CISTER
Research Centre in
Real-Time & Embedded
Computing Systems

BEng Thesis

Smart Maintenance of Home Appliances

Orientação científica: Luis Lino Ferreira, Coorientação: Cláudio Maia e Rafael Rocha

Rita Sousa

CISTER-TR-191214

Smart Maintenance of Home Appliances

Rita Sousa

CISTER Research Centre

Polytechnic Institute of Porto (ISEP P.Porto)

Rua Dr. António Bernardino de Almeida, 431

4200-072 Porto

Portugal

Tel.: +351.22.8340509, Fax: +351.22.8321159

E-mail:

https://www.cister-labs.pt

Abstract

In several countries around the world, home appliances have a minimal warranty of two years since the date they are bought. Due to their electromechanical nature, during its lifecycle several failures can occur, some are related to poor usage, others are related to the malfunctioning of components or even due to the natural wear and tear of certain parts. The project 1cSmart Maintenance in Home Appliances 1d (Smart-PDM) aims to prevent and detect home appliance failures by acquiring its energy consumption pattern and detecting variations or abnormal patterns. Any variation that is abnormal is a good indicator that something is not working properly, requiring external intervention in order to identify the failure and possibly repair it.The full project implementation is described in the report including the study done to learn the areas of Internet of Things and Machine Learning. An Android Application was developed to install the sensors on the personal Wi-Fi network and to view the energy consumption of home appliances. A Web Application was implemented with the main functions of storing the consumption information of the home appliances and invoking a Data Analysis component. This component, in addition to analyzing and processing the data, involved the implementation of Machine Learning techniques to predict or classify the health status of home appliances.Finally, the report presents the conclusions drawn essentially from the Machine Learning area, such as what is the best machine learning technique (namely, classification algorithms and pattern matching) for each situation.

Smart Maintenance of Home Appliances

CISTER - Research Centre in Real-Time and Embedded Computing Systems

2018 / 2019

1161227 Rita Sousa

Smart Maintenance of Home Appliances

CISTER - Research Centre in Real-Time and Embedded Computing Systems

2018 / 2019

1161227 Rita Sousa

Degree in Computer Engineering

September 2019

ISEP Advisor: **Luís Lino Ferreira**

External Supervisors: **Cláudio Maia, Rafael Rocha**

‹‹To my partner, my family and my big boss ››

Acknowledgments

Firstly, I have to thank Cláudio Maia, for always giving me the opportunity to learn something (in the most spectacular ways possible) and helping me to be "a top engineer", and Bruno Pinho for all patience, companionship, help and for always encouraging me not to give up.

Thanks a lot Luis Lino Ferreira for guiding me during this project, Rafael Rocha for all tireless support, discussions and disagreements (which helped me grow a lot) and everybody at CISTER for making my work environment fun, enjoyable and for always being available for help.

Thanks to everybody at CDIO, specially Margarida Guerra, and to my baboons for all the knowledge, friendship and distractions that they provided me during my course in Computer Engineering Degree making all of these possible or else I would have gone crazy.

Last but not least, I would like to thank my family for their undying support all my life!

Abstract

In several countries around the world, home appliances have a minimal warranty of two years since the date they are bought. Due to their electromechanical nature, during its lifecycle several failures can occur, some are related to poor usage, others are related to the malfunctioning of components or even due to the natural wear and tear of certain parts. The project “Smart Maintenance in Home Appliances” (Smart-PDM) aims to prevent and detect home appliance failures by acquiring its energy consumption pattern and detecting variations or abnormal patterns. Any variation that is abnormal is a good indicator that something is not working properly, requiring external intervention in order to identify the failure and possibly repair it.

The full project implementation is described in the report including the study done to learn the areas of Internet of Things and Machine Learning. An Android Application was developed to install the sensors on the personal Wi-Fi network and to view the energy consumption of home appliances. A Web Application was implemented with the main functions of storing the consumption information of the home appliances and invoking a Data Analysis component. This component, in addition to analyzing and processing the data, involved the implementation of Machine Learning techniques to predict or classify the health status of home appliances.

Finally, the report presents the conclusions drawn essentially from the Machine Learning area, such as what is the best machine learning technique (namely, classification algorithms and pattern matching) for each situation.

Keywords (Theme): Smart Maintenance, Internet of Things, Machine Learning.

Keywords (Technologies): ASP.NET, C, R, Python, KNIME.

Resumo

Em vários países do mundo, os eletrodomésticos têm uma garantia mínima de dois anos desde a data em que foram comprados. Devido à sua natureza eletromecânica, durante o seu ciclo de vida, várias falhas podem ocorrer, algumas relacionadas com o seu mau uso, outras relacionadas com o mau funcionamento de componentes ou mesmo devido ao desgaste natural de certas peças. O projeto "Manutenção Inteligente em Eletrodomésticos" tem como objetivo prevenir e detetar falhas de eletrodomésticos adquirindo seu padrão de consumo elétrico e detetando variações ou padrões anormais. Qualquer variação anormal é um bom indicador de que algo não está a funcionar corretamente, exigindo intervenção externa para identificar a falha e, se possível, repará-la.

A implementação completa do projeto é descrita no relatório, incluindo o estudo realizado para aprender as áreas da *Internet of Things* e *Machine Learning*. Uma aplicação Android foi desenvolvida para instalar os sensores na rede Wi-Fi pessoal e visualizar o consumo de energia dos eletrodomésticos. Uma aplicação Web foi implementada com as principais funções de armazenar as informações de consumo dos eletrodomésticos e chamar um componente de Análise de Dados. Esse componente, além de analisar e processar os dados, envolveu a implementação de técnicas de *Machine Learning* para prever ou classificar o funcionamento do consumo de eletrodomésticos.

Por fim, o relatório apresenta as conclusões tiradas essencialmente da área de *Machine Learning*, como qual é a melhor técnica de *Machine Learning* (algoritmos de classificação e reconhecimento de padrões) para cada situação.

Palavras-chave (Tema): Manutenção Inteligente, *Internet of Things*, *Machine Learning*.

Palavras-chave (Tecnologias): ASP.NET, C, R, Python, KNIME.

Table of Contents

Index of Figures

Index of Tables

Notation and Glossary

AI	Artificial Intelligence. Field of study where machines perform tasks like humans, making machines act with some intelligence.
API	Application Programming Interface. A set of clearly defined methods of communication between various software components.
CISTER	Research Centre in Real-Time and Embedded Computing Systems.
DL	Deep Learning. An approach to Machine Learning which implements Artificial Neural Networks to learn and make intelligent decisions.
GUI	Graphical User Interface.
IoT	Internet of Things. A network of objects (such as sensors) that can capture data autonomously and self-configure intelligently based on physical world events, allowing these systems to become active participants in various public, commercial, scientific, and personal processes.
ML	Machine Learning. Field of study that allows software applications to become more accurate in predicting outcomes without being explicitly programmed.
MQTT	Message Queuing Telemetry Transport. A messaging protocol for machine-to-machine communication.
REST	REpresentational State Transfer or RESTful Web services. Way of providing interoperability between computer systems on the Internet.
Smart-PDM	Smart Predictive Maintenance.
UI	User Interface.

1 Introduction

This section is divided into three different sections: Section 1.1 gives the project context; Section 1.2 explains what the problem is, presents the goals that would be achieved and its approach; Section 1.3 shows how is it organized the rest of the report.

1.1 Project Context

This project is performed in the context of the Smart-PDM [1] project which has the goal to perform the remote maintenance of home appliances in near real-time. This project is part of a vast portfolio of projects developed at CISTER's research unit.

CISTER focuses on real-time and embedded computing systems, contributing with seminal research works in programming paradigms, modelling and analyzing temporal behavior, resource management in energy-aware computation, among other subjects. Over time, CISTER has been developing projects in Predictive Maintenance where the objective is mainly to detect failures on devices and even predict when some of these failures will happen, thus scheduling adequate maintenance before that failure happens, like in the MANTIS project [2].

The student's motivation to accept this challenge came from the fact that this project can be a vital contribution to society, since it allows end users to save money on maintaining their home appliances, gives important contribution to the implementation of a the circular economy concept and also because it is a challenging project that uses technologies such as Internet of Things (IoT) and Machine Learning (ML).

1.2 Problem Description

The problem to be solved in this project consists in the identification of patterns and potential failures in home appliances with the help of IoT devices capable of acquiring the energy consumption of home appliances (which will hereafter be referred by its Smart-PDM name, that is Smart Connector) and technologies like Machine Learning. Smart Connectors are devices that can acquire energy consumption data and send this data to cloud-based systems using a communication protocol. The work described in this report fits in the scope of the Smart-PDM project. Thus, Figure 1 displays an overview of the main components[1] that are part of the Smart-PDM project, which are described next:

[1] In Section 1.2, main components are written in italic for easier comprehension and relation between the problem description and Figure 1.

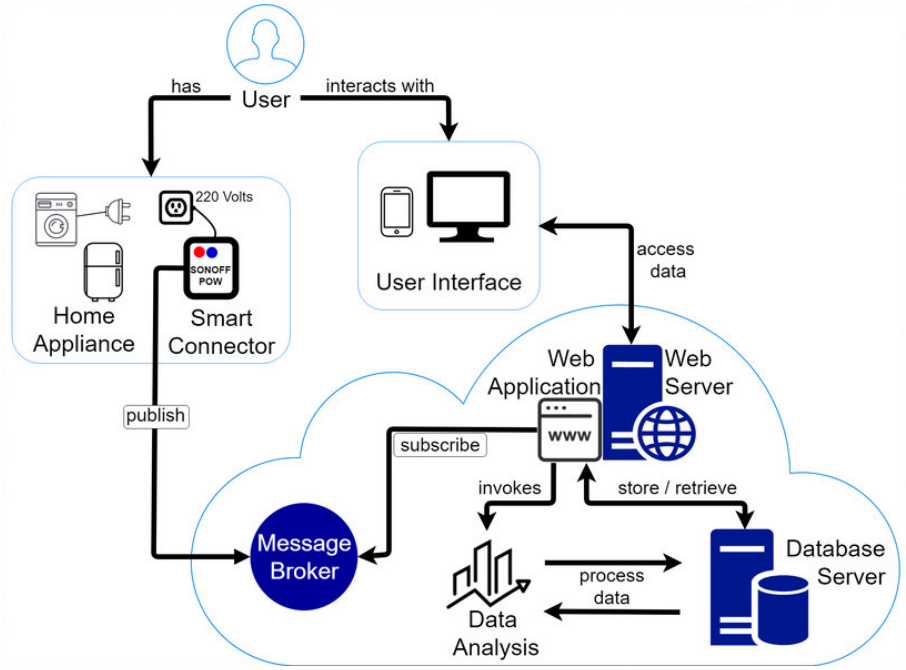

Figure 1 - System diagram of Smart-PDM's project

- *User* – A home appliances' user which has *Home Appliances* and *Smart Connectors* and accesses home appliances' data through a *User Interface*.
- *User Interface* - Can be accessed by the *User*, in any occasion, to see the consumption of his *Home Appliances*, configure or install the *Smart Connector* and receive notifications if one or more of his home appliances are malfunctioning, via a web or mobile application.
- *Home Appliance* - Can be plugged to a *Smart Connector* and the *Smart Connector* plugs directly into a power socket.
- *Smart Connector* – Device which senses *Home Appliances*' consumption and send (i.e. publish) that information to *Message Broker*.
- *Message Broker* – Implements a specific communication protocol and manages the message flow between publishers (i.e. the *Smart Connector*) and subscribers (i.e. the *Web Application*) – detailed in Section 2.1.2.
- *Web Application* - Hosted in a *Web Server*, it is the one who receives the messages published by the *Smart Connector* and stores them on database. At the appropriate times, invokes the *Data Analysis* component.
- *Data Analysis* - Component using Machine Learning algorithms, with the purpose of identifying a *Home Appliance*'s energy consumption patterns, and later, detect anomalies determining if the *Home Appliance* is functioning properly. The data stored in the *Database Server*, is processed in this component and after is being conveniently treated it is saved in database again.

1.2.1 Goals

The overall goal for the project described in this report is the creation of a system that can predict and detect failures based on the energy consumption pattern of home appliances and notify a failure to the end user and/or the maintenance provider.

In order to fulfill the student's goals, it was necessary to update some of the functionalities already prototyped which are presented in goals 1 and 2 (described in Section 3). The main goals are described in 3 and 4 (described in Section 4).

1. Acquire a home appliance's power consumption through a Smart Connector to the cloud;
2. Develop interfaces, i.e. a *Web Application* that can be used as a cloud service, providing supporting features (e.g. showing home appliances' consumption patterns), storing consumption information;
3. Develop a framework to detect home appliances failures, based on Machine Learning techniques, which would be able to discover patterns in a home appliance's consumption, and later, detect anomalies in near real-time;
4. Allow the system to monitor the home appliances by comparing new data with the learned consumption patterns.

The student's work fits the Smart-PDM project context, specifically targeting the *Data Analysis* component since the problem to be solved is related to the application of ML algorithms. Additionally, it may be needed to reconsider other components as part of the changes that are necessary for the inclusion of *Data Analysis*. For instance, it may be needed to modify the *Web Server*, *Database Server* and the *Message Broker*. The Smart-PDM project had a non-relational database and it was only possible to flows messages in the same network. Since the project needs a *Data Analysis* component, there were advantages in construct a relational database and flow messages remotely to get consumption information's from different Wi-Fi networks.

1.2.2 Contributions

The realization of Smart-PDM project will help users detect and predict failures, thus hopefully prolonging the useful life of home appliances, and consequently reducing equipment costs as it is usually more expensive to replace the whole equipment than a single component.

The main contributions are:

- Reduction on the energy consumption;

- Reduction on labor costs associated with maintenance;
- A system capable of communicating with the *Smart Connectors* on multiples houses;
- Framework capable of evaluating the health status of *Home Appliances*, by monitoring them and detecting failures.
- The avoidance of electronic waste, thus contributing to the circular economy [3] which turns down air pollution, water pollution, soil pollution, information security, and even human exploitation, at an environmental level.

1.2.3 Approach

The approach of this project is somewhat conditioned since other decisions have already been made and implemented[2] (more details in Section 3.1). For instance:

- The communication protocol to transfer the consumption's information between the *Smart Connector* and *Web Application* (the protocol is Message Queuing Telemetry Transport, MQTT);
- The *Smart Connector* to sense the *Home Appliance's* consumption (the smart connectors are Sonoff Pow version 2.0 and Sonoff Pow R2 [4]);
- The mobile application to configure the *Smart Connector* to a Wi-Fi network was already chosen as well.

Another restriction of this project is the exclusive use of open-source tools, which is a requirement of the project partner. Considering the goals, the approach followed in this project starts with the acquisition of energy consumption, through the *Smart Connector*. Then, as the *Smart Connector* collects information, it needs to send it for further processing. This dictates the next phase of the project which involves the data transition from the *Smart Connector* to the *Web Application* and the data storage in the *Web Application*. After storing the consumption data, the data processing is made using the *Data Analysis* component. This analysis phase not only processes the data, but also employs several algorithms and selects the one with the best statistical performance to detect consumption patterns and/or possible failures.

1.2.4 Work planning

For work planning, it was implemented an Agile methodology, namely, Scrum. Every week, it is presented the tasks accomplished or not, defined in each sprint. Daily scrum was also made to synchronize the status of the sprint tasks or to plan the workday.

[2] Even though it was already predefined what would be used, other hypotheses on Smart Connector and Communication Protocols themes were studied (more details in Section 2.1).

In Figure 2 is presented a Gantt's chart with the tasks performed during the internship.

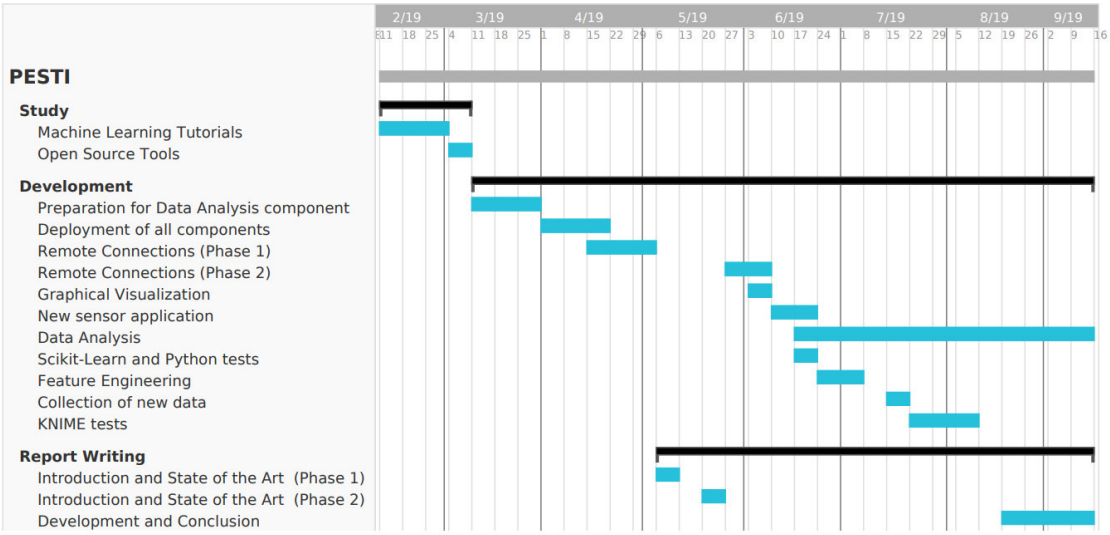

Figure 2 - Gantt chart [5]

1.3 Report structure

This section presents the report's structure and a small description of each chapter's contents.

1. Introduction: Provides a general perspective of the problem under study while also describing the project and student's goals, contributions, approach and work planning.
2. State of the art: Describes the current technologies used to solve the problem statement. Internet of Things and Machine Learning are the main topics of this chapter. Related projects mention documents of what is currently being explored in IoT field, and Machine Learning subchapter describes existing technologies with a focus on open source solutions.
3. Preparation for Data Analysis component: Presents the work to accomplish goals 1 and 2, i.e. the updates made to the existing prototype and its deployment.
4. Data analysis of consumption patterns: Shows the development to accomplish the goals 3 and 4, i.e. the analysis made to the home appliances data consumptions and the experiments/tests made to discover the best the ML algorithms to use.
5. Conclusions: This chapter presents the conclusions regarding the project highlighting the strengths and shortcomings of the solution, are referred to and setbacks are justified, and approaches and solutions are suggested for possible future work on the system.

2 State of the art

This state of the art is divided by topics, according to the scope of this project, namely Internet of Things for the Smart Connector and Message Broker components, and Machine Learning for the Data Analysis component. Each of these topics is covered in the next sections.

2.1 Internet of Things

Several researchers use a common definition for Internet of Things which is "a dynamic global network infrastructure with self-configuring capabilities based on standard and interoperable communication protocols where physical and virtual 'Things' have identities, physical attributes, and virtual personalities and use intelligent interfaces, and are seamlessly integrated into the information network" [6] [7] [8]. US National Intelligence Council, considered IoT as one of the six "Disruptive Civil Technologies" [8], meaning that IoT has potential impacts, a fact that is being confirmed by the increasing numbers of real applications [7] and the forecast of billions of connected devices to the internet by 2020 [8].

IoT has a few perspectives to create three, four or five-layered architectures based on technology, technical requirements and business needs. From a functionality perspective, a four-layered architecture seems the best architecture, with interface, service, networking and sensing layers [7], as can be seen in Figure 3.

Interface layer provides interaction methods to users and other applications since many devices are from different manufacturers/vendors who cannot follow the same standards/protocols.

Service layer can be able to identify common application requirements and provide APIs and protocols to support required services, applications, and user needs. This layer also processes all service-oriented issues, including information exchange and storage, data management, search engines, and communication.

Networking layer allows wireless or wired connections to share information or transfer data between the things connected to each other. Section 2.1.2 presents several communication protocols which can be used in this project.

Sensing layer senses/controls the physical world, i.e. smart plugs, which are equipped with intelligent sensors or radio-frequency identification and can be connected and controlled remotely. In Section 2.1.1 the specifications of the sensor required to this project is presented.

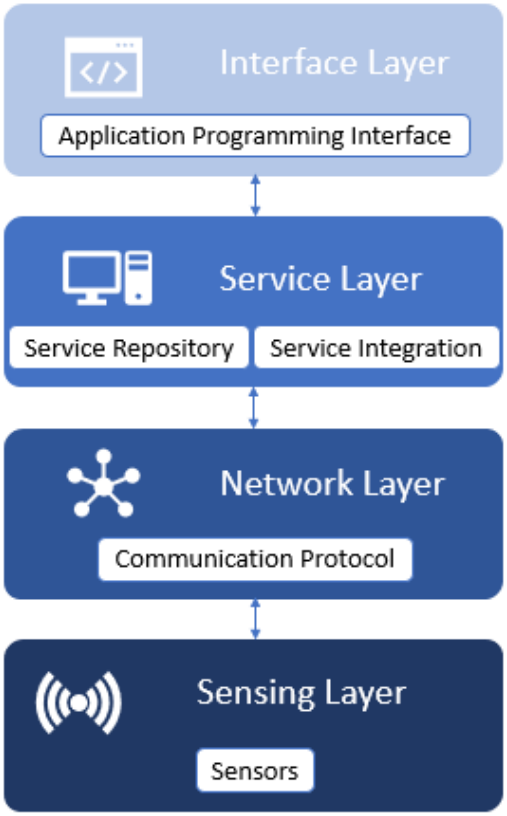

Figure 3 - IoT architecture with four layers [7]

2.1.1 Smart Plug

IoT technologies enable support the operation of a smart plug capable of measuring and communicate a home appliance's power consumption over the internet. In the market we can find several devices with this kind of capabilities.

Cloogy [9], is a technology developed by Virtual Power Solutions, capable of measuring the power consumption of home appliances, communicating to Cloogy gateway, using 802.15.4 protocol, which then connects to the internet. It enables real-time communication with the cloud and the user. And, it is also possible to turn on or off his associates home appliance.

Kisense [10] is basically an industrial grade solution of Cloogy, targeted to the industrial and business sector, it is capable of monitoring, anytime and anywhere, energy consumption to support people and their organizations in reducing energy consumption, targeted for the business sector and decrease its associated costs.

Eyedro Home Eletricity Monitors [11] helps with commitment to energy reduction where end users can see when home appliances are using the most electricity and which ones are the culprits, in real-time.

Wi-Fi Smart Plug with Energy Monitoring [12] has few features, such as, it controls devices connected to the Smart Plug, schedules the Smart Plug to automatically power electronics on and off, analyzes a device's real-time and historical power consumption.

The most similar plug to the Smart-PDM project is Sense Home Energy [13]. Sense consists in a device installed on the home's electrical panel and provides information on energy consumption through iOS, Android and web applications. Sense identifies patterns in energy consumption to help the user be more efficient, informed and secure. The advantage of Smart-PDM over Sense is that it does not require as many minimum installation and usage requirements and the cost will hopefully be less.

In an extensive list of smart plugs, for this project, Sonoff Pow version 2.0 and, later, Sonoff Pow R2 were chosen [4]. Both support 802.11 b/g/n wireless frequencies [14], a maximum current of 16 Amperes, a maximum power of 3500 Watts, a voltage range between 90 and 250 Volts Alternate Current input and a temperature range between 0ºC and 40 ºC.

The Sonoff Pow contains an important component in its circuitry for the power monitoring, namely HLW8012 (for version 2.0) and CSE7766 (for R2). Both sensors measure the current and the voltage and can output values of active power (W), voltage (V), current (A), apparent power (VA) and power factor (%) which demonstrates that it is enough for the purpose of this project since it only needs electrical current, active power value. For the programming of the Sonoff, Arduino IDE was installed with ESP8266 core and the HLW8012 and CSE7766 libraries were imported. The eWelink firmware and Android/iOS application [15] can be used as a remote control to turn on/off the device, monitors energy usage, shows historical energy consumption, act as a programmable timer, and more. However, the Smart-PDM project includes a proper mobile application where it is possible to do the requests necessary that are the addition of a home appliance and, eventually, the visualization of its consumption.

2.1.2 Communication Protocols

The development of an IoT network requires the selection of protocols that are efficient in transmitting data as well as having non-functional properties such as reliability, security, among others.

The most used protocols on IoT networks are built upon TCP/IP protocols guaranteeing the orderly delivery of all packets between devices [16] [17]. There are numerous protocols capable of handling the efficiency in transmitting data. For this project the following protocols have been chosen mostly due to their market acceptance: MQTT (Message Queue Telemetry Transport), CoAP (Constrained Application Protocol), AMQP (Advanced Message Queuing

Protocol), REST (Representational State Transfer), HTTP (Hyper Text Transport Protocol) and DDS (Data Distribution Service). Table 1 contains a summary description of the five protocols, which contains the protocol's architecture, network communication model (abstraction), transport protocol, the number of levels of Quality of Service (QoS)[3] (if it is supported), security protocol and if it conforms to the REST architectural style. These characteristics were chosen since they identify focal points of each protocol that are significant for the project context.

Table 1 - Comparison between MQTT, CoAP, AMQP, REST HTTP e DDS protocols [18] [19]

	MQTT	CoAP	AMQP	REST HTTP	DDS
Architecture	Client/Broker	Client/Server Client/Broker	Client/Server Client/Broker	Client/Server	Client/Broker
Abstraction	Publish/Subscribe	Request/Response Publish/Subscribe	Request/Response Publish/Subscribe	Request/Response	Publish/Subscribe
Transport	TCP	UDP	TCP	TCP	TCP/UDP
QoS	3 levels	Limited	3 levels	-	Extensive
Security	TLS/SSL	DTLS	TLS/SSL	TLS/SSL	TLS/DTLS/DDS
RESTful	No	Yes	No	Yes	No

From these protocols, one had to be selected for the Smart-PDM. The protocol was selected based in comparisons between message size and overhead, bandwidth and latency, reliability/QoS and security and Machine-to-Machine(M2M)/IoT usage.

The message size required for the Smart-PDM is short, around 105 bytes, and message overhead should be lower since it is necessary to receive messages at least every two seconds, although the target is to reach one message every 30 milliseconds. Due to the high frequency of incoming messages, low bandwidth consumption and latency are required. The reliability and the QoS are very important because without reliable data, it is more difficult to predict the home appliance's operation, so it is necessary high reliability/QoS. Additionally, since the data being transmitted has privacy issues a certain level of security is required, in order to ensure that the data has not been altered and that data cannot be analyzed by third parties.

According to papers [18], [19], [20], [21], [22], [23] and the Table 1 and Table 2, it was possible to conclude that MQTT is the most appropriated protocol. In Table 2, the protocols are sorted in ascending order according to each corresponding parameter (i.e., CoAP has the lowest message size, while HTTP has the highest). As such, the reasons for why MQTT was chosen are presented:

[3] A QoS level defines the guarantee of delivery for a specific message depending on the level chosen

- MQTT has a limitation on header size however it does not contribute to message overhead;
- Although CoAP presented lower latency and bandwidth, TCP was valued instead of UDP transport protocol due to its guarantee to orderly deliver all packets. So, as CoAP only uses UDP, it was set aside. Furthermore, CoAP is mostly suited for client-server communication for transferring state information like in REST/HTTP, so it is not purely event based like MQTT which is suited for many-to-many communication between multiple clients;
- DDS has higher reliability than MQTT since it has an extensive amount of QoS parameters against MQTT's three QoS levels, however it was easier to implement MQTT than DDS and open source DDS is limited in security and is quite expensive, so DDS was set aside too;
- Among the evaluated protocols, MQTT is the poorest in terms of security, since it does not include any security protocol by default. However, TLS/SSL encryption can be later used in conjunction with MQTT, providing data integrity. This means that TLS assures that the data preserves its accuracy and consistency [24];
- Lastly, MQTT is the most used in M2M and IoT projects by many organizations, such as IBM and Facebook.

Table 2 - Comparison of parameters between MQTT, CoAP, AMQP, REST HTTP e DDS protocols [18] [19]

Parameter	Lower	←			Higher
Message Size/Overhead	CoAP	MQTT	DDS	AMQP	HTTP
Bandwidth/Latency	CoAP	MQTT	AMQP	DDS	HTTP
Reliability/QoS	HTTP	CoAP	AMQP	MQTT	DDS
Security	MQTT	CoAP	HTTP	AMQP	DDS
IoT usage	HTTP	DDS	CoAP	AMQP	MQTT

In the MQTT protocol, the MQTT server is called a "broker" that handles the publishing/subscribing of actions to the target topics. MQTT clients are the connected devices. The process of sending messages is called publishing, and to receive messages an MQTT client must subscribe to a topic [25], see in the Figure 4. One of the most popular brokers that implement the MQTT protocol is the Mosquitto broker [26]. Mosquitto is an open source message broker that implements the MQTT, it's lightweight and suitable for Internet of Things messaging. Figure 3 shows a MQTT Client (*Smart Plug*) publishing a message (*75W*) with a topic (*consumption*) to the MQTT Broker and another MQTT Client that subscribes to the MQTT Broker with a certain topic (*consumption*) which waits for the published message/s (*75W*).

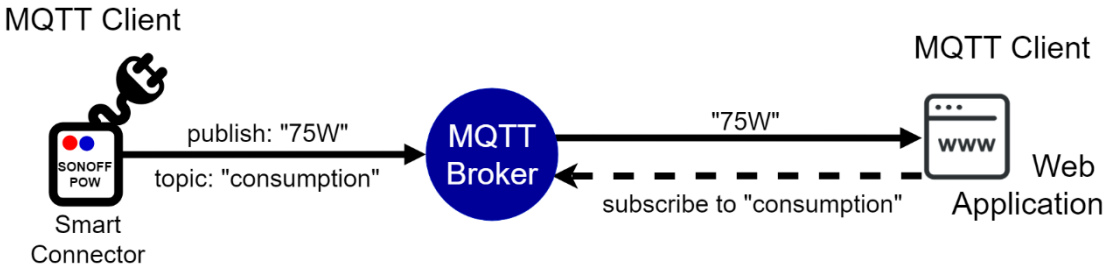

Figure 4 - MQTT protocol

In this context, the Figure 4 represents the process to get data through the smart connector and save it in the database for further analysis.

2.2 Machine Learning

Machine Learning is a field of Artificial Intelligence, as seen in Figure 5, devoted to the study of algorithms that parse data, learn from it, and then decide or predict something from the data [27]. Artificial Intelligence is a field of study where machines perform tasks similarly to humans, making machines act with some intelligence. Deep Learning is an approach to ML which implements Artificial Neural Networks to learn and make intelligent decisions (see Figure 5).

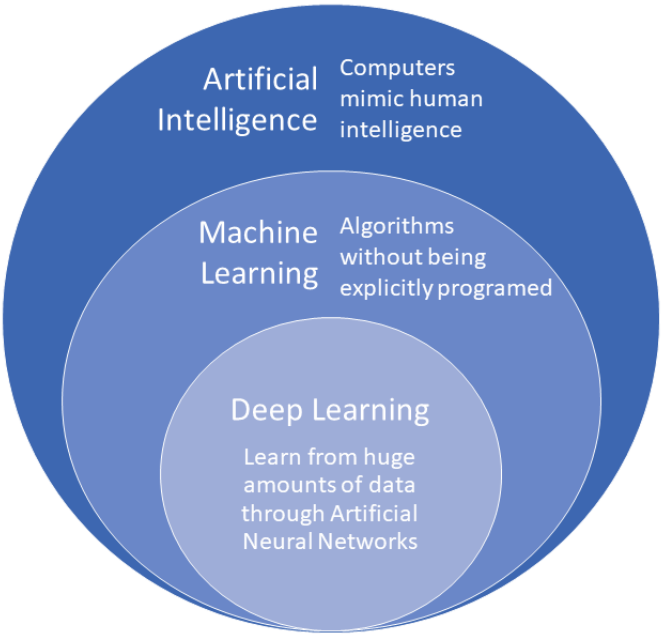

Figure 5 - Artificial Intelligence, Machine Learning and Deep Learning at a glance [28] [29]

Andrew Ng, chooses other definitions for Machine Learning in his ML course (available in Coursera [30]), referencing Arthur Samuel for his older definition, “Field of study that gives computer the ability to learn without being explicitly programmed”, and Tom Mitchell for his newer definition, “A computer program is said to learn from experience E with respect to

some class of tasks T and performance measure P, if its performance at tasks in T, as measured by P, improves with experience E".

In Figure 6, it is possible to see different types of ML algorithms which are Supervised, Unsupervised, Semi-supervised and Reinforcement Learning.

In Supervised Learning (SL) the data is labeled, i.e. the correct output is already known, and the algorithms learn to predict data from the input data [31].

Otherwise, in Unsupervised Learning (UL), all data is unlabeled, and the algorithms learn to inherent structure from the input data [31].

Semi-Supervised Learning (SSL) sits between supervised and unsupervised learning, i.e. some data is labeled but most of it is unlabeled [31].

Reinforcement Learning (RL) enables an agent to learn in an interactive environment by trial and error using feedback from its own actions and experiences [32].

Each of these types has, in the white rectangles, two examples of real-life use-cases (e.g. Supervised Learning can be used for predicting house prices or tumors).

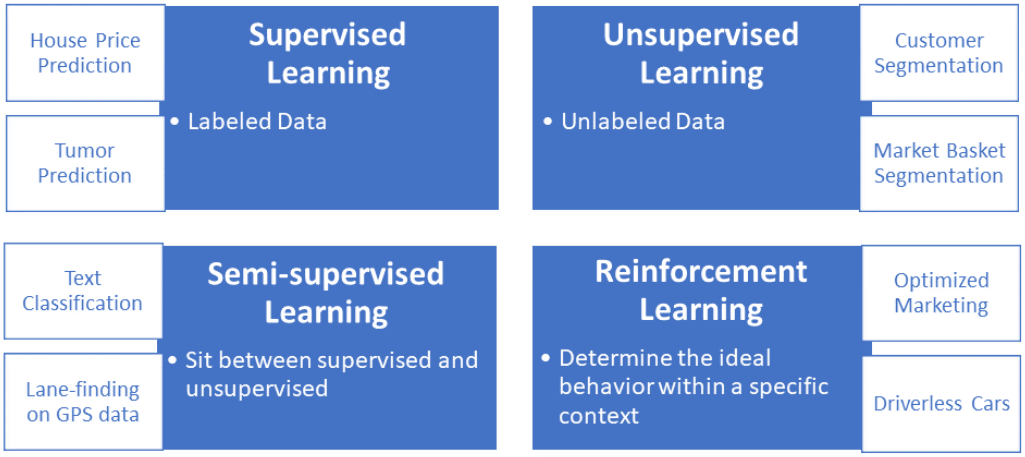

Figure 6 - Types of Machine Learning's Algorithms [33] [34]

The project's requirements only need supervised and unsupervised learning knowledge which will be explained in the Sections 2.2.1 and 2.2.2, respectively. Section 2.2.3 presents the Neural Networks algorithm due to the fact that it can be either supervised, unsupervised or semi-supervised, depending on the data, i.e. if it is either labeled, not labeled, or partially labeled [34] [35].

There are a few steps to be followed when choosing an ML algorithms [29] [36]. Firstly, it is necessary to pick one or a list of algorithms which, theoretically, it will resolve the problem and then evaluate it with the most appropriate metrics and optimize it.

- Whichever algorithm is chosen, after acquiring a dataset, the goal is to achieve a function h called, for historical reasons, hypothesis. For every input value, this function will predict an output value, for example, in house price prediction, giving the living area it predicts its price or in tumor prediction, given the tumor's size it predicts if it is malign or benign.
- After having a model representation, or in other words a good hypothesis, then for the evaluation, there are multiple metrics such as accuracy/error rate, precision and recall, squared error, likelihood, F1 score, cost, among others [37], to measure the performance of the algorithm, i.e. if the hypothesis is predicting well. While not all will be addressed, most of them can be measured through a confusion matrix presented in Figure 7. Table 3 describes the formula and the definition of four metrics (accuracy, precision, recall and F1 score), and the result of applying the formula to the values presented in Figure 7.

N = 200		**Predicted**	
		Positive	Negative
Actual	Positive	True Positive (TP) **100**	False Positive (FP) **10**
	Negative	False Negative (FN) **5**	True Negative (TN) **85**

True Positives: data points labeled as positive that are actually positive
False Positives: data points labeled as positive that are actually negative
True Negatives: data points labeled as negative that are actually negative
False Negatives: data points labeled as negative that are actually positive

Figure 7 - Confusion matrix

Table 3 - Performance metrics

Metrics	**Accuracy**	**Precision**	**Recall**	**F1 score**
Formula	$\frac{TP+TN}{N}$	$\frac{TP}{TP+FP}$	$\frac{TP}{TP+FN}$	$2 * \frac{Precision * Recall}{Precision + Recall}$
Result	$\frac{100+85}{200} \approx 0.925$	$\frac{100}{100+10} \approx 0.909$	$\frac{100}{100+5} \approx 0.952$	$2 * \frac{0.909 * 0.952}{0.909+0.952} \approx 0.465$
Definition	the degree to which values were well identified	the degree to which values were well identified in the actual values	the degree to which values were well identified in the predicted values	single metric that combines recall and precision using the harmonic mean (a method for measuring the mean)

- Afterwards, the model is optimized based on performance metrics to reach higher efficiency/successes rate. This optimization process has different methods, most of which are presented in Figure 8. The most used optimization method is gradient descent [30]. Despite the advantages of the other methods not needing to manually pick variables and often being faster than gradient descent, these have the disadvantage of being more complex.

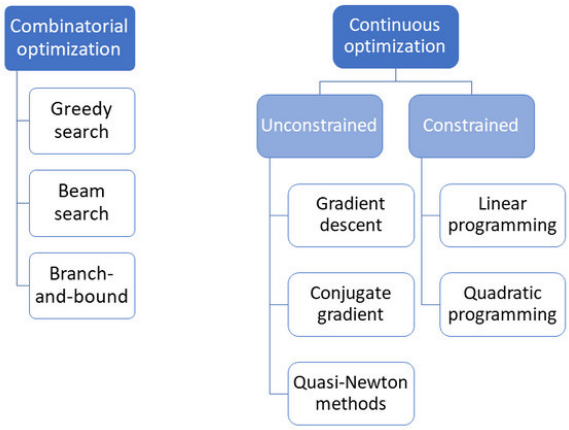

Figure 8 - Methods of optimization [36]

After completing these three steps, it is possible to compare algorithms and choose one in the immense variety of learning algorithms available.

2.2.1 Supervised Learning

Supervised Learning contains a set of algorithms where the data is labeled, that is, every input value has a desired output value, hence the model is predictive because it already knows what the correct output should look like.

There are two types of problems, regression and classification. Regression has a continuous output (e.g. house price prediction) while classification has a small number of discrete outputs (e.g. tumor prediction, is it malign or benign).

Some of the supervised algorithms are Linear Regression, Support Vector Machine and Decision Tree [30] [38]. These algorithms can be more appropriate for classification or regression problems, but, in the majority, they are not specifically for one of those two types of problems.

Linear Regression is a typical regression problem where the core idea is to find a linear function that, for the maximum independent data values, predicts the dependent values with the lowest possible error. In this case, the error is the distance between the data point (the blue points in Figure 9) and the hypothesis (the red line in Figure 9).

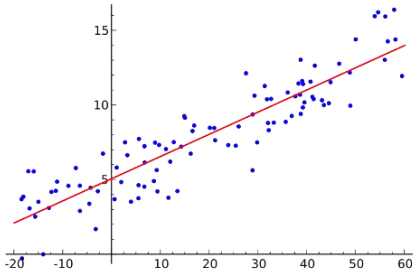

Figure 9 - Linear Regression Algorithm [39]

Support Vector Machine (SVM), as shown in Figure 10, is mainly used for classification problems where the classification is separated by a hyperplane (the red line) with the largest margin possible (the yellow-shaded area Figure 10). The margins (the dashed lines), also known as support vectors, are defined so that the distance between the different classified data (the blue and green points Figure 10) is maximum, minimizing the classification error.

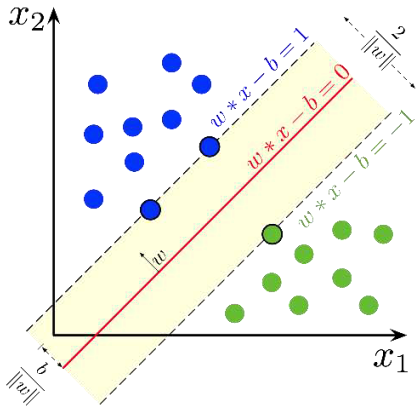

Figure 10 - Support Vector Machine Algorithm [40]

Decision Tree is often used for classification problems. It consists in splitting the data based on the features[4] (which, in the example presented in Figure 11, are: “outlook”, “humidity” and “windy”), and concluding an outcome that can be a categorical or continuous value (in the case of Figure 11, categorical: “play” or “don’t play”).

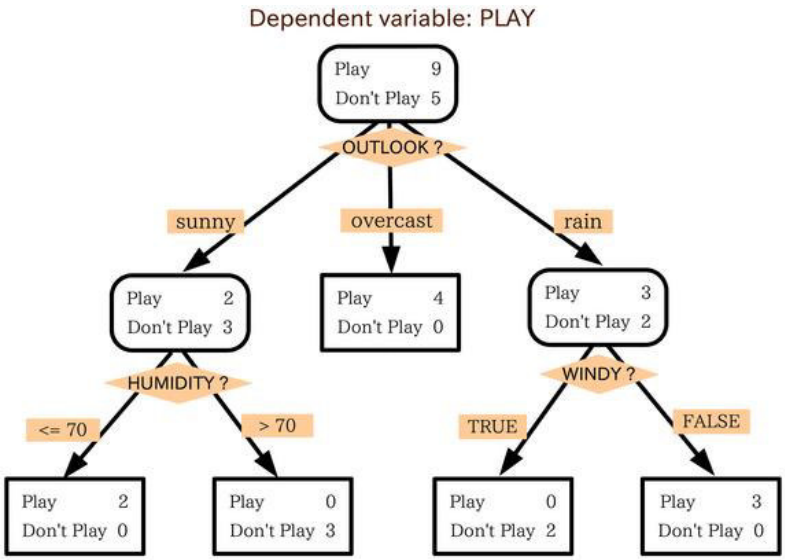

Figure 11 - Decision Tree Algorithm [41]

[4] A feature is a measurable variable for detecting data patterns by a Machine Learning algorithm. Selecting the best features will lead to the best predictions of an outcome.

2.2.2 Unsupervised Learning

Since unsupervised learning deals with unlabeled data, this type of algorithm mines for rules, detects patterns and groups the input values, i.e., clusters the data. There are also two types of problems in unsupervised learning: clustering and non-clustering, also known as association rule learning. The most used algorithms are K-Means Clustering [42] and the set of algorithms of association rule learning such as Apriori, Eclat and FP-growth [31].

K-Means Clustering clusters the data that have identical characteristics. The inputs for this algorithm are the dataset and k clusters, where k is a number, which is the reason why the algorithm is called K-Means. Then, k points are randomly initialized, called cluster centroids. For each data value, the distance to the cluster centroids is calculated, with the data point being allocated to the centroid with the lowest distance. Afterwards, the average of the location of all data values attributed to one centroid is calculated, and finally the cluster centroid is then moved to the averaged location. This process is easier to understand with the support of Figure 12.

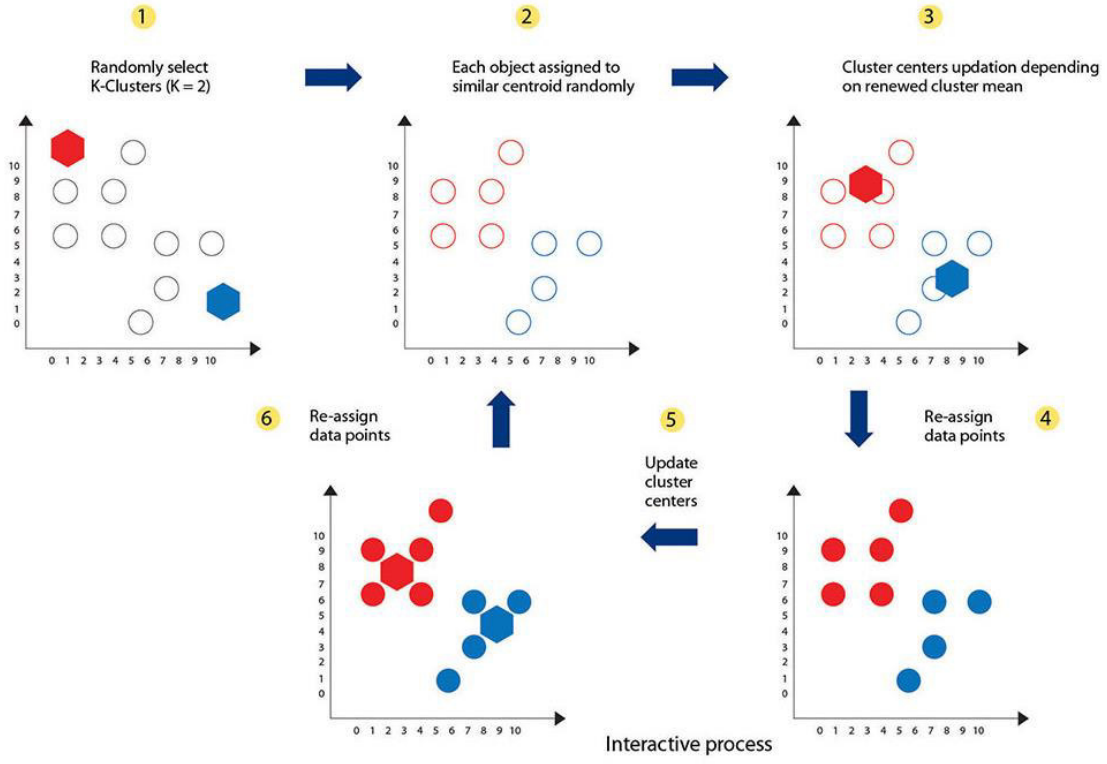

Figure 12 - K-Means algorithm process [43]

2.2.3 Neural Networks

Neural Networks is used in problems like self-driving cars, image recognition software or recommender systems. Neural Networks, also referred as Artificial Neural Networks, try to

mimic the brain, working like the human nervous system. The left side of Figure 13 shows a neuron, while the right side shows a neural networks representation. A neuron works as follows: dendrites take the input from other neurons; the axon generates inferences from the inputs and transmits the inferences over a distance; finally, the axon terminals transmit outputs [44].

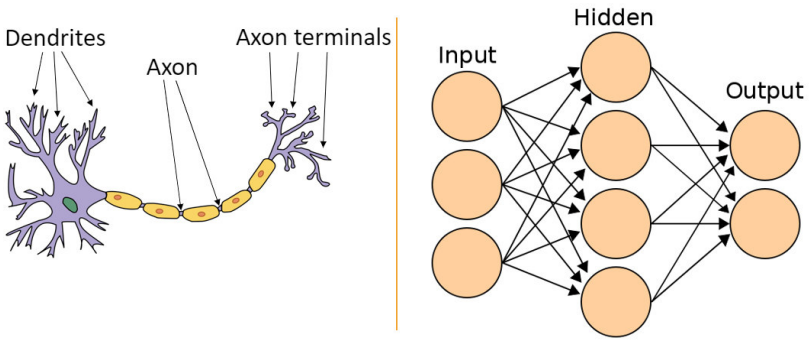

Figure 13 - Neuron representation and Neural Networks algorithm representation

In the supervised Neural Networks, the output layer of the input is already known. Comparing the neuron with the representation of the neural networks' algorithm, the dendrites are like the input features and the axon terminal would be the equivalent of the hypothesis function. The axon will be the hidden layer that takes the features, applies its weights, adds biases, applies the activation function, computes the composite prediction or probabilities and then passes the data to the next layer, the axon terminal. The process described is also known as feed forward.

Meanwhile in unsupervised learning, the Neural Networks categorizes the data regarding inputs received and groups them. Hence, the algorithm must discover the patterns and the features from the input data.

Artificial Neural Networks can be divided in feed forward and in feed backward also known as backpropagation [45]. In backpropagation, the neurons which are contributing more to the error are minimized. So, after one iteration of feed forward, optimization functions can be applied to help with finding the weights which yield a smaller loss in the next iteration [46].

2.2.4 Important concepts

This section presents important concepts in ML that are arguably more important than selecting an algorithm, including what dangers to avoid and important issues to focus on.

Machine learning models need to generalize well to new cases that the model has not seen in practice. An algorithm can fit a training set well but does not mean it is a good hypothesis. It could overfit and, as a result, its predictions on the test set would be poor.

As can be seen in Figure 14, the blue and red points represent the training data and the black and green line represent two possible hypothesis functions. The green line would be an example of overfitting since it fits excessively the available data and does not generalize as the black hypothesis does.

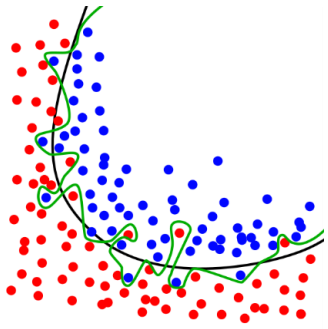

Figure 14 - Overfitting [47]

Usually, the initial dataset (to construct a model) is divided (randomly or not) in two portions, 70% to train the model/algorithm, the training set, and 30% to test the performance, the test set. In this way, the error of the hypothesis measured on the training set will be lower than the error on any other data set, therefore the training set's error is not a good predictor.

Another model validation technique is cross-validation. Cross-validation in practice consists in dividing 60% of the data set to training, 20% for validation and 20% to test randomly. The training data for a number of polynomial degrees optimizes the parameters of the hypothesis, i.e., the weight that each feature has; then in cross-validation, find the polynomial degree of hypothesis function with the least error; finally, estimate the generalization error with the test set, i.e. predict outcomes for unseen data.

Through the process described in the last three paragraphs, it is possible to know what problems the model/hypothesis is suffering from. The most known problem is overfitting. This can be solved if the bias and variance concepts are introduced, because they both give a very strong indicator for the changes needed to improve the algorithm.

Bias calculates the distance between the prediction values and the actual values. Variance tells how spread the data is, i.e., the variability of model prediction for a given data point. Figure 15 gives a better understanding of bias vs. variance, using an analogy with throwing darts at a board, where the black points represent prediction values and the dartboard's bullseye represents the actual values.

When variance is high, the model suffers from overfitting. In contrast, when bias is high, it means that the model suffers from underfitting. While overfitting pertains to the model

following excessively its training data and not being able to generalize, underfitting is the opposite, i.e. the hypothesis oversimplifies the data and predicts values poorly. Using cross-validation, it is possible to know that the bias is high when the training and validation errors are high, while if it has low training error and high validation error then it means that the model has high variance.

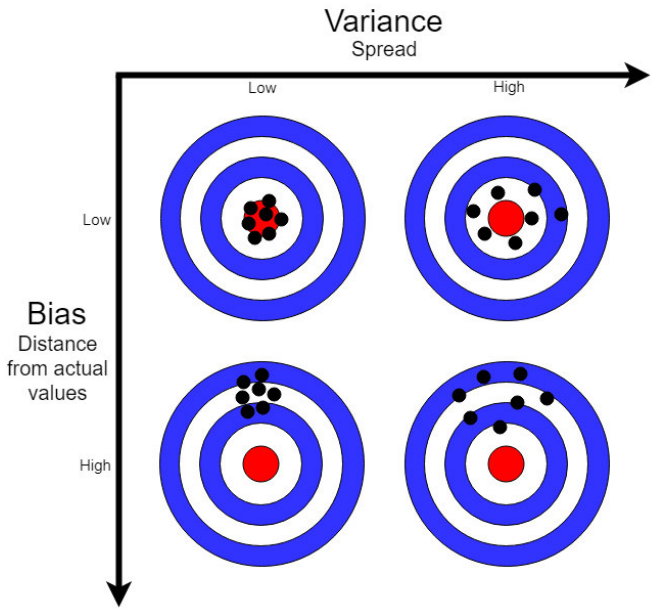

Figure 15 - Bias vs. Variance [22]

The most popular methods to address the issue of overfitting are reducing the number of features (thus generalizing the data) or adding a regularization term to the hypothesis function. The regularization keeps all features, reducing their weight. If bias is high, then decreasing the regularization term or adding features will help. In the case of high variance, the regularization term should be increased, trying small sets of features or adding more training examples [30]. To sum up, adding or removing features or altering the weights of existing features, promotes better performance for the algorithms. This process is known as Featuring Engineering.

Feature Engineering is one of the most important steps to take in ML. It must be applied after gathering and cleaning data and before defining the model/hypothesis and training, testing model/hypothesis. ML does not consist in just gathering data and applying an algorithm. Understanding the concept of feature engineering might be easy, applying it, however, can be harder due to the necessity of constructing new features. The path to follow in feature engineering consists in brainstorming about features, creating the features that might work well in the hypothesis, verifying if said feature is working properly and predicting well with new data, improving it if needed, and repeating the process as long it is necessary.

2.2.5 Existing Technologies

This section shows which are the more appropriate languages, frameworks, libraries and platforms for the Smart-PDM project. The section is divided by the following topics: Languages, Integrated Development Environment (without much details), Frameworks & Libraries and Platforms.

During this project, a survey was made about existing technologies and the results of this survey were compared against the work reported in [48]. This work consisted in posing several questions related to the field of machine learning with the goal of obtaining answers from the people that work with these technologies in a daily basis, i.e., individuals working in various business sectors (software and academic) and with different roles such as software engineers, data scientists and students**.** The work is large and diverse (23,859 responses) since it includes people from all around the world, including all age groups.

Each section will present the topics covered in both the survey and the previously described work [48], concluding with a decision on the most appropriate option for each topic, considering the Smart-PDM project goals.

2.2.5.1 Languages

Python is the most suggested language to use in Machine Learning. According to [49], [50] and [51], generally, the chosen language for a project depends on the type of project and on the developer's background. Based on the articles analyzed, the two best languages are Python and R.

R is similar to Python and it is designed for statistical analysis and visualizations. Python's syntax is like other languages while R's syntax is different. Python is a full-fledged language so it will integrate with other components better than R. In regard to visualizing data on charts, R is better (e.g. ggplot2, htmlwidgets, Leaflet) than Python (e.g. Matplotlib) since it provides more options.

Figure 16 presents a word cloud of popular programming languages, where it is possible to see that Python is the most popular language among newbies and experts. Figure 17 demonstrates that Matplotlib (Python) is more popular than ggplot2 (R).

Figure 16 - Popular Programming languages [48]

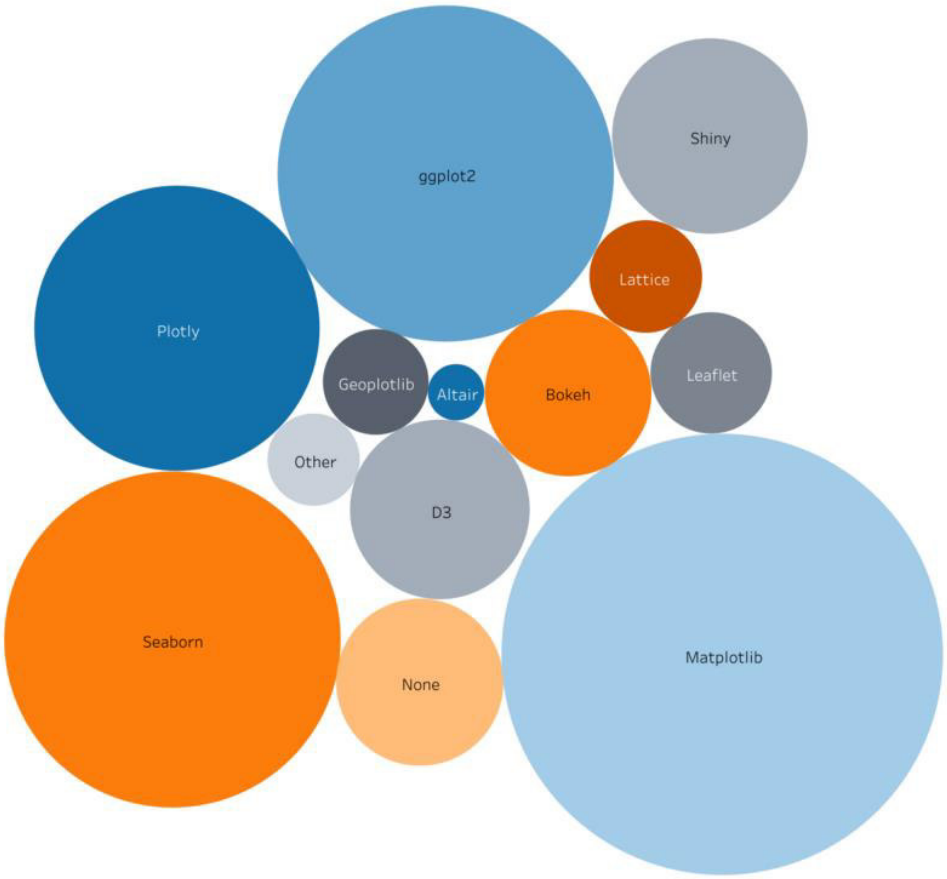

Figure 17 - Visualization Frameworks [48]

So, even though R is better suited to visualize data on charts, the best choice for the Smart-PDM project is Python because:

- it is widely used in data mining (which is good since the project will deal with Big Data);
- it comes with a few popular libraries (like NumPy and SciPy);
- it is simple and easy to learn.

2.2.5.2 Integrated Development Environment

Initially, the Machine Learning part of this project did not require the implementation of code (since it was originally planned to only use established data mining platforms) so the research in this topic was not very deep. Anyway, in general, the machine learning community uses notebooks kernels like Jupyter Notebook, Kaggle Kernels and Google Colab. Notebook kernels provide an interactive environment for exploring, analyzing and conceptualizing data and an easy way to document and share findings in more than one format [48].

Jupyter Notebook is a web-based application for authoring documents that combine live-code with narrative text, equations and visualizations [52]. Kaggle Kernels is a free platform to run Jupyter Notebooks in the user's browser [53]. Google Colab is a free service which provides

CPU resources and access to a GPU unit and is similar to Jupyter Notebook but stored in Google Drive [54].

Comparing the study with the article, it is safe to conclude that the decision for an IDE is not the most important to make, due to that fact the majority of Data Science and Machine Learning community does not even use any notebook, as it can be seen in Figure 18.

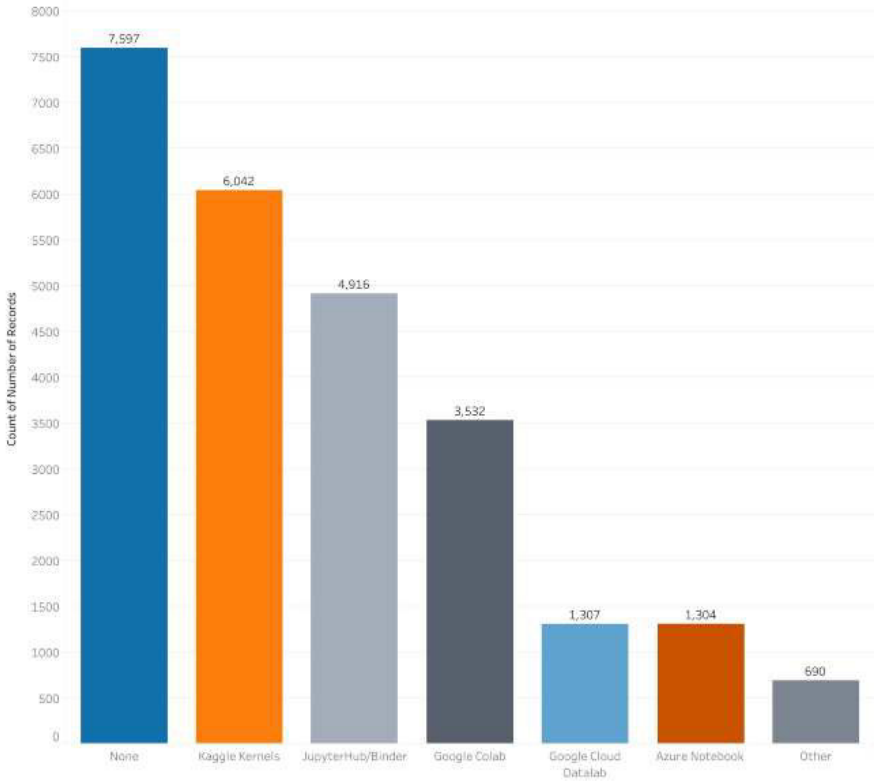

Figure 18 - Notebook Kernels suggested by article's respondents [48]

2.2.5.3 Frameworks & Libraries

The knowledge of frameworks and libraries is fundamental because they provide reusable and expandable code pallets that make it easier to explore, visualize and train datasets.

One of the reasons for Python to be the most appropriate language for the Smart-PDM project is that most frameworks and libraries are in Python.

According to [52], [55], [56] and Figure 19, the most generally chosen framework/library is TensorFlow. Scikit-Learn is recommended for beginner data analysts since it supports most of the classical supervised and unsupervised learning algorithms. However, if the chosen algorithm is Neural Networks, then TensorFlow has greater support.

Other frameworks and libraries are referenced but it seems that they have a minimal purpose. Pandas is for fetching and preparing data for later use in other machine learning libraries like

Scikit-learn or TensorFlow. NumPy is a core component of Scikit-learn and Pandas, it supports multi-dimensional arrays and matrices along with all core linear algebra operations. PyTorch is better for developing Deep Learning projects quickly and easily. Keras is designed to simplify the creation of deep learning models and is a direct competitor to PyTorch, because they both strive to provide a simple API for networking with Neural Networks.

Figure 19 - Machine Learning Frameworks [48]

The framework/library that is best suited for the Smart-PDM project is Scikit-learn, for the short-term, and TensorFlow for the long term, because Scikit-learn focuses on data mining and data analysis and when it is necessary to support bigger datasets, it is possible to port the model to TensorFlow. The learning curve of Scikit-learn is also shorter than TensorFlow.

2.2.5.4 Platforms

Platforms are used in the machine learning community for data preparation and analytical tasks like visualization, interactive exploration, deployment, performance engineering data preparation and data access. So, according to [57], [58], [59] and [60], the most referenced platforms are KNIME and H2O.

KNIME has a drag and drop GUI, it is very similar to RapidMiner (a proprietary/closed-source platform used by the last person working on the Smart-PDM project), it is easy to do ETL operations, it has a rich algorithm set, and also integrates with other languages like R and Python. H2O is well-documented, has easy to use algorithms and is easy to connect to cluster machines.

For this topic, there is no best or most appropriate platform for the project, thus this should be chosen based on which one the developer is most comfortable with. KNIME and H2O do not diverge that much, and both are the Leaders on a Gartner Magic Quadrant [57]. Gartner Magic Quadrant is a quadrant where the axes are “completeness of vision” and “ability to execute”. The Leaders quadrant represents the best of both axes. Thus, KNIME and H2O are the platforms that execute well against their current vision and are well positioned for tomorrow.

2.3 Related projects

This section presents works that are similar in nature with Smart-PDM but do not target the same market sector.

Predictive Maintenance for Milling Machines [61] applies special sensors, like ultrasonic or vibration sensors, to identify patterns of a fragile spindle and alert situations relevant to the current state of the machine. The benefits are greater process transparency, lower maintenance costs and reduced machine downtime.

Predictive Maintenance for Heat Exchangers [62] consists in measuring the temperature differential, analyzing them and setting the limit values for proper operation. If there is a malfunction, for example a clog, there is the possibility of notifying the users. The benefits are the early warning of anomalies indicating possible blockages and the reduction of machine downtime and less waste of materials.

Predictive Maintenance for the health of robots [63] [64], collects many parameters such as CPU temperature, positioning, overload, among others, for further analysis and evaluation. The benefits are awareness of the machine's health, intervention before the machine is damaged, increased uptime and early recognition of wear.

Mantis [65] provides a proactive maintenance service platform architecture based on Cyber Physical Systems to predict and prevent imminent failures and to schedule proactive maintenance.

Flexigy [66] manages and balances energy costs for the purpose of reducing the tariffs paid by consumers and promoting the expansion of renewable energy. Despite the difference of purposes between this project and Smart-PDM, there is a similarity in the process of data collection and pattern recognition.

Yujin Tang, Kunpei Sakai, Shixin Luo and Yiliang Zhao developed an end-to-end demo system on Google Cloud Platform that learns how to accurately identify home appliances' (e.g. electric kettles and washing machines) operating status using smart power readings, together with modern machine learning techniques such as long short-term memory (LSTM) models [67].

The book *Distributed Computing and Artificial Intelligence* contains a chapter which describes the fault detection mechanism of a predictive maintenance system developed for the metallurgic industry. Real-world sensor data was used, which was obtained from monitoring different machine components and parameters. Imminent faults were predicted by estimating autoregressive integrated moving average models [68].

3 Preparation for Data Analysis component

This section will be divided in two main sections. Section 3.1 presents the prototype status and the improvements done to it, and Section 3.2 describes the deployment of the updated prototype.

3.1 Prototype Analysis

Smart-PDM project began as a challenge done by a home appliance distribution company between September 2018 and January 2019 and, as a result, a prototype was developed.

The major difference between the prototype and the actual work was the lack of any Data Analysis component. Figure 20 presents a more correct assessment of the system diagram displayed in Figure 1, which is the component diagram.

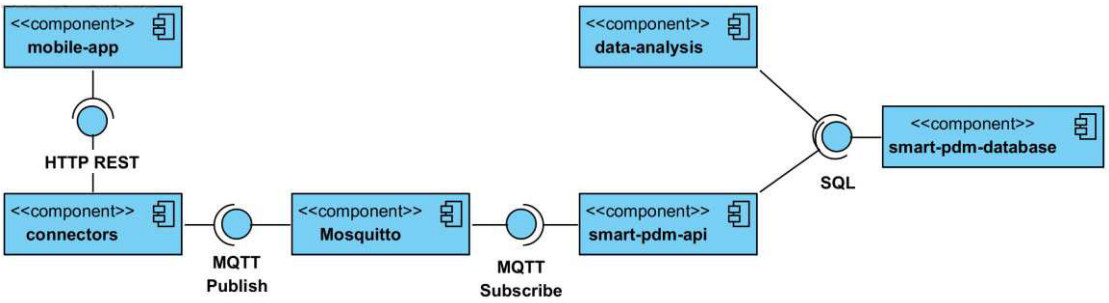

Figure 20 - Component diagram of actual work

Despite the lack of Data Analysis component, the communication between the components did not suffer big modifications. The Sections 3.1.1 and 3.1.2 describe, respectively, the state of the prototype and the improvements made to the prototype until the prototype reaches the current project state.

3.1.1 Prototype Status

The Smart-PDM project already had some code developed. Indeed, the work already developed was:

- The mobile application to connect the Smart Connector to a Wi-Fi network;
- The Message Broker – the bridge between the Smart Connector and the Web Application;
- A Web Application functioning as MQTT Client;
- A No-SQL database;
- A Java application to label/classify the dataset.

The mobile application is only available for Android users (for this reason, from here on, it shall be referred to as “Android application”) and did not suffer any modifications or gained any new features. Nevertheless, some modifications were originally planned for it – in fact, Section 0 goes into detail about all the work that was not fully achieved due to time constraints.

For the Message Broker, the code already developed in the previous work was used, but due to project needs, it was necessary to implement new code for a different Smart Connector, Sonoff Pow R2.

The Web Application was considered too simple/poor for the problem in hand and had two databases while one was enough. In brief, Web Application behaved like an MQTT Client, subscribing to all topics, and established a connection to the No-SQL database to save each message received from the Smart Connector of all topics. The Entity Framework ORM was used for the database, with just three objects without any relationship, as seen in Figure 21: User, which was not used; Plug, in the Smart-PDM context, the Smart Connector, with the id and the MQTT topic; and Consumptions, to save the consumption of a specific Plug.

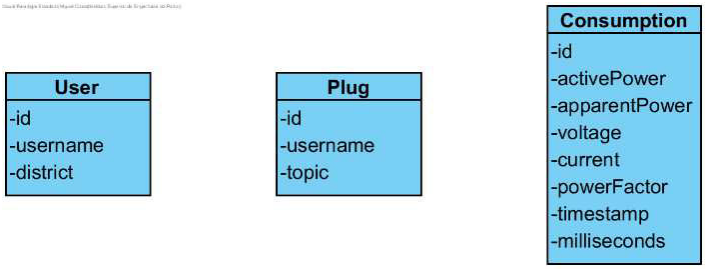

Figure 21 - Previous Domain Model of Web Application component

In actuality, the Web Application existed to provide a visual component of the home appliance’s consumption. As such, the Web Application queried data from the No-SQL database, created new object instances based on this data, and saved them in the SQL database (namely, SQL Server). The No-SQL database had two collections. One to save all information from an MQTT message, represented in Figure 22, plus the sensor’s identifier, the Plug ID. The other collection saved the topics pertaining to the district where the user lives, as can be seen in Figure 23. Thus, the MQTT topic becomes larger since it includes another level[5] (in this case, the country’s district). Consequently, message forwarding becomes more scalable, since a subscriber can specify the consumption of a district, instead of subscribing to the consumption of the entire country.

[5] An MQTT topic can have multiple levels. The levels are separated by slashes, e.g. the topic “car/BMW” has two levels.

```
_id: ObjectId("5b51eb40d7f53e1dc06502f9")
Plug ID: "67"
Active W: "573"
V: "223"
I: "7.92"
Apparent W: "1699"
Power %: "33.00"
Timestamp: "1532095292"
Milliseconds: "46"
```


Figure 22 - Document example from Consumption collection


```
_id: ObjectId("5b3f36d69a893b2ae83ebcdb")
topic: "consumption/Braga/+"

_id: ObjectId("5b3f36d69a893b2ae83ebcdc")
topic: "consumption/Vila Real/+"
```


Figure 23 - Documents examples from District collection

Lastly, the Java application became obsolete since, in the actual work, the data is no longer extracted from the No-SQL database. However, some of the feature construction ideas were reused (see in Section 4.1.2).

3.1.2 Changes/Improvements made to the prototype

From the previous section it was concluded that it was necessary to implement code for Sonoff Pow R2 (Section 3.1.2.1) and a new Web Application (Section 3.1.2.2) that was prepared for any use case that could arise.

3.1.2.1 Sonoff Pow R2

Both Smart Connectors (Sonoff Pow version 2.0 and Sonoff Pow R2) have been previously described in Section 2.1.1. There are some differences between the Smart Connectors, as can be seen in Figure 24. The only real difference, at a programming level, between Sonoff Pow version 2.0 from Sonoff Pow R2 is the sensor, which is highlighted in red in Figure 24. The biggest difficulty was to discover this difference and find the proper documentation/libraries for the CSE7766 sensor.

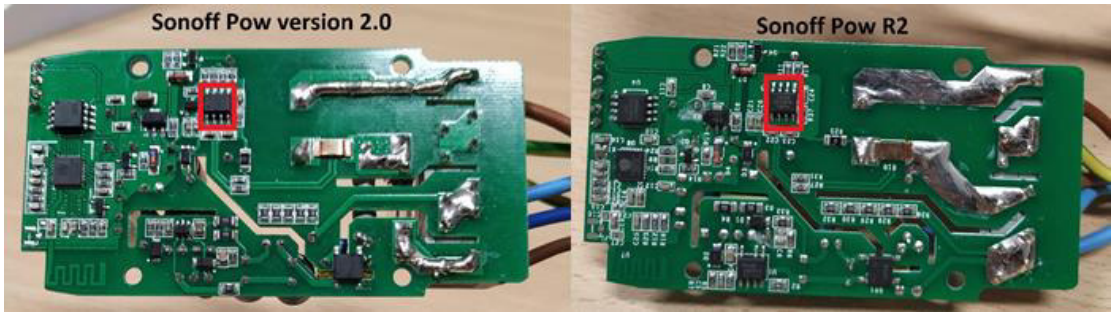

Figure 24 - Smart Connectors boards

All the tasks related to the Smart Connectors consisted in programming them. As an additional task and also to have the minimal guarantee that the energy consumption data is reliable, the values measured by the Sonoff Pow version 2.0 were verified with a multimeter. Figure 25 presents the wiring diagram assembled with the multimeter and the Smart Connector to verify the voltage and the electric current, since they were the only variables that the multimeter measured.

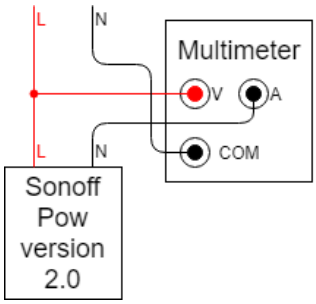

Figure 25 - Wiring diagram for calibration

Table 4 presents the interval of the values measured from a lamp by the Sonoff Pow version 2.0 sensor and the multimeter simultaneously for five minutes, approximately.

Table 4 - Values measured by Sonoff Pow and the multimeter

	Sonoff Pow version 2.0	Multimeter
Electric Current (A)	230 - 239	240 - 241
Voltage (V)	0,35 - 0,36	0,38

Despite the difference between the devices and the variance in Sonoff Pow version 2.0, the results of Sonoff Pow version 2.0 were not considered bad/unacceptable.

When this verification was made, Sonoff Pow R2 was not implemented yet and it was not though that it would be and, as R2 has better measurements than version 2.0 [69], the R2 was not submitted to these verifications.

3.1.2.2 Web Application

Figure 26 presents the updated domain model of the Web Application component. Comparing with the previous domain model in Figure 21, the User, Plug and Consumption were preserved but renamed as HomeUser, Sensor and ConsumptionRaw, respectively. A more detailed explanation of domain model, it is described later using the database schema (in Figure 27).

Initially, a MySQL database was set up, while the MQTT Client (which subscribed to and saved the information in the database) and the User Interface (UI) (which displayed the home appliances' consumption) were implemented in Python.

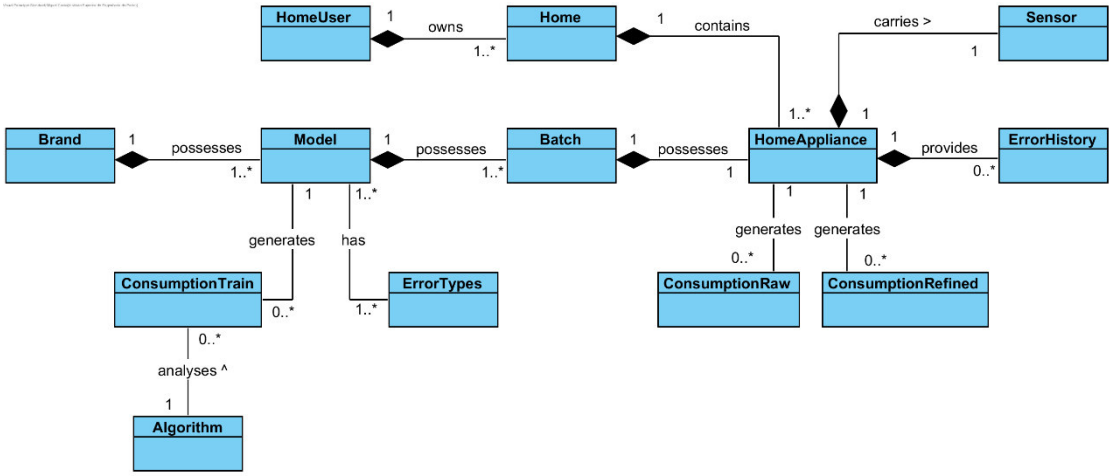

Figure 26 - Domain Model of Web Application component

Even though this project was being developed by two people with different technologies, an effort was made to implement a solution with a unique database, with the goal to sell the solution in two versions, the free, open-source version which is presented in this report, and the proprietary, paid version presented in other work [70], only using Azure technologies. Before the reconstruction of the database, i.e., the switch from MySQL to SQL Server, some restrictions were found on the Azure technologies. Basically, it was not possible to save more than 8000 messages per day using Azure's IoT Hub, through the students account. This is a major restriction because it is extremely important to get the data for the subsequent analysis work using Machine Learning. As such, since the data is sent every 2 seconds, this means that it is only possible to store approximately 4 hours and 27 minutes of data per day, for only one home appliance.

After some brainstorming, it was decided to develop a new web application in ASP.NET in accordance to the new database[6], not just to display home appliance's consumption but also to give more options to the user, such as adding or removing new home appliances. However, all the previous work (with the MySQL database and code in Python) was not wasted since the code in Python which subscribed to and saved the data, was reused by the Azure solution developer to circumvent the 8000-message restriction.

Figure 27 gives some context of the SQL database and, consequently, the domain objects that were developed for the API and were presented in Figure 26. From hereafter, the entities represented in Figure 27 will be referred to in bold. An explanation of the process for the users that will use this product is subsequently presented and the process of the system as well.

[6] This database was built considering only the refrigerator. For more details about the database, see Section 5.4

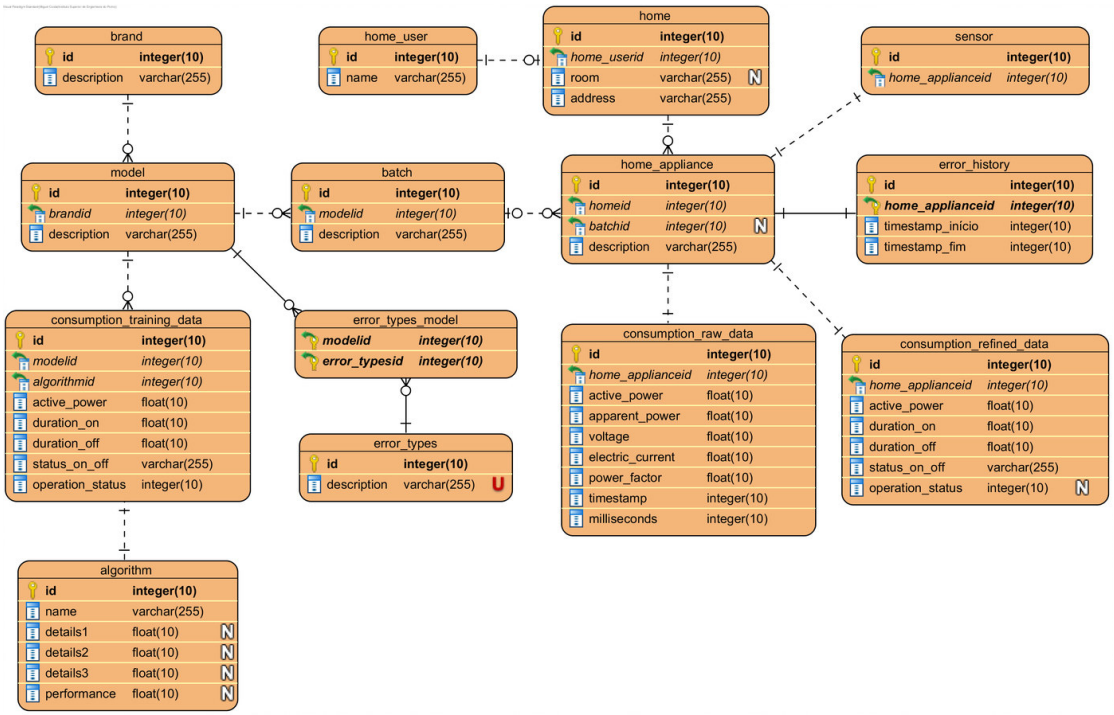

Figure 27 - Database schema

First, the **home_user** buys the **sensor(s)** he wants and installs them at home. Through the Android application, the **home_user** will be registered in the system specifying his address[7] and, if he wants, insert the room where the **home_appliance** is located. Before specifying the **home_appliance**, the **home_user** will connect his **sensor(s)** to his Wi-Fi network. Once the **sensor** is connected to the internet, it registers itself in the system, creating a new **sensor** entry. From this moment onward, the **home_user** stops having any direct influence on the system.

The **sensor** will start publishing messages with the values measured by the sensor, which are then stored in **consumption_raw_data**. The table **consumption_raw_data** is required since it is not possible to process the data before it is saved in the database because of the sensor's memory restrictions.

Every home appliance has a batch, a model and a brand. Therefore, the **home_appliance** has the information of its **batch**, the **batch** is related to a **model** and the **model** to a **brand**. Lastly, the **home_appliance** has a history of failures registered in **error_history**.

[7] The **home_user** specifies his address for scalability reasons referred in Section 3.1.1**Error! Reference source not found.** (the construction of the MQTT topic).

Certainly, there will be errors which can be common between the different type of home appliances[8], so these are all stored in **error_types** and each error can have a small description of it or a possible procedure that the home user can take. The **error_types_model** table stores the errors associated to each home appliance model.

The **model** is related to the **consumption_training_data** and to **error_types_model** due to the fact that the consumption will be the same for each type of home appliances with the same model, i.e. brand X has two models, A and B; all model A home appliances will have the same consumption and all model B home appliances will have the same consumption as well but this does not mean that models A and B have the same consumption.

Finally, the tables **consumption_train_data** and **consumption_refined_data** have data processed from **consumption_raw_data**, i.e. data that has been through the feature engineering process. As a consequence of using Supervised Learning, it is necessary to classify the data. Thus, the resulting classification label will be stored in the column operation_status in both tables. The most obvious difference between both tables is that the column operation_status can be null in **consumption_refined_data** while in the **consumption_train_data** cannot be null since it is necessary to give classified data to train the Machine Leaning model which will predict the health status of the home appliance. Once a classified dataset is available for analysis, a set of ML algorithms will be tested, and their performance will be stored in the table **algorithm**. The algorithm with the best performance is then used to create a ML model based on the data stored in **consumption_train_data**. The resulting model will then complete the column operation_status in **consumption_refined_data** with its predictions.

The **consumption_train_data** is only used for the initial phase of constructing the ML model for a specific home appliance model. Afterwards, that table will never be used for that home appliance model, and the **consumption_refined_data** table will then store the data for the rest of the home appliance's lifecycle.

The operation_status column in the **consumption_refined_data** table is the key for the notifications of bad home appliance functioning. The idea is that the number 0 will represent a good functioning and the numbers 1, 2, 3, and so on, will be the errors. If a new entry in this column is different than 0, a notification will be sent to the Android application.

[8] Whenever "type of home appliances" is used in this report, this does not serve to differentiate or categorize the home appliances i.e. big or small home appliances, but to differentiate between fridges, washing machines, cookers, ovens, among others, without considering the brand and the model.

3.2 Deployment of the Data Collection components

The Data Collection process is performed by the Message Broker, Web Application and Database, whose roles consist in acquiring and storing the data for future analysis. The deployment of these components was to collect the data outside the development environment and also to test the developed system in a real scenario. The Data Analysis component deployment is described in Section 4.1.3.3.

3.2.1 Message Broker

The Smart Connector (MQTT client) publishes messages to a broker in a loop. Thus, to deploy this component, it is necessary that the message broker (that could be standardized as server) is always available. As such, Mosquitto [26] was used, which is a free open source MQTT broker that runs on Windows and Linux. Mosquitto was configured with the Internet Protocol (IP) address and the port number of on an ISEP server.

3.2.2 Web Application and Database

The Database was created through Entity Framework, so these two components were joined. Since CISTER is part of ISEP and due to the restriction of using only open source tools, for the hosting server, a virtual machine hosted by ISEP's servers, with external access through a port, was used.

Regarding the actual deployment, Internet Information Services (IIS) was used to host and monitor the Web Application. This part was very challenging since the process of saving data into the Database worked well in the development environment, while in the production environment it did not. To solve this situation, given that in a production environment there is no output console to check execution errors, a logger was implemented (namely *Apache log4net*). In the end, this helped to identify errors that occurred only in the production environment and not in the development environment. Afterwards, threads, timers and callbacks were implemented to invoke the data analysis component. This part of the deployment process cost almost a month that could have been better spent in researching the best Machine Learning algorithm to detect energy consumption patterns and anomalies. This work was not wasted because in the development environment is implemented and the process logic between components is already coded, e.g. the Data Analysis component cannot be invoked, if there is no data.

4 Data analysis of consumption patterns

The purpose of this section is to identify which machine learning algorithm or pattern matching technique is best suited for the problem at hand, by selecting the one with best performance.

Firstly, it was necessary to extract and process the data. To this end, the project partner offered a washing machine and CISTER made available a refrigerator for testing. For these two home appliances, the external parameters that could influence their energy consumption were defined. It should be mentioned that these parameters were specifically chosen to meet the project's secondary objectives, such as not disturbing the user to ask for other parameters that could lead to a better prediction (e.g. the weight of the clothes put in the washing machine or the actual temperature of the refrigerator).

Both datasets were created at CISTER. Each dataset row represents a consumption reading from the sensor at a specific time (Timestamp + Milliseconds) and contains the following features: Active Power (Watts), Apparent Power (Watts), Voltage (Volts), Electric Current (Amperes), Power Factor (%), Timestamp and Milliseconds represented in the table's columns in Figure 28. Each dataset column could be used as a feature for the ML algorithms.

ActivePower	ApparentPower	Voltage	EletricCurrent	PowerFactor	Timestamp	Milliseconds
1	12	236	0.05	15.96	1563481045	346177
0	0	236	0.00	100.00	1563481047	348178
3	13	236	0.06	29.04	1563481049	350179

Figure 28 - Dataset structure

This section is divided into two sections, each corresponding to each of the two home appliances. In the case of the washing machine (Section 4.1), the problem consists in detecting energy consumption patterns, while in the case of the refrigerator (Section 4.2), the problem consists in detecting potential failures.

4.1 Washing Machine

This section is divided into three subsections: Section 4.1.1 describes how the data is structured and the experiments that were done with the perspective of finding the best features to classify the data and to discover consumption patterns; Section 4.1.2 presents the analysis done to the experiments (described in Section 4.1.1), i.e. presents the cleaning and visualizing of the data and how the features influence the behavior of the consumption

pattern; finally, Section 4.1.3 shows the results of the application of supervised machine learning algorithms and pattern matching techniques using the data processed.

4.1.1 Data

Due to the number of different washing machine programs and other parameters, like temperature and centrifugation, it was decided that only four washing machine programs were to be studied. For any washing machine program, the data measurement was made every two seconds and the chosen parameters that could influence energy consumption were:

- Water temperature chosen by the user;
- Weight of the clothes inside the washing machine;
- Whether centrifugation was selected or not for the washing process.

The four chosen programs have the following names: “14 minutes”, “30 minutes”, “Coloured” and “Cottons + Prewash”. The experiments consist in running a washing machine program and varying the parameters previously cited, i.e. temperature, weight and centrifugation. For example, experiment 1 may represent the experiment of running the "14 minutes" program at 30 degrees with weight and centrifugation, while experiment 2 may be the experiment of running the "14 minutes" program at 20 degrees with weight and centrifugation.

Table 5 shows an overview of the experiments and repetitions which were made for each program, e.g. the program “14 minutes” had 12 different experiments, with each experiment being repeated 6 times. Due to the long duration of the programs “Coloured” and “Cottons + Prewash” (about two to three hours), fewer experiments and repetitions were made. The “30 minutes” program had fewer repetitions, as the consumption pattern did not change much from the “14 minutes” program.

Table 5 - Washing Machine dataset information

Program	14 minutes	30 minutes	Coloured	Cottons + Prewash
No. of experiments	12	12	8	4
No. of repetitions of each experiment	6	2	3	2

The experiments distributed among the 4 programs are equivalent to, at least, 93 hours of washing machine execution, approximately.

Appendix 7.1 presents the information regarding all the experiments made. To each washing machine program experiment was attributed a number and a description, and the date it was measured was registered. The experiments will be referred to mostly by their number, not by their description.

4.1.2 Analysis

4.1.2.1 Consumption pattern and visualizing data

Firstly, the consumption pattern for the “14 minutes” program is shown in Figure 29. The colored zones represent the three phases of the washing program.

In the yellow sections, when the active power (watts) goes down and back up, it means that these rotations switched direction from clockwise to counterclockwise or vice-versa. In red, it is possible to see the moment when the active power is very high relative to the rest of the program, which corresponds to the water heating process, in this case, 30 degrees. Finally, in green, the centrifugation process. This phase is the most inconsistent part in every program, i.e., its consumption pattern is similar but the moment when it starts is unpredictable.

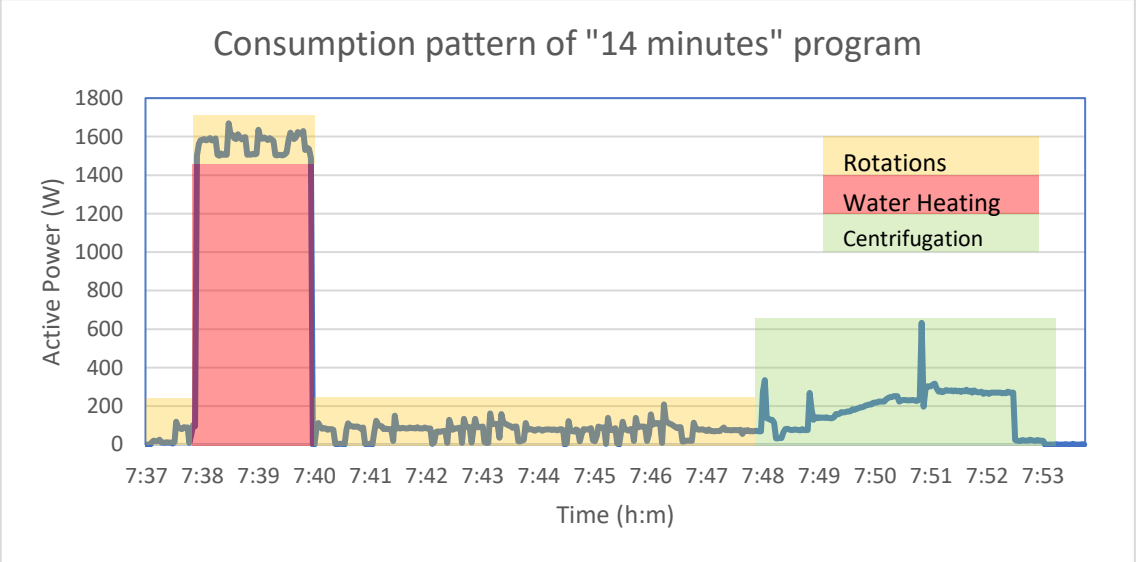

Figure 29 - Consumption pattern of "14 minutes" program

During the data extraction of the washing programs the data acquired from performing multiple repetitions of each experiment was not enough to find a clear consumption pattern. After some analysis of this data, it was discovered that some information was lost from the data that was sent every 2 seconds (the reason is explained in Section 0). However, an effort was made to mend these holes in the acquired data. As the time interval without data was rarely greater than 4 seconds, to mend the holes, the mean of the previous and next measures was calculated for all features measured by the sensor (better explained in Section 4.1.2.2). Due to the project’s deadline, the “14 minutes” program was given the bigger focus since this program is the shortest one available in the washing machine; hence it has more repetitions. The beginning and end of the experiments/programs were registered in a spreadsheet. To fill in this data, it was recorded the dates and times (in hours and minutes) when the washing

machine program was started (manually) and when the washing machine beeped that it had just run a particular program. In the spreadsheet, the dates and times were converted to Epoch timestamp and then these values were copied to R to extract the experiments/programs from the database, as shown later in Figure 30.

4.1.2.2 Processing data

To facilitate the analysis, multiple functions in R were developed for each program to:

- **Extract data from the database:** After a database connection is established, all experiments are extracted by calling a function which carries as parameters the timestamps when the experiment starts and ends, e.g. in Figure 30, the last line represents the extraction of the first repetition of experiment 24.

```
connection <- 'driver={SQL Server};server=DESKTOP-DPUOLFO;database=smartpdm;trusted_connection=true'
dbhandle <- odbcDriverConnect(connection)

getTableFromProgramDB <- function(timeStart,timeEnd){
  query <- sprintf('SELECT * FROM [smartpdm].[dbo].[ConsumptionRaws] WHERE [IdHomeAppliance] = 1
                   AND [Timestamp] > %d AND [Timestamp] < %d',timeStart, timeEnd)
  table <- sqlQuery(dbhandle, query)
  return(table)
}

e24_first <- getTableFromProgramDB(1563306180,1563306840)
```

Figure 30 - Extracting data from the database

- **Mend the data holes:** Since the data is extracted every two seconds and the experiments were compared between themselves (to understand the washing machine behavior), the dataset cannot have holes. Figure 31 shows the loop where these fixes were made, i.e. if a difference bigger than 2 seconds is found between the timestamps A and A+N (where N > 2), the mean of the values at these timestamps is calculated and inserted between them.

```
while(nrow(aux[which(diff(aux$Timestamp)>2),]) > 0){
  aux <- aux[order(aux$Timestamp),]
  erro <- sprintf("There are %d data hole(s) to resolve on %s",nrow(aux[which(diff(aux$Timestamp)>2),]), obj)
  print(erro)
  list <- aux[which(diff(aux$Timestamp)>2),"IdConsuRaw"]
  for(id in list){
    dado <- which(aux$IdConsuRaw == id)
    novoDado <- data.frame(IdConsuRaw = paste0(aux[dado,"IdConsuRaw"],"_1"),
                           IdHomeAppliance='NA',
                           ActivePower=mean(c(aux[dado,"ActivePower"],aux[dado+1,"ActivePower"])),
                           ApparentPower=mean(c(aux[dado,"ApparentPower"],aux[dado+1,"ApparentPower"])),
                           Voltage=mean(c(aux[dado,"Voltage"],aux[dado+1,"Voltage"])),
                           EletricCurrent=mean(c(aux[dado,"EletricCurrent"],aux[dado+1,"EletricCurrent"])),
                           PowerFactor=mean(c(aux[dado,"PowerFactor"],aux[dado+1,"PowerFactor"])),
                           Timestamp=(aux[dado,"Timestamp"])+2,
                           Milliseconds=mean(c(aux[dado,"Milliseconds"],aux[dado+1,"Milliseconds"])))
    aux <- rbind(aux, novoDado)
  }
}
```

Figure 31 - Fixing the data holes

- **Adjust the beginning and end of the experiments to facilitate, in the data visualization, the comparison between the experiments done:** All the experiments

made had an initial active power value greater than 17 watts. Under those circumstances, five values (ten seconds) were maintained before the experiment's beginning. If the experiment did not accomplish that, a manual fix was made. Looking at Figure 30, instead of the experiment beginning at 1563306180, it begins at 1563306120, i.e. 1 minute before. Considering whether the centrifugation was activated or not, the maximum duration of all experiments was calculated, and this value was used as a reference to add or remove values in the end of each experiment. This process can be seen in Figure 32.

```
aux <- aux[order(aux$Timestamp),]

if(which.first(aux$ActivePower >= 17) > 5){
  aux <- tail(aux,-(which.first(aux$ActivePower >= 17)-5))
} else if(which.first(aux$ActivePower >= 17) < 5){
  cat("!!!!!!!!!!!!!!!!!!! NEEDS FIXES",obj,"!!!!!!!!!!!!!!!!!!!\n")
}

finalLines <- nrow(aux)-centrifugacao

if(finalLines <= 0){
  aux[nrow(aux):(nrow(aux) + abs(finalLines)),] = tail(aux,1)
}else {
  aux <- head(aux,-(nrow(aux)-centrifugacao))
}
```

Figure 32 - Adjustments of experiments' duration

- **Construct the features:** Figure 33 presents the construction of water heating duration and experiment duration features (more details about these features are in Section 4.2.3). The time interval that the active power values were greater than 1000 watts correspond to the water heating duration. To get the experiment duration, it was calculated the difference between the last active power value greater than 7 and the first value greater than 17, i.e. the 5th value of the experiment dataset.

```
aux$ConstEpoch <- aux$Timestamp[which.last(aux$ActivePower>7)] - aux$Timestamp[5]
aux$ConstAquecimento <- aux$Timestamp[which.last(aux$ActivePower>1000)] -
                        aux$Timestamp[which.first(aux$ActivePower>1000)]
aux$Epoch = aux$Timestamp - aux$Timestamp[1]
aux$Aquecimento <- 0
iniAquecimento <- aux$Timestamp[which.first(aux$ActivePower>1000)]
for(index in which(aux$ActivePower>1000)){
  aux$Aquecimento[index] <- aux$Timestamp[index] - iniAquecimento
}
```

Figure 33 - Feature construction

- **Classify the data for training the models:** Through the experiment number, the data is classified by adding a column "Classification", as seen in Figure 34, with the information of the program ("14min", "30min", "Coloured" or "CottonsPrewash"), the temperature ("20", "30" or "40") and the centrifugation ("WC" or "WOC", i.e. With Centrifugation or Without Centrifugation).

```
if(as.numeric(gsub("\\D", "", obj)) %in% c(1,2,5,7,12,13,14,15,21,22,23,24)){
  aux$Classification <- "14min_"
  if(as.numeric(gsub("\\D", "", obj)) %in% c(21,22,23,24)){
    aux$Classification <- paste0(aux$Classification,"20_")
  } else if(as.numeric(gsub("\\D", "", obj)) %in% c(1,5,12,14)){
    aux$Classification <- paste0(aux$Classification,"30_")
  } else if(as.numeric(gsub("\\D", "", obj)) %in% c(2,7,13,15)){
    aux$Classification <- paste0(aux$Classification,"40_")
  }
 if(as.numeric(gsub("\\D", "", obj)) %in% c(1,2,5,7,21,22)){
    aux$Classification <- paste0(aux$Classification,"WC")
  } else {
    aux$Classification <- paste0(aux$Classification,"WOC")
  }
}
```

Figure 34 - Data classification

- **Plot various programs for a better comparison:** While the temperature comparison function is presented in Figure 35, more functions like this were constructed, such as the comparison between the repeated experiments, and the comparison with and without centrifugation/weight. All functions previously described plot the number of experiments that are needed to compare (in the “14 minutes” program there are three: 20, 30 and 40 degrees) and then export a PNG file with the resulting image of the plot.

```
compareTemperatureInOneGraph <- function(graph1,graph2,graph3,titleGraph){
  par(mar=c(5,5,3,1)+.1,cex.lab=2.5, cex.axis=2.3, cex.main=2.5)
  plot(graph1$Epoch,graph1$ActivePower, type="l", col = "green", main = titleGraph,
       xlim = c(0,max(graph1$Epoch,graph2$Epoch,graph3$Epoch)),
       ylim = c(0,max(graph1$ActivePower,graph2$ActivePower,graph3$ActivePower)+10),
       xlab = "Time (seconds)", ylab = "Active Power (W)")
  lines(graph2$Epoch,graph2$ActivePower, type="l", col = "blue")
  lines(graph3$Epoch,graph3$ActivePower, type="l", col = "red")

  legend("topright", legend=c("20 ºC","30 ºC","40 ºC"),
         col=c("green","blue","red"), lty=1, cex=1.8)
  dev.copy(png,sprintf("finalComparationsTemperature_%d_%d_%d_time%d.png",
                       as.numeric(gsub("\\D", "", deparse(substitute(graph1)))),
                       as.numeric(gsub("\\D", "", deparse(substitute(graph2)))),
                       as.numeric(gsub("\\D", "", deparse(substitute(graph3)))),
                       stringToTime(titleGraph)), width = 1500, height = 800)
  dev.off()
}
```

Figure 35 - Function example to plot 3 experiments

However, it was still complicated to find a consumption pattern on the washing machine programs. All the analysis is presented in the next section with image support.

4.1.2.3 Analysis and feature engineering

Figure 36 displays six repetitions of the “14 minutes” program, with the water temperature at 40 degrees, centrifugation enabled, and no weight, i.e. experiment 2.

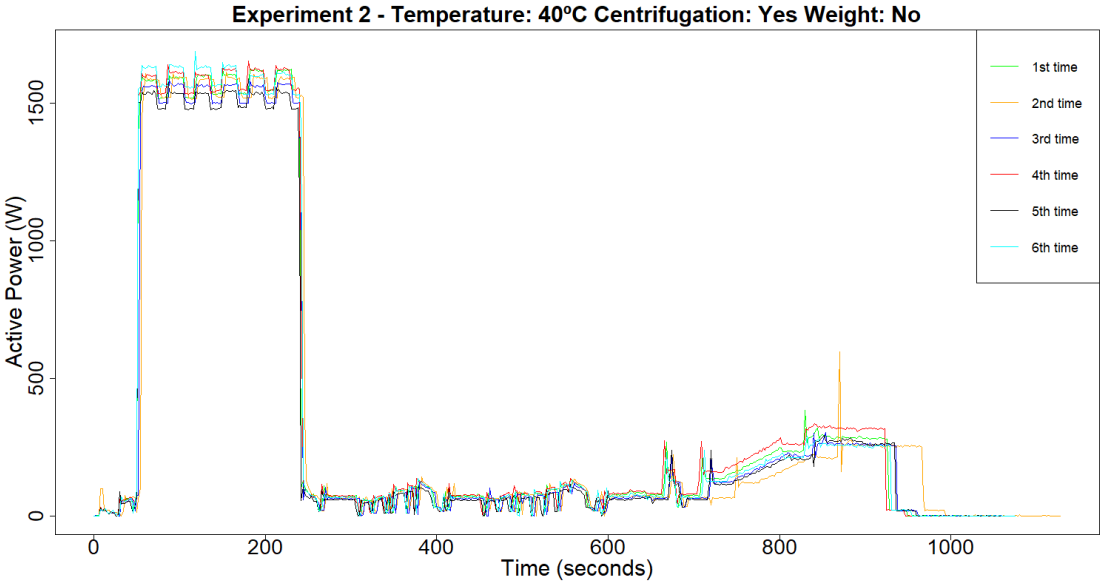

Figure 36 - Experiment 2 of "14 minutes" program

When there is no weight or when the weight is the same between experiments, it is possible to define a pattern, although when the weight varies, the consumption pattern changes considerably, as shown in Figure 37 and Figure 38 in the fifth and first repetitions, respectively.

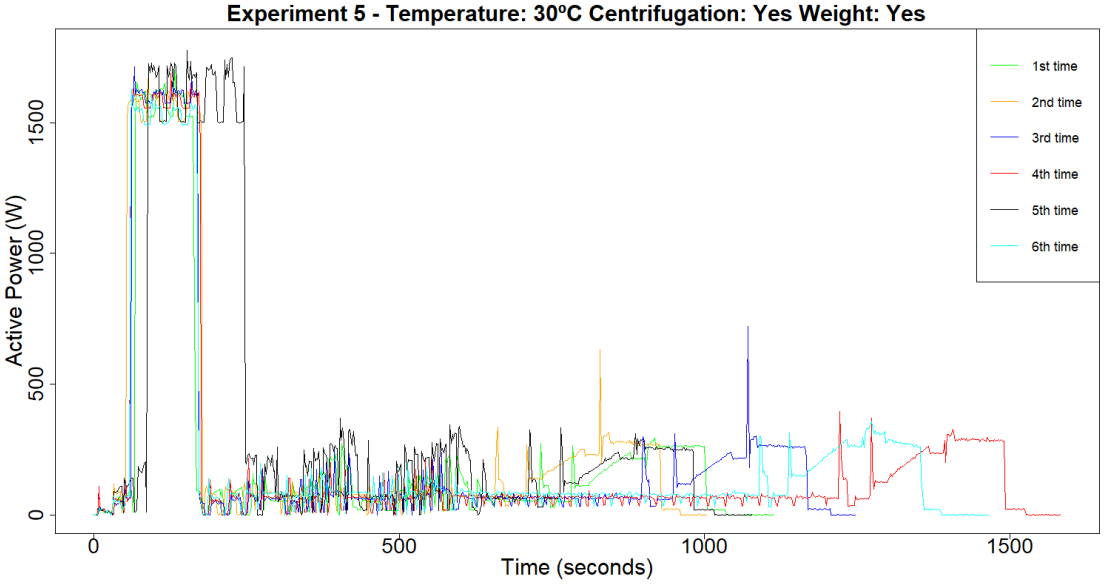

Figure 37 - Experiment 5 of "14 minutes" program

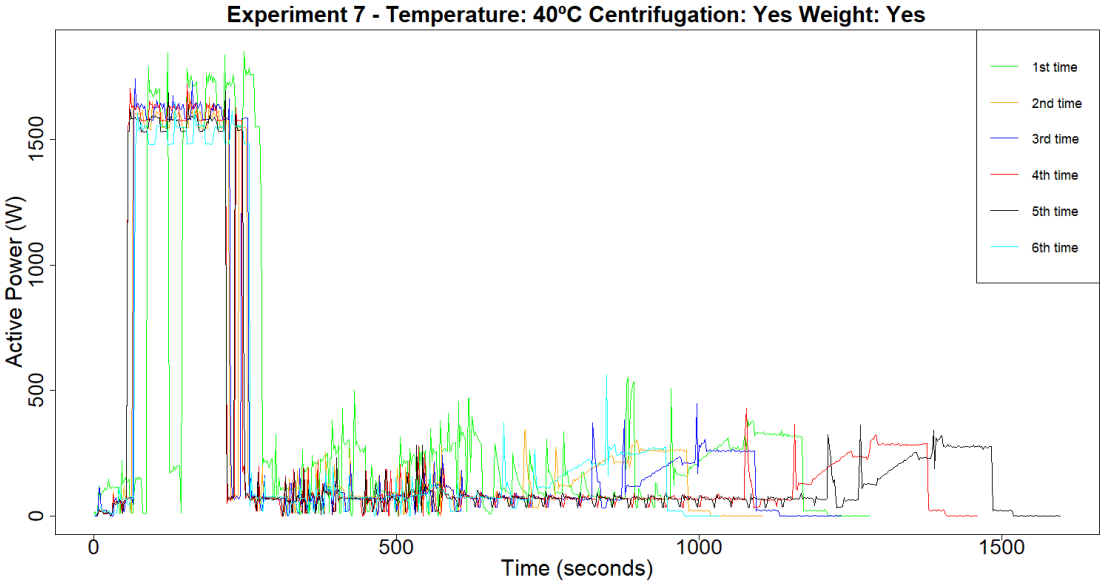

Figure 38 - Experiment 7 of "14 minutes" program

In these two last figures, all repetitions were made with a bed coverlet that filled the entire drum, except for the fifth and first repetitions which had different kind of clothes that partially filled the drum. Beyond weight, it is possible to conclude that the instant for when the centrifugation is started is always the same only when the washing machine drum is empty; in truth, there is a consumption pattern, but it seems that it is moved (probably) because of the weight.

Nevertheless, the problem of the centrifugation phase being moved also occurs in the water heating phase, as it can be seen in the Figure 39, and in all experiments made with weight (clothes inside) and without centrifugation.

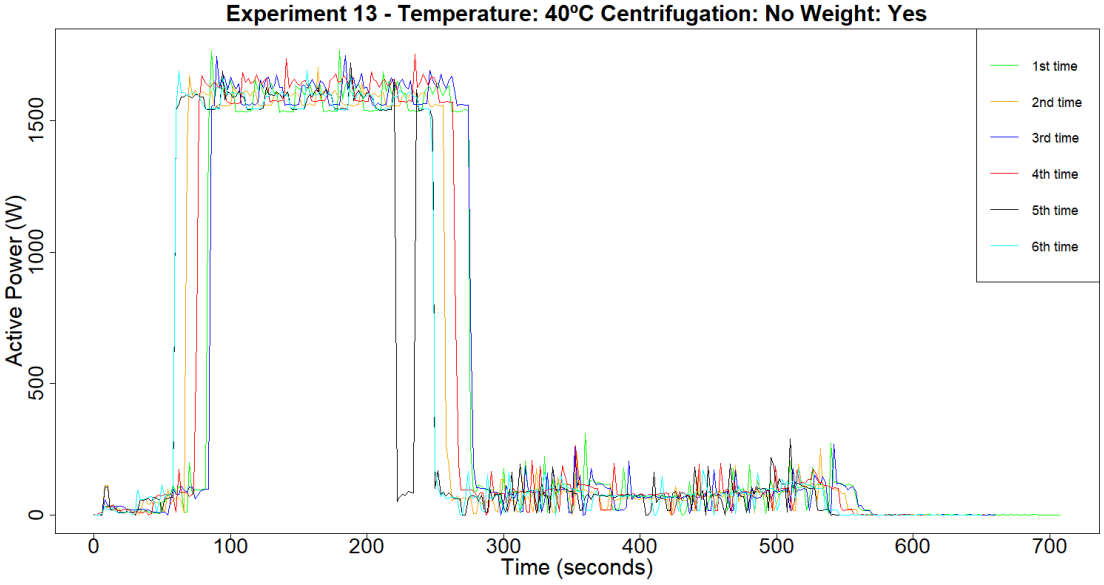

Figure 39 - Experiment 13 of "14 minutes" program

Thus, because it is hard to define a consumption pattern while changing the variables, it was opted to first classify a program and two external parameters, temperature and centrifugation.

Figure 40 shows that it seems possible to distinguish the temperature, at least on the “14 minutes” program; the higher the temperature, the longer the consumption is, with active power values greater than, at least, 1000W.

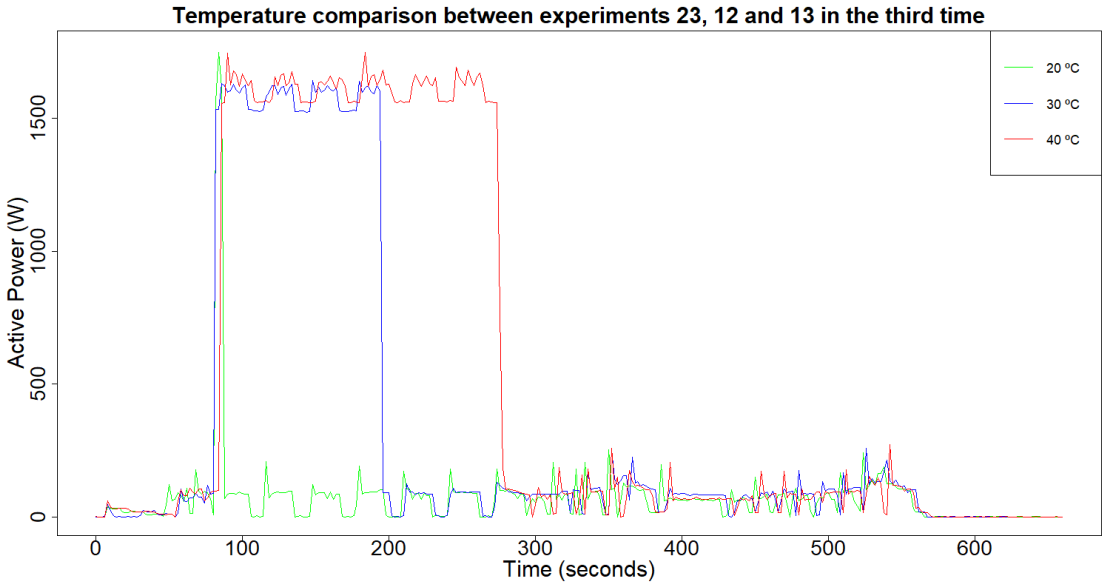

Figure 40 - Temperature comparison of "14 minutes" program experiments without weight

Figure 41 shows a more realistic consumption pattern (since the washing machine has weight unlike of Figure 40) and demonstrates that the beginning of the water heating phase and its duration are a little uncertain, i.e. while in Figure 40 shows that the water heating duration to 30 degrees is half of the water heating duration to 40 degrees, approximately, in Figure 41 shows that the water heating to 30 and 40 degrees was almost the same.

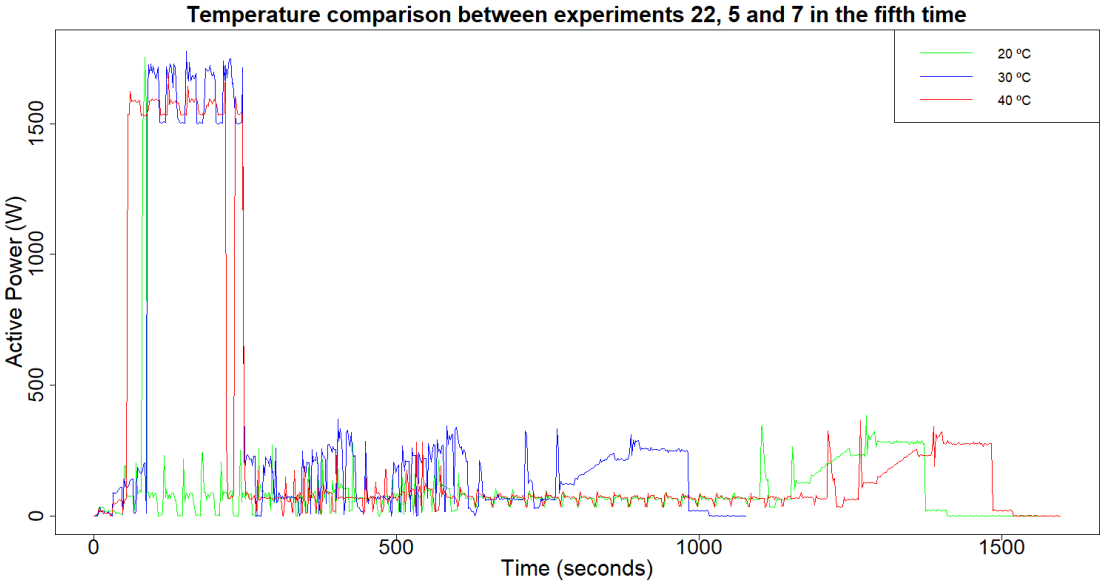

Figure 41 - Temperature comparison of "14 minutes" program experiments with weight

However, looking at the differences at the start and the duration of the water heating phase, the durations of the water heating phases in each temperature look different, and it was based on this fact that new features were created. Along with the already existing features (Active Power, Voltage, Electric Current, Apparent Power and Power Factor), two more features were added: the duration of the program and the duration of the water heating phase. It was expected that the duration of the program would be enough to identify if centrifugation was enabled or not, as seen in Figure 42, and the duration of the water heating phase to discover which temperature was chosen.

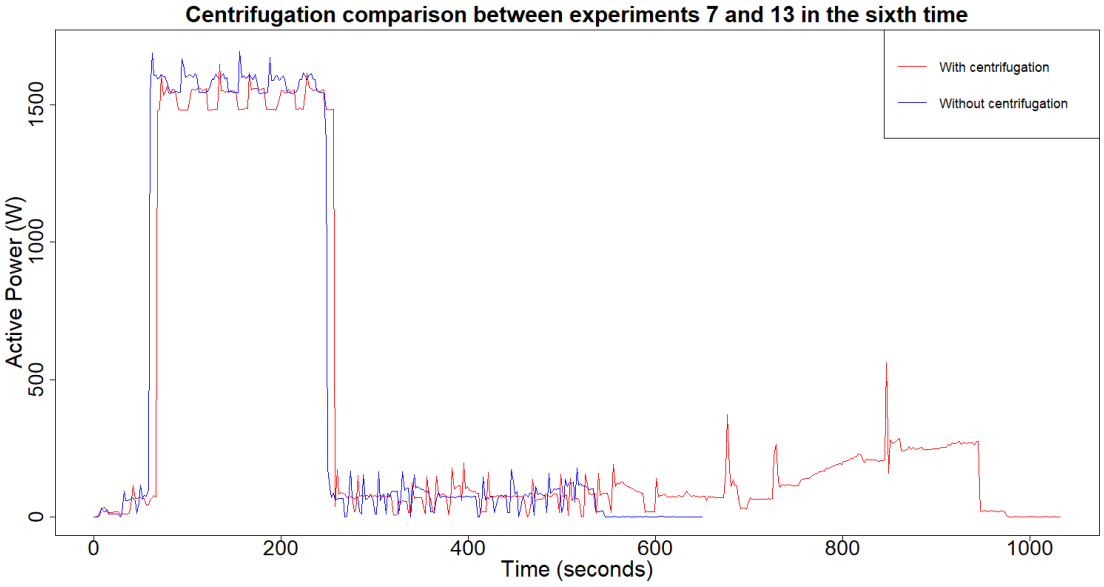

Figure 42 - Centrifugation comparison of "14 minutes" program experiments

The main reason found as to why the consumption pattern changes a lot between experiments is due to the clothes' weight. As can be seen in Figure 43, when the experiment was run with weight, the rotations phase could be longer or there are more peaks of consumption, among other.

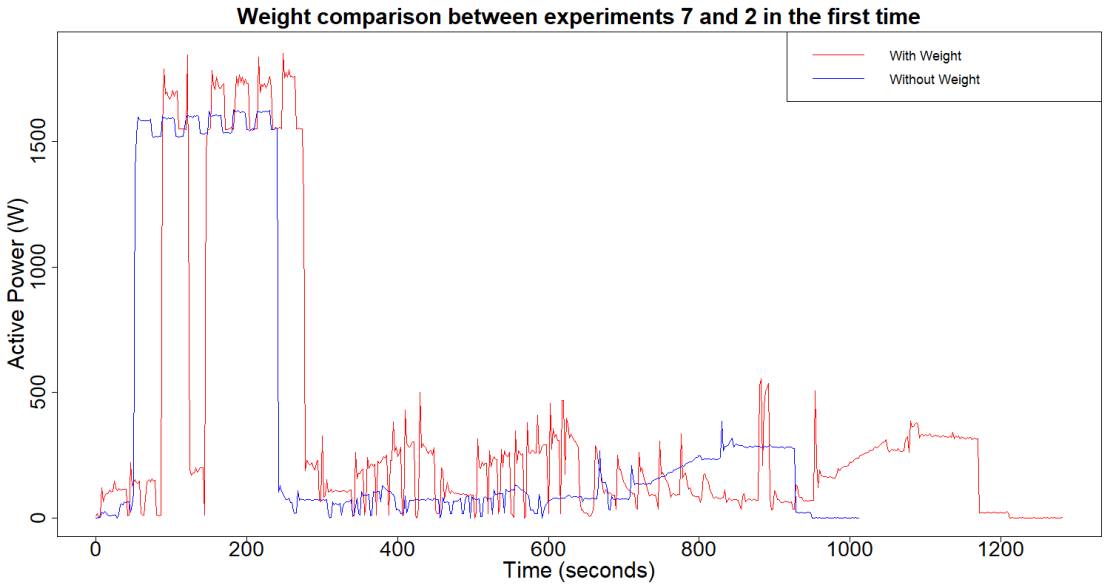

Figure 43 - Weight comparison of "14 minutes" program experiments

The sensor is only capable of acquiring electrical measurements, and it would be very intrusive to ask the user to input the weight of the clothes that were put inside the machine for every washing program. Another solution would be to consider using a larger confidence interval for classifying the consumption pattern as good, which implies dangerous consequences. Since the pattern of faulty energy consumption is not known, there is a danger of classifying a consumption pattern as good when, in reality, it is bad.

4.1.3 Building and testing models

The results of the tested classification algorithms are subsequently presented in order to describe how well these classify the washing program, the water temperature and if centrifugation was enabled or not.

4.1.3.1 Supervised Machine Learning algorithms

The "14 minutes" program was the first one to be given focus. Classification seemed the best way to go since the objective was to categorize the data into temperature and centrifugation. Logistic Regression, Naïve Bayes, K-Nearest Neighbor, Decision Tree, Random Forest and Support Vector Machine are the most commonly used classification algorithms; however, it was decided to test only four of these because of time constraints. These tests were made in

the KNIME platform to decide which classification algorithm seems to be the best for the classification of the washing machine program. The key word here is “seems”, because there is a possibility that different data mining tools or machine learning libraries could give different results.

4.1.3.1.1 Algorithms’ overview

As previously explained, only four prediction algorithms were tested. Logistic Regression was excluded since the predicted variable can only be binary and, while it could work for detecting if centrifugation was enabled or not, it will not work for detecting which temperature was used since the prediction can be 20, 30 or 40 degrees. Afterwards, since Random Forest uses multiple Decision Trees for its predictions, it can be more accurate than using a Decision Tree in cases of having large datasets [71]. Thus, for this reason, Decision Tree was excluded.

The next topics present a small description and the motivation to use Naïve Bayes [72], Support Vector Machine [73], K-Nearest Neighbors [74, 75] and Random Forest [76].

- Naïve Bayes:

This algorithm is based on Bayes’ theorem with an assumption of independence among every pair of features. In simple terms, a Naïve Bayes classifier assumes that the presence of a particular feature in a class is unrelated to the presence of any other feature. Even if these features depend on each other or upon the existence of the other features, all of these properties independently contribute to the probability. Naïve Bayes model is easy to build and particularly useful for very large data sets. Since the better features are unknown and according to the Naïve Bayes description, a correlation matrix was made (displayed in Figure 44).

The values measured by the sensor are Active Power, Electric Current and Voltage.

Active Power is highly correlated to Apparent Power since the Apparent Power is the square root of the sum of the square of Active Power and Reactive Power (this last variable is not important for this project).

Apparent Power and Electric Current is “1” because Apparent Power is the result of Voltage's multiplication with Electric Current and the Voltage values are almost constants (it varies between 230 and 241 volts).

Power Factor has the Active and Apparent Power as the highest features correlated because the Power Factor is the division between them, respectively, with some previous verification.

Timestamp and Epoch has high correlation since both refers to the time.

Heating is the duration of the water heating phase, i.e. when the active power was greater than 1000 watts, so the higher the Active Power, the higher the Heating is.

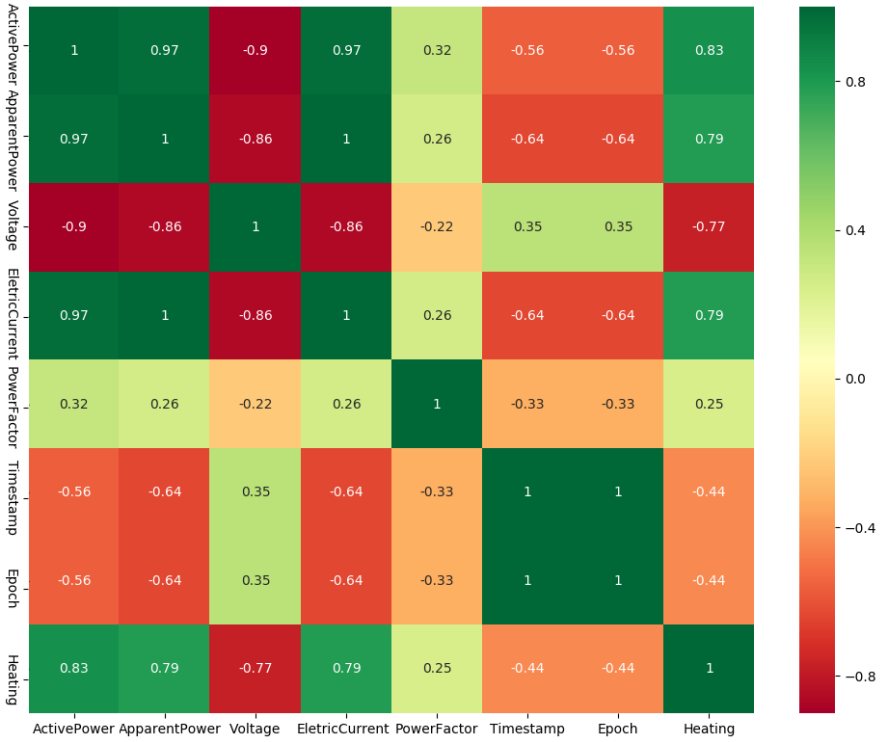

Figure 44 - Correlation matrix

Naïve Bayes algorithm assumes that each feature is independent of one another. However, it was proved previously that some of them are not, namely Active Power, Apparent Power and Electric Current. Table 6, Table 7 and Table 8 present the accuracy results of each algorithm, depending on the features provided. As can be seen across these results, initially, the models present low accuracies, but as the dependent features are reduced, the accuracy increases.

- Support Vector Machine (SVM):

The idea of SVM is to create a line or a hyperplane which separates the data into classes. The values closest to the hyperplane should be as far from the hyperplane as possible, since if this does not happen, the data becomes more difficult to classify and the probability that the data will be misclassified is higher. The effectiveness of an SVM depends upon three parameters that reduce error and overfitting. In Table 6, Table 7 and Table 8, SVM were less effective since the three parameters were not optimized.

- K-Nearest Neighbors (KNN):

K-Nearest Neighbors uses the k nearest/closest labelled points to learn how to label the new points, where the “k” is the number of neighbors it checks. To label a new point, it looks at

the labelled points closest to that new point (those are its nearest neighbors), and has those neighbors vote, so whichever label most of the neighbors have is the label for the new point.

- Random Forest

Random Forest operates by constructing a multitude of decision trees at training time and outputting the class that is the mode of the classes (classification) or mean prediction (regression) of the individual trees. Random decision forests correct the decision trees' habit of overfitting to their training set.

4.1.3.1.2 Algorithms' results

As seen previously in Figure 40, the higher the temperature, the higher is the duration of high values of consumption, so knowing the duration of water heating phase, the data could be separated by the three temperatures (in the case of the "14 minutes" program). The same goes for centrifugation, in Figure 42 there is a duration difference in the whole program.

Given the short project deadline, a quick decision was needed for the problem at hand: which prediction algorithm better classifies the data? Thus, it was decided that, firstly, the four algorithms would be tested without making any feature engineering. This leads to poor accuracy results for Naïve Bayes and SVM, and excellent accuracy results for KNN and Random Forest, presented in Table 6.

Table 6 - Accuracy results of the models without feature engineering

Algorithm	Naïve Bayes	SVM	KNN	Random Forest
Accuracy	25.9 %	8.9 %	99.8 %	92.5 %

The fact that there is no constant value in the entire program which causes the algorithm to predict the program every two seconds, i.e. when given a certain program, one and only one answer was not obtained, as can be seen comparing the two columns in the Figure 45.

14min_30_WC	14min_40_WOC
14min_30_WC	14min_40_WOC
14min_30_WC	14min_40_WOC
14min_30_WC	14min_40_WOC
14min_30_WC	14min_40_WOC
14min_30_WC	14min_40_WC
14min_30_WC	14min_20_WC
14min_30_WC	14min_20_WC
14min_30_WC	14min_30_WC
14min_30_WC	14min_20_WC
14min_30_WC	14min_20_WC
14min_30_WC	14min_20_WC
14min_30_WC	14min_20_WC
14min_30_WC	14min_30_WC
14min_30_WC	14min_20_WC

Figure 45 – Actual and prediction results of experiment 1

As previously analyzed in Section 4.1.2.3, the full duration of the program and the water heating phase were highly considered to be good features, so they were calculated, a new model was created by adding these two features, and the results improved considerably, as shown in Table 7Table 7 - Accuracy results of the models with feature engineering.

Table 7 - Accuracy results of the models with feature engineering

Algorithm	Naïve Bayes	SVM	KNN	Random Forest
Accuracy	51.3 %	37.8 %	99.8 %	100 %

Thus, the performance of the four algorithms was tested only with the Active Power, Water Heating and Program durations features. As mentioned in the previous section, reducing the number of features since there are dependencies between them will probably help the performance of the algorithm. The results still improved further as Table 8 shows.

Table 8 - Accuracy results of the models with the features number reduced

Algorithm	Naïve Bayes	SVM	KNN	Random Forest
Accuracy	100 %	92.4 %	99.9 %	100 %

As seen in **Error! Reference source not found.**, the "14 minutes" program was repeated six times for each experiment. Three repetitions were used as the dataset to construct the models partitioning 70% for training set and 30% to test set. After seeing the values in Table 8, the models created were saved and the other repetitions were used to check if the models' accuracy remained very good or not. Due to time constraints, only the Random Forest were checked, and it predicts all programs well.

4.1.3.1.3 Error simulation to new algorithm evaluation

In a best-case scenario, an integrated system would have been implemented and data from malfunctioning home appliances would be acquired, but this was not possible due to time constrains.

It was important to make sure that the prediction models would be able to correctly classify bad consumptions. Therefore, all repetitions of all collected experiments were observed again, and some repetitions were found in a set of experiments (1, 5, 7, 15, 22 and 23) that somewhat escaped all other repetitions, as shown in following 3 topics:

- In the 1st repetition of experiment 1 in Figure 46 and in the 5th repetition of experiment 5 in Figure 47, the water heating phase took longer than it was supposed to.

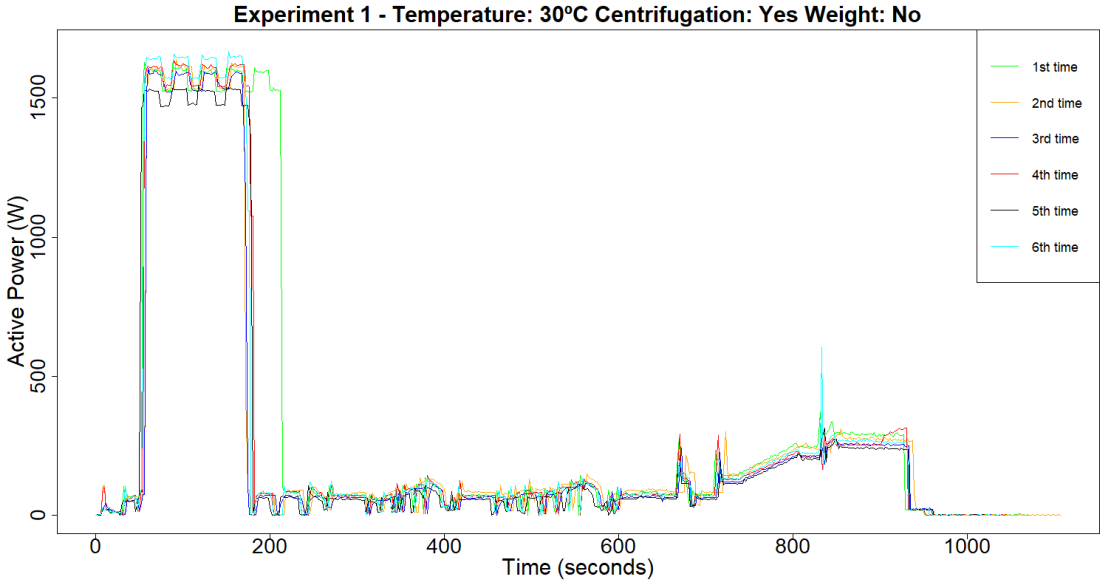

Figure 46 - All repetitions of experiment 1

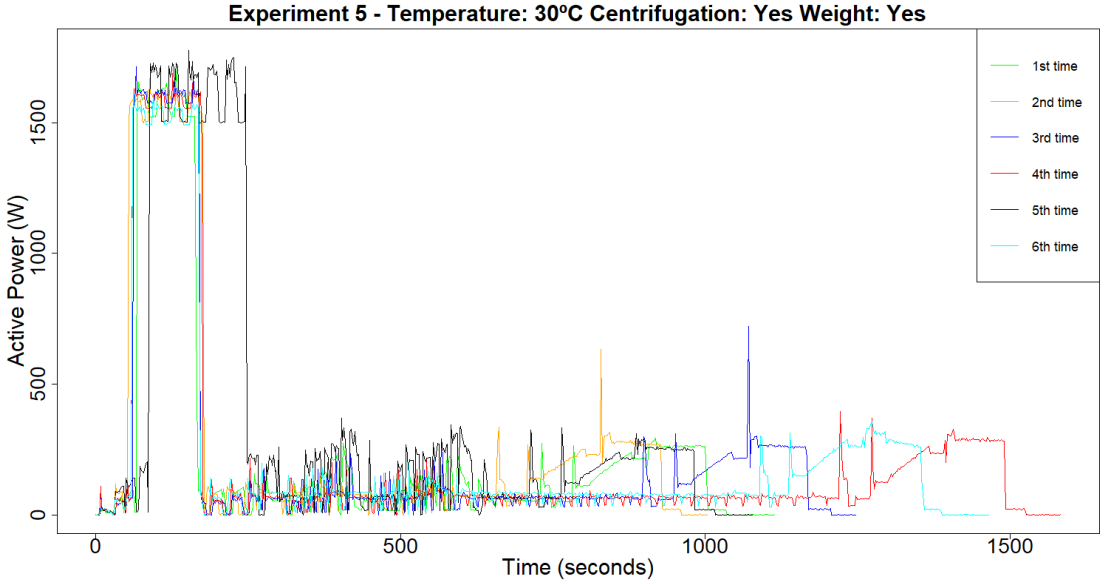

Figure 47 - All repetitions of experiment 5

- In the 1st repetition of experiment 7 in Figure 48 and in the 2nd repetition of experiment 15 in Figure 49, it seems that the experiment was moved in time.

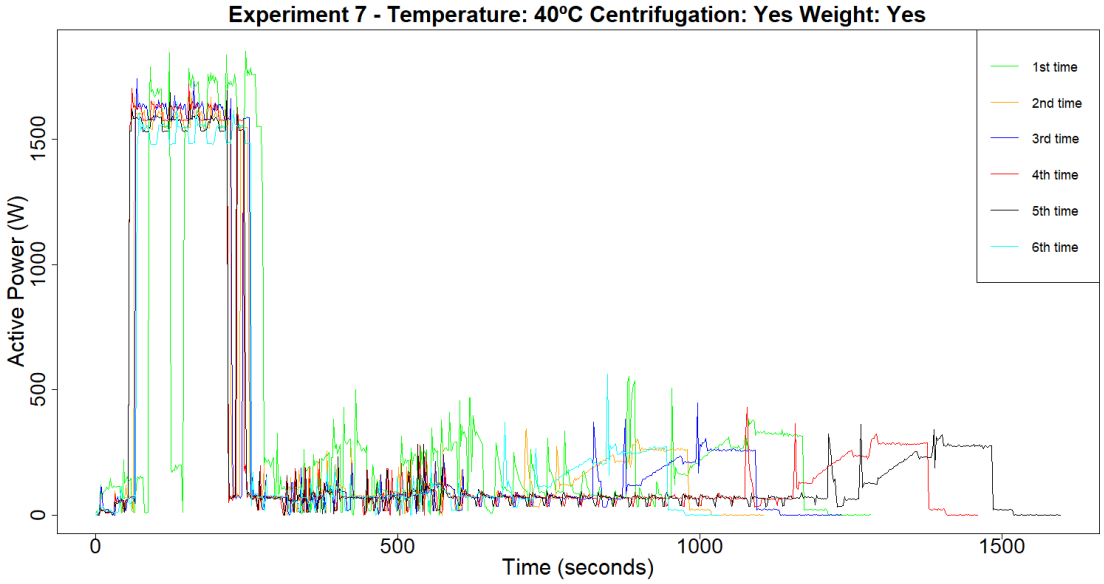

Figure 48 - All repetitions of experiment 7

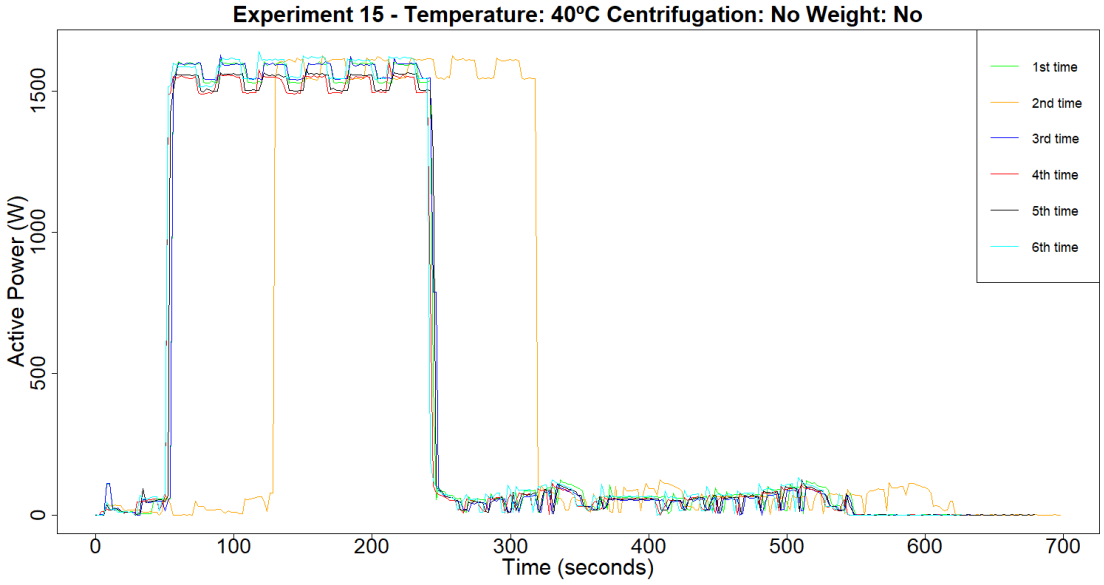

Figure 49 - All repetitions of experiment 15

- In the 6th repetition of experiment 22 in Figure 50 and in the 1st repetition of experiment 15 in Figure 51, the water heating phase started earlier than the rest.

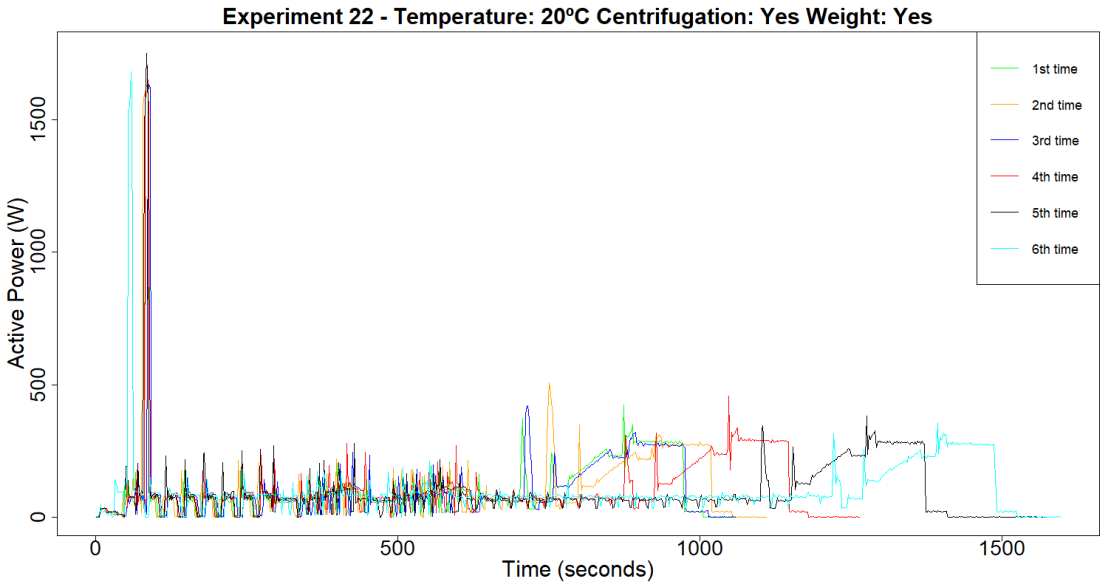

Figure 50 - All repetitions of experiment 22

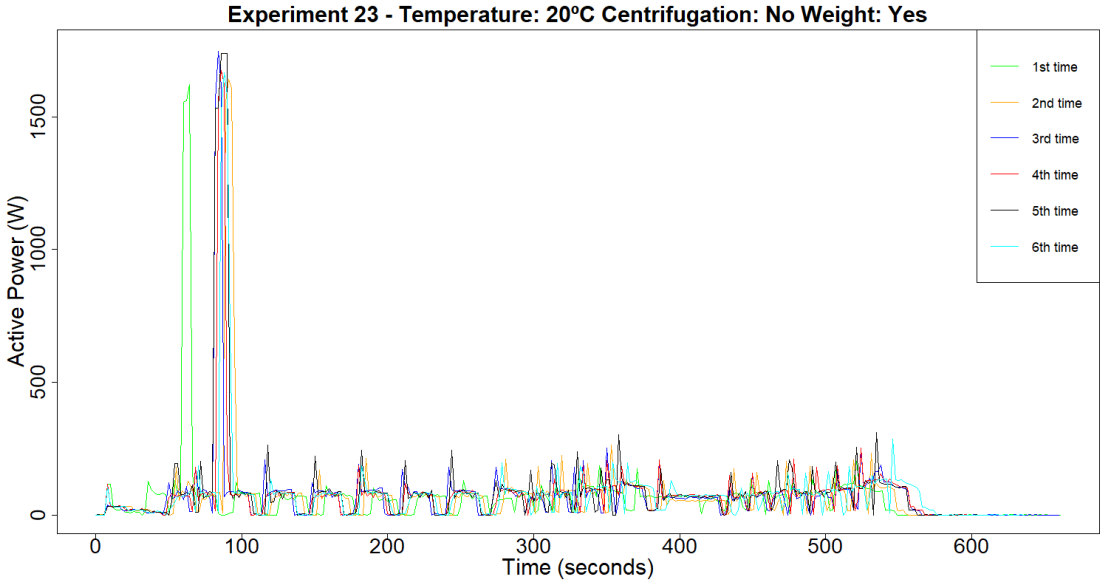

Figure 51 - All repetitions of experiment 23

Each of the above set of experiments repetitions referred were classified as “Error” and three new prediction models were constructed. In Table 9, the first two rows specify the dataset used to create the three prediction models, i.e. the first model was created with all experiments made in the first, second and third repetitions. The values written in blue means that the prediction model already knew that experiment. The last six rows present the accuracy of the predictions made by the model to classify the experiments written in the first column, i.e. in the third prediction model that was built with the 1st repetition of all experiments, the prediction model was able to classify the 1st repetition of experiment 1 well, as expected since the model had already seen that experiment at the time of its construction.

Fortunately, it also was able to classify the 5th repetition of experiment 5 well even though it had never seen it. The results discussion can be found after Table 9.

Random Forest was the chosen algorithm since the accuracies were always high independently of the features. KNN was also a good candidate, but the platform used to speed up the testing process of these algorithms does not have an option to save the KNN model, which made it impossible to make any test. One way to ensure that the model is not overfitting is giving data to the prediction model which the prediction model has never seen and if it is misclassifying, it is an indication that the model suffers from overfitting, i.e. the prediction model does much better on training data than on test set; but if it classifies well, in principle the model is good. As KNN always had excellent accuracies but cannot save the model, it was decided not to use it as there would always be a doubt whether the model suffered from overfitting or not.

Table 9 – Accuracy results of the models with simulated errors

Prediction model:	1st	2nd	3rd
Dataset used:	1st, 2nd and 3rd repetitions	2nd, 5th and 6th repetitions	1st repetition
1st rep. of experiment 1	100%	11.25%	100%
5th rep. of experiment 5	8.88%	100%	100%
1st rep. of experiment 7	96.13%	0%	100%
2nd rep. of experiment 15	94.17%	100%	0%
6th rep. of experiment 22	0%	100%	0%
1st rep. of experiment 23	100%	0%	100%

Generally, the results were poor mainly because there were few examples of errors provided to the model, and so the prediction model struggles between either classifying the data as an error or as the program that it actually was. Analyzing by prediction models:

- 1st: In the six experiments classified as error, the prediction model already knew 4 (experiment 1, 7, 15 and 23) of the 6 experiments. Regardless, even after already knowing 4 of these experiments, only 2 of them had 100% accuracy (experiment 1 and 23), while in the other half (experiment 7 and 15), there was a small percentage that incorrectly classified it as a normal program and not as a program with an “Error”. These 6 experiments are grouped two by two, by error similarity as described above. In the case of experiment 5, the model could not classify it as an error, even after

being trained with experiment 1's dataset, which contains the same problem of the water heating phase taking longer than it is supposed to. This is perhaps due to a disproportionate amount of data from good and bad consumptions. In the case of experiment 6, the reason why the prediction model did not classify it as "Error" may have been due to the existence of a centrifugation phase between experiments 22 (which had centrifugation) and 23 (which did not have centrifugation), as shown in the Figure 50 and Figure 51;

- 2nd: The reason as to why the model fails to correctly classify experiment 1 is, once again, perhaps due to training data disproportion (i.e., the dataset had a much higher amount of good consumption examples, than of bad consumption examples). As for experiments 7 and 23, the same centrifugation issue present in the 1st model happens again in the 2nd model. Namely, one example of the error has centrifugation enabled (experiment 7 and 22), while the others do not (experiment 15 and 23);
- 3rd: Since the proportion of good and bad data was more evenly balanced, the model was able to classify experiment 5 as "Error". For experiments 15 and 22, the accuracy ratio was the same as given for experiments 7 and 23 in the 2nd prediction model for the same reasons.

Furthermore, while other types of hypothetical errors could have been created (by altering the consumption data) in order to tune the prediction models, it was instead decided to try a different approach using pattern matching. While pattern matching knowledge was very limited, the goal of this approach was to check if it was possible to recognize a consumption pattern through convolutions or correlation.

4.1.3.2 Pattern Matching techniques

There are two simple pattern matching techniques: convolution and correlation [77]. A convolution is an integral that expresses the amount of overlap of one function *g* as it is shifted over another function *f*. Mathematically, a convolution is defined as a product of functions *f* and *g*. The mathematical calculation of correlation is same as convolution in time domain, except that the signal is not reversed, before the multiplication process [78]. In paper [77], it is presented a study made to conclude which method (between convolution and correlation) is better to count the number of objects that are present inside an image. The conclusion was that convolution has greater speed. For this reason, convolution was used to test if it is possible to find the centrifugation phase to distinguished if an experiment has centrifugation or not.

Consider that, from Figure 52 to Figure 54, *f* is the original pulse, *g* is the filter impulse response and the result of the convolution is the filtered signal.

For the first problem, it was given an entire program (see Figure 52). Unexpectedly, when the filtered impulse response passes through the water heating phase, it was given very high values and it was concluded that to know if there was centrifugation or not, the better thing to do is cutting the high values of the water heating phase, as shown in Figure 53.

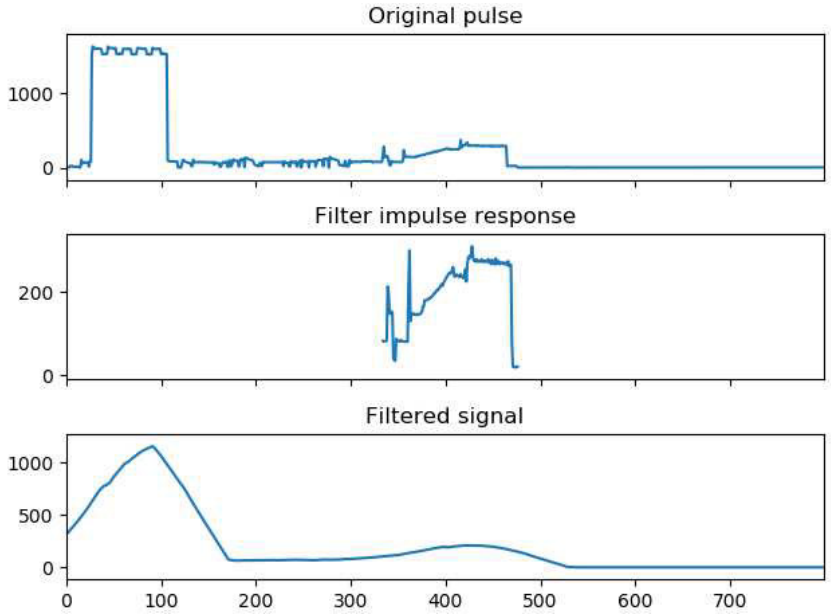

Figure 52 - Convolution to find centrifugation phase

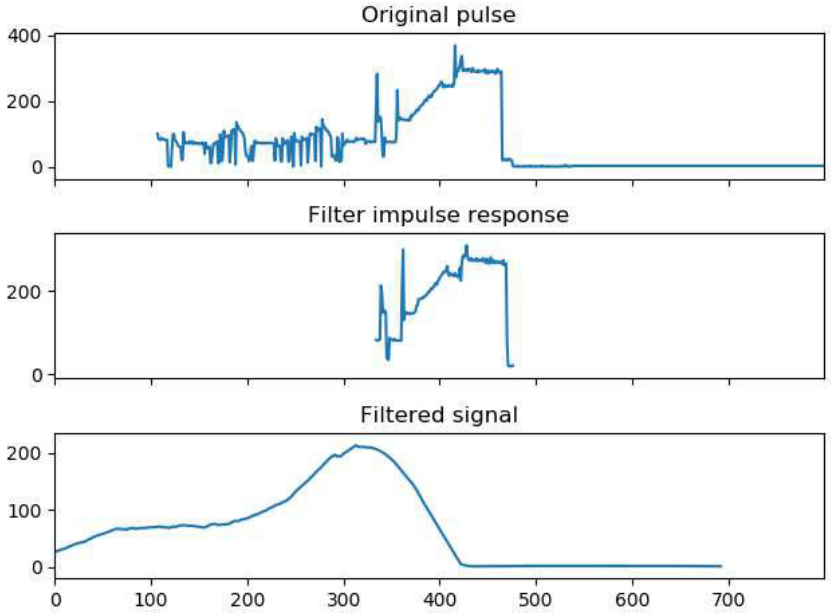

Figure 53 - Convolution to find centrifugation phase without water heating phase

This way, there is a slight peak that reaches values near of 200. To be sure that convolution results, it was given a program without the water heating and centrifugation phase (see Figure

54). There was a peak too, however the maximum value was near to 50. So, if it was defined a limit, e.g. 100 and then it was verified if when applying convolution, the results of its application exceeded 100, it means that in that program/experiment the centrifugation was activated.

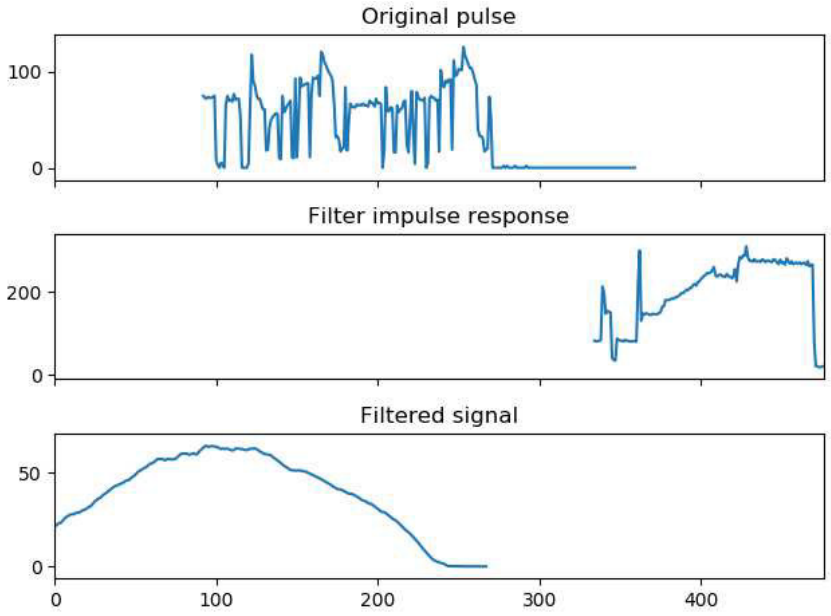

Figure 54 - Convolution to find centrifugation phase without water heating and centrifugation phase

The results for the centrifugation phase search in the other programs are in Appendix 7.2.

Since it was possible to identify the centrifugation phase through convolution technique, it was speculated that this technique would be capable to detect programs, e.g. in a day. Unfortunately, the results were not as predicted and so, a different path was taken, described in the next section.

4.1.3.3 Washing machine programs classification

After all the attempts previously described to identify which program were run per day and to classify them, it was concluded that the most successful technique was the supervised machine learning algorithm Random Forest.

It was first necessary to distinguish when the machine was running a program from when it was not. Since the start time and end time of the program is not defined by anyone, the consumption "Running a program" of "No running program" was distinguished based on the appliance's energy consumption. All programs analyzed had the particularity of when they started running a program, the first value was over 17 watts. During the rotation phase, there were time intervals where the consumption was in the range 0 to 3 watts. When the program

ended, the consumption values varies between 0 to 3 watts as well. So, it was calculated the maximum time interval, during the rotations phase, where the consumption values ranged between 0 and 3. Then, it was defined a limit higher than the maximum time interval. If any other interval was higher than that limit, it meant the program ended.

The beginning and end of a program were defined like this and after obtaining these two timestamps it was possible to give the model the consumption of a program that ran later to classify it.

As shown in Table 5, there are two programs that were only repeated twice ("30 minutes" and "Cottons + Prewash"), and to make sure that the model would classify the data well but not because of knowing it, only the first repetition of all experiments for each program was given to train the model. In 128 measured experiments, the predictive model already knew 36 programs since it was trained with the first experiment of each program, which means that for "14 minutes" and "30 minutes" was 12 experiments, "Coloured" was 8 experiments and "Cottons + Prewash" was 4 experiments, so 12+12+8+4=36. So, there was 128-36=92 programs to predict. By the amount of data that was given to the model to learn to distinguish the 4 programs, the temperatures and whether there was centrifugation or not, the model even behaved well (see Table 10).

Table 10 – Accuracy results percentage of model's creation with only first repetition

Pro-gram	14 minutes							30 minutes			Coloured				Cottons + Prewash		
Time		1st	2nd	3rd	4th	5th	6th		1st	2nd		1st	2nd	3rd		1st	2nd
Experiments	1							1			1				1		
	2							2			2				2		
	5							3			3				3		
	7							4			4				4		
	12							5			5						
	13							6			6						
	14							7			7						
	15							8			8						
	21							9									
	22							10									
	23							11									
	24							12									

Accuracy Range

0-20%	20-40%	40-60%	60-80%	80-100%

To ensure that the data analysis component did its job, a preliminary solution was developed. This solution consisted in writing a Python script that gets invoked by the Web Application. However, this alternative lacks testing and polish, and therefore was not used for this project.

4.2 Refrigerator

This section is divided into three subsections: Section 4.2.1 which describes the experiments that were done with the refrigerator; Section 4.2.2 that presents the analysis done of three refrigerators consumption patterns and the data processing of one of the three; and Section 4.2.3 shows a possible behavior of a supervised machine learning algorithm using the data processed.

4.2.1 Data

The refrigerator operation can depend on a several variables:

- The temperature inside the refrigerator;
- The temperature outside the refrigerator;
- The number of products inside the refrigerator;
- The number of times that the refrigerator door was opened.

These variables are introduced because if they change over a certain period of time and this change is reflected in the consumption pattern, this difference cannot be classified as a failure. However, only the temperature variation inside the refrigerator was submitted to experiments. To vary the temperature outside the refrigerator, it will be required a large amount of effort since the refrigerator is located in an open space and it was not possible to move it to another place. For the number of products inside the refrigerator, it was decided to ignore this parameter, since the only way to get that information, in a real environment, was to prompt the user. Lastly, the number of times that the refrigerator door was opened was recorded for two days, however this action did not cause much difference in the consumption pattern, being that the only difference noted was when the compressor was stopped. Furthermore, to acquire this type of data the system would have to prompt the user, which would go against the project's goals.

Briefly, the experiments analyzed in this report consisted in simulating failures, so that it would then be possible to conclude which failures could be detected through the refrigerator's consumption. The experiments made were:

Table 11 - Refrigerator experiments

Experiment number	Experiment description
1	Open refrigerator door.
2	Open freezer door.
3	Open both (freezer and refrigerator) doors.
4	Change the inside temperature through the controller.

In the first three experiments, the doors were opened for about 5 minutes. In the fourth experiment, the inside temperature of the refrigerator was measured with a digital thermometer.

Most of the time, the data was measured from CISTER's Floor 0 refrigerator. However, the consumption of two other refrigerators was measured for two days. From here on, the refrigerators will be referred by its location to distinguish between them, i.e. CISTER's Floor 0 refrigerator will naturally be referred to as "CISTER's Floor 0 refrigerator", and the other two will be "CISTER's Floor 1 refrigerator" and "Residence refrigerator".

The latter two refrigerators were not tested any longer because no experiment was possible/conceivable to test due to their locations, and also due to the fact that the use of these refrigerators could not be controlled/supervised at all times. In addition, there was no similarity in brand or model (see Table 12), which did not help because it should have been proved that within the same model, the consumption pattern was nearly the same. Although, the consumption pattern of the three refrigerators is presented in Section 4.2.2.

Table 12 - Refrigerators' brand and model

Refrigerator	Brand	Model
CISTER's Floor 0	INDESIT	RAA 24 N
CISTER's Floor 1	WHIRLPOOL (IKEA)	CB304W
Residence	FAGOR	– [9]

4.2.2 Analysis

As shown in Figure 55, Figure 56 and Figure 57, the refrigerators have a similar consumption pattern. For each of the three cases, two situations of the compressor running are presented, e.g. in the case presented in Figure 55, the compressor was running from 23:38 until 23:59 and from 00:46 to 01:07. When the compressor starts running, the active power values can reach 1000 watts therefore, Figure 55, Figure 56 and Figure 57 were limited to 200 watts in the Active Power axis for a better visualization of the consumption curve[10]. In Figure 56, the peaks at 11:58, 12:03, 12:08 and 12:12 are not errors. Sometimes there are some energy peaks that happen due to the times when the refrigerator door was opened for a short period of time, i.e. the more times the refrigerator is used, the more peaks there will be.

[9] Unable to obtain Residence refrigerator model.

[10] Consider that the consumption curve is from the peak energy point (from when compressor started) to the point immediately before returning to zero energy consumption (when compressor stops run).

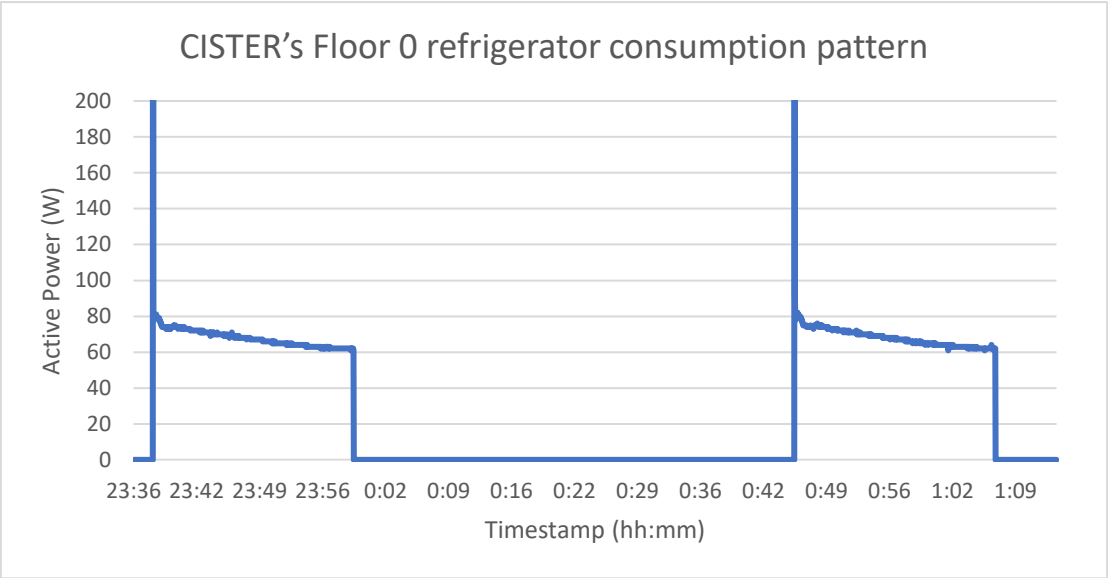

Figure 55 – CISTER's Floor 0 refrigerator consumption pattern

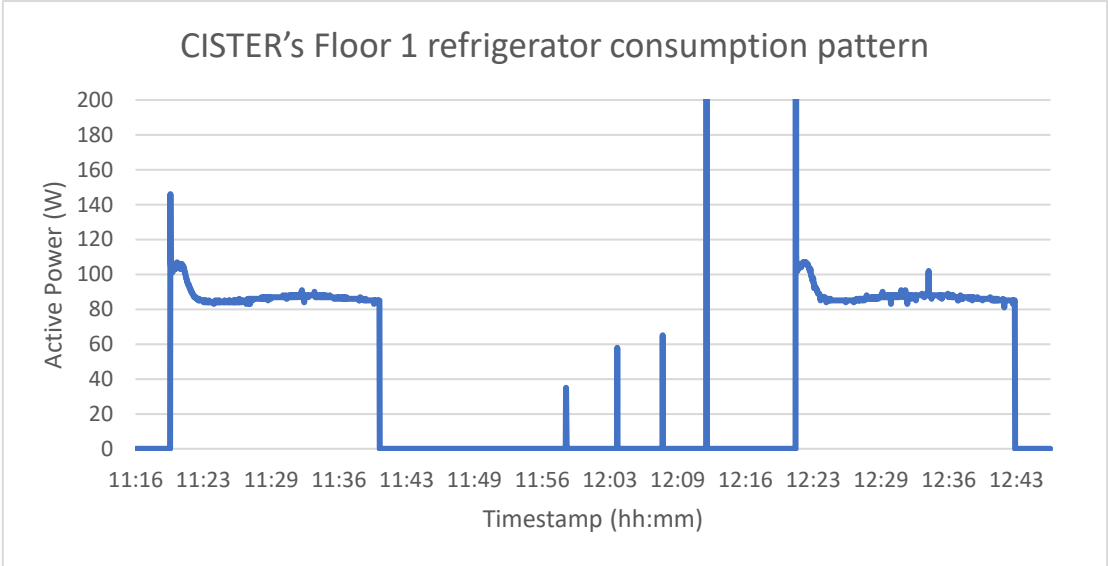

Figure 56 – CISTER's Floor 1 refrigerator consumption pattern

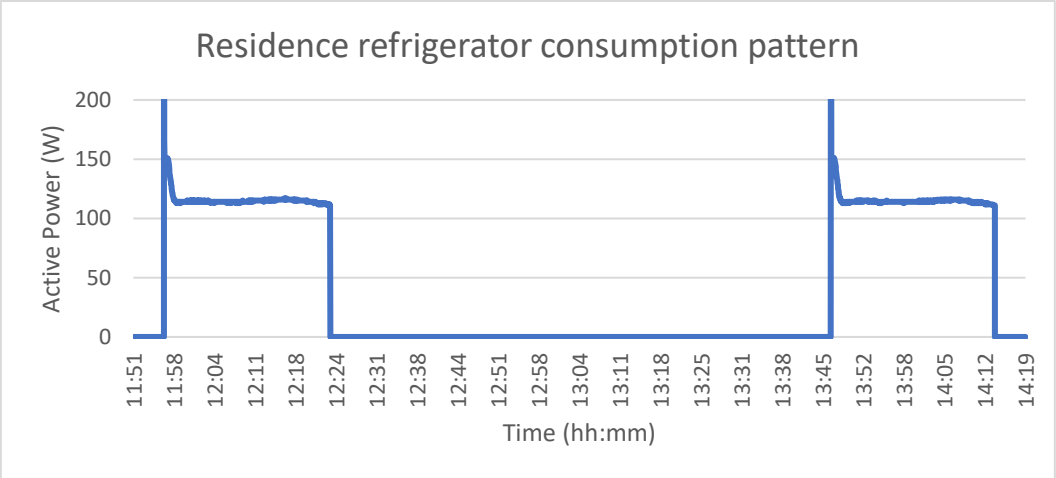

Figure 57 – Residence refrigerator consumption pattern

In essence, the main differences between refrigerators are the consumption curve(s) and the duration while the compressor is running or not running. After some analysis of this data, it was discovered that some information was lost. In this case, there was a bigger effort since the refrigerator functioning is continuous and there was a loss of data for about 20 seconds (and more) in some situations.

To avoid the risk of using a dataset with loss of data, some R code was developed to separate the consumption by days or hours (depending on the on the size of the data loss). After that division, the dataset was verified again for timestamps greater than 3, then, if this did not happen, a new spreadsheet file was created where the spreadsheet name was the start and end date and time of the data stored in excel, as shown in Figure 58.

```
day11_12 <- allData[275806:293557,]
day11_12 <- day11_12[order(dia11_12$Timestamp),]
plot(day11_12$Timestamp,day11_12$ActivePower)
day11_12[which(diff(day11_12$Timestamp) > 3),]
allData <- allData[0:275806,]
path1 = paste0(dirname(rstudioapi::getSourceEditorContext()$path),
               "/11_06_17h27__12_06_03h19.csv")
write.table(day11_12[,2:8], path1, sep=",", row.names = FALSE)
```

Figure 58 - Data processing example

The data was initially stored in a spreadsheet as it facilitated data sharing for the Azure solution developer to circumvent the 8000-message restriction.

When one of the three first experiments were made, the error column was rewritten with the value 1, as in the two last lines of Figure 59.

```
refrigeratorData$Error <- 0
refrigeratorData[17751:17901,10] <- 1
refrigeratorData[18827:18976,10] <- 1
```

Figure 59 - Data classification

In an initial phase, the value “0” in the error column corresponds to a correct function of the refrigerator while “1” signals that something went wrong. However, the objective is that the values different from 0 correspond to a specific error, as previously referred in Section 3.1.2.2.

The refrigerator has an advantage over the washing machine: it is easier to simulate failures. While with the washing machine, it is not possible to simulate failures without breaking some components, with the refrigerator it is possible to leave the door open which can be considered a failure that should be notified to the user. Figure 60 presents the results of two experiments, 3 and 1 respectively. At 11:59, both doors were opened (experiment 3), and at 12:04 were closed. At 13:24, the refrigerator door was opened (experiment 1), and at 13:29 the door was closed. Given these two facts and taking into account Figure 60, it can be concluded that this failure is detectable only through energy consumption, since the

consumption increases and decreases at almost the same time as the door was opened and closed (it takes 30 seconds approximately to see the consumption variation). The consumption between having the refrigerator or the freezer door open is different. When the refrigerator door is open, the active power is higher than expected during that time. While when the freezer door is open, the active power increases slightly (not exponentially) and the duration when the compressor is running increases, hence the reason why the consumption curves shown in Figure 60 have different durations.

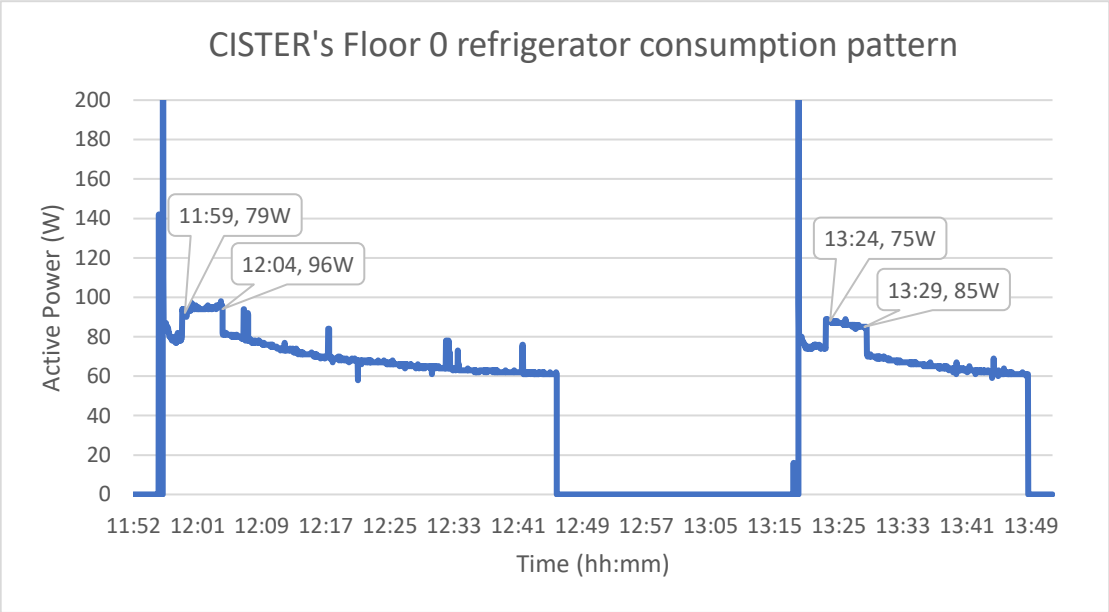

Figure 60 - CISTER's Floor 0 refrigerator consumption pattern with failures

4.2.3 Building and testing models

As previously explained, there are two people working on this project and due to time constraints, the building and testing of the prediction models for each home appliance was divided. This report has a more complete version of the models built and tested for the washing machine (details on Section 4.1.3), while [70] has those for the refrigerator. However, before the work division took place, a Polynomial Regression model had already been built and observed without major conclusions as to how viable it would be. Admittedly, the best algorithms to apply would be those specific for time series data [70].

The idea for the Polynomial Regression was rather complex. Beyond the features presented in Figure 28 (i.e., Active Power, Apparent Power, Voltage, Electric Current, Power Factor, Timestamp and Milliseconds), the duration of both compressor states (running or not) was calculated. Regardless of whether the compressor is running or not, the duration is verified. If the compressor is running, Polynomial Regression will evaluate the consumption curve, and if

there was a considerable difference between the values measured and the values of the Polynomial Regression model, a notification will appear. The data of one day was sufficient to test this theory. In R, the data was divided in good or bad consumption curves. In Python, the good consumption curves were used to create the Polynomial Regression model and the bad consumption curves were used to discover if there was enough difference to find the failures. The goal with Polynomial Regression was to overfit the model so that small variations between good consumption curves would not be reported but when the difference exceed a certain limit, would be reported. Figure 61 shows part of 6 curves, 4 with good consumption curve (blue, red, green and yellow lines) and 2 where the door was left open for about 5 minutes called bad consumption curve (black and cyan lines). As it is can be seen, it would be possible to define a hypothesis function that will determines what the normal consumption was (ignoring the peak) while the compressor is running. If it was set that the received values exceeded 10-15 watts of what is supposed, a notification would appear. It was said 10-15 watts since the difference between the black/cyan line(s) and the blue/red/green/yellow line(s) derives approximately in that interval. Unfortunately, it was not possible overfit the model, so this theory was not viable. Although, even if it works, it was necessary to make enough calculus. As it was said previously, time series would be the next type of algorithms to be tested, but a new division of work was made, and the results are presented in the report [70].

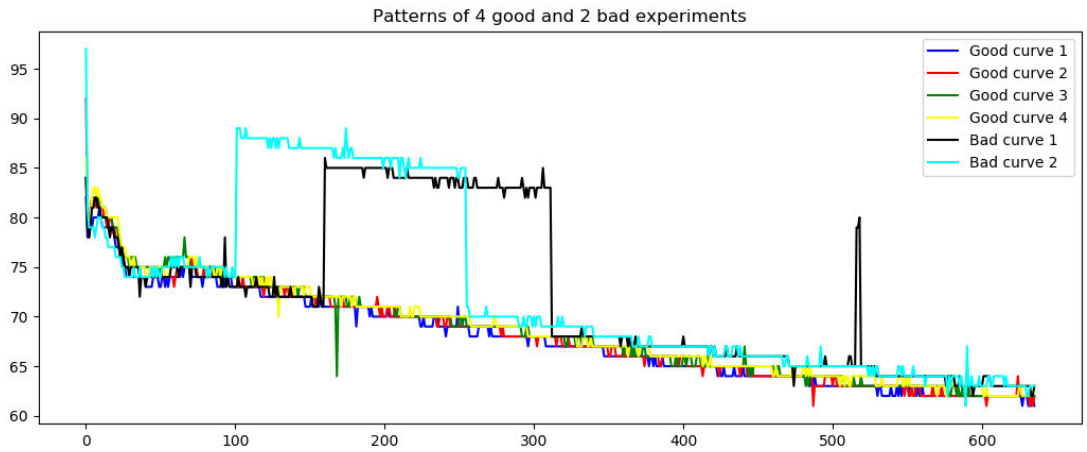

Figure 61 - Comparison between good and bad consumptions curves

5 Conclusions

The overall goal for the project was the creation of a system that was capable to identify energy consumption patterns and (potential) failures in home appliances, focusing on the development of the Data Analysis component.

5.1 Summary

First, an analysis of an initial prototype was made (Section 3.1.1) and it was concluded that the prototype had to be modified and updated. A better Smart Connector was integrated into the system, capable of giving more accurate consumption measurements (Section 3.1.2.1), and a new Web Application was implemented (Section 3.1.2.2). The Web Application is capable of receiving and storing data from multiple Smart Connectors, regardless of their location, and also provides other functionalities to the user, such as adding, editing or removing a home appliance or viewing its consumption.

Then, the developed components were deployed in a production environment in order to acquire data from the home appliances, thus leading to the start of the data collection phase (Section 3.2).

After obtaining the data, regardless of whether it was in a development or production environment, everything was ready for consumption pattern analysis, data processing, and ultimately, to identify the best Machine Learning techniques for classifying consumption patterns or identifying/predicting failures.

A great deal of time was spent cleaning up the acquired data from the initial prototype and taking advantage of it. This data was from a refrigerator; however, this was the first appliance to be analyzed and given the simplicity of the consumption pattern, it facilitated the understanding of the consumption pattern. Section 4.2.2 presents the behavior of the refrigerator's consumption pattern. Due to time constraints, the construction and performance analysis of the models was not performed.

However, for the washing machine, four ML classification algorithms were tested and found that Random Forest worked quite well. Pattern matching techniques were also applied where they also had good results for certain purposes.

5.2 Goals

The degree of the goals accomplishment are presented in Table 13, The first and second goals were described on Section 3.1.2. The third goal has been partially achieved since the solutions developed do not guarantee that identify or predict failures (Section 4.1.3). The accomplishment of the last goal is presented in Section 4.1.3.3.

Table 13 - Goals' accomplishments

Goal	Degree of Accomplishment
Acquire a home appliance's power consumption through a *Smart Connector* to the cloud.	Accomplished
Develop interfaces, i.e. a *Web Application* that can be used as a cloud service, providing supporting features (e.g. showing home appliances' consumption patterns), storing consumption information.	Accomplished
Develop a framework to detect home appliances failures, based on Machine Learning techniques, which would be able to discover patterns in a home appliance's consumption, and later, detect anomalies in near real-time.	Partially accomplished (75% completion)
Allow the system to monitor the home appliances by comparing new data with the learned consumption patterns.	Accomplished

5.3 Limitations

The project suffered from a few limitations since there are no examples of malfunctioning home appliance consumption data, therefore it was difficult to come to a final conclusion on Machine Learning or Pattern Matching techniques that were viable enough to detect anomalies. This limitation also undermined the achievement of the third goal.

Another limitation would be the number of home appliances tested. In the case of the refrigerator, it was possible to obtain the data from 3 different refrigerators. While in the washing machine, it was not possible to obtain the data from another washing machine.

Lastly, it was very important to receive as much data as possible in the shortest possible timeframe, and the fact that the data was measured only every 2 seconds may have hindered pattern identification and hence error identification.

However, all the work done will be used in the future as there is already a thorough understanding of consumption patterns and what may affect them.

5.4 Future Work

In Section 3.1, the Android application was mentioned as one of the components that could have suffered some improvements. At this point, when the button "SELECT YOUR NETWORK" is clicked, the application opens another window with a list of SSIDs of the Wi-Fi networks that the smartphone was able to find. After selecting a network, the application returns to the main page shown in Figure 62 and the home user enters the password. However, some Wi-Fi networks may also require a username, not just a password. That said, it would be best that when the button "SELECT YOUR NETWORK" was clicked, the app would redirect (if possible) to the smartphone's own native Wi-Fi list, removing the network selection responsibility from the Android application.

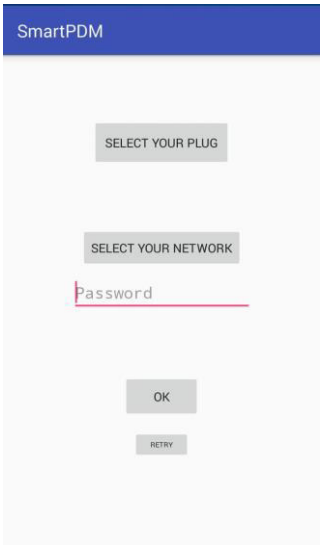

Figure 62 - Android application state

For the construction of the MQTT topic it was necessary to get more information from the user, namely the address. Therefore, a prototype of the Android application is displayed in Figure 63, where it is possible to register or login a user, save the address for the MQTT topic, give a name/small description to the home appliance, link the smart connector to the corresponding home appliance, connect the smart connector to a Wi-Fi network and, if it was not possible to identify the brand and the model through the appliance's energy consumption, the application will already support input text fields to save the brand, the model and, if necessary, the batch.

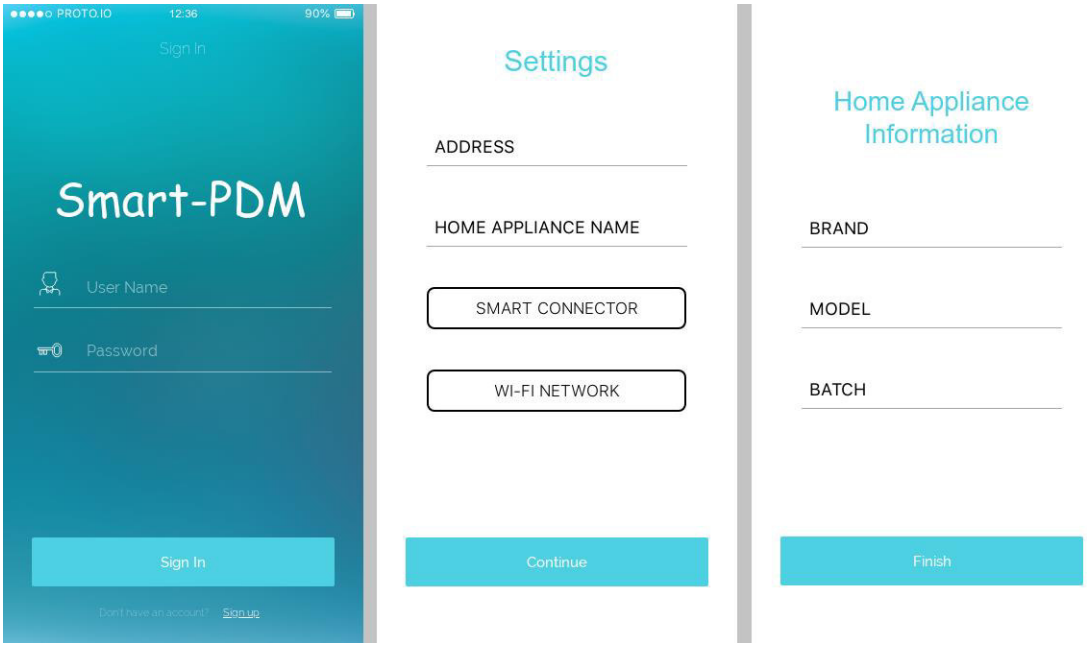

Figure 63 - Android application prototype

In Section 3.1.2.2, it was stated that the database was built considering only the refrigerator. It is necessary to know all the home appliances that the project partner wants to include in this project. After knowing that, new columns will be added to two tables only, since the features (i.e., the table's columns) will depend on each type of home appliance.

Another improvement would be the reprogramming of the Smart Connectors so that it could cut the power to the home appliance. Through this, the user could for example, in the case of a washing machine, cancel a washing program away from home, in case something goes very badly.

Regarding the data loss issue mentioned in Section 4.1.2 and 4.2.2, the cause for this happening was briefly studied and would be at the level of communication protocols, since it was necessary to use a Quality of Service (QoS) level 2, but the PubSubClient library [79] does not support it. QoS level 2 guarantees that each message is only received by the intended recipients, but the MQTT client, with the library used, only supports publishing at QoS 0 and subscribing at QoS 0 or QoS 1 [80]. A project which consists in the implementation of one library to publish and subscribe in the three QoS levels would be an asset.

6 References

[1] ITEA, “SMART-PDM A Smart Predictive Maintenance Approach based on Cyber Physical Systems,” ITEA, [Online]. Available: https://itea3.org/project/smart-pdm.html. [Acedido em 2019].

[2] “MANTIS Cyber Physical System Based Proactive Collaborative Maintenance,” [Online]. Available: https://www.cister.isep.ipp.pt/projects/mantis/. [Acedido em 02 05 2019].

[3] Circular Economy Portugal, “Projectos,” Circular Economy Portugal, [Online]. Available: https://www.circulareconomy.pt/projetos/.

[4] ITEAD Intelligent Systems Co.Ltd., “Sonoff Pow WiFi Switch With Power Consumption Measurement,” [Online]. Available: https://www.itead.cc/sonoff-pow.html. [Acedido em 14 05 2019].

[5] TeamGantt, “Intuitive and Beautiful Project Planning.,” TeamGantt, [Online]. Available: https://www.teamgantt.com.

[6] R. v. Kranenburg, The Internet of Things: A Critique of Ambient Technology and the All-Seeing Network of RFID, Amsterdam: Institute of Network Cultures, 2007.

[7] L. D. Xu, W. He e S. Li, “Internet of Things in Industries: A Survey,” *IEEE Transactions on Industrial Informatics,* vol. 10, nº 4, pp. 2233-2243, 2014.

[8] P. P. Ray, “A survey on Internet of Things architectures,” *Journal of King Saud University - Computer and Information Sciences,* vol. 30, nº 3, pp. 291-319, 2018.

[9] Virtual Power Solutions, “Cloogy,” [Online]. Available: https://www.vps.energy/cloogy. [Acedido em 22 05 2019].

[10] Virtual Power Solutions, “Kisense,” [Online]. Available: https://www.vps.energy/kisense. [Acedido em 22 05 2019].

[11] Eyedro Green Solutions, “Eyedro Home Electricity Monitors,” MyEyedro, 21 04 2017. [Online]. Available: http://eyedro.com/home-electricity-monitors/. [Acedido em 13 05 2019].

[12] TP-Link Technologies Co., “Wi-Fi Smart Plug with Energy Monitoring HS110,” [Online]. Available: https://www.tp-link.com/pt/home-networking/smart-plug/hs110/. [Acedido em 22 05 2019].

[13] “Sense,” Sense, 2019. [Online]. Available: https://sense.com. [Acedido em 13 05 2019].

[14] IEEE , “IEEE GET Program,” [Online]. Available: https://ieeexplore.ieee.org/browse/standards/get-program/page/series?id=68. [Acedido em 14 05 2019].

[15] eWeLink , “eWeLink Smart Home Center,” [Online]. Available: http://www.ewelink.cc/en/. [Acedido em 14 05 2019].

[16] NETSCOUT Systems, “Difference between TCP and UDP,” 25 09 2014. [Online]. Available: https://enterprise.netscout.com/edge/tech-tips/difference-between-tcp-and-udp. [Acedido em 14 05 2019].

[17] A. Hochstadt, “TCP vs UDP: Understanding the Difference,” [Online]. Available: https://www.vpnmentor.com/blog/tcp-vs-udp/. [Acedido em 14 05 2019].

[18] N. Naik, “Choice of Effective Messaging Protocols for IoT Systems: MQTT, CoAP, AMQP and HTTP,” pp. 3-6, 01 10 2017.

[19] J. Dizdarevic, F. Carpio, A. Jukan e X. Masip-Bruin, “A Survey of Communication Protocols for Internet of Things and Related Challenges of Fog and Cloud Computing Integration,” pp. 14-17, 27 02 2019.

[20] P. Patierno, “MQTT & IoT protocols comparison,” 19 02 2014. [Online]. Available: https://pt.slideshare.net/paolopat/mqtt-iot-protocols-comparison. [Acedido em 08 05 2019].

[21] Solace, “Understanding IoT Protocols – Matching your Requirements to the Right Option,” [Online]. Available: https://solace.com/blog/understanding-iot-protocols-matching-requirements-right-option/. [Acedido em 08 05 2019].

[22] N. Sakovich, “Internet of Things (IoT) Protocols and Connectivity Options: An Overview,” 22 08 2018. [Online]. Available: https://www.sam-solutions.com/blog/internet-of-things-iot-protocols-and-connectivity-options-an-overview/. [Acedido em 08 05 2019].

[23] S. P. Jaikar e K. R. Iyer, “A Survey of Messaging Protocols for IoT Systems,” *International Journal of Advanced in Management, Technology and Engineering Sciences,* vol. 8, nº 2249-7455, pp. 510-514, 2018.

[24] IBM Knowledge Center, “Why Use SSL?,” [Online]. Available: https://www.ibm.com/support/knowledgecenter/en/SSYKE2_8.0.0/com.ibm.java.security.component.80.doc/security-component/jsse2Docs/whyssl.html. [Acedido em 23 05 2019].

[25] S. Cope, “MQTT Publish and Subscribe Beginners Guide,” 10 10 2018. [Online]. Available: http://www.steves-internet-guide.com/mqtt-publish-subscribe/. [Acedido em 14 05 2019].

[26] Eclipse, “Eclipse Mosquitto,” [Online]. Available: https://mosquitto.org. [Acedido em 22 05 2019].

[27] M. Copeland, “What’s the Difference Between Artificial Intelligence, Machine Learning, and Deep Learning?,” 29 07 2016. [Online]. Available: https://blogs.nvidia.com/blog/2016/07/29/whats-difference-artificial-intelligence-machine-learning-deep-learning-ai/. [Acedido em 14 05 2019].

[28] E. Zhao, “Explaining the terms AI, ML, DL, DS,” 12 10 2018. [Online]. Available: https://medium.com/ds3ucsd/explaining-the-terms-ai-ml-dl-ds-b0ac43e99f55. [Acedido em 14 05 2019].

[29] D. Faggella, “What is Machine Learning?,” 19 02 2019. [Online]. Available: https://emerj.com/ai-glossary-terms/what-is-machine-learning/. [Acedido em 14 05 2019].

[30] A. Ng, “Machine Learning,” Stanford University, [Online]. Available: https://www.coursera.org/learn/machine-learning. [Acedido em 08 05 2019].

[31] J. Brownlee, “A Tour of The Most Popular Machine Learning Algorithms,” 25 November 2013. [Online]. Available: https://machinelearningmastery.com/a-tour-of-machine-learning-algorithms/.

[32] S. Bhatt, “Reinforcement Learning 101,” 19 March 2018. [Online]. Available: https://towardsdatascience.com/reinforcement-learning-101-e24b50e1d292.

[33] J. Brownlee, “Supervised and Unsupervised Machine Learning Algorithms,” 16 03 2016. [Online]. Available: https://machinelearningmastery.com/supervised-and-unsupervised-machine-learning-algorithms/. [Acedido em 09 05 2019].

[34] D. Fumo, “Types of Machine Learning Algorithms You Should Know,” 15 06 2017. [Online]. Available: https://towardsdatascience.com/types-of-machine-learning-algorithms-you-should-know-953a08248861. [Acedido em 09 05 2019].

[35] J. Frählich, “Supervised and unsupervised learning,” 2004. [Online]. Available: https://www.nnwj.de/supervised-unsupervised.html. [Acedido em 14 05 2019].

[36] P. Domingos, “A few useful things to know about machine learning,” *Communicarions of the ACM,* vol. 55, nº 10, pp. 18-87, 2012.

[37] A. Mishra, “Metrics to Evaluate your Machine Learning Algorithm,” 24 02 2018. [Online]. Available: https://towardsdatascience.com/metrics-to-evaluate-your-machine-learning-algorithm-f10ba6e38234. [Acedido em 14 05 2019].

[38] J. Brownlee, “Supervised and Unsupervised Machine Learning Algorithms,” 16 March 2016. [Online]. Available: https://machinelearningmastery.com/supervised-and-unsupervised-machine-learning-algorithms/.

[39] Sewaqu, “File:Linear regression.svg,” 5 11 2010. [Online]. Available: https://commons.wikimedia.org/wiki/Category:Linear_regression#/media/File:Linear_regression.svg. [Acedido em 15 05 2019].

[40] Larhmam, “File:SVM margin.png,” 19 10 2018. [Online]. Available: https://commons.wikimedia.org/wiki/File:SVM_margin.png. [Acedido em 15 05 2019].

[41] T-kita, “File:Decision tree model.png,” 8 10 2005. [Online]. Available: https://commons.wikimedia.org/wiki/File:Decision_tree_model.png. [Acedido em 15 05 2019].

[42] A. Dey, “Machine Learning Algorithms: A Review,” *International Journal of Computer Science and Information Technologies,* vol. 7, nº 3, pp. 1174-1179, 2016.

[43] BrandIdea Consultancy, “K-Means and X-Means Clustering,” 2017. [Online]. Available: https://www.brandidea.com/kmeans.html. [Acedido em 15 05 2019].

[44] Khan Academy, “Neuron action potentials: The creation of a brain signal,” [Online]. Available: https://www.khanacademy.org/test-prep/mcat/organ-systems/neuron-membrane-potentials/a/neuron-action-potentials-the-creation-of-a-brain-signal. [Acedido em 14 05 2019].

[45] O. IsaacAbiodun et al., “State-of-the-art in artificial neural network applications: A survey,” *ScienceDirect,* vol. 4, nº 11, 2018.

[46] A. Al-Masri, “How Does Back-Propagation in Artificial Neural Networks Work,” [Online]. Available: https://towardsdatascience.com/how-does-back-propagation-in-artificial-neural-networks-work-c7cad873ea7. [Acedido em 15 05 2019].

[47] “Overfitting,” 08 2017. [Online]. Available: https://en.wikipedia.org/wiki/Overfitting#/media/File:Overfitting.svg. [Acedido em 11 05 2016].

[48] R. S. Vuppuluri, “State of Data Science & Machine Learning,” Towards Data Science, 25 February 2019. [Online]. Available: https://towardsdatascience.com/state-of-data-science-machine-learning-e8bdd4f21b6b. [Acedido em 12 March 2019].

[49] C. Voskoglou, “What is the best programming language for Machine Learning?,” Towards Data Science, 5 May 2017. [Online]. Available: https://towardsdatascience.com/what-is-the-best-programming-language-for-machine-learning-a745c156d6b7. [Acedido em 8 March 2019].

[50] S. Deoras, “Top 10 Programming Languages For Data Scientists to Learn In 2018,” Analytics India, 25 January 2018. [Online]. Available: https://www.analyticsindiamag.com/top-10-programming-languages-data-scientists-learn-2018/. [Acedido em 12 Marh 2019].

[51] K. Some, “Top 5 Machine Learning Programming Languages You Should Master,” Analytics Insight, 29 June 2018. [Online]. Available: https://www.analyticsinsight.net/top-5-machine-learning-programming-languages-you-should-master/. [Acedido em 12 March 2019].

[52] S. Boiko, “Best Libraries and Tools to Start off with Machine Learning and AI,” Railsware, 9 August 2018. [Online]. Available: https://railsware.com/blog/2018/08/09/best-libraries-and-tools-to-start-off-with-machine-learning-and-ai/. [Acedido em 12 March 2019].

[53] Y. Guo, “Introduction to Kaggle Kernels (AI Adventures),” Google Cloud Platform, 12 December 2017. [Online]. Available: https://www.youtube.com/watch?v=FloMHMOU5Bs. [Acedido em 12 March 2019].

[54] J. VanderPlas, “Get started with Google Colaboratory (Coding TensorFlow),” TensorFlow, 30 January 2019. [Online]. Available: https://www.youtube.com/watch?v=inN8seMm7UI. [Acedido em 12 March 2019].

[55] M. J. Garbade, “Top 8 open source AI technologies in machine learning,” opensource.com, 15 May 2018. [Online]. Available: https://opensource.com/article/18/5/top-8-open-source-ai-technologies-machine-learning. [Acedido em 12 March 2019].

[56] “12 Best Machine Learning Tools In 2019,” RankRed Media Private, 25 December 2018. [Online]. Available: https://www.rankred.com/best-machine-learning-tools/. [Acedido em 12 March 2019].

[57] D. Weldon, “16 top platforms for data science and machine learning,” SourceMedia, 8 March 2018. [Online]. Available: https://www.information-management.com/slideshow/16-top-platforms-for-data-science-and-machine-learning. [Acedido em 12 March 2019].

[58] D. Hoffman, “18 Machine Learning Platforms For Developers,” DZone AI Zone, 29 January 2018. [Online]. Available: https://dzone.com/articles/18-machine-learning-platforms-for-developers. [Acedido em 12 March 2019].

[59] “H2O.ai vs. KNIME,” IT Central Station, [Online]. Available: https://www.itcentralstation.com/products/comparisons/h2o-ai_vs_knime. [Acedido em 12 March 2019].

[60] S. Shanin, “10 of the Best Platforms for Data Science and Machine Learning,” Towards Data Science, 16 May 2018. [Online]. Available: https://medium.com/eteam/10-of-the-best-platforms-for-data-science-and-machine-learning-36a61ec1a676. [Acedido em 12 March 2019].

[61] M. S. Lande, P. M. Sirsat e R. S. Tupkar, “A Case Study On Predictive Maintenance Of Oj/ 5522 Dt- 40 Cnc Milling Machine,” *International Journal of Advanced Research and Publications,* vol. 1, nº 3, pp. 27-30, 2017.

[62] Costruxioni Industriali CIVIDAC, “Predictive Maintenance for Heat Exchangers,” [Online]. Available: https://www.cividac.com/news/predictive-maintenance-for-heat-exchangers.html. [Acedido em 13 05 2019].

[63] B. Tawfik, A. B. Hidri, B. Neef e M. S. Naceur, “Data analytics for predictive maintenance of industrial robots,” em *International Conference on Advanced Systems and Electric Technologies (IC_ASET)*, Braunschweig, 2017.

[64] J. Hider, “Predictive Maintenance Service Reduces Robot Breakdowns,” Modern Machine Shop, 27 03 2018. [Online]. Available:

https://www.mmsonline.com/products/predictive-maintenance-service-reduces-robot-breakdowns. [Acedido em 13 05 2019].

[65] MANTIS, "What is Mantis?," [Online]. Available: http://www.mantis-project.eu/about-mantis/what-is-mantis/. [Acedido em 23 05 2019].

[66] "FLEXIGY Energy Flexibility Services Platform," [Online]. Available: https://www.cister.isep.ipp.pt/projects/flexigy/. [Acedido em 02 05 2019].

[67] Y. Tang, K. Sakai, S. Luo e Y. Zhao, "AI in depth: monitoring home appliances from power readings with ML," 2019 February 25. [Online]. Available: https://cloud.google.com/blog/products/ai-machine-learning/monitoring-home-appliances-from-power-readings-with-ml.

[68] M. Fernandes, A. Canito, J. M. Corchado e G. Marreiros, "Fault Detection Mechanism of a Predictive Maintenance System Based on Autoregressive Integrated Moving Average Models," em *Distributed Computing and Artificial Intelligence*, 16th International Conference, 2019, pp. 171-180.

[69] E. Zhou e X. Pérez, "SONOFF S31- A WORLD APART," ITEAD Intelligent Systems Co.Ltd., 10 May 2018. [Online]. Available: https://www.itead.cc/blog/sonoff-s31-a-world-apart.

[70] T. Coelho, "Smart Maintenance for Home Appliances," Porto, 2019.

[71] J. Ali, R. Khan, N. Ahmad e I. Maqsood, "Random Forests and Decision Trees," *International Journal of Computer Science Issues(IJCSI),* vol. 9, 2012.

[72] R. Gandhi, "Naive Bayes Classifier," 5 May 2018. [Online]. Available: https://towardsdatascience.com/naive-bayes-classifier-81d512f50a7c.

[73] R. Gandhi, "Support Vector Machine - Introduction to Machine Learning Algorithms," 5 June 2018. [Online]. Available: https://towardsdatascience.com/support-vector-machine-introduction-to-machine-learning-algorithms-934a444fca47.

[74] O. Harrison, "Machine Learning Basics with the K-Nearest Neighbprs Algorithm," 10 September 2018. [Online]. Available: https://towardsdatascience.com/machine-learning-basics-with-the-k-nearest-neighbors-algorithm-6a6e71d01761.

[75] T. Srivastava, "Introduction to k-Nearest Neighbors: A powerful Machine Learning Algorithm (with implemetation in Python & R)," 26 March 2018. [Online]. Available: https://www.analyticsvidhya.com/blog/2018/03/introduction-k-neighbours-algorithm-clustering/.

[76] T. Yiu, "Understanding Random Forest," 12 June 2019. [Online]. Available: https://towardsdatascience.com/understanding-random-forest-58381e0602d2.

[77] J. K. Dharpure, M. B. Potdar e M. Pandya, "Counting Objects using Convolution based Pattern Matching Technique," *International Journal of Applied Information Systems (IJAIS),* vol. 5, nº 8, pp. 14-19, 2013.

[78] E. W. Weisstein, “Convolution,” MathWorld--A Wolfram Web Resource, [Online]. Available: http://mathworld.wolfram.com/Convolution.html. [Acedido em 06 September 2019].

[79] N. O'Leary, “Arduino Client for MQTT,” 22 May 2019. [Online]. Available: https://github.com/knolleary/pubsubclient.

[80] N. O’Leary, “Arduino PubSubClient - MQTT Client Library Encyclopedia,” 13 September 2015. [Online]. Available: https://www.hivemq.com/blog/mqtt-client-library-encyclopedia-arduino-pubsubclient/.

[81] N. Drake, “Best cloud computing services of 2019,” TechRadar, 29 January 2019. [Online]. Available: https://www.techradar.com/news/best-cloud-computing-service. [Acedido em 12 March 2019].

[82] “Top 5 Cloud Platforms and Solutions to Choose From,” New Generation Applications Pvt Ltd, 7 December 2017. [Online]. Available: https://www.newgenapps.com/blog/top-5-cloud-platforms-and-solutions-to-choose-from. [Acedido em 12 March 2019].

[83] CISTER RESEARCH CENTRE, “Regulations,” 08 06 2016. [Online]. Available: https://www.cister.isep.ipp.pt/info/regulations/. [Acedido em 27 04 2019].

[84] M. Rouse, “Big Data,” 11 2018. [Online]. Available: https://searchdatamanagement.techtarget.com/definition/big-data. [Acedido em 27 04 2019].

[85] M. Rouse, “internet of things (IoT),” 03 2019. [Online]. Available: https://internetofthingsagenda.techtarget.com/definition/Internet-of-Things-IoT. [Acedido em 27 04 2019].

[86] Fiix Software, “Predictive maintenace,” [Online]. Available: https://www.fiixsoftware.com/maintenance-strategies/predictive-maintenance/. [Acedido em 3 5 2019].

[87] Thales Group, “Understanding IoT Modules,” 03 03 2019. [Online]. Available: https://www.gemalto.com/iot/inspired/iot-modules. [Acedido em 14 05 2019].

[88] Cloogy, “Cloogy,” Virtual Power Solutions, 2019. [Online]. Available: https://www.cloogy.pt. [Acedido em 13 05 2019].

[89] X. Wu et al., “Top 10 algorithms in data mining,” *Knowledge and Information Systems,* vol. 14, nº 1, pp. 1-37, 2007.

[90] J. Gubbi, R. Buyya, S. Marusic e M. Palaniswami, “Internet of Things (IoT): A Vision, Architectural Elements, and Future Directions,” *Future generation computer systems,* vol. 7, nº 29, pp. 1645-1660, 2013.

[91] L. Atzori, A. Iera e G. Morabito, “The Internet of Things: A survey,” *Computer Networks,* vol. 54, nº 15, p. 2787–2805, 2010.

[92] D-Link, “Home Wi-Fi Motion Sensor DCH-S150,” [Online]. Available: https://eu.dlink.com/uk/en/products/dch-s150-motion-sensor. [Acedido em 22 05 2019].

[93] M. Fouad, N. Oweis, T. Gaber, M. Ahmed e V. Snasel, “Data Mining and Fusion Techniques for WSNs as a Source of the Big Data,” em *International Conference on Communication, Management and Information Technology*, Prague, 2015.

[94] S. RAY, “Commonly used Machine Learning Algorithms (with Python and R Codes),” 9 September 2019. [Online]. Available: https://www.analyticsvidhya.com/blog/2017/09/common-machine-learning-algorithms/.

[95] L. Chen, S. Yu e M. Yang, “Semi-supervised convolutional neural networks with label propagation for image classification,” em *International Conference on Pattern Recognition (ICPR)*, Beijing, China, 2018.

[96] A. Gupta, “25 Questions to test a Data Scientist on Support Vector Machines,” 5 October 2017. [Online]. Available: https://www.analyticsvidhya.com/blog/2017/10/svm-skilltest/.

7 Appendix

7.1 Washing Machine experiments information

Table 14, Table 15, Table 16 and Table 17 show all experiments made with Washing Machine "30 minutes", "Coloured" and "Cottons + Prewash" programs. These programs are underreported or test targets, however, these tables give a better insight into other experiments done with other programs.

Table 14 - Experiments of "14 minutes " program

Number	Description	Date of measured experiment					
		1st	2nd	3rd	4th	5th	6th
1	"14 minutes" with centrifugation at 30 C	1/7/19 9:38	15/7/19 18:01	19/7/19 8:57	24/7/19 18:16	25/7/19 23:16	22/7/19 11:37
2	"14 minutes" with centrifugation at 40 C	1/7/19 9:53	15/7/19 18:21	19/7/19 9:40	24/7/19 18:37	26/7/19 0:38	22/7/19 11:59
5	"14 minutes" with centrifugation and weight at 30 C	2/7/19 9:45	16/7/19 7:37	17/7/19 17:02	23/7/19 13:44	23/7/19 14:25	26/7/19 0:11
7	"14 minutes" with centrifugation and weight at 40 C	3/7/19 15:02	16/7/19 9:37	17/7/19 17:25	23/7/19 14:25	23/7/19 17:46	25/7/19 23:51
12	"14 minutes" with weight at 30 C	16/7/19 10:27	16/7/19 10:39	17/7/19 15:56	23/7/19 12:35	23/7/19 12:35	25/7/19 23:39
13	"14 minutes" with weight at 40 C	16/7/19 10:51	16/7/19 22:12	17/7/19 16:16	23/7/19 12:46	23/7/19 12:46	22/7/19 14:34
14	"14 minutes" at 30 C	16/7/19 17:18	16/7/19 19:56	17/7/19 14:56	25/7/19 22:29	26/7/19 1:06	22/7/19 13:21
15	"14 minutes" at 40 C	16/7/19 17:30	16/7/19 20:17	17/7/19 15:08	25/7/19 22:42	26/7/19 1:19	22/7/19 13:33
21	"14 minutes" with centrifugation at 20 C	16/7/19 19:25	17/7/19 13:50	17/7/19 15:34	24/7/19 17:56	25/7/19 22:55	22/7/19 13:00
22	"14 minutes" with centrifugation and weight at 20 C	16/7/19 21:52	17/7/19 9:22	17/7/19 16:42	23/7/19 13:18	23/7/19 15:55	22/7/19 14:05
23	"14 minutes" with weight at 20 C	16/7/19 21:38	17/7/19 9:10	17/7/19 16:28	23/7/19 13:03	23/7/19 13:03	22/7/19 14:49
24	"14 minutes" at 20 C	16/7/19 19:43	17/7/19 14:10	17/7/19 15:21	24/7/19 18:57	26/7/19 0:52	22/7/19 13:49

Table 15 - Experiments of "30 minutes" program

Number	Description	Date of measured experiment	
		1st	2nd
1	“30 minutes” at 20 C	31-07-19 18:03	01-08-19 22:42
2	“30 minutes” at 30 C	31-07-19 18:27	01-08-19 23:05
3	“30 minutes” at 40 C	31-07-19 18:51	01-08-19 23:30
4	“30 minutes” with centrifugation at 20 C	01-08-19 08:21	01-08-19 20:54
5	“30 minutes” with centrifugation at 30 C	01-08-19 08:59	01-08-19 21:34
6	“30 minutes” with centrifugation at 40 C	01-08-19 09:37	01-08-19 22:09
7	“30 minutes” with weight at 20 C	01-08-19 12:12	01-08-19 17:25
8	“30 minutes” with weight at 30 C	01-08-19 12:36	01-08-19 17:56
9	“30 minutes” with weight at 40 C	01-08-19 13:02	01-08-19 18:27
10	“30 minutes” with centrifugation and weight at 20 C	01-08-19 13:31	01-08-19 18:54
11	“30 minutes” with centrifugation and weight at 30 C	01-08-19 16:40	01-08-19 19:33
12	“30 minutes” with centrifugation and weight at 40 C	01-08-19 15:48	01-08-19 20:18

Table 16 - Experiments of "Coloured" program

Number	Description	Date of measured experiment		
		1st	2nd	3rd
1	“Coloured” at 30 C	1-7-19 15:16	15-7-19 19:45	18-7-19 14:38
2	“Coloured” at 40 C	2-7-19 0:04	15-7-19 21:11	18-7-19 19:04
3	“Coloured” with centrifugation at 30 C	2-7-19 15:09	16-7-19 8:56	18-7-19 10:34
4	“Coloured” with centrifugation at 40 C	16-7-19 12:18	2-8-19 13:44	18-7-19 16:05
5	“Coloured” with weight at 30 C	16-7-19 14:43	17-7-19 13:14	18-7-19 17:31
6	“Coloured” with weight at 40 C	16-7-19 18:50	17-7-19 10:54	18-7-19 12:12
7	“Coloured” with centrifugation and weight at 30 C	16-7-19 21:31	17-7-19 8:58	18-7-19 9:20
8	“Coloured” with centrifugation and weight at 40 C	16-7-19 23:24	16-7-19 13:30	17-7-19 18:48

Table 17 - Experiments of "Cottons + Prewash" program

Number	Description	Date of measured experiment	
		1st	2nd
1	"Cottons + Prewash" with centrifugation at 90 C	04-07-19 09:13	02-08-19 10:42
2	"Cottons + Prewash" with centrifugation at 60 C	01-08-19 10:15	02-08-19 08:00
3	"Cottons + Prewash" with centrifugation and weight at 90 C	04-07-19 11:58	02-08-19 02:04
4	"Cottons + Prewash" with centrifugation and weight at 60 C	05-08-19 18:09	02-08-19 04:30

7.2 Convolution experiments

The Figure 64, Figure 65, Figure 66, Figure 67 and Figure 68 prove that, through convolution, it is possible to detect the centrifugation phase in the other washing machine programs, not only in "14 minutes" program. The process to understand this is the same presented in Section 4.1.3.

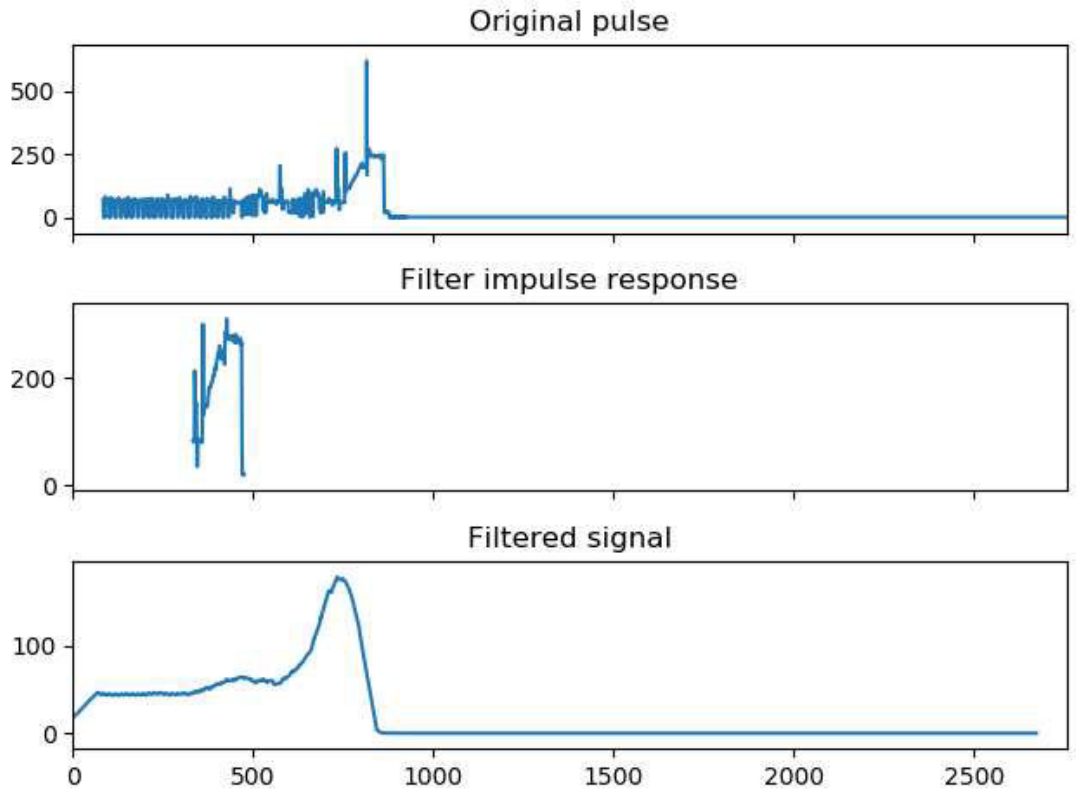

Figure 64 - Convolutions test for program "30 minutes" with centrifugation phase

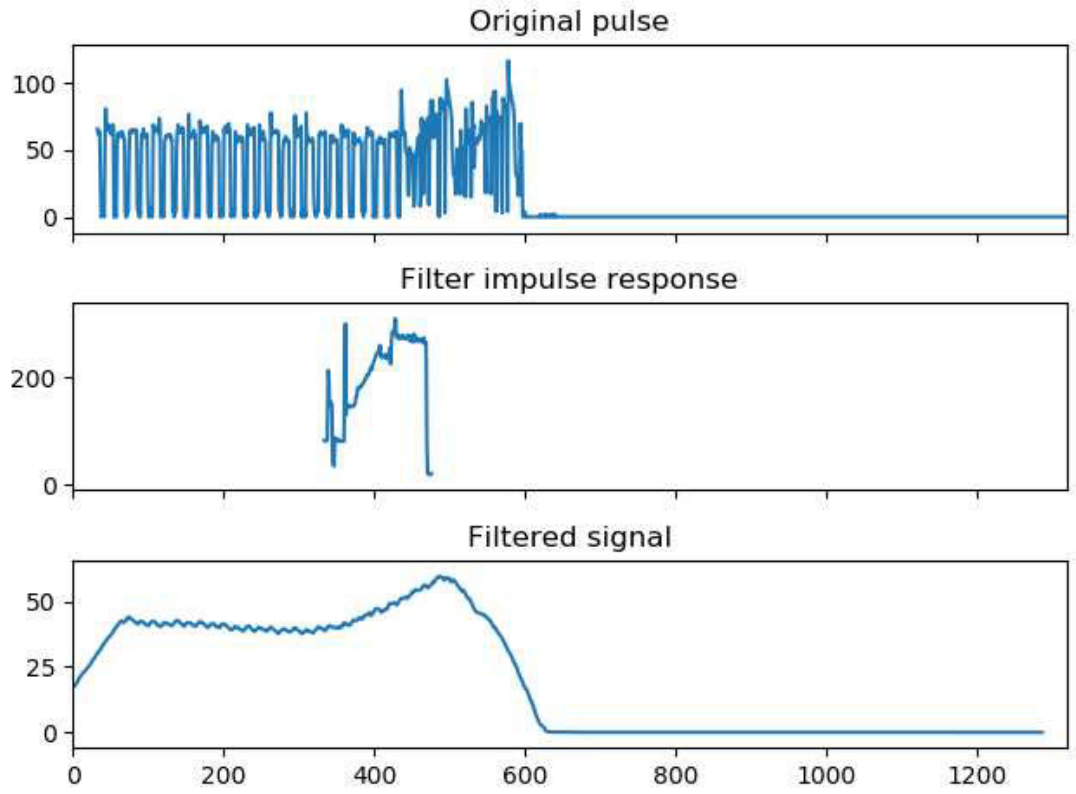

Figure 65 - Convolutions test for program "30 minutes" without centrifugation phase

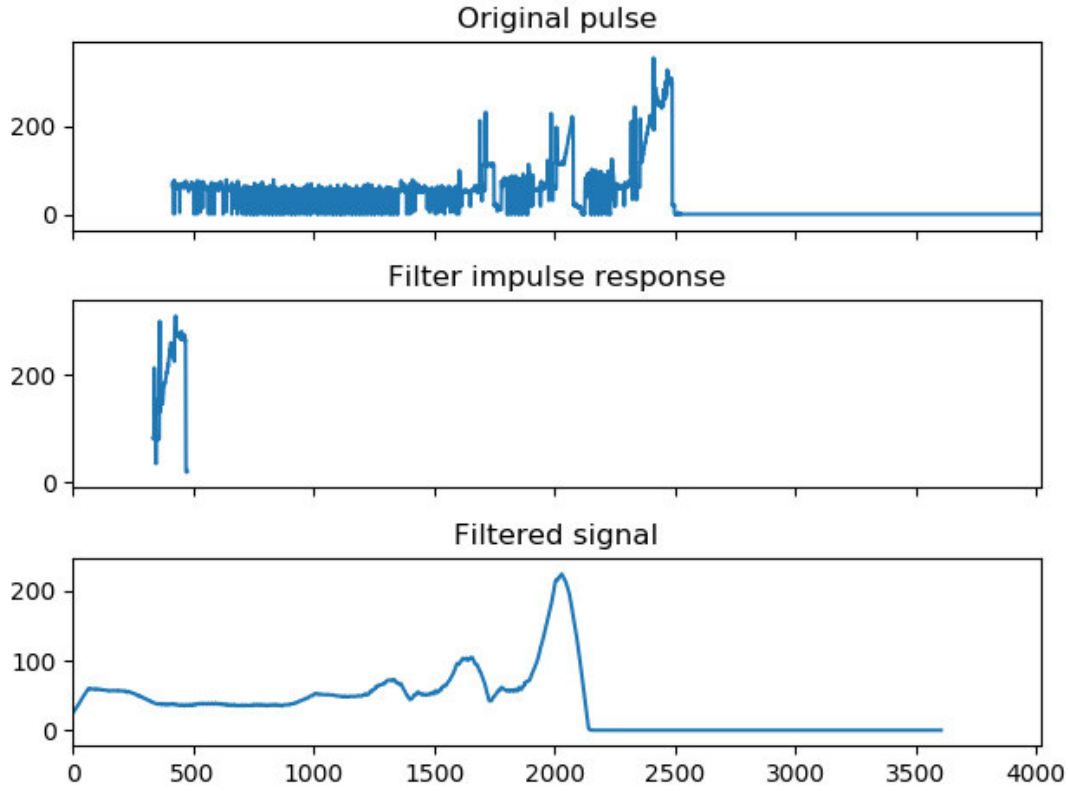

Figure 66 - Convolutions test for program "Coloured" with centrifugation phase

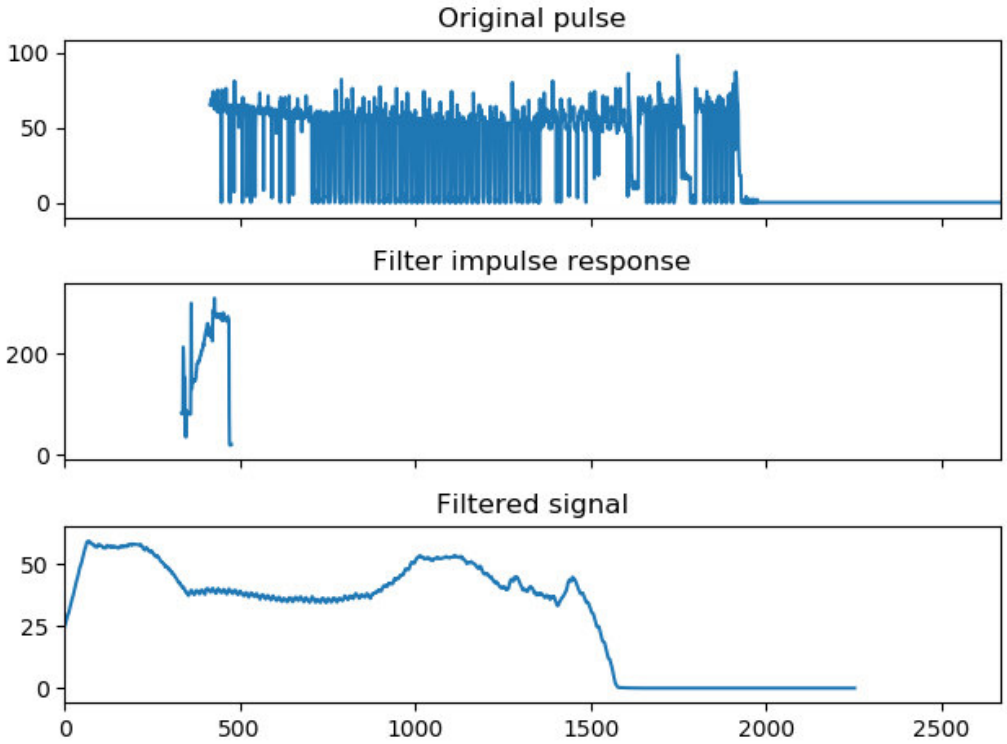

Figure 67 - Convolutions test for program "Coloured" without centrifugation phase

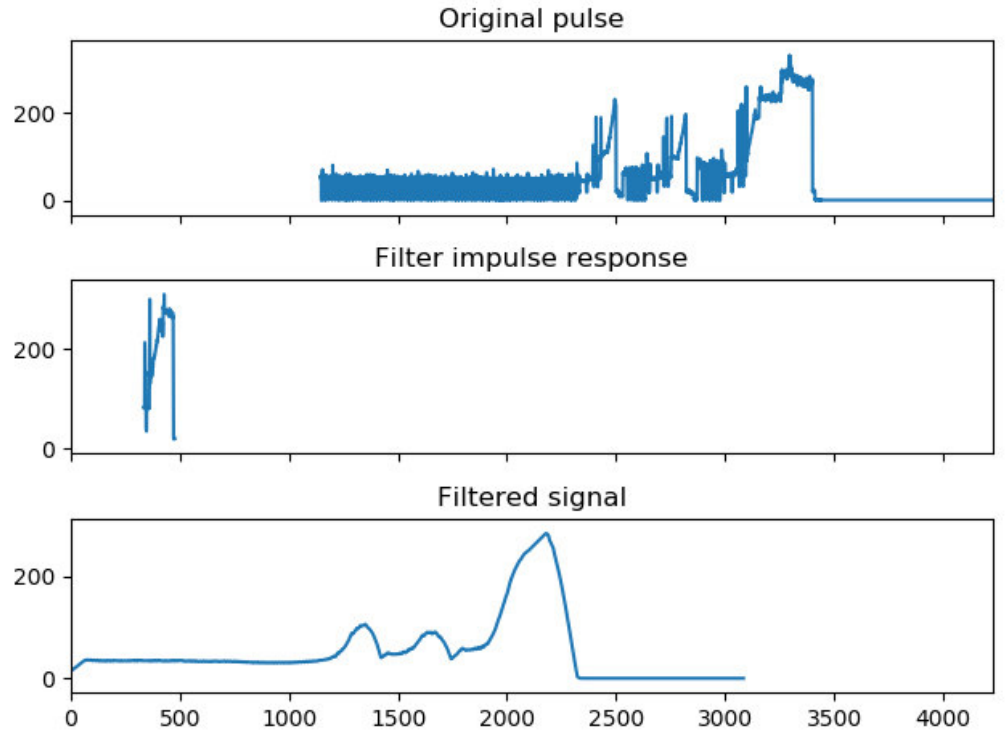

Figure 68 - Convolutions test for program "Cottons+PreWash" with centrifugation phase