team has implemented hardware components such as sensory boards (using arduino, raspberry and generic sensors) including a smart switch prototype, which opens up the possibilities of producing our own product.

3.4 Conclusion

This chapter showed a series of products and patents similar to or related to SmartOffice. Likewise, we reviewed the state of the art of prediction algorithms for intelligent environments. It was observed that most of the prediction algorithms are based on the user's actions and use Markov chains. Finally, we present a breaf discussion about our contribution which takes advantage over the current existing patents, products and prediction algorithms for automation systems.

Chapter 4

Indoor Automation System

This chapter introduces our indoor automation system, which is required to apply our proposal. Section 4.1 presents the system functionality and the requirements to achieve the proposal. Section 4.2 presents a general overview of the components related to ManIoT and our automation system. Section 4.3 details all the hardware components (gateway, sensors and actuators), how we used them and some other considerations. Section 4.4 shows the software components, which include the interface used for user interaction, the database model and software requirements. Lastly, Section 4.5 concludes the chapter.

4.1 Proposed functionality and requirements

In this section, we present, through a story, how our system should work and the requirements it must accomplish.

SmartOffice: The new toy of Victor

"It is the first day that Victor comes to work, he finds the office very dark and very hot, sensations that he reports through the SmartOffice mobile application, then he opens the blind shades and turns on the air conditioning. During his working day, Victor decreases the temperature and answers the questions about his satisfaction on several occasions. Every time Victor leaves the office, the system closes the blind shades, and turns off the light and air conditioning unit. At nightfall, Victor closes the blind shades and turns on the light; but he struggles with the air conditioning, because if it is turned on the room gets cold very fast, however if it is turned off, the room becomes hot."

As the days go by...

"SmartOffice learns about Victor's preferences and the situation changes. Every morning, when Victor enters his office, the blind shades open and the air conditioner turns on. Then, Victor expresses his satisfaction using the SmartOffice questionnaire. After a time, when an increase in temperature is detected, the cooling of the environment is intensified. At nightfall, the blind shades close, the light turns on and the air conditioner

regulates the temperature constantly. The number of times Victor requested changes in the office decreased and his satisfaction scores increase."

As this story shows, the system requires an autonomous agent that manages the actions of an automation system to adapt them to the preferences of the user.

About scenarios, actuators and mobile application

To achieve our proposal, we define the *improvement evaluation scenarios* (also called *scenarios*), which are situations where the system can intervene to improve the user experience. Therefore, each *scenario* must be linked to some actuators that allow its improvement. According to the concepts studied in section 2.4, there are environmental factors that affect the productivity of workers and the most influential are: *luminosity*, *temperature*, and *noise*. The latter is very relevant because high levels of noise are the main cause of distraction in tasks that require mental concentration. Nevertheless, there is no actuator that regulates noise. Hence, in this work, we consider scenarios for *temperature* and *luminosity*.

In order to automate the office, it is necessary to convert each of these actuators into smart actuators. This means that they must be connected to a logical network to be able to communicate with other devices, exchange information and execute actions automatically. In addition, as shown in Victor's story, we need a mobile application to request actions from the actuators and to evaluate the system (inform the satisfaction of the user with the system's behavior).

About sensors

To make decisions, the autonomous agent needs information about the environment. The most magnitudes we track, the more complex and rich the learning is. For that reason, we require sensors related to our *scenarios*, hence, the main sensors are for temperature and luminosity. Nevertheless, we can consider other magnitudes.

Other requirements

Furthermore, we need (a) a mechanism to identify when the user is present and when he is absent; (b) a database to store the data of sensors, actions, evaluations and configurations of the system; and (c) a IoT gateway to allow the communication between devices (actuators and sensors), database and clients (mobile application and autonomous agent). The complete list of requirements is shown in Table 4.1.

Requirement	**Functionality**
Smart ACU	Regulate air cooling. Includes ACU device and controller, which should execute actions remotely
Smart Shades	regulate entrance of natural light. Includes traditional shades which are manipulated through a motor, which should be controlled remotely.
Smart Switch	Turn on/off the luminaries. Includes a switch and a remote control mechanism.
External luminosity sensor	Tracks luminosity. It should be located outside the room near to the window.
Internal luminosity sensor	Tracks luminosity. It should be located in the middle of the room.
Air quality sensor	Tracks different gases concentration inside the room.
Humidity sensor	Tracks relative humidity inside the room.
Noise sensor	Tracks noise level inside the room.
Pressure sensor	Tracks pressure inside the room.
Window status sensor	Tracks the status (open/closed) of the window.
Door status sensor	Tracks the status (open/closed) of the door.
Presence sensor	Tracks the user presence status.
Presence detection mechanism	Identifies the user presence.
Mobile device	Sends action requests and evaluates the system.
Mobile application	Sends action requests and evaluates the system.

Table 4.1: SmartOffice requirements

4.2 General overview of the implementation

We adapted ManIoT, an IoT device management platform, to our office automation application[1]. Our system preserves some important intrinsic ManIoT properties such as *modularity* and *device heterogeneity*. These characteristics ensure that sensors, actuators, and other products found in the market can be added to the platform. Figure 4.1 shows our adaptation.

[1]We implemented the prototype the room 6315 of the Computer Science Department (DCC) in the Federal University of Minas Gerais (UFMG), Belo Horizonte, Brazil.

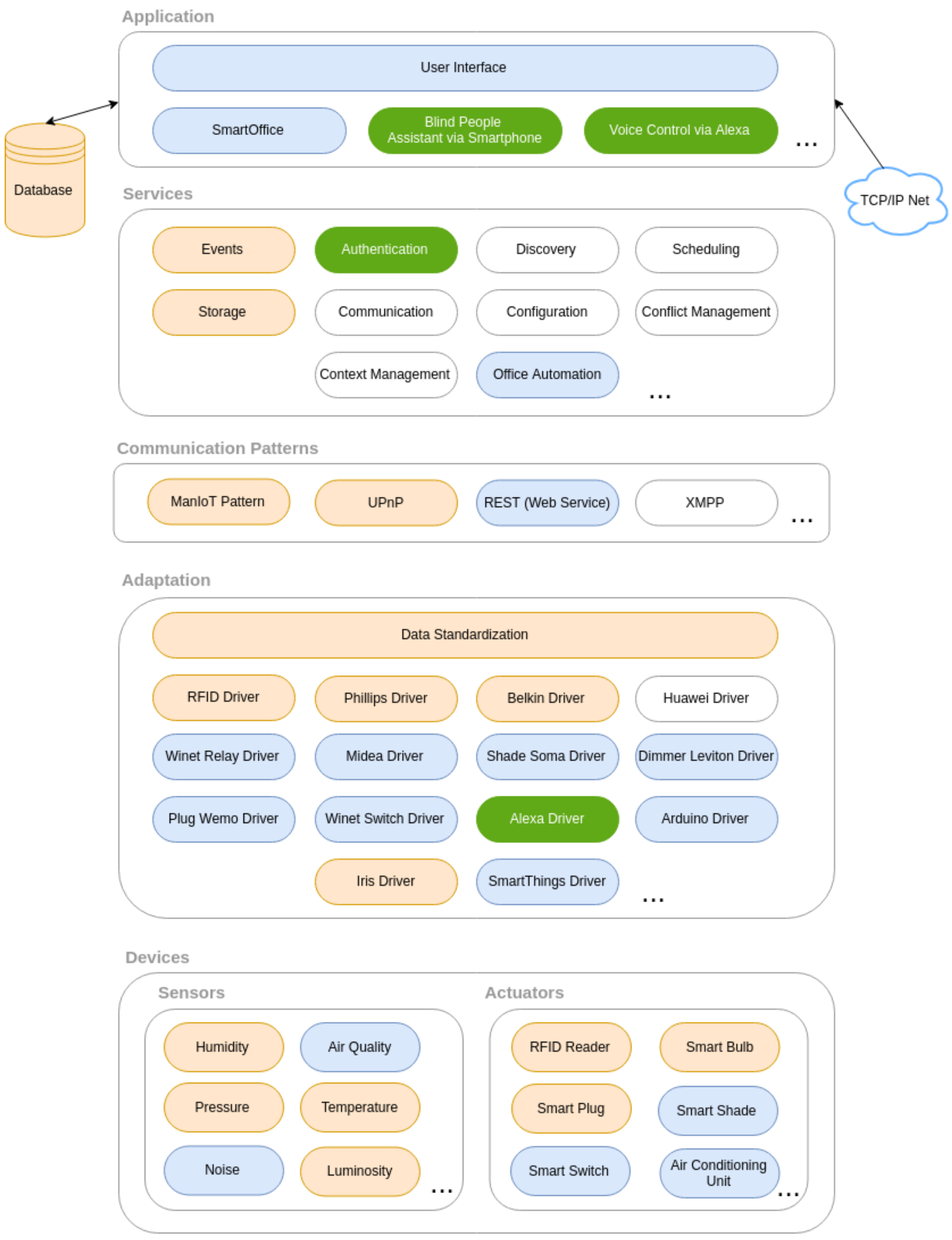


Figure 4.1: SmartOffice components. Orange means developed before our work; green, parallel work based on ManIoT; blue, developed for SmartOffice and white, future work.

To get started, the components colored orange in the Figure were implemented before our proposal. Components that were already available include the basic components for testing its model of local and global management, basic sensing, data standardization, implementation of drivers for some actuators, storage, and communication.

Blue components are those developed for this work. The most relevant ones are: a) the specification and implementation (hardware and software) of our own *smart switch*; b) the implementation of drivers for the smart shade, smart switch and air conditioning unit; c) the implementation of the web service REST, which allows the communication between any client (cellphone, Alexa, etc) and ManIoT; and d) the *office automation application* that includes the prediction algorithm which provides intelligent automation; and e) the creation of an Android application, called SmartOffice, to facilitate the user interaction.

In addition, we highlight some green components that were implemented in parallel (final thesis of undergraduate and graduate students): *Blind People Assistant via Smartphone* [33], *Voice Control via Alexa* [34], and *Authentication Manager* [24]. Finally, the white components are still unimplemented and left for future works. The complete list of collaborators is shown in Table A.1 of appendix A.

4.3 Hardware components

We need many hardware components to accomplish our sensing, actuation, data processing, and machine learning requirements. It includes an IoT gateway, sensors, actuators, micro-controller boards, and hubs. All these elements are detailed in subsections 4.3.1, 4.3.2 and 4.3.3.

4.3.1 IoT gateway

Our prototype was implemented in a virtual machine running on a MAC desktop with processor i3-4160, 3.60GHz and 16GB RAM, with operating system Linux Ubuntu 16.04. The VM runs over Oracle VM VirtualBox version 5.2.8. The virtual machine has 2GB of memory and 15GB of disk space. The gateway also has installed MySQL Distrib 10.1.25-MariaDB, Java Openjdk version 1.8, Python 3.5 and Scikit-learn packages for running XGBoost.

The gateway is physically connected via serial port to an *arduino module*, an *iris module* and a *smartThings module*. In addition, a *bluetooth dongle* is also connected to establish the communication with the *smart shades*, a wifi communication to a *smart switch*. On the other hand, the control of the air conditioning unit is through HTTP requests. The detail of all these modules are explained in detail in Section 4.3.2, 4.3.3.

4.3.2 Sensors

As we want to demonstrate the system flexibility, we use different modules for sensing. Hence, we distributed all the sensing tasks in three different modules: Arduino, Iris and SmartThings modules. Most of the sensors are implemented using Arduino because of its flexibility and efficiency. The Iris module is used just for tracking the room luminosity because the MDA100CB default board has a luminosity sensor. Furthermore, the SmartThings module provides information about the windows and door status, and it also tracks the user presence. All these modules are used to acquire the required sensor values used in our prediction model. The completed list of sensors is presented in Table 4.2.

The *arduino module* is composed by one Arduino Mega board, shown in Figure 4.2e, and four sensors: HIH-4030, MQ-135, BMP-085 and MAX-9812. The HIH-4030, shown in Figure 4.2a, measures humidity [51]. MQ-135, shown in Figure 4.2b, returns a value that represents the air quality, which in reality means the concentration of gases such as NH_3, alcohol, benzene, smoke, CO_2 among others [39]. BMP-085, shown in Figure 4.2c, measures barometric pressure and temperature. Because pressure changes with altitude, this sensor could also be used as an altimeter [20]. MAX-9812, shown in Figure 4.2d, is a microphone used for measuring the noise level [73].

The *iris module* is used for tracking the room luminosity in different positions. In the mounted room we have one iris mote in the center of the room to measure inside luminosity and another one in the window to measure outside luminosity. The module is composed by the base station MIB520, sensor board MDA100CB and an iris mote. The MIB520 Mote Interface Board, shown in Figure 4.3c, sends the sensor network data onto a computer platform via a serial/USB interface. MEMSIC also offers a stand-alone gateway solution, the MIB600 for TCP/IP-based Ethernet networks. The MDA100CB board, shown in Figure 4.3b, is a sensor and data acquisition board which has a precision thermistor, a light sensor/photocell and general prototyping area. It is designed for use with the IRIS, MICAz and MICA2. The iris mote, shown in Figure 4.3a, is used for sending and receiving data between the sensor boards and the base station.

The *smartThings module* helps to incorporate easily new off-the-shelf sensors compatible with SmartThings. In our case, we just use the multipurpose sensor, shown in Figure 4.4b, which is a magnetic sensor used to sense if doors/windows are opened or closed [91]. This module also includes a SmartThings Hub, shown in Figure 4.4c, which connects the SmartThings sensors to a computer network, and it also offers a mobile interface for the user [92].

Our platform has an independent module for user presence detection, which helps to know if there is someone inside the room and who is that user. This is essential to request the specific user preferences and to trigger the calculated predictions. The same infrastructure was shared by two users which for future references we will call *user1* and *user2*. This helps us to generate more interesting results because we can demonstrate how different users have different preferences and the how the system predictions are adapted to each one. To recognize the two different users and configure the room properly, we assigned one *user identification device* to each user.

Actually, our system can recognize the users through two identification methods: with a *smart cellphone* and with a *smart keychain*. The first one is based on the TCP/IP protocol, basically the system is constantly verifying if the cellphone is connected to the SmartOffice network. The details of the operation of the presence module will be see in section 4.4.2. The second one, shown in Figure 4.4a, uses the *smartthings platform* and its *keychain arrival sensor*, which uses the ZigBee protocol to communicate with the *smartthings hub* [90].

Sensor Name	**Physical quantity**	**Measurement Unit**	**Range of Values**
MDA100CB	Int. luminosity	lux	0-20000
MDA100CB	Ext. luminosity	lux	0-200000
MQ-135	Air quality	ppm	0-55000
HIH-4030	Relative Humidity	RH%	0-100
MAX-9812	Noise	decibel (dB)	0-180
BMP-085	Pressure	hectopascal (hPa)	300-1100
BMP-085	Temperature	celsius	15-29
Multipurpose Sensor	Window status	categorical	0,1
Multipurpose Sensor	Door status	categorical	0,1
Nexus 5[2]	Presence	categorical	0,1

Table 4.2: Description of sensors used in our automation system.

4.3.3 Actuators

In this section, we give technical details about the air conditioning unit, smart shade, and smart switch in sections 4.3.3.1, 4.3.3.2 and 4.3.3.3, respectively. At the end, in Table 4.3, we show a summary of the actuators.

4.3.3.1 Air conditioning unit

The chosen room had installed a Floor Ceiling Type R407C air conditioning unit (ACU), shown in Figure 4.5a. This model is manufactured by the Midea Group. This ACU is not a completed HVAC (Heating, ventilation, and air conditioning) system, it just conditions the air by freezing it.

The R407C offers several options to the user. It allows to turn on/off the device, to change the temperature from 17 to 26 Celsius degrees, to change the fan to three different speeds: *low*, *medium*, *high*. Choosing any of these options indicates a constant speed. Additionally, it has a fourth speed option *auto* that maintains the other configurations activated until the required temperature is reached. This device also allows to activate or to deactivate the *swing*, which is an oscillating movement of the fan blades [62]. As was mentioned previously, this device does not offer the option of heating, thus the unique alternative for warming the environment is limited to turn the ACU off.

For controlling this device, the user can use different methods. The two most conventional are by using the panel, which is physically connected to the unit; and by using the remote control using infrared. A more sophisticated method is by a web application[3]. Unlike the previous two methods, this one allows to query the device status, i.e. any client can ask the status of a unit through a web service.

We developed a driver to manage the ACU. It was implemented in java and it interacts with the Midea web application at the DCC web server. Our driver requests the information and sends the execution orders via HTTP requests. The problem with this solution is the dependence of the DCC Midea server and internet connection. For that reason we plan to develop a generic driver that communicates via the infrared technology.

[3]Developed by the Centro de Recursos Computacionais (CRC) of the Computer Science Department of Federal University of Minas Gerais. The application is available in https://painel.dcc.ufmg.br/midea

4.3.3.2 Smart shades

Shades are an important tool for providing different light intensity in a room, and nowadays they can become smart by connecting them to a motor. This motor fulfills the function of moving the shade rope, replacing a manual human intervention.

In a manual shade, one rope is used for moving the shade as a curtain, from both sides to the centre or vice-versa; the other rope is used for moving every slat over its axis. As part of this work, we mounted a smart engine over the existing manual shade inside the room. We use the Soma Smart Shade [93], shown in Figure 4.5b, to move the slat rope. We chose this rope because it provides homogeneous luminosity changes, a characteristic that is not given by opening the shade from center to the sides. Soma Smart Shade moves the slat in a range of 0 to 100: when the status is 0, it means that the slat is totally opened (90 degrees in relation to the window); when the status is 100, it means that the slat is completely closed (180 degrees in relation to the window).

For using the Soma Smart Shade it is necessary to pair the smart engine with a smartphone which should have installed its mobile application. The communication is established via Bluetooth Low Energy (BLE), that is why the smartphone should have the Bluetooth activated. As we needed our central processing unit to have a direct connection to the device, we use a Bluetooth dongle, so we can send execution orders and send queries to know the shade status.

We also develop a driver to control the shades. In this case, we establish four well-defined states: Opened (value=0), More opened than closed (value=33), More closed than opened (value=66) and Closed (value=99). We do that for two reasons: a) when there is an small difference i.e. five, the device does not perform the action; and b) in empirical tests we identified that small changes in the angle of the slat do not generate a significant effect.

4.3.3.3 Smart switch

Similar to the *smart shades*, the *smart switch* is essential for implementing our proposal. Four different solutions were tested to decide which one to use.

The first option was to use the Philips Hue bulbs, which already had a driver implemented in ManIoT and have the ability to change color as well as adjust its luminosity. It also can be controlled remotely via a smartphone over WiFi. The problem with these bulbs was that their luminous flux was not enough for our demands. Each bulb can generate just 800 lumens, while a luminary irradiates 3800 lumens. Considering that the room has three luminaries, we would need 15 hue bulbs to reach the same luminosity. This solution was very expensive and limited, for that reason we discarded it.

The second option was to use the Wemo Plug, which also had a driver in ManIoT. We built a serial circuit with two conglomerations of three fluorescent bulbs. Although the luminosity was better than the Philip Hue bulb, it didn't reach the expected luminosity. For a time we used this solution until we acquired a Leviton DZ6HD Dimmer [57].

The Leviton Dimmer is a Z-Wave Plus enabled universal dimmer designed for residential lighting applications and controls the luminaries with a mobile application. This was an interesting solution until the luminaries reactor broke down and was replaced with a new one. After this point, the dimmer started to behave strangely: the reaction time increased and many times it was unable to turn the lights on. After analyzing the reason of its behaviour, we concluded that power draw of the new reactor was higher than the older and as it is a dimmer it needed too much effort to accomplish the actions.

The fourth and final solution was to implement our own smart switch named *Switch Winet*, it is shown in Figure 4.5c. It is composed by a relay *Keyes_SR1y*, 5v model and a NodeMCU ESP32 microcontroller, which is an open source IoT platform. NodeMCU has a WiFi interface for communication and is capable to send and receive data via the HTTP protocol. It also controls the relay through a logic port allowing to turn on/off the luminaries.

Scenario	**Actuator Name**	**Device**	**Available Options**
Temperature	Air Conditioning Unit	Floor Ceiling Type R407C	Mixture of: a) Turn on/off device, b) change temperature, c) change speed (auto, low, medium, high), d) turn on/off swing
Luminosity	Smart Shades	Soma Smart Shade	Four positions. a) Opened, b) Most opened than closed, c) Most closed than opened, d) Closed
Luminosity	Smart Switch	Switch Winet	Turn on/off the light

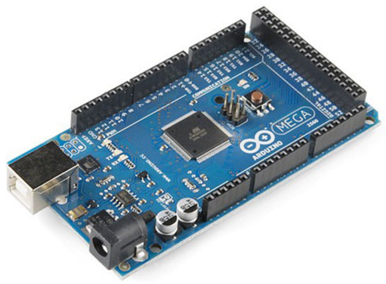
Table 4.3: Actuator list

(a) HIH-4030. Humidity sensor.

(b) MQ-135. Air quality sensor.

(c) BMP-085. Temperature sensor

(d) MAX-9812. Noise sensor.



(e) Arduino Mega. Microcontroller board

Figure 4.2: Arduino module components

4.4 Software Components

As we mentioned in the previous section, in this work were developed some modules for sensing. The purpose of these modules is to standardize the method of communication between the different sensor platforms and the central processing unit. Hence, the *arduino module* and the *iris module* periodically get the data from the boards via serial port and store it into the database. The *smartthings module* also reads data regularly, but it queries the information through the smartthings platform[4].

The data model, presence module and mobile application need more discussion, for this reason we will describe them in separate subsections 4.4.1, 4.4.2 and 4.4.3. Finally,

[4]The module accesses the site https://www.smartthings.com/ where the Smartthings devices are previously registered and configured.

(a) Iris mote. Module used in wireless sensor networks for transmission of data.

(b) MDA100CB. Sensor and data acquisition board with a general prototyping area.

(c) MIB520. Base Station for Wireless Sensor Networks.

Figure 4.3: Iris module components.

the software is also composed by the intelligent automation model of Smart Office. This part will be described in chapter 5 in more details.

4.4.1 Data model

We defined the data model for the implementation of the SmartOffice platform. The data model helps to specify the data in sufficient detail to store and/or transmit the information [71]. Hence, the data model was based on the ManIoT information model, resulting in tables created in the MySQL database management system.

Figure 4.6 shows part of the relation-entity model for SmartOffice. As was mentioned, this is based on ManIoT but has some adaptations. The two more important ones are management of a) presence and b) learning data. We create a table `m_presence_sensor` to represent the relation between a user and a sensor that can be used for presence recognition. On the other hand, we store the learning data in two tables `m_action` and `m_evaluation`, both with data prepared to be the input in the machine

(a) SmartThings Arrival Sensor in format of Keychain used for presence recognition

(b) SmartThings Multipurpose Sensor used for identifying open/close status of doors or windows.

(c) SmartThings Hub used for controlling other SmartThings devices.

Figure 4.4: SmartThings module components.

learning model which is detailed in chapter 5.

4.4.2 Presence module

Another challenge was to deal with the presence identification. We executed several tests to define which should be the most appropriate method to detect the user presence. These tests included RFID, Bluetooth, WiFi and Smart Keychain technologies.

RFID was the first technology that we used. It was considered because of its popularity for recognizing people. This is because it is easy to use and it is low cost. Our hypothesis was that the user must carry an RFID tag in the format of an identification card or a keychain. Inside the room, there is an RFID reader and one or two RFID antennas. The user should not pass the tag over the reader as in a typical RFID personal identification system. Hence, our first attempt was to use an antenna, and the system should identify his presence automatically, the same when he leaves the room. Our tests

(a) Midea Air Conditioning Unit

(b) Soma Shade

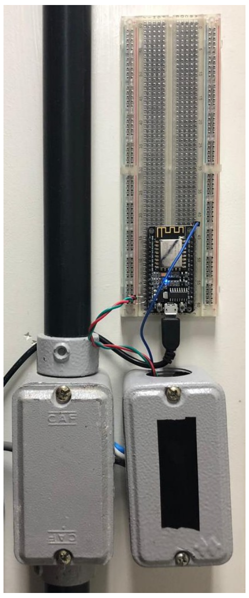

(c) Switch Winet

Figure 4.5: Actuators used in our automation system.

demonstrated that this model does not work, many times the user passed very fast next to the antenna and it simply did not scan the tag. In a second attempt to use RFID, we tried to add another antenna to complement the existing one, and the results were similar. The third attempt was to put the antenna near to the user seat, so that it should scan constantly the tag, but the reading was very unstable. All these tests helped us to conclude that RFID was not the best technology to fulfill our requirements.

Bluetooth was an interesting solution because everyone has a cellphone and most of these devices have a Bluetooth interface for data transmission. The idea was to use a Bluetooth dongle to scan the user cellphone. The problem with this solution was that the

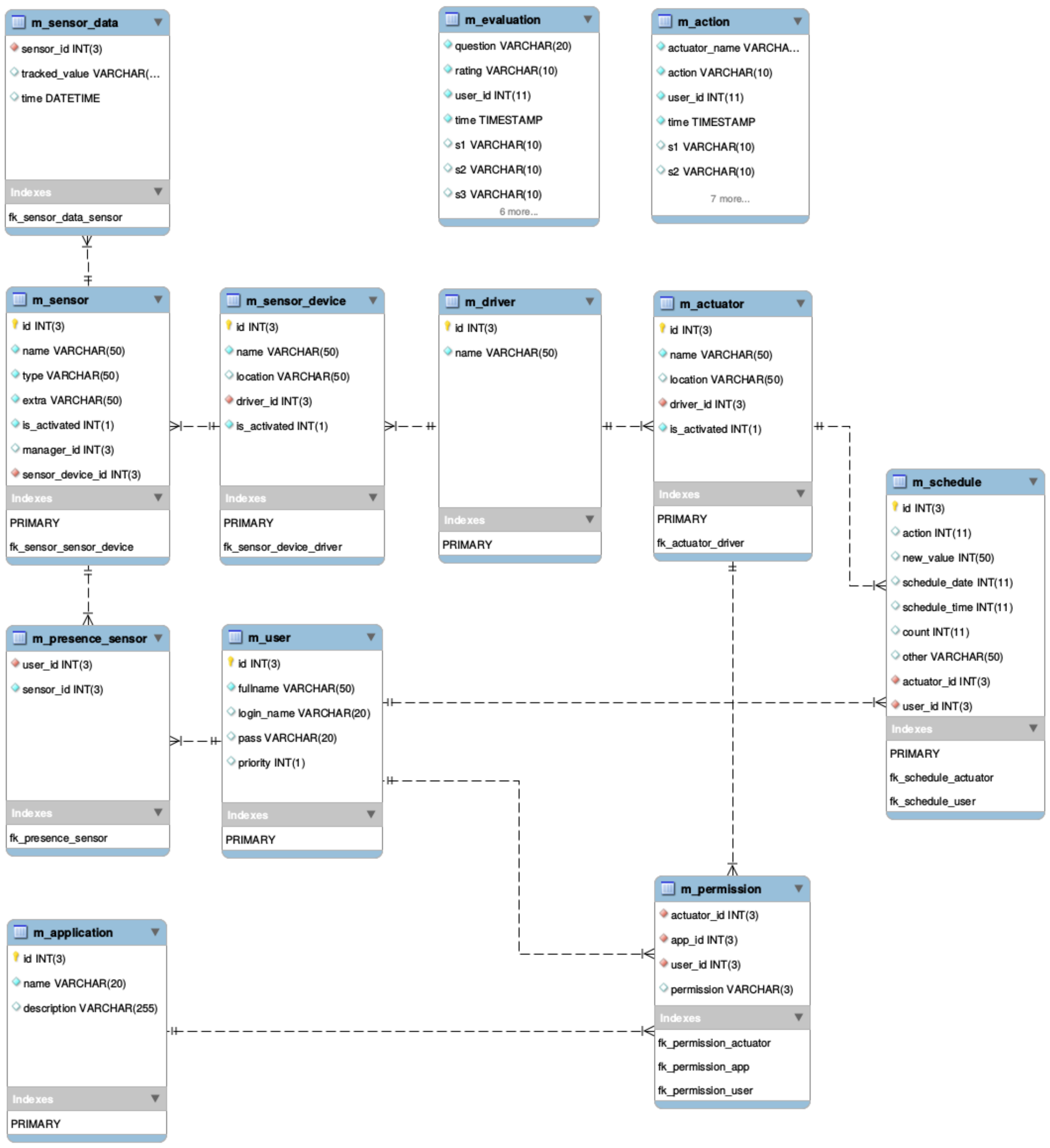


Figure 4.6: SmartOffice database model

cellphones should have the Bluetooth screen opened during the connection time period, otherwise the Bluetooth is disabled.

WiFi gave us a temporary and stable solution. The user presence can be determined by verifying if the user cellphone is connected to the local network. Similar to the Bluetooth solution, the cellphone should have the WiFi activated. We have two problems with this, at first, the most sophisticated cellphones as the latest iPhones have very drastic security and energy-saving policies, such that WiFi is not connected all the time. Second, there are moments (seconds) in which the WiFi signal is lost. Despite these difficulties, we selected WiFi as our user identification method. We use a Nexus 5 that was flexible to configure the WiFi connection as permanent and we defined an algorithm to deal with the connection drops. The algorithm is shown in Algorithm 1. It shows the use of four presence status values: *long_time_outside*, *recently inside*, *long_ time_inside*, *recently_outside*. The definition of these values is useful for the *orchestrator*, which will be explained in section 5.

At the beginning, the system assumes that the user has a long time outside the room. Then it verifies constantly the existence of a connection between the cellphone and the network. Each time a connection is detected it is verified if the number of total successful connections is equal to a pre-defined *presence_window* that represents the maximum number of connections required to ensure that the user is present. If the condition is true, then the status is changed to *recently inside*. In the next queries is defined a status *long_time_inside* automatically, until an absence of the user is detected (*recently outside*). This status is achieved if the number of failed connections surpasses the *absence_window*. Similar to *long_time_inside*, it confirms the user absence after an exhaustive verification. In this case, the status *long_time_outside* is established after *recently_outside* and it will remain in that status until a new presence is detected.

For tuning the algorithm, it is necessary to define the value of two parameters: *presence_windows* and *completed_absence_windows*. The first one represents the presence verification level, the number represents the number of verifications. The second one is the absence verification level. In our experiments, we conclude that it is better to have more absence verification than presence verification because we can ensure a stable performance of the system when the user is inside the room.

Algorithm 1 Define WiFi Presence Algorithm

```
1:  procedure DEFINE PRESENCE
    Input: User phone IP; Last status
    Output: User presence Status
2:      is_connected = try_wifi_connection(IP)
3:      last_connections = shift_right_connections(is_connected)
4:      if $last\_status == LONG\_TIME\_OUTSIDE$ then
5:          if $count\_connections(last\_connections) == presence\_window$ then
6:              $status = RECENTLY\_INSIDE$
7:      else if $last\_status == RECENTLY\_INSIDE$ then
8:          $status = LONG\_TIME\_INSIDE$
9:      else if $last\_status == LONG\_TIME\_INSIDE$ then
10:         if $count\_connections(last\_connections) == 0$ then
11:             $all\_falses\_counter = all\_falses\_counter + 1$
12:             if $all\_falses\_counter > completed\_absence\_windows$ then
13:                 $status = RECENTLY\_OUTSIDE$
14:     else if $last\_status == RECENTLY\_OUTSIDE$ then
15:         $status = LONG\_TIME\_OUTSIDE$
```

4.4.3 Mobile application

As most popular home systems, SmartOffice has an interface that enables the user to interact with the system by configuring options, review sensor status, evaluate feedback and request new actuations. We developed this interface in Android Marshmallow, which is compatible with most mobile devices (phones and tablets). Below we detail each of the screens.

The first screen, shown in Figure 4.7b, lists all the actuators options. For the use cases that we consider in this work, the system was configured to recognize three actuators: air conditioning unit, light and shades. Hence, the user can turn on/off the air conditioning, change the temperature, change the fan speed (auto, low, medium, high) and activate/deactivate the swing. It also can turn on/off the room luminaries and regulate the sunshade to obtain different levels of luminosity.

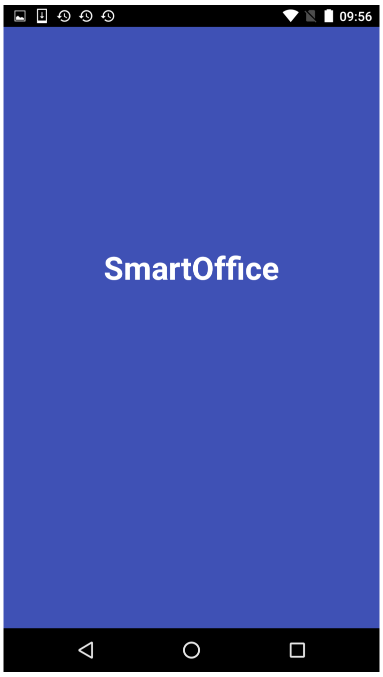
In the second screen, shown in Figure 4.7c, the user can verify the sensors status. We can find information of the temperature, pressure, humidity, noise, air quality, internal luminosity, external luminosity, door status and window status. We call internal luminosity to the tracked value in the center of the room and external luminosity to the tracked value located on the window. Both values are very important because with them we can infer for example if the sunshade is open or closed.

The most relevant screen for our proposal is the third one. It allows the user to evaluate the different scenarios. The first one can be evaluated through the question *how do you feel about the room temperature?* and has seven possible options for temperature comfort. The screen used for its evaluation is shown in Figure 4.7d. Similarly, the second one is evaluated with the question *how do you feel about the room luminosity?* with seven options as well. This screen is shown in Figure 4.7e.

Finally, we created a settings screen, shown in Figure 4.7f. Here, the user can set some system configurations such as change the server IP or the type of presence device to be used.

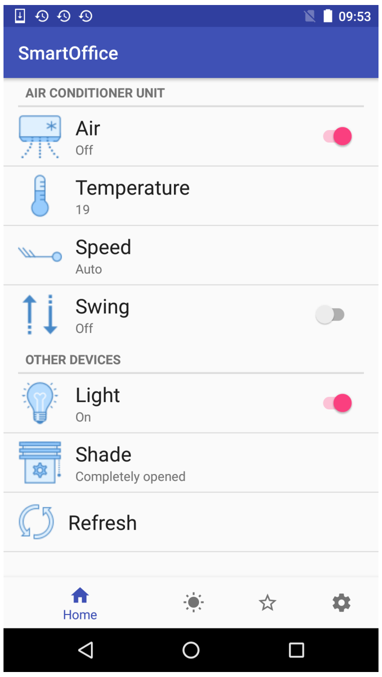
4.5 Conclusion

This chapter described our system architecture, identifying which components were already implemented, which ones we implemented as well as those that are being implemented by others in the WINET laboratory. In addition, we present the hardware and


09:56

SmartOffice

(a) Opening Screen


09:53

SmartOffice

AIR CONDITIONER UNIT

Air
Off

Temperature
19

Speed
Auto

Swing
Off

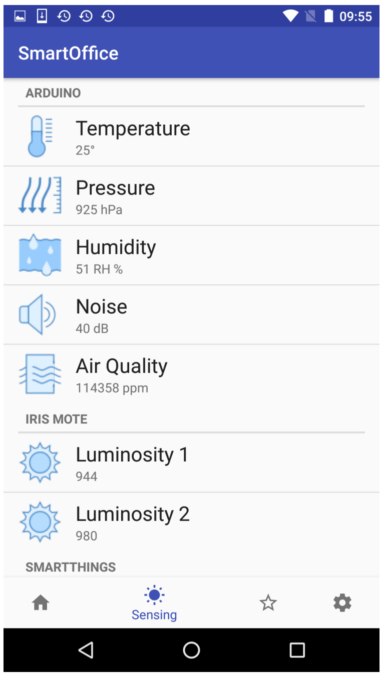
OTHER DEVICES

Light
On

Shade
Completely opened

Refresh

(b) Actuators Screen


09:55

SmartOffice

ARDUINO

Temperature
25°

Pressure
925 hPa

Humidity
51 RH %

Noise
40 dB

Air Quality
114358 ppm

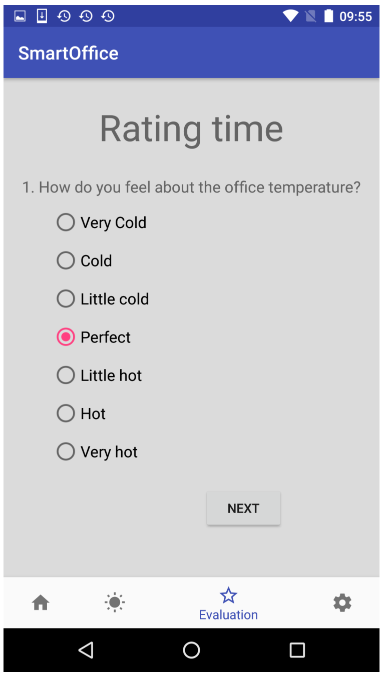
IRIS MOTE

Luminosity 1
944

Luminosity 2
980

SMARTTHINGS

(c) Sensors screen


09:55

SmartOffice

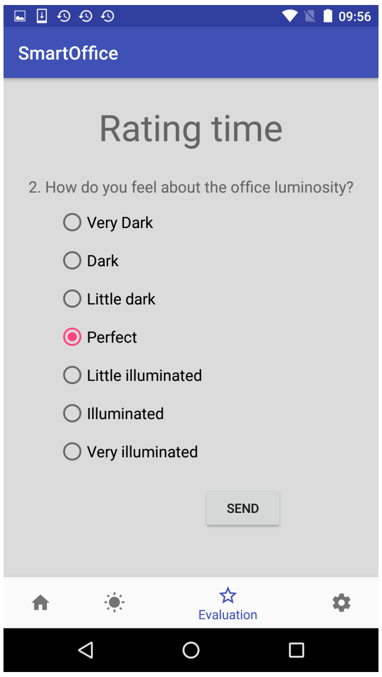
Rating time

1. How do you feel about the office temperature?

- Very Cold
- Cold
- Little cold
- [✓ Perfect]
- Little hot
- Hot
- Very hot

NEXT

(d) Rating temperature screen


09:56

SmartOffice

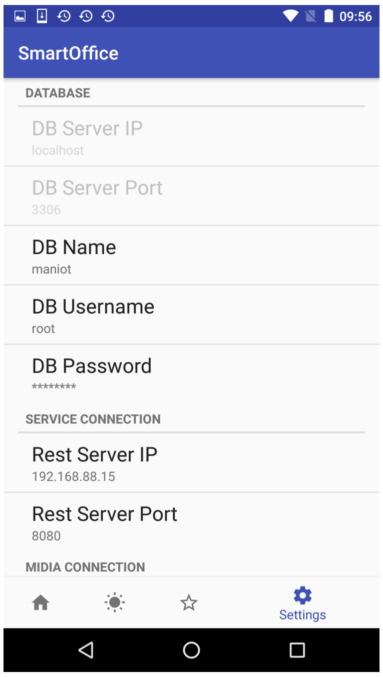
Rating time

2. How do you feel about the office luminosity?

- Very Dark
- Dark
- Little dark
- [✓ Perfect]
- Little illuminated
- Illuminated
- Very illuminated

SEND

(e) Rating luminosity screen


09:56

SmartOffice

DATABASE

DB Server IP
localhost

DB Server Port
3306

DB Name
maniot

DB Username
root

DB Password
****

SERVICE CONNECTION

Rest Server IP
192.168.88.15

Rest Server Port
8080

MIDIA CONNECTION

(f) Settings screen

Figure 4.7: SmartOffice phone application screens

software components used in smartoffice. Finally, we describe our data model, showing the entity-relation model and some extra adaptations of the ManIoT for accomplish the SmartOffice objectives. A front photography of the room in which the system is installed is shown in Figure 4.8. The next chapter presents the last part of the system, which is the software that measures comfort and acts based on the evaluated comfort.

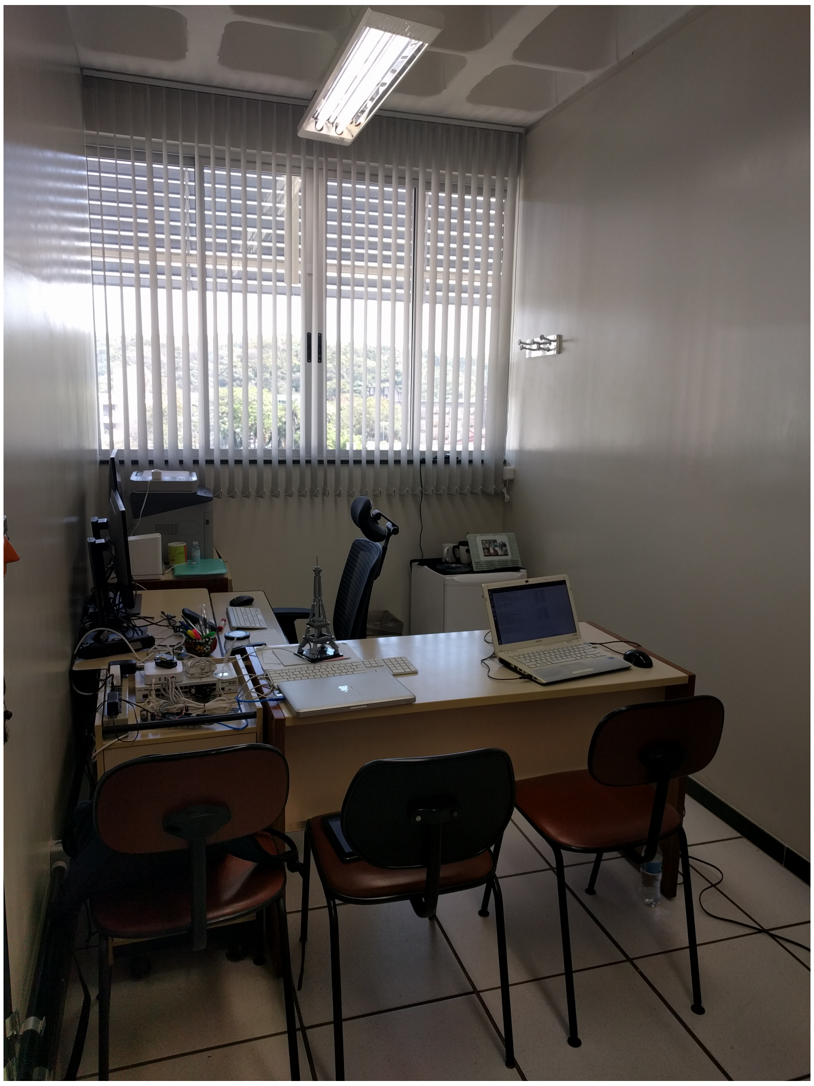


Figure 4.8: Front photography of the office used for implement the office automation system

Chapter 5

Intelligent Automation Model

One of the most interesting challenges of indoor environment automation is to customize the actuation automatically, in order to increase the user comfort. In fact, there is no general pattern of preferences, since people have different inclinations, for example some people like hot environments with more or less luminosity. In Section 6.2, we demonstrate this hypothesis is true with the support of a tool for the analysis of learning models. Today, we find some solutions based on pre-configuration or preferences selected by the user, however, there are no solutions based on the user's own subjective preferences.

In this chapter, we introduce our proposal, which is an intelligent automation model that utilizes user evaluations to make decisions and readjust the actuations. This implementation required: (a) an automation system and (b) a smart agent for the decision-making. The developed automation system, presented in Chapter 4, allows sensing, manipulation of the different actuators and data collection. On the other hand, the smart agent, also called *orchestrator*, uses these resources for learning about how the user prefers the environment and in this way is capable of generating a prediction, which will be translated into action commands, the flowchart is presented in Figure 5.1. We will discuss each stage separately in the following sections. In Sections 5.1 and 5.2 we describe the data preparation and prediction training, respectively. Section 5.3 shows the conditions that tell the orchestrator to determine the future actions. Section 5.4 presents the *level of discomfort* concept and how it is relevant for the application of the rules, explained in Section 5.5. Section 5.6 indicates the execution of the decided actions. Finally, Section 5.7 concludes the chapter.

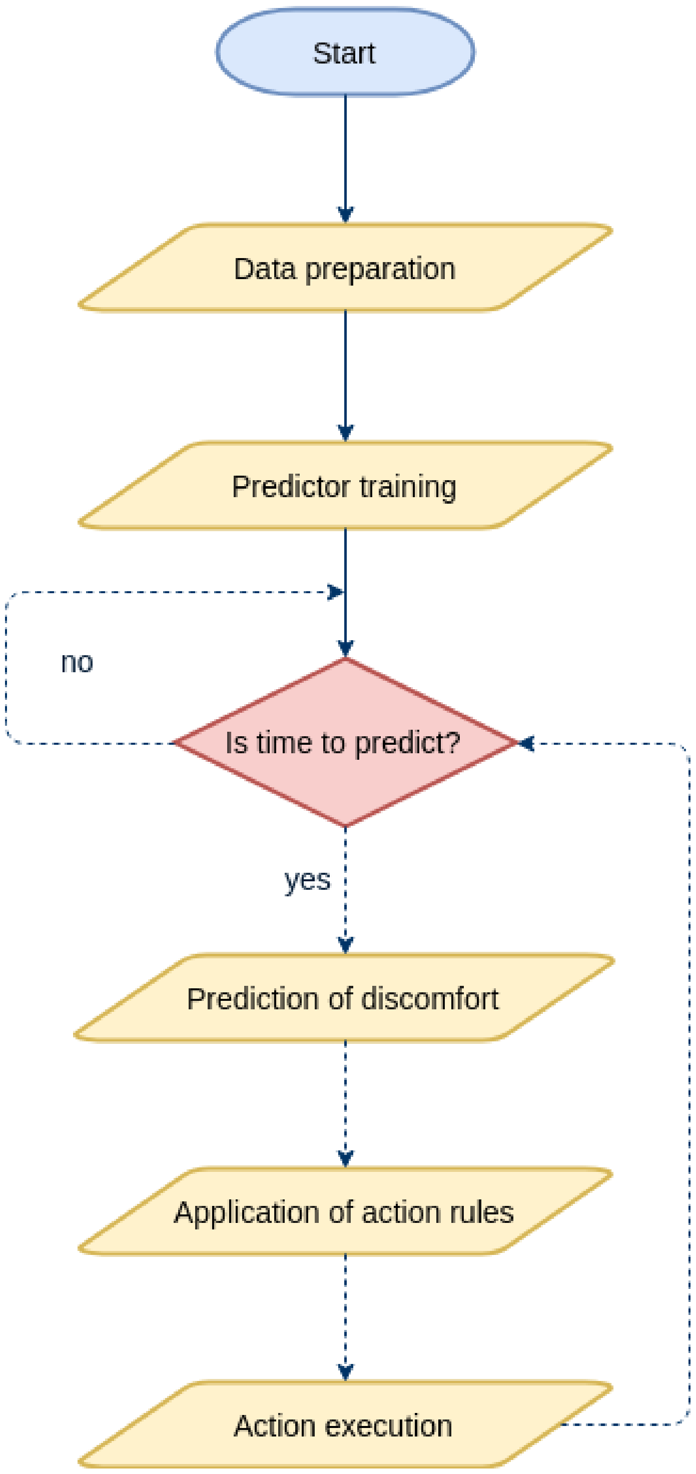


Figure 5.1: General automation flowchart

5.1 Data preparation

In this stage, we format, clean and choose data that will be used as input in the prediction model. The automation system stores raw data in a table in database and it is necessary to format it to be readable for the prediction algorithm. After training, we also need a process of cleaning because our system was susceptible to several problems provoking corrupted data. Furthermore, we choose updated data for our purpose.

A record of data in database is composed by the user which evaluated the system, the evaluated scenario, the rating, date, time and the values of all the sensors and the moment of the evaluation. Based on this information, we classified the data by users and different scenarios. In instance, data for the *user 1* in the *temperature* scenario. Hence, the predictor learns the comfort that a user is experimenting in certain environmental circumstance

Several hardware issues such as lack of battery power in certain devices, blackout, and lack of memory generated corrupted data, for this reason data cleanliness became essential. This cleaning consists of verifying that the values of all the attributes are correct. A value is correct if it is in the corresponding range. To know the values, check Table 4.2. Data that presented these values were discarded.

It is important to highlight that each day the system is trained with updated data. Evaluations of the current day are added to input data and oldest data is removed maintaining a time window of 30 days. This decision is taken because the user can change his preferences with the passage of time, for example in the changes of season.

5.2 Predictor training

In this part of the flow, the prediction model trains a machine learning predictor to estimate the user preferences. Every prediction model requires a set of entries (input) and a target value (output). In our case, the entries are all the sensor values, which represent the environment status, recorded at the moment of user evaluation. Table 4.2 shows the completed list of these entries. Furthermore, the expected prediction output represents the user evaluation provided via the mobile application.

The mobile application provides categorical options to evaluate the system. The reason is that they are more intuitive than numbers and also can capture: (a) the *intensity of discomfort*, where a larger variation between the evaluation of the user and the perfect evaluation (value=3) represents more discomfort; and (b) *the reason why the user chooses that evaluation*, i.e., the user does not like the internal room conditions because is too cold, or because it is too hot. Both *intensity* and *reason* play an important role for the orchestrator to know how to act. Nevertheless, as we choose a regressor for evaluation prediction, the training evaluations are stored and processed as continuous numbers between 0 and 6. The mapping of values for temperature and luminosity is shown in Table 5.1.

Continuous Eval.	**Temperature Eval.**	**Luminosity Eval.**
0	Extremely cold	Extremely dark
1	Very cold	Very dark
2	Little cold	Little dark
3	Perfect	Perfect
4	Little hot	Little illuminated
5	Very hot	Very illuminated
6	Extremely hot	Extremely illuminated

Table 5.1: Mapping of evaluations from categorical to continuous values in different scenarios.

Our predictor should use a regression model because of the continuous nature of the evaluation value, which indicates intensity, a characteristic that cannot be expressed with categories. Besides that, we can observe with more precision to which category (evaluation) the prediction is closest. To decide which model to use we compare different regression algorithms, as will be detailed in Section 6.1.

5.3 Prediction trigger

Once the predictor is trained, the orchestrator enters into a state of *activity* where it constantly decides if it is necessary to execute new actions. On the other hand, we determine that when the user is absent, the system enters into a state of *inactivity*, where all the devices are powered-off (for saving energy) and the only concern is to detect the next user activity (presence). Hence, there are two different presence situations: a) when the user just enters into the room after a long time of absence; and b) when the user is inside the room for a long period, this one is verified periodically.

The cyclic verification is reflected on the dotted dash arrows in Figure 5.1.When required to decide to act, the decision is made using Algorithm 2. Three input parameters are defined in the algorithm: (a) `presence_status` is obtained through a database query and indicates the most recent status of the user presence, (b) `prediction_interval`, which refers to how much time the system should wait until running the next prediction, and (c) `begin` stores the initial time of the cycle. Its value is updated when it is detected that the user is outside the room recently.

Initially, the orchestrator assumes that the user is outside the room and the first time that the user presence is identified, the presence status is redefined with the constant `RECENTLY_INSIDE`. In the next iteration, it switches to `LONG_TIME_INSIDE` to confirm the presence. This status is kept until the user absence is detected when the status is `RECENTLY_OUTSIDE`. At this moment, the system restarts the presence verification cycle assigning the current time to the variable `begin`.

Algorithm 2 Orchestrate

```
1:  procedure ORCHESTRATE
    Input: $presence\_status$, $prediction\_interval$, $begin$
    Output: $is\_time\_to\_predict$
2:      if $presence\_status == RECENTLY\_INSIDE$ then
3:          $return\ true$
4:      else if $presence\_status == LONG\_TIME\_INSIDE$ then
5:          if $(now - begin) >= prediction\_interval$ then
6:              $return\ true$
7:      else if $presence\_status == RECENTLY\_OUTSIDE$ then
8:          $begin = now()$
9:          $return\ false$
10:     else
11:         $return\ false$
```

5.4 Prediction of discomfort

For applying the rules for office automation, it is necessary to calculate the *level of discomfort* for each scenario. It helps the orchestrator to decide what to do because it transmits the physical sensation (hot, cold, darkness) and its intensity. The *level of discomfort* LD_{su} is calculated with Formula 5.1, where the evaluation of a user u in a specific scenario s is denoted by E_{su}. For this step, the E_{su} is represented by the predicted evaluation. PEV_s represents the *perfect evaluation value*. In our case, both, temperature and luminosity scenarios have a PEV_s of 3.

$$LD_{su} = E_{su} - PEV_s \tag{5.1}$$

In order to classify the prediction, as shown in Equation 5.2, this value is compared with a discomfort threshold *DT*. If the *level of discomfort* surpasses the threshold, it is inferred that a change is required and depending on the signal of the value, it is possible to decide which scenario option applied. For instance, if it necessary a change in the temperature and the *level of discomfort* value is negative, the orchestrator will warm the environment, otherwise, it will cold it.

$$classify\_discomfort(LD_{su}) = \begin{cases} no\ change & \text{for } |LD_{su}| <= \text{DT} \\ change & \text{otherwise} \end{cases} \tag{5.2}$$

As future work, we plan to use the magnitude of the discomfort to define the intensity of the actuation. For example, a value of -2 will generate a more aggressive change in the temperature than a discomfort level of -1.

5.5 Application of actuation rules

As mentioned before, the predicted evaluation gives the knowledge of what the user feels, then we use this value to update the actuators status. To determine what actions must be executed by the orchestrator, first, it is important to identify which actuators are available to use and which are the scenarios considered in the model (temperature and for luminosity). In our case we have three actuators: a) an ACU, used only to cold the air; b) a smart switch that controls the room's lamps; and c) a smart shade that regulates the natural light that enters into the room. Hence, the ACU is the unique actuator that can regulate the temperature, meanwhile, the smart switch and smart shades offer several options to regulate the internal room luminosity.

Based on our scenarios and their related actuators, several actuation rules are defined. They are presented in Sections 5.5.1 and 5.5.2.

5.5.1 Application of the rules for temperature

Figure 5.2 shows the process to adapt the temperature scenario. Once the orchestrator determines the level of discomfort is very high, it verifies if the cause is cold or heat. In the first case, the orchestrator will execute a warming rule, in the second case, a cooling one.

Due to the characteristics of our ACU, we define rules just for cooling and limit us to turn off the device when a warming action is required. Our cooling rules are the result of a combination of the different ACU options: status on/off, temperature, speed and swing on/off. These combinations were ordered using the intensity of cold that they can reach, from the one with less intensity to the one of greater intensity. This sorting is important because each time that a *cooling order* is required, the ACU can increase the cold level gradually, passing from a specific status to a colder one. If the orchestrator demands to *cold environment*, then it determines the current *cooling level (CL)* and executes the respective action. The list of temperature rules is shown in Table 5.2. In the table, the maximum level is 14, after it, there are no more options to cool the environment. Columns in the *Status* group represent the current ACU configuration and columns in the *Action* group mean the configuration to be applied.

	Status				**Action**			
CL	**ST**	**TP**	**SP**	**SW**	**ST**	**TP**	**SP**	**SW**
1	off	-	-	-	on	26	low	off
2	on	26	low	off	on	25	low	off
3	on	25	low	off	on	24	low	off
4	on	24	low	off	on	23	low	off
5	on	23	low	off	on	22	low	off
6	on	22	low	off	on	21	low	off
7	on	21	low	off	on	20	low	off
8	on	20	low	off	on	19	low	off
9	on	19	low	off	on	18	low	off
10	on	18	low	off	on	17	low	off
11	on	17	low	off	on	17	medium	off
12	on	17	medium	off	on	17	high	off
13	on	17	high	off	on	17	high	on
14	on	17	high	on	no changes			

Table 5.2: Temperature rules when is very hot.

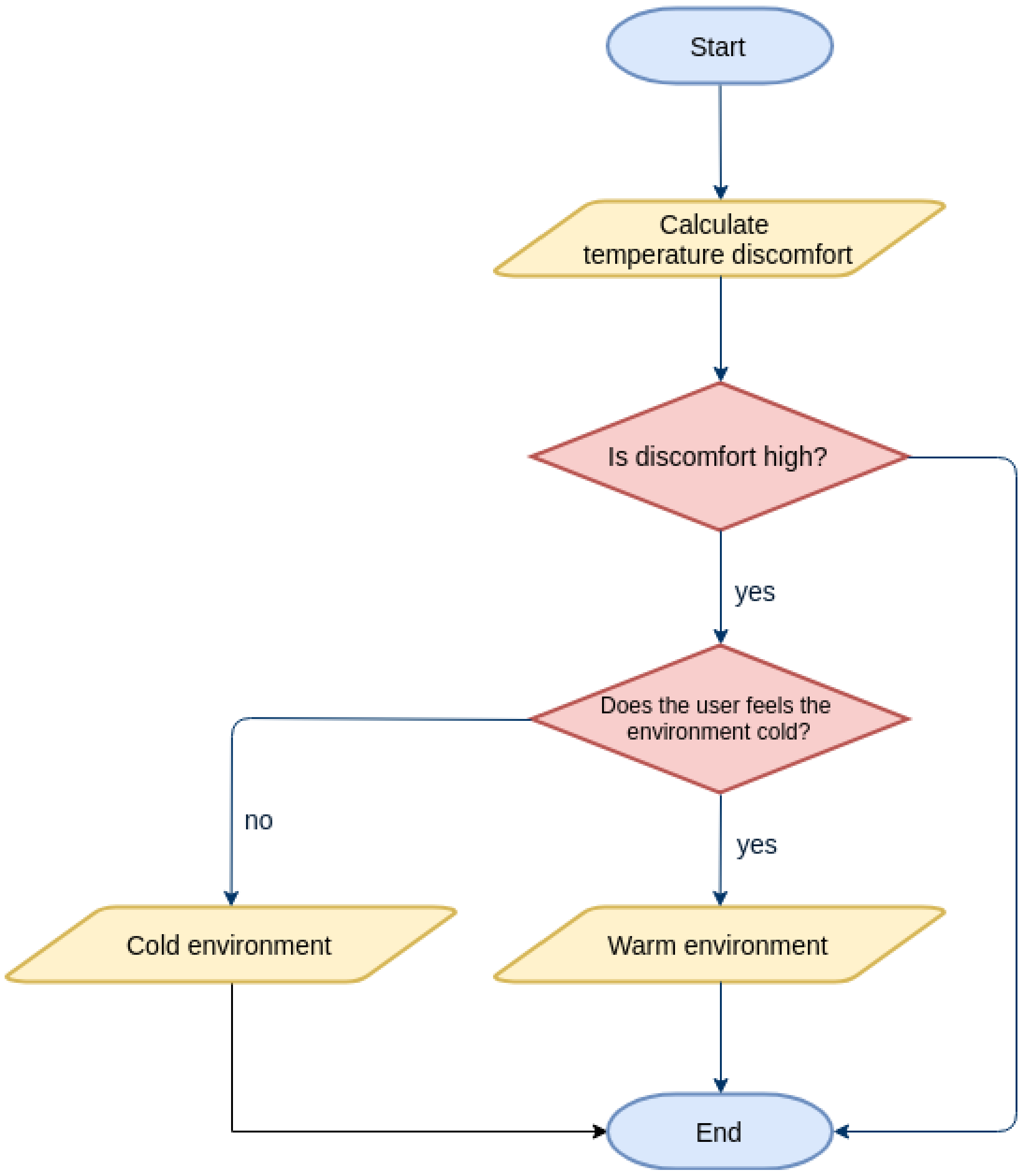

Start
Calculate temperature discomfort
Is discomfort high?
yes
no
Does the user feels the environment cold?
no
yes
Cold environment
Warm environment
End

Figure 5.2: Flow of application of the rules in the temperature scenario. For cooling the room use rules of Table 5.3.

5.5.2 Application of the rules for luminosity

Similarly to temperature, we use some rules for luminosity scenario, but instead of verifying if it is cold, it is verified if the low satisfaction is because of an excess of luminosity or for a darkness sensation. Figure 5.3 shows the flow to follow. In case of a feeling of excess lighting, we apply rules for darkening (Table 5.4). On the contrary, in case of a feeling of darkness, we apply rules for increasing the luminosity (Table 5.3). Furthermore, it depends on the *night condition*. This concept is relative, for the model *night* is defined by two sensor values: internal luminosity, l_{int} and external luminosity, l_{ext} . If $l_{int} \leq l_{int}$ it is inferred that it is night, otherwise it is *day*. Depending on this condition, the use of *smart switch* has more or less priority of use in relation to the use of smart shades. The *night condition* is just used when the switch is powered off and the shades are totally closed. In the table, the hyphen(-) means that the condition is not considered. If *day* is defined, shades have priority to be opened. If *night* is defined, the switch has the priority.

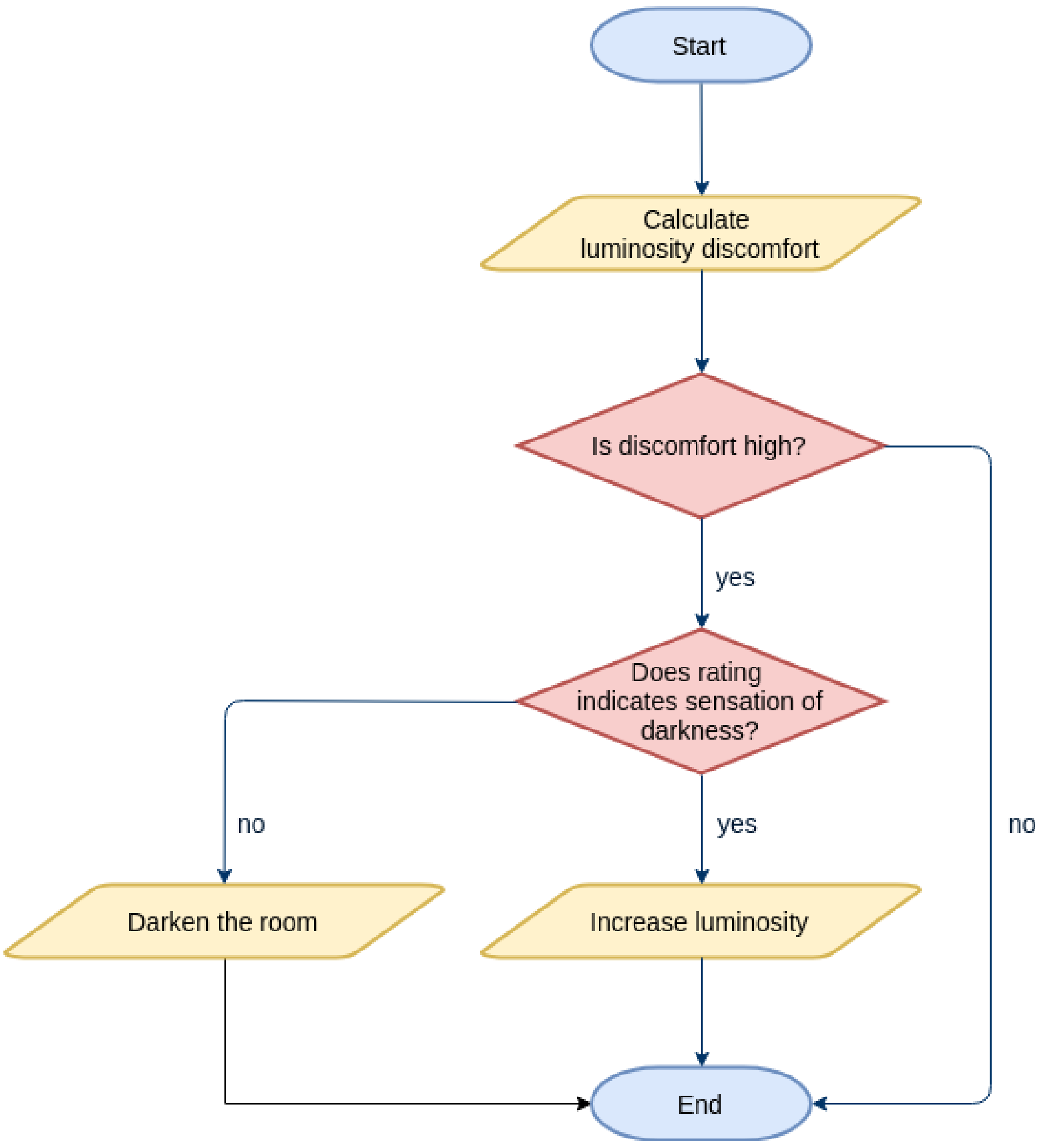

Start
Calculate luminosity discomfort
Is discomfort high?
yes
Does rating indicates sensation of darkness?
no
yes
no
Darken the room
Increase luminosity
End

Figure 5.3: Flow of application of the rules in the luminosity scenario. For increasing luminosity of the room use rules of Table 5.3 and for darkening the room use rules of Table 5.4.

	Status		**Action**	
DL	**Switch**	**Shade**	**Switch**	**Shade**
0	on	totally opened	no changes	close
1	off	totally opened	no changes	close
2	on	totally closed	turn off	no changes
3	off	totally closed	no changes	no changes

Table 5.3: Luminosity rules when is very illuminated.

	Status			**Action**	
IL	**Switch**	**Shade**	**Night**	**Switch**	**Shade**
1	on	totally opened	-	no changes	no changes
2	off	totally opened	-	turn on	no changes
3	on	totally closed	-	no changes	open
4	off	totally closed	yes	turn on	no changes
5	off	totally closed	no	no changes	open

Table 5.4: Luminosity rules when is very dark.

5.6 Action execution

Finally, after defining the actions to be applied, the orchestrator sends a command to the different actuator driver, i.e. ACU, smart switch and smart shades. The automation system executed these actions in real time and keep verifying constantly if is necessary a new fix in the room environment.

5.7 Conclusion

This chapter presented the proposed intelligent automation model. It required an adaptation of the ManIoT platform to interact with an orchestrator, which is the agent in charge of the room automation. For executing the automation, the orchestrator uses user evaluation prediction and action rules. The next chapter will evaluate which regression algorithm to use based on its precision. Furthermore, the next chapter will also evaluate the system in real conditions, with two real users.

Chapter 6

Evaluation of the Intelligent Automation Model

In this chapter we analyze (a) regression models to define which one to use as the predictor (Section 6.1); (b) feature relevance to reveal the importance of an adaptive predictor (Section 6.2); and (c) the performance of a system with predictor and without the predictor (Section 6.3). Finally, we close the chapter with a general conclusion.

6.1 Evaluation of Regression Models

Our proposal requires an algorithm to predict the user evaluation at a specific time. Therefore, in this section, we present experiments related to the choosing of the best method. First, we give the details about data. Second, we indicate which were the selected algorithms. Third, we explain how we compared the algorithms. Finally, we show and discuss the results.

As was mentioned in Section 5.2, our training data was generated by an user who used the SmartOffice mobile application to evaluate the automation system. The user was a member of the WINET laboratory and did not have a regular activity inside the room. There were days when the user worked all day and others in which he stayed in the room only for a few hours. Hence, the quantity of hours and the times he enters into the room were varied. To obtain the amount of data needed to train our predictors, we encouraged the user to evaluated the system periodically (some alarms were set to remain a new evaluation).

At the moment of choosing the best algorithm, we consider data of the system without the application of our proposal. In this stage, we collected 205 evaluations in 102 effective hours distributed in 25 days. Data was captured from August 17, 2018 to October 10 of the same year. Nevertheless, at the final of this research we recalculated the results adding data of the system with proposal. This with the purpose of obtaining more accurate results. The new data was tracked from October 17 to January 24, 2019 and also was distributed in 25 days. In this stage, the user evaluated the system 172 times in 84 effective hours. Table 6.1 shows the schedule of data tracking [1].

Aug 2018				Set 2018				Oct 2018				Nov 2018				Dec 2018				Jan 2019			
w1	w2	w3	w4	w1	w2	w3	w4	w1	w2	w3	w4	w1	w2	w3	w4	w1	w2	w3	w4	w1	w2	w3	w4
		■	■	■	■	■	■	■	■														
											■	■	■	■	■	■	■	■	■	■	■	■	

Table 6.1: Schedule of data tracking

The algorithms selected for the analysis were Random Forest regressor (RF), XGBoost regressor (XGB), Logistic regressor (LR), kNeighbors regressor (KNN) and Multi Layer Perceptron regressor (MLP). We chose them because they are well-known regression algorithms and they are based on different principles. Consequently, we can explore different forms of prediction. RF and XGB are ensemble methods, LR is a statistical model, KNN is a non-parametric method, and MLP is a method based on neural networks.

To determine which one is the most appropriate regressor, we apply the Root Mean Square Error (RMSE). The error was normalized to the range $[0, 1]$ because evaluations have values between 0 and 6. For running the tests, we use the cross-validation method holdout. We separated a quantity of data which is used to train all the regressors, and then use the rest to tests the predictions. We calculated the RMSE for all the predictions and as a result the average RMSE for each regressor.

Figure 6.1 shows the results in our scenarios. Blue bars represents the values for temperature and red bars for luminosity. In the first case, almost all the algorithms obtained RMSE $= 0.0458$ and the only different result is from the XGBoost regressor that obtained RMSE $= 0.0426$. In the second case, the situation is similar. The XGBoost regressor obtained RMSE $= 0.0457$, while the other algorithms got RMSE $= 0.041$.

Although in both scenarios, the difference in results is small, we can reach two conclusions. First, the XGBoost regressor is better than the other algorithms. Second, in all cases the result of the RMSE was less than 0.1, which indicates that they are good predictive models. This means that any one of the five algorithms could be used in our model.

[1] In the table, the second row represent the weeks of a month, for example w1 would be the first week.

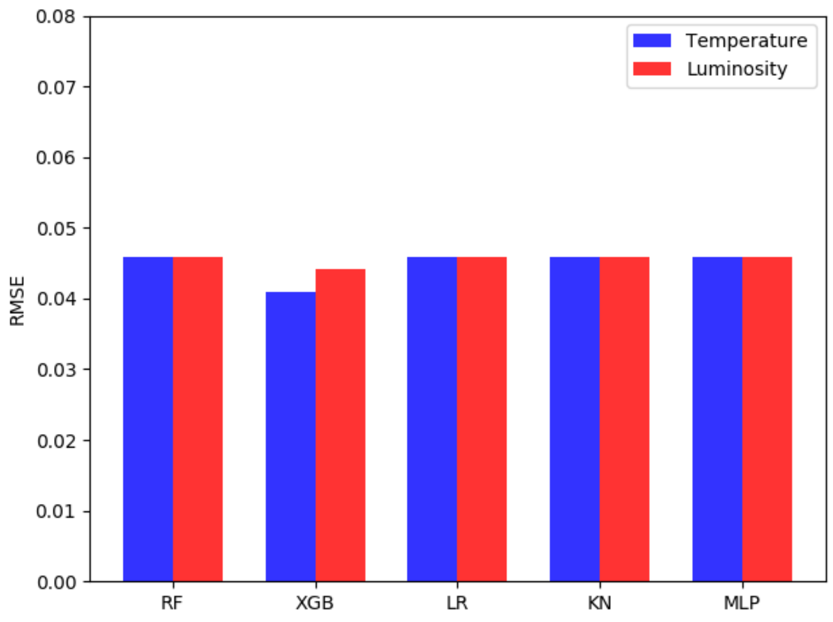


Figure 6.1: Comparison of regression algorithms using RMSE

6.2 Data Analysis with SHAP

As mentioned in Chapter 2, SHAP is a tool used to analyze how a model behaves with certain data. It offers advanced options of visualization to deploy specific features, revealing their degree of relevance and correlation with other features. By using SHAP, we aim to demonstrate that different factors can cause discomfort for different users. For instance, a high concentration of gases can give a greater sensation of heat to one user and not another, which may feel more heat when there is more humidity.

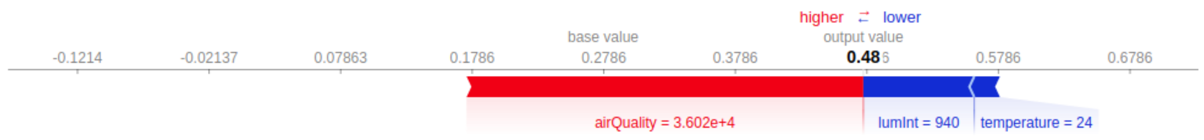
As we already explained, each evaluation record is made up of the values of the sensors and the value of the evaluation. Nevertheless, to simplify the SHAP analysis, we decided to binarize the target value (evaluation). The binarization generated two classes: (a) *change* (value=1), which demands an immediate change (in the room) because of the high level of discomfort; and (b) *nochange* (value=0) that indicates that no action is required. Values lower than 2 and higher than 4 belong to the class *change* and the other to class *nochange*.

For our analysis, we choose three kinds of SHAP data visualization: (a) `summary_plot`, (b) `summary_plot_bar`, and (c) `force_plot`. The first two give a general idea about which features are the most influential in the model, and the third one indicates the proportion of influence and if the feature value is directly or inversely proportional to user comfort. Hence, we show in the groups of figures 6.2, 6.3, 6.4, 6.5, the visualizations per scenario and user.

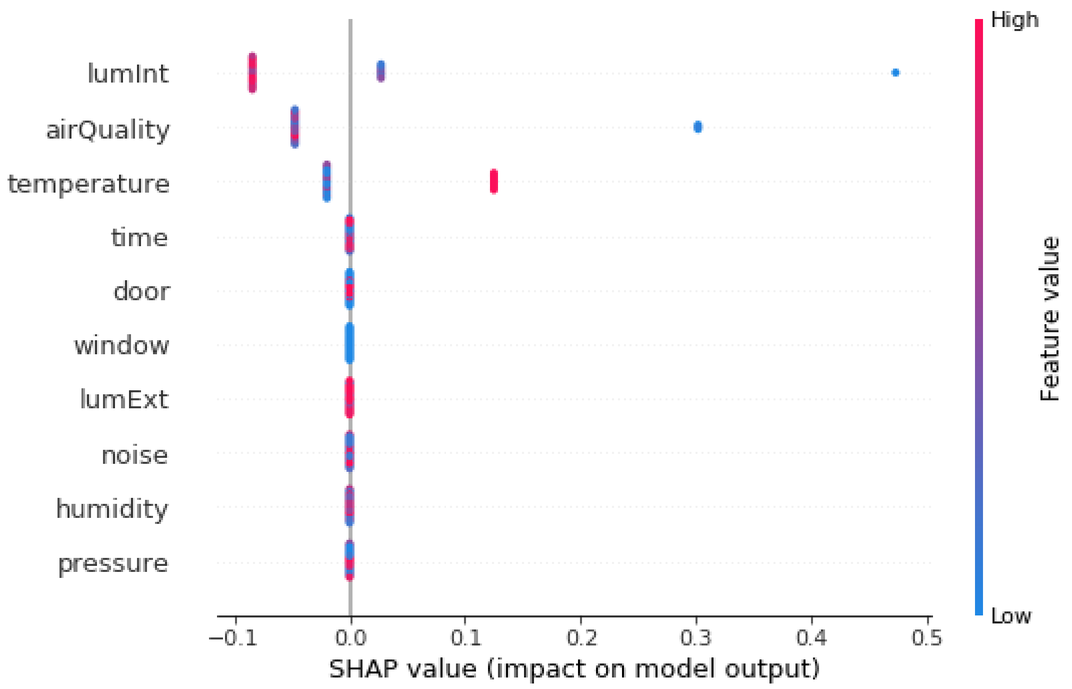
First, we discuss the results for *user 1*. In the *temperature scenario*, as shown in figures 6.2b, 6.2c, the most relevant features are *internal luminosity*, *temperature* and *air quality*. Nevertheless, we can infer from Figure 6.2a that air quality is the most crucial and when it has higher values, *user 1* feels more discomfort. On the other hand, something that could not be identified intuitively is that a higher *internal luminosity* influences positively this user in the *temperature scenario*. Additionally, a higher *temperature* also provides comfort to *user 1*, but not as much as expected. In *luminosity scenario*, figures 6.3b, 6.3c and 6.3a reveal that *time*, *humidity* and *internal luminosity* are the unique features that determine the user comfort. *Time (hour)* is the main factor for this scenario, since in later times of the day, the user perceives the environment darker, a situation that he does not like.

Second, we analyze the scenarios for *user 2*. In the *temperature scenario*, `summary_plot` and `summary_plot_bar`, in figures 6.4b, 6.4c show that almost all the features are involved to determine the comfort of this user. The unique feature that does not intervene is *window status*. The `force_plot` graph, shown in Figure 6.4a, gives a deepest explanation. There, we can observe that *pressure* pushes the model output higher, i.e., a higher value influences negatively the user comfort. Contrarily, *time*, *humidity*, *air quality* and *door status* contribute to improve the comfort. The first two features are the most influential, specially *time*. From the graph we can deduce that *user 2* feels more comfort (in relation to temperature) when it is later. In *luminosity scenario*, figures 6.4b and 6.5b indicate that just four features affect the user comfort: *internal luminosity*, *air quality*, *external luminosity* and *humidity*, being the first one the most influential. `Force_plot`, shown in Figure 6.5a, explains better the feature influence. We can interpret that this user likes to work in a well illuminated room, but prefers the light of luminaries than the natural light, or maybe extremely high solar radiation bothers him. In addition, a high humidity influences negatively his comfort, but a poor air quality influences positively his perception of the luminosity.

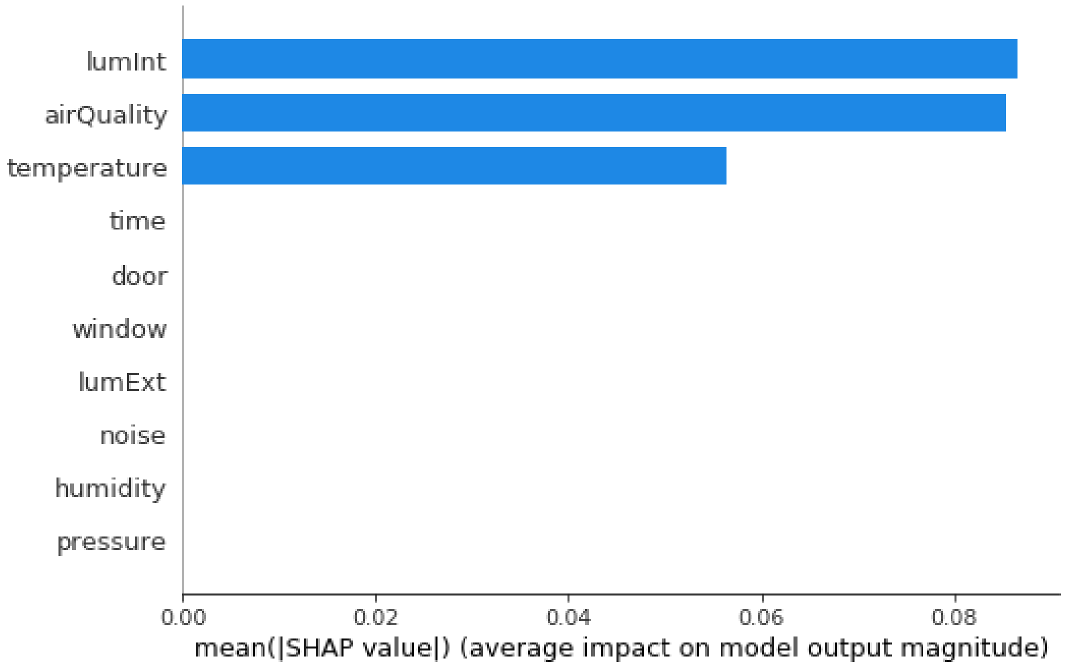
After analysing the features that influence the two users, we can conclude that both are very different. In the temperature scenario, the air quality, the internal luminosity and temperature are very relevant for *user 1*, while pressure, time and humidity are for *user 2*. In the luminosity scenario, *user 1* is highly influenced by the time and internal luminosity, and *user 2* is influenced by the internal luminosity, air quality, external luminosity and humidity. Hence, each user should have a personalized automation, which can be reached by using different instances (for each user) of the predictor.



(a) SHAP `force_plot` for *user 1* in temperature scenario.

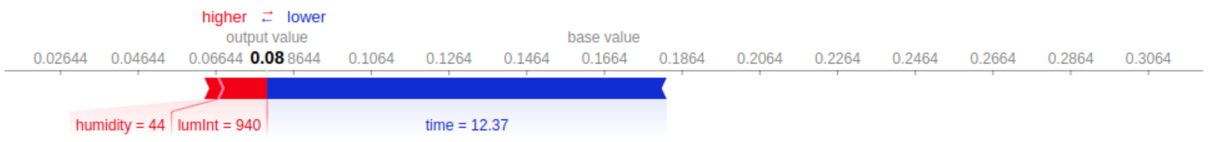


(b) SHAP `summary_plot` for *user 1* in temperature scenario.

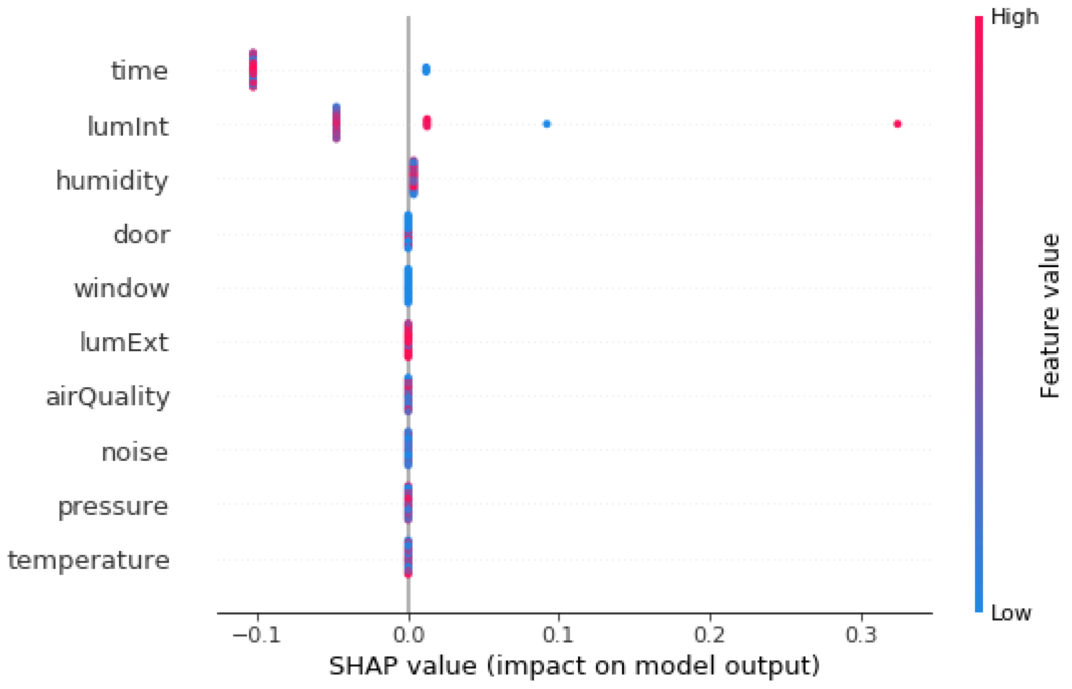


(c) SHAP `force bar_plot` for *user 1* in temperature scenario.

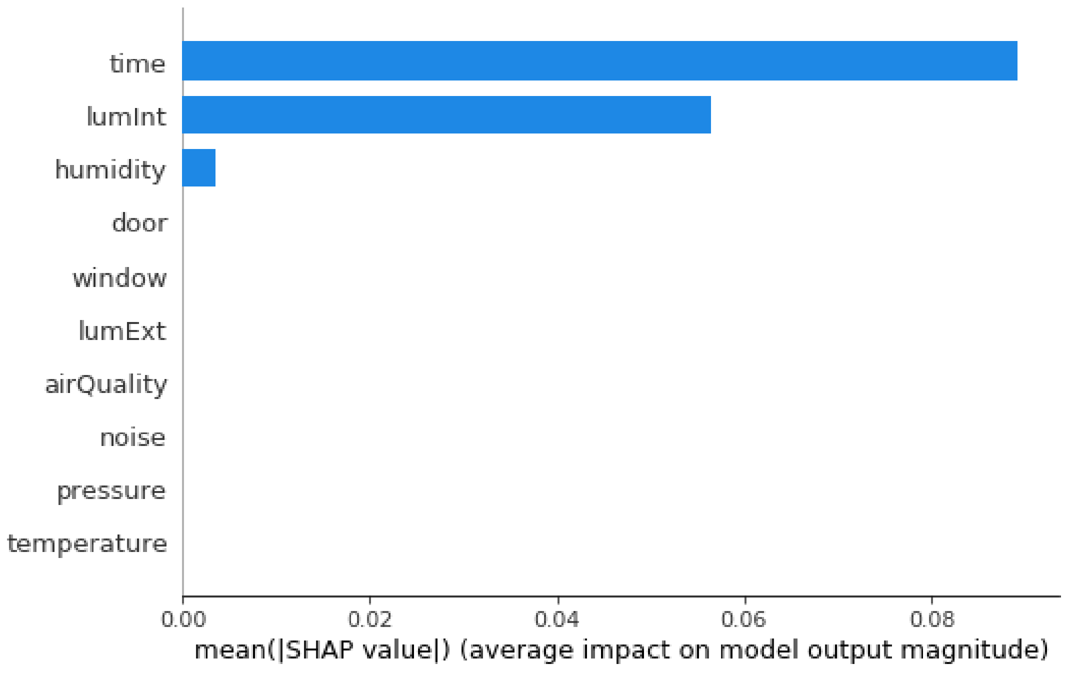
Figure 6.2: SHAP plots for *user 1* in temperature scenario.



(a) SHAP force_plot for *user 1* in luminosity scenario.

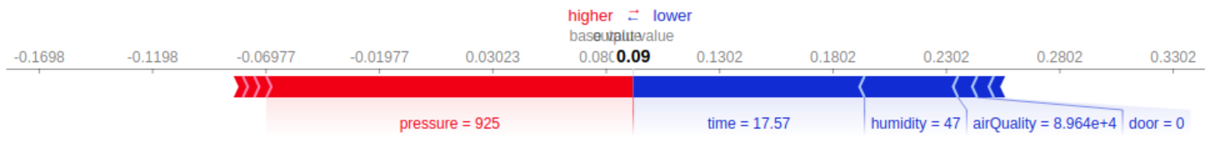


(b) SHAP summary_plot for *user 1* in luminosity scenario.

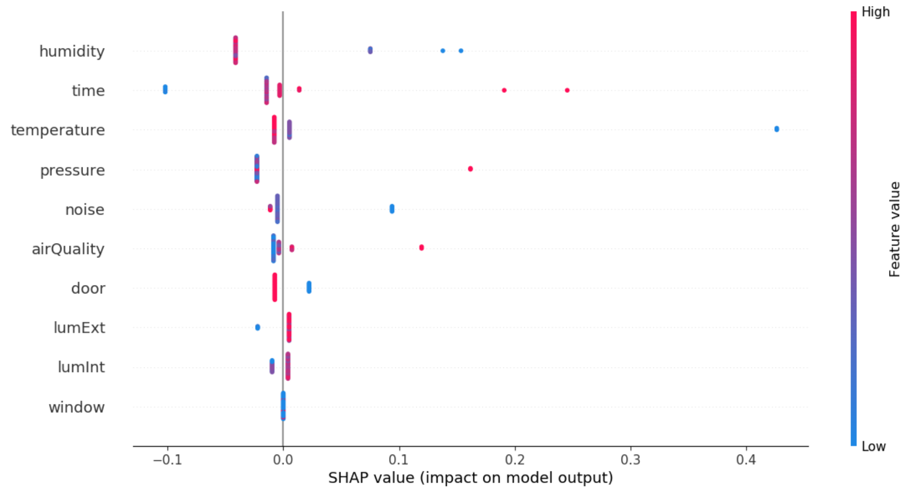


(c) SHAP force bar_plot for *user 1* in luminosity scenario.

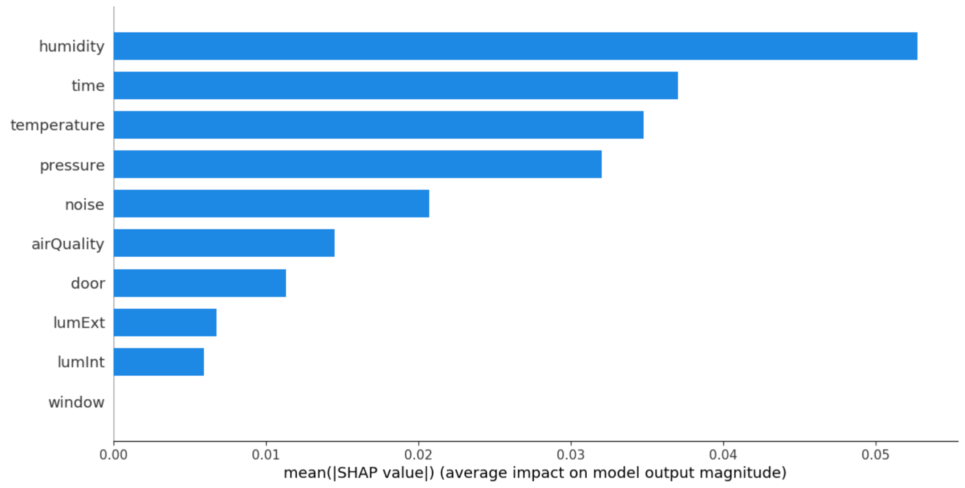
Figure 6.3: SHAP plots for *user 1* in luminosity scenario.



(a) SHAP `force_plot` for *user 2* in temperature scenario.

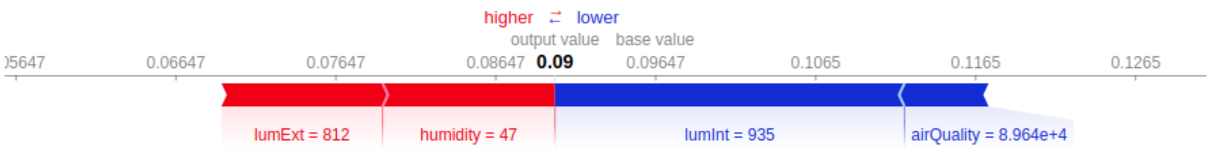


(b) SHAP `summary_plot` for *user 2* in temperature scenario.

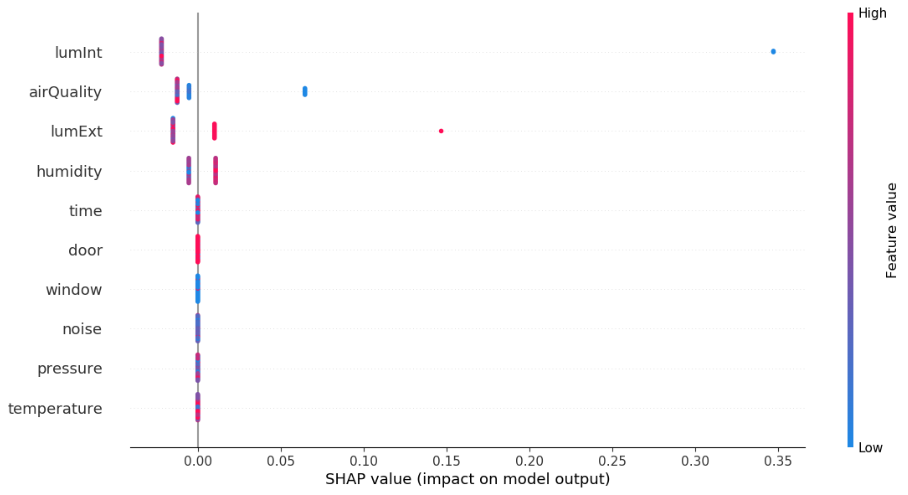


(c) SHAP `force bar_plot` for *user 2* in temperature scenario.

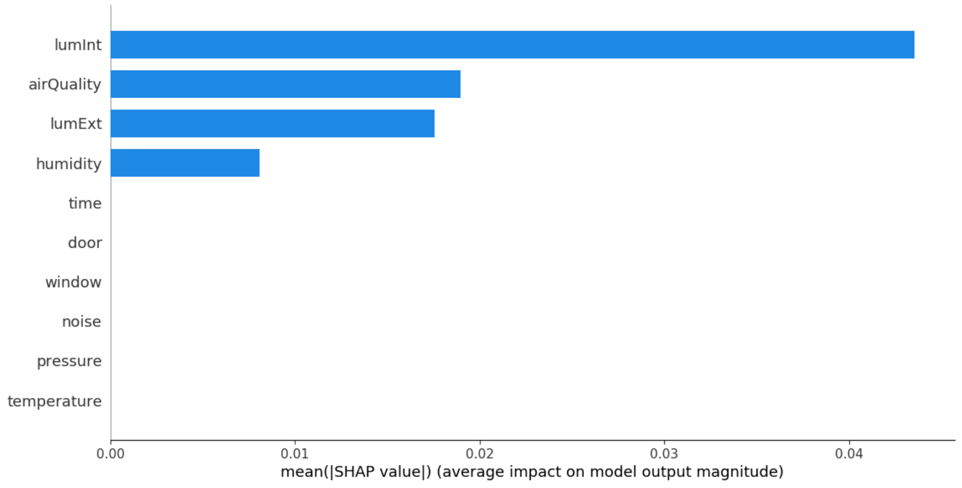
Figure 6.4: SHAP plots for *user 2* in temperature scenario.



(a) SHAP force_plot for *user 2* in luminosity scenario.



(b) SHAP summary_plot for *user 2* in luminosity scenario.



(c) SHAP force bar_plot for *user 2* in luminosity scenario.

Figure 6.5: SHAP plots for *user 2* in luminosity scenario.

6.3 Evaluation of the SmartOffice System

To justify the importance of this work, we demonstrate that the automation system with our proposal offers a better user experience than an automation system without our orchestrator. The notion of user experience can be very abstract and subjective, however, we define two metrics that help us capture this idea: (a) *metric based on discomfort*, and (b) *metric based on interactions*. These metrics take advantage of the user feedback, which can be explicit or implicit. Furthermore, they are calculated in relation to a specific day. The reason is that the usefulness of the system can be very variable, some days the user can stay in the office for a long uninterrupted time, other days he can be present several times for just a few minutes or even be absent for whole days.

We use the *student's test-t* for unpaired observations to support our metrics. First, we determine if the two systems are significantly different. This occurs when the confidence interval of $A - B$ does not include zero. And second, we compare the average metrics, the system with lower average is considered the better, since the objective is to minimize both metrics.

To calculate our metrics we utilized data from a single user and we chose 25 days of data. The system without proposal was used from August to October and the system with the smartoffice orchestrator from November to January. The explanation of the metrics, their results and conclusions are detailed in Sections 6.3.1 and 6.3.2.

6.3.1 Metric based on discomfort

One way to know if the user experience improved under any situation is by directly asking the user about his opinion. As seen in recommendation systems, one way to measure the degree of user satisfaction in relation to a service or product is through an evaluation, either through a form or simply assigning a note [17]. In our case, the automation system is the service and the degree of satisfaction will be known through the discomfort metric. In our case, we capture it using a questionnaire in the mobile application whose responses are translated as *the level of discomfort*, a concept that is used for: (a) the application of rules, and (b) the calculation of this metric that we called *daily intensity of discomfort (DID)*. Basically, it represents numerically the user's intensity of discomfort throughout the whole day.

Table 5.1 shows a mapping between the categorical rating value to a numeric rating value. If the user wants to express discomfort with a lot of intensity about any of the scenarios (luminosity and temperature), he chooses an option such as *Extremely hot*, *Extremely cold*, *Extremely dark* or *Extremely illuminated*. These options have the maximum intensity value (3.0) in contrast to the minimum (0.0), which represents a perfect evaluation.

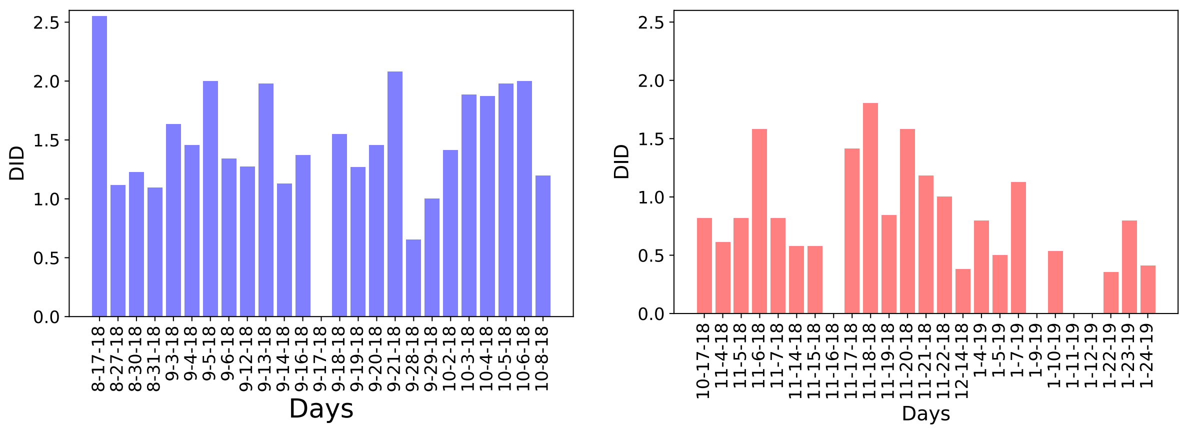
We calculate DID of a specific day d using Equation 6.1, where n_d represents the number of evaluations on day d and $LD_{i,d}$ is the level of discomfort calculated from one of the evaluations at day d. DID is always non-negative, and a value of 0 indicates that the user did not feel any discomfort during the day. In general, a lower DID_d is better than a higher one, hence, the objective of our proposal is to minimize its value.

$$DID_d = \sqrt{\frac{\sum_{i=0}^{n_d} (LD_{i,d})^2}{n_d}} \tag{6.1}$$

Figures 6.6a, 6.6b, 6.7b, 6.7a contrast the results of the system without our proposal (in blue) with the system with the proposal (in red), for temperature and luminosity evaluation scenarios.

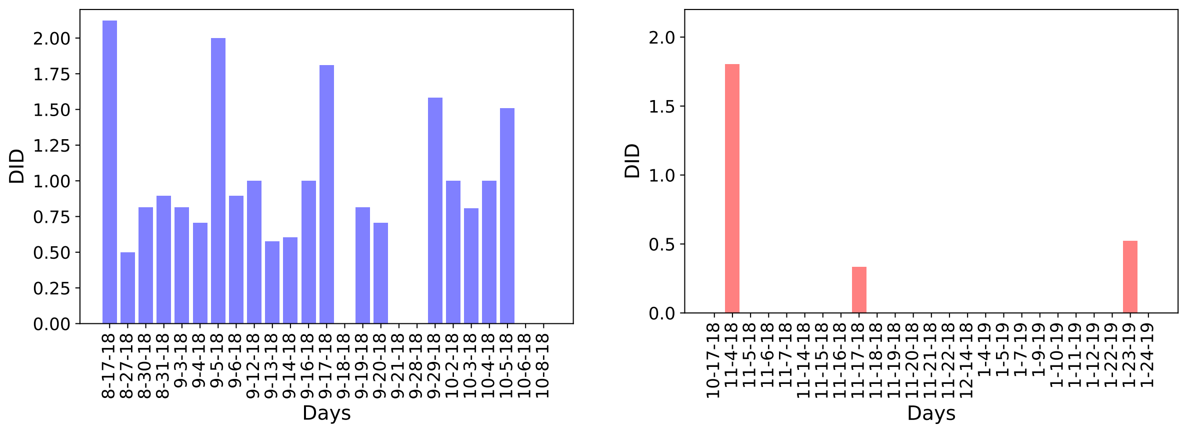
In figures 6.6a and 6.6b, we compare the system versions for the temperature scenario. We observed that the system without our proposal rarely recorded values between 0.0 and 1.0 for the DID metric, however with the automation proposal 18 of the 25 days achieved this range. In addition, the system without our proposal reached the worst result, $DID = 2.55$. By using the *student's test-t* with 90% confidence level, we obtain $< 0.53, 0.91 >$ as a confidence interval for the difference. The interval does not include zero, hence, we can affirm that the systems are significantly different and that the proposed system reduces the level of discomfort. Our proposal allows to achieve an average $DID = 0.74$, lower than $DID = 1.46$ reached if we do not apply it. In conclusion, by using the proposal we offer an improvement to the user comfort with respect to the temperature scenario.

As shown in figures 6.7b and 6.7a, the results of this metric in the luminosity scenario are more conclusive than those of temperature. We can clearly see that on almost every trial day with the proposal, the DID value was zero, which indicates that the user was very satisfied with our predictions. This did not happen with the system without the proposal, in which most days (17 of 25) the values of DID were greater than 0.5. Further, the worst value of DID = 1.8 was obtained without our proposal. Similar to the temperature scenario, we applied the *student's test-t* and obtained a confidence interval $< 0.53, 0.91 >$, and an average $DID = 0.85$ for the system without proposal and an average $DID = 0.11$ for the proposal. Hence, we can confirm that our proposal was significantly superior in this scenario.



(a) Without the proposal (b) With the proposal

Figure 6.6: DID values for temperature



(a) Without the proposal (b) With the proposal

Figure 6.7: DID values for luminosity

6.3.2 Metric based on user interactions

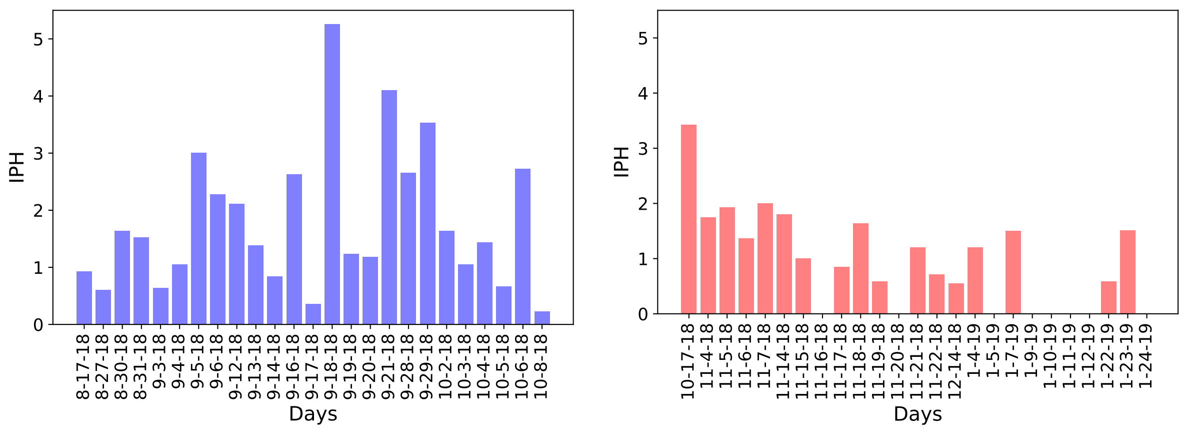
One of the purposes of this proposal is for the user to execute the least number of manual actions, thereby saving time and gaining concentration to have a continuous and efficient work. For that reason, we define the metric based on user interactions. In our case, an user interaction is each one of actuator changes, which are requested through the mobile application. Unlike the metric based on discomfort, this metric assumes that the fewer actions are required, the better the user experience. In other words, automation will be perfect when the user does not need to intervene anymore in the environment.

The metric below represents the average of interactions over the time the user is inside the room during a day d as shown in Equation 6.2. Time is expressed in *minutes* but it is converted to *hours*.

$$IPH_d = \frac{I_d}{time_d} * 60 \tag{6.2}$$

Figures 6.8, 6.9b, 6.9a, 6.10b, 6.10a, 6.11b and 6.11a shows the comparison between the system versions using the IPH metric. Similarly to the DID results, the blue line represents the version without proposal and the red line the system using our proposal. In the figure, the X axis represents the day and the Y axis, the IPH.

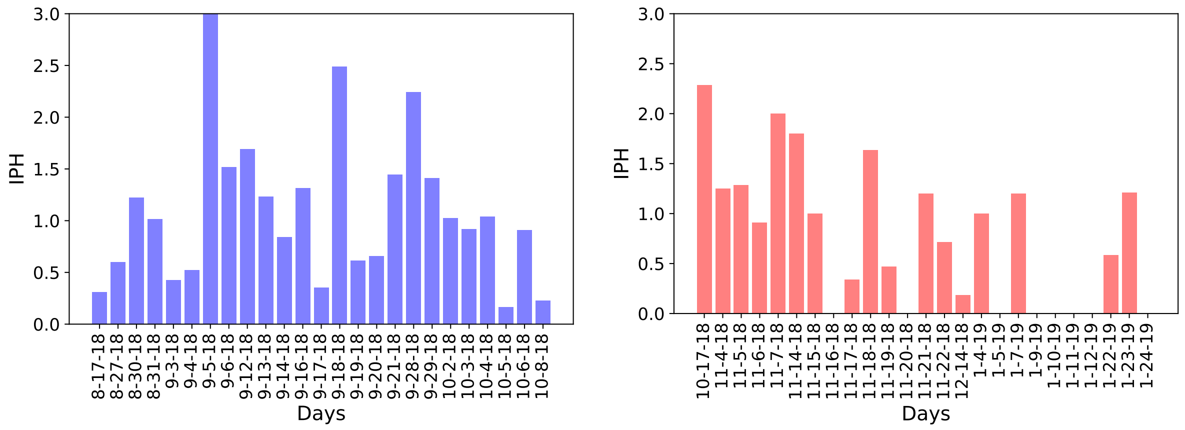
In general, figure 6.8 shows that the system using our proposal obtained an average IPH=0.94, lower than 1.79 of the system without the proposal. Besides, it was registered a day with a high value IPH=3.43, the IPH values of almost all the the days is lower 2.0, and eight of them reached IPH=0. On the other hand, the system without the proposal achieved only 13 out of 25 days with values in this range and there is no day when it registered an IPH=0. This means that by using the proposed system the user always requests at least one action to the automation system. The following paragraphs analyze the results separately, i.e. presenting a comparison for each actuator.



(a) Without the proposal (b) With the proposal

Figure 6.8: Overall IPH of the systems

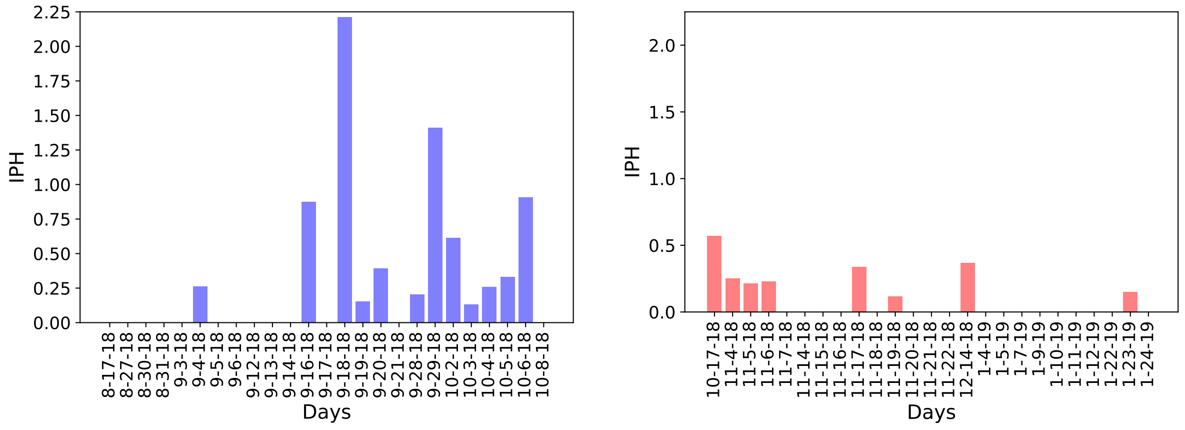
Figures 6.9a and 6.9b shows the comparison of the versions using the metric IPH for AC actuator. In this case, our proposal achieved an average IPH=0.76 and the system without the proposal achieved IPH=1.09. This means that the difference is not vast but is favorable for our proposal, which got more than a half of days with a IPH lower than 1.0, versus the eleven days obtained without orchestration. In addition, the proposal achieved eight days without user interactions, which was the best case, while the system without orchestration never reached that result. The *student's test-t* defines a confidence interval ¡0.06, 0.59¿, which indicates that our proposal has a better IPH.



(a) Without the proposal (b) With the proposal

Figure 6.9: IPH metric for the AC actuator

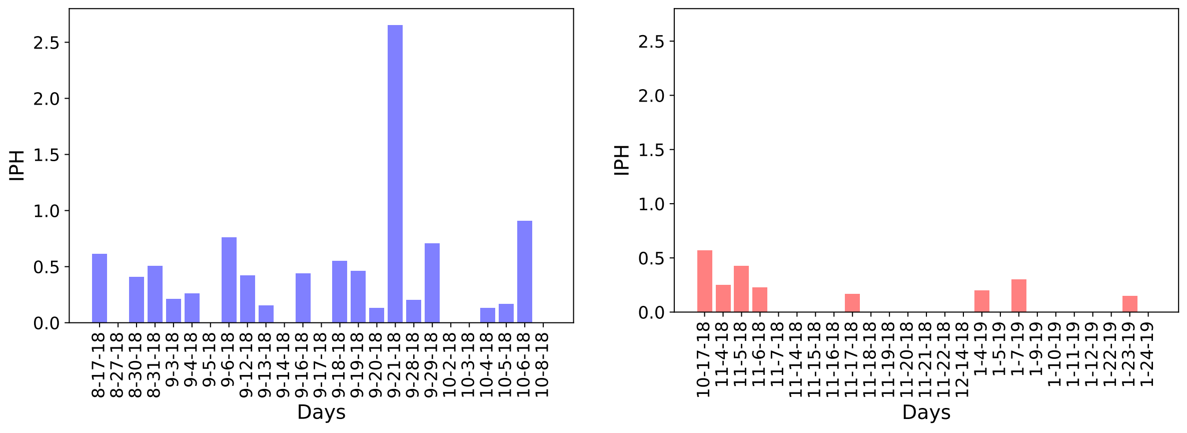
The improvement is more prominent with the switch actuator. We can see in figures 6.10a and 6.10b that 17 of 25 days mark IPH=0 with our automation proposal, while the base system just achieves seven. The base system also obtained the worst result, IPH=2.65. In general, the system without our proposal has an average IPH=0.39 and the system with proposal an average IPH=0.09. The *student's test-t* indicates a confidence interval ¡0.15, 0.44¿ with 90% of confidence level. Therefore, we can conclude that our system has a positive impact in smart switch usage, reducing the interactions required by the user.



(a) Without the proposal (b) With the proposal

Figure 6.10: IPH metric for the Smart Switch actuator

The results for the opening and closing of the smart shades, shown in Figure 6.11a and 6.11b, are also better for the proposal. It reaches 17 out of 25 days with IPH = 0 and most of the remaining values do not exceed 0.40, with the exception of a day that was 0.57. In the case of the system without our proposal, and IPH = 0 was recorded in 13 days, but there were four days in which a value greater than 0.6 was reached, even reaching the highest value of 2.21. Finally, the average IPH for the proposal is 0.09 and for the version without orchestration is 0.31. *Student's test-t* confirms a relevant difference between the systems with a confidence interval ¡0.07, 0.37¿. Then, we can conclude that our proposal also has a positive impact over the number of actions performed on the smart shades.



(a) Without the proposal (b) With the proposal

Figure 6.11: IPH metric for the Smart Shade actuator

6.4 Conclusions

This chapter evaluates the importance of our proposal by comparing it with a smart office system that offers a basic automation without intelligence. For this comparison we defined two metrics (DID and IPH), which were supported by the *student's test-t* to confirm the superiority of our proposal. We used the SHAP library to reaffirm that each user perceives the environment in a different way in relation to another, so for one user the sensation of heat is closely related to humidity and temperature, but for another, darkness also influences the perception of the environment. For this reason, our proposal provides customized solutions. We observe through the DID metric that the user feels more comfortable using the automation system with our prediction proposal. In addition, it was verified (with the IPH metric) that the user executes fewer manual actions, since the system does it for him.

Chapter 7

Conclusions

As we have seen, the customization of an intelligent office implies learning about the user in order to change some parameter of the office. This brought great challenges, from the analysis of the components that were considered, through the implementation in a real office, the selection of an appropriate prediction algorithm, to the logic to orchestrate the devices in the best way.

The implementation of the automation system in a real office was really challenging. In the beginning, we defined the essential requirements to be able to use it as a basis for our proposal. These requirements were: the implementation of a sensing system, the controllers of the actuators, the user interface, the system for identifying the presence of the user, the mechanism for sending and obtaining information, an orchestrator that predicts and executes the next user action, a gateway where the controllers, the database and the algorithms processing are centralized. However, with calibration of the sensors, the stability of the connection, the persistence of the data, choosing the best presence identification mechanism were some of the challenges that we faced in the hardware part. In addition, we have had a great difficulty to achieve a stable system to collect the training data. One frequent issue was the lack of memory, which suggests an optimization in the implementation of the prediction part.

One of the main challenges of this work was the number of tasks that had to be accomplished before running the complete system. Data collection could only start after all the office was automated, and during data collection we were at the mercy of the weather, and the fact that users interact with the environment a few times a day. Because of that, our database was not representative of the whole year (differences in temperature, sunrise and sundown times, periods of long rains), and the amount of collected data was limited.

Despite those limitations, we could verify that each user has different sensations with respect to the climate of the room and this impacts strongly the predictions of the actions of each user. This supports the importance of learning about each different user and customizes the office with his/her own preferences. It also reinforces the importance of a metrics based on discomfort, we can map the different users sensations in an numeric value and consequently measure the approval of different office automation systems.

After running the experiments to compare the old system with the new system, we verified that our proposal improved the user experience. On the one hand, the smart office obtained an average of daily intensity of discomfort (DID) that is considerable smaller than the office without the proposal. Likewise, we observed a considerable improvement in the average number of Interactions per Hour (IPH), that is, with the smart office the user decreased the number of interactions with the environment.

7.1 Future work

Some future works are:

- Use temporal methods to predict the evaluation rating. Thus, the machine learning algorithm can learn about changes over the time. For instance, if during a period it is detected that the temperature is increasing, then it is expected that the next days will be more hot.
- Incorporate more physical sensors for providing data that can enrich the prediction of comfort evaluations. For example, sweat sensors, a sensor to measure the dilation of the pupil or ultraviolet ray sensors.
- Consider the city climate parameters. This parameters can be incorporated from virtual sensors implemented in several online applications.
- Consider different periods for training, i.e. after a day, after a week, after ten days. It depending on whether there is a radical change in climate like a change of season.
- Definition of more effective rules for the orchestration of the actuators. Currently, we do not consider the intensity of discomfort to define the intensity of the actuation. We could use the degree of discomfort to calculate how many degrees or lumens should be changed at the time of action.
- Explore new predictions algorithms such as unsupervised algorithms or reinforcement learning.
- Propose an orchestrator that learns the user preferences with time. This is important for a commercial product, since the system should ship with a basic comfort model. This model would be refined to match the preferences of each user.
- Assuming that this work could become a commercial product, new options can be added in the mobile application: a section for user configuration, screens to

add new users, methods to give permissions to certain devices, assigning the user identification method, etc. In the same way, a more flexible management of the devices (sensors and actuators) is needed, allowing them to be added and removed. In addition, a web version could be implemented.

- Implementation of a better mechanism for user identification. Our mechanism is based in the wifi connection, which is limited to the internet connectivity, cellphone security restrictions and range of the device. The mechanism based on a key chain, which was also implemented, was useful too, but also has its limitations mainly in coverage, it has a very wide range. Hence, future work could consider the fusion of multiple sensors to model user presence.
- Implementation of a generic driver for controlling a vast quantity of air conditioning devices. According to the bibliographic review, there are applications that control several manufacturers, as long as they communicate via infrared protocol.
- Due to the fact that in shared environments there are microclimates of both temperature and luminosity, it is possible to create a system that helps to plan the best seat location to each one of the users. Hence, the user that usually feels cold will be located in a warm position.

Bibliography

[1] Ambi climate. https://www.ambiclimate.com/, 2018. Accessed: 2018-09-05.

[2] Distech controls' smart room control solution. http://www.distech-controls.com/, 2018. Accessed: 2018-09-05.

[3] Evapolar personal air control. https://evapolar.com/, 2018. Accessed: 2018-09-05.

[4] Kapsul. https://www.kapsulair.com, 2018. Accessed: 2018-09-05.

[5] Klevernes. http://www.kleverness.com/, 2018. Accessed: 2018-09-05.

[6] Mommit cool. https://www.momit.com, 2018. Accessed: 2018-09-05.

[7] Nature remo. https://nature.global/, 2018. Accessed: 2018-09-05.

[8] Nest learning thermostat. https://nest.com/, 2018. Accessed: 2018-09-05.

[9] Sensibo sky. https://sensibo.com/, 2018. Accessed: 2018-09-05.

[10] Muhammad Raisul Alam, Mamun Bin Ibne Reaz, and Mohd Mohd Ali. Speed: An inhabitant activity prediction algorithm for smart homes. *Systems, Man and Cybernetics, Part A: Systems and Humans, IEEE Transactions on*, 2012.

[11] N. S. Altman. An introduction to kernel and nearest-neighbor nonparametric regression. *The American Statistician*, 46(3):175–185, 1992.

[12] Yali Amit and Donald Geman. Shape quantization and recognition with randomized trees. *Neural Comput.*, 1997.

[13] Josué Batista Antunes. Uma plataforma para gerenciamento e aplicações em internet das coisas. Master's thesis, Programa de Pós-Graduação em Ciência da Computação da Universidade Federal de Minas Gerais, 2016.

[14] Luigi Atzori, Antonio Iera, and Giacomo Morabito. The internet of things: A survey. *Comput. Netw.*, pages 2787–2805, 2010.

[15] Sebastian Bach, Alexander Binder, Grégoire Montavon, Frederick Klauschen, Klaus-Robert Muller, and Wojciech Samek. On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation. *PLOS ONE*, 2015.

[16] Piotr Batog and Marek Badura. Dynamic of changes in carbon dioxide concentration in bedrooms. *Procedia Engineering*, pages 175 – 182, 2013.

[17] Shlomo Berkovsky, Iván Cantador, and Domonkos Tikk. *Collaborative Recommendations: Algorithms, Practical Challenges and Applications*. 04 2019.

[18] Amiya Bhattacharya and Sajal Das. Lezi-update: An information-theoretic approach to track mobile users in pcs networks. 1999.

[19] Shen Bin, Zhang Guiqing, Wang Shaolin, and Wei Dong. The development of management system for building equipment internet of things. *2011 IEEE 3rd International Conference on Communication Software and Networks*, pages 423–427, 2011.

[20] Bosch. Bmp-085 digital pressure sensor. https://www.sparkfun.com/datasheets/Components/General/BST-BMP085-DS000-05.pdf, 2009. Accessed on 13.09.2018.

[21] Leo Breiman. Random forests. *Machine Learning*, 2001.

[22] D.E. Broadbent. Recent advances in understanding performance in noise. In *Proceedings 4th International Congress on Noise as a Public Health Problem*, 1983.

[23] Jr Brown, Jr Robert, James D Romanowiz, and Charles W Staples. Energy management and home automation system, 1996.

[24] Tiago O. Castro, Daniel F. Macedo1, and Aldri Santos. Controle de acesso iot escalável e ciente de contexto suportando múltiplos usuários. *Simpósio Brasileiro de Redes de Computadores*, page 14, 2018.

[25] Tianqi Chen and Carlos Guestrin. Xgboost: A scalable tree boosting system. In *Proceedings of the 22Nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 2016.

[26] CISCO-IBSG. The internet of things. how the next evolution of the internet is changing everything. 2011.

[27] Diane J. Cook, Manfred Huber, Karthik Gopalratnam, and Michael Youngblood. Learning to control a smart home environment. *In Innovative Applications of Artificial Intelligence*, 2003.

[28] Diane J. Cook, G. Michael Youngblood, Edwin O. Heierman III, Karthik Gopalratnam, Sira Rao, Andrey Litvin, and Farhan Khawaja. Mavhome: An agent-based smart home. In *PerCom*, pages 521–524. IEEE Computer Society, 2003.

[29] E.N. Corlett. Ergonomics and sitting at work. *Work*, pages 235–238, 2009.

[30] T. Cover and P. Hart. Nearest neighbor pattern classification. *IEEE Trans. Inf. Theor.*, 13(1):21–27, 2006.

[31] Jeong Dae-young. Cloud computing based smart office system and server for managing the same and method for managing the same, 2012.

[32] A. Datta, S. Sen, and Y. Zick. Algorithmic transparency via quantitative input influence: Theory and experiments with learning systems. In *2016 IEEE Symposium on Security and Privacy (SP)*, 2016.

[33] Alison de Oliveira Souza. Internet das coisas: Interação entre objetos para criar ambientes acessíveis, July 2018.

[34] Laila Matuck de Paula Reis. Controle de ambiente domótico à distância por meio de comandos de voz, July 2018.

[35] Huang Dechang, Li Bo, Zhang Yuejin, Yang Hui, Gao Xincheng, Li Zhenbang, Lin Hongcheng, Wang Yifan, and Liang Shu. Intelligent office desk, 2015.

[36] Ofer Dekel. From online to batch learning with cutoff-averaging. In D. Koller, D. Schuurmans, Y. Bengio, and L. Bottou, editors, *Advances in Neural Information Processing Systems 21*, pages 377–384. Curran Associates, Inc., 2009.

[37] Thomas G. Dietterich. An experimental comparison of three methods for constructing ensembles of decision trees: Bagging, boosting, and randomization. *Machine Learning*, 2000.

[38] R Djukanovic, P Wargocki, and Po Fanger. Cost-benefit analysis of improved air quality in an office building. In *Proc. of Indoor Air 2002, vol 1 , pp 808-813, Indoor Air 2002, Inc*, 2002.

[39] Elecrow. Mq-135 gas sensor. https://www.elecrow.com/download/MQ-135.pdf, 2018. Accessed on 13.09.2018.

[40] Anthony Fadell, Matthew Rogers, Edwin Satterthwaite, Ian Smith, Daniel Warren, Joseph Palmer, Shigefumi Honjo, Grant Erickson, Jonathon Dutra, and Hugo Fiennes. User-friendly, network connected learning thermostat and related systems and methods, 2013.

[41] Hongqing Fang and Jinjin Ruan. An improved position prediction algorithm based on active lezi in smart home. pages 1733–1736, 08 2012.

[42] Institute for Work & Health. At iwork. *Information on workplace research from the Institute for Work & Health*, 2003.

[43] T H E Free, Electron Laser, Power Source For, Publication Date, and Copyright Information. Control of temperature for health and productivity in offices. 2008.

[44] Yitzhak Fried, Samuel Melamed, and Haim A. Ben-David. The joint effects of noise, job complexity, and gender on employee sickness absence: An exploratory study across 21 organizations - the cordis study. *Journal of Occupational and Organizational Psychology*, pages 131–144, 2002.

[45] Adrian Furnham and Lisa Strbac. Music is as distracting as noise: the differential distraction of background music and noise on the cognitive test performance of introverts and extroverts. *Ergonomics*, pages 203–217, 2002.

[46] Karthik Gopalratnam and Diane J. Cook. Active lezi: An incremental parsing algorithm for sequential prediction. In *FLAIRS Conference*, pages 38–42. AAAI Press, 2003.

[47] Majorkumar Govindaraju, Arunkumar Pennathur, and Anil Mital. Quality improvement in manufacturing through human performance enhancement. *Integrated Manufacturing Systems*, pages 360–367, 2001.

[48] E. Gulian and J. R. Thomas. The effects of noise, cognitive set and gender on mental arithmetic performance. *British Journal of Psychology*, pages 503–511, 1986.

[49] Richard Hagale Anthony, Ernest Kelley Jason, and Rozich Ryan. Rfid smart office chair, 2005.

[50] Tin Kam Ho. Random decision forests. In *Proceedings of the Third International Conference on Document Analysis and Recognition (Volume 1)*. IEEE Computer Society, 1995.

[51] Honeywell. Hih-4030/31 series - humidity sensors. `https://www.sparkfun.com/datasheets/Sensors/Weather/SEN-09569-HIH-4030-datasheet.pdf`, 2008. Accessed on 13.09.2018.

[52] International Telecommunication Union (ITU). Itu-t y.2060 - overview of the internet of things. `http://www.itu.int/internetofthings`, 2018. Accessed: 2018-10-20.

[53] Richard J. Samworth. Optimal weighted nearest neighbour classifiers. *The Annals of Statistics*, 40, 2011.

[54] Jens Kober, J. Andrew Bagnell, and Jan Peters. Reinforcement learning in robotics: A survey. *The International Journal of Robotics Research*, 32(11):1238–1274, 2013.

[55] Jérémy Lapalu, Kevin Bouchard, Abdenour Bouzouane, Bruno Bouchard, and Sylvain Giroux. Unsupervised mining of activities for smart home prediction. *Procedia Computer Science*, 2013.

[56] Victor Lesser, Michael Atighetchi, Brett Benyo, Bryan Horling, Anita Raja, Régis Vincent, Thomas Wagner, Ping Xuan, and Shelley Xq. Zhang. The intelligent home testbed. 1999.

[57] Leviton. Leviton. https://www.leviton.com/en/products/dz6hd-1bz, 2018. Accessed on 13.09.2018.

[58] Stan Lipovetsky and Michael Conklin. Analysis of regression in game theory approach. *Applied Stochastic Models in Business and Industry*, 2001.

[59] Scott M. Lundberg, Gabriel G. Erion, and Su-In Lee. Consistent individualized feature attribution for tree ensembles. *CoRR*, 2018.

[60] J. Manyika. *The Internet of Things: Mapping the Value Beyond the Hype*. 2015.

[61] Yoky Matsuoka, E Astier, Frank, Rangoli Sharan, David Sloo, and Anthony Michael Fadell. Energy efficiency promoting schedule learning algorithms for intelligent thermostat, 2013.

[62] Midea. Midea. http://www.midea.com/global/products/air_conditioning/residential_air_conditioner/ac_split/, 2018. Accessed on 13.09.2018.

[63] Daniele Miorandi, Sabrina Sicari, Francesco De Pellegrini, and Imrich Chlamtac. Internet of things: Vision, applications and research challenges. *Ad Hoc Networks*, pages 1497–1516, 2012.

[64] Christoph Moeslinger, Thomas Schmickl, and Karl Crailsheim. A minimalist flocking algorithm for swarm robots. In *Advances in Artificial Life. Darwin Meets von Neumann*. Springer Berlin Heidelberg, 2011.

[65] Michael C. Mozer. The neural network house : An environment that adapts to its inhabitants. 2002.

[66] U.S. Department of Labor Occupational Safety and Health Administration. Ergonomics: The study of work. https://www.osha.gov/Publications/osha3125.pdf, 2000. Accessed on 25.10.2018.

[67] Romanelli Pat. Air temperature control system, 1972.

[68] Lu Peisen, Li Xin, Chen Bihao, Wei Xiaohui, and Yi Chunyang. Office and learning health care furniture combination capable of being intelligently controlled, 2013.

[69] Charith Perera, Chi Harold Liu, Srimal. Jayawardena, and Min Chen. A survey on Internet of Things from industrial market perspective. *IEEE Access*, pages 1660–1679, 2014.

[70] June J. Pilcher, Eric Nadler, and Caroline Busch. Effects of hot and cold temperature exposure on performance: a meta-analytic review. *Ergonomics*, pages 682–698, 2002.

[71] A. Pras and J. Schoenwaelder. Rfc3444: On the difference between information models and data models. http://tools.ietf.org/html/rfc3444, 2003. Accessed: 2018-09-05.

[72] Kevin L. Priddy and Paul E. Keller. *Artificial Neural Networks: An Introduction (SPIE Tutorial Texts in Optical Engineering, Vol. TT68)*. SPIE- International Society for Optical Engineering, 2005.

[73] Maxim Integrated Products. Max9812/max9813. https://datasheets.maximintegrated.com/en/ds/MAX9812-MAX9813L.pdf, 2014. Accessed on 13.09.2018.

[74] Weijun Qin, Qiang Li, Limin Sun, Hongsong Zhu, and Yan Liu. Restthing: A restful web service infrastructure for mash-up physical and web resources. pages 197–204, 2011.

[75] Sunil K Rao and Raman K Rao. Home automation and smart home control using mobile devices and wireless enabled electrical switches, 2014.

[76] Parisa Rashidi and Diane J. Cook. Keeping the resident in the loop: Adapting the smart home to the user. *IEEE Trans. Systems, Man, and Cybernetics, Part A*, 2009.

[77] Craig W. Reynolds. Flocks, herds and schools: A distributed behavioral model. *SIGGRAPH Comput. Graph.*, 1987.

[78] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. Why should i trust you?: Explaining the predictions of any classifier. In *Proceedings of the 22Nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD 16, pages 1135–1144. ACM, 2016.

[79] Daniele Riboni, Claudio Bettini, Gabriele Civitarese, Zaffar Haider Janjua, and Viola Bulgari. *From lab to life: Fine-grained behavior monitoring in the elderly's home*, pages 342–347. Institute of Electrical and Electronics Engineers Inc., 2015.

[80] Carlos Riquelme Ruiz and David Mata Valdes. Smart office desk interactive with the user, 2016.

[81] F. Rosenblatt. *The Perceptron, a Perceiving and Recognizing Automaton Project Para*. Report: Cornell Aeronautical Laboratory. Cornell Aeronautical Laboratory, 1957.

[82] Ran Roth and Omer Enbar. Method and apparatus for controlling an hvac system, 2015.

[83] Abhishek Roy, Soumya K. Das Bhaumik, Amiya Bhattacharya, Kalyan Basu, Diane J. Cook, and Sajal K. Das. Location aware resource management in smart homes. In *Proceedings of the First IEEE International Conference on Pervasive Computing and Communications (PerCom'03),March 23-26, 2003, Fort Worth, Texas, USA*, 2003.

[84] Nirmalya Roy, Abhishek Roy, Sajal Das, and Kalyan Basu. A cooperative learning framework for mobility-aware resource management in multi-inhabitant smart homes. In *The Second Annual International Conference on Mobile and Ubiquitous Systems: Networking and Services*, 2005.

[85] Nirmalya Roy, Abhishek Roy, and Sajal K. Das. Context-aware resource management in multi-inhabitant smart homes: A nash h-learning based approach. In *PerCom*. IEEE Computer Society, 2006.

[86] Ando Saabas. Tree interpreter. http://blog.datadive.net/interpreting-random-forests/, 2019. Accessed on 22.01.2019.

[87] Alok Saklani and Shweta Jha. Impact of ergonomic changes on office employee productivity. 2011.

[88] Shai Shalev-Shwartz and Shai Ben-David. *Understanding Machine Learning: From Theory to Algorithms*. Cambridge University Press, New York, NY, USA, 2014.

[89] Avanti Shrikumar, Peyton Greenside, and Anshul Kundaje. Learning important features through propagating activation differences. In Doina Precup and Yee Whye Teh, editors, *Proceedings of the 34th International Conference on Machine Learning*. PMLR, 2017.

[90] SmartThings. Smartthings arrival sensor (2015 model). https://support.smartthings.com/hc/en-us/articles/205382134-SmartThings-Arrival-Sensor-2015-model-, 2015. Accessed on 13.09.2018.

[91] SmartThings. Multipurpose sensor. https://www.smartthings.com/products/smartthings-multipurpose-sensor, 2018. Accessed on 13.09.2018.

[92] SmartThings. Smartthings hub. https://www.smartthings.com/products/smartthings-hub, 2018. Accessed on 13.09.2018.

[93] Soma. Soma simple smart home. https://www.somasmarthome.com/, 2018. Accessed on 13.09.2018.

[94] Steven L Sunyich. Personalized smart room, 2003.

[95] Serge Thomas Mickala Bourobou and Younghwan Yoo. User activity recognition in smart homes using pattern clustering applied to temporal ann algorithm. *Sensors (Basel, Switzerland)*, 2015.

[96] M.C Torrance. Advances in human-computer interaction: The intelligent room. *In Working Notes of the CHI 95 Research Symposium*, 1995.

[97] Ovidiu Vermesan and Peter Friess. *Internet of Things: Converging Technologies for Smart Environments and Integrated Ecosystems*. River Publishers Series in Communication. River, 2013.

[98] Erik Štrumbelj and Igor Kononenko. Explaining prediction models and individual predictions with feature contributions. *Knowl. Inf. Syst.*, 2014.

[99] Li Wei. Multifunctional intelligent office system, 2015.

[100] P. J. Werbos. *Beyond Regression: New Tools for Prediction and Analysis in the Behavioral Sciences*. PhD thesis, Harvard University, 1974.

[101] Hans Wigo and Igor Knez. Psychological impact of air velocity variations in a ventilated room. *Ergonomics*, pages 1086–1096, 2005.

[102] WSU-Energy-Program. Measuring carbon dioxide inside buildings – why is it important? 2013. Washington State University - Energy Program.

[103] S. Wu, J. B. Rendall, M. J. Smith, S. Zhu, J. Xu, H. Wang, Q. Yang, and P. Qin. Survey on prediction algorithms in smart homes. *IEEE Internet of Things Journal*, 4(3), 2017.

[104] Zhu Xiaojuan. An intelligent control office furniture combination of learning health, 2018.

[105] Yang Min Huang Yingsheng. Intelligent adjusting desk and control method thereof, 2015.

[106] ZWave. Smart home and home automation: what's the difference? https://www.z-wave.com/blog/smart-home-and-home-automation-what-s-the-difference, 2016. Accessed on 25.10.2018.

[107] Ding Zenghua, Zhu Xi, Dong Jinyu, and Chen Jinye. Intelligent office energy-saving control system, 2011.

[108] Yongjun Zhang and Tingting Qi. Research on mining association behavior of smart home users based on apriori algorithm. In *International Conference on Materials Engineering and Information Technology Applications (MEITA 2015)*, 2015.

[109] Zhi-Hua Zhou. *Ensemble Methods: Foundations and Algorithms.* Chapman & Hall/CRC, 2012.

[110] Jacob Ziv and Abraham Lempel. Compression of individual sequences via variable-rate coding. *IEEE Trans. Information Theory*, 24:530–536, 1978.

Appendix A

List of collaborators

Collaborator	**Academic Historical**[1]	**Contribution**
Guillermo Ponce	Computer Science Master Student	ManIoT refactoring, Presence Driver, SmartOffice Mobile Interface, Office automation (Orchestrator)
Vitor Reis	Electrical Engineering Undergraduate Student	Hardware support, data preprocessing
Gustavo Guedes	Electrical Engineering Undergraduate Student	Smart Switch Winet and drivers for SmartThings and Dimmer Leviton
Alison Souza	Computer Science Undergraduate Student	Blind People Assistant via Smartphone
Laila Matuck	Control and Automation Engineering Undergraduate Student	Alexa driver
Guilherme Jácome	Electrical Engineering Undergraduate Student	Drivers for Air Midea, Shade Soma, Arduino, Iris and ManIoT refactoring
István Leal	Electrical Engineering Undergraduate Student	Phillips and RFID drivers
Tiago Oliveira	Computer Science Master Student	Development of ManIoT arquitecture
Josué Batista	Computer Science Master Student	Creator of ManIoT

Table A.1: ManIoT Collaborators