UCLouvain

École polytechnique de Louvain

A pragmatic strategy for reducing energy consumption via sustainable automation technology

Authors: **Gautier Radar, Hugo Rolin**
Supervisor: **Ramin Sadre**
Readers: **Ynan Cao, Cristel Pelsser**
Academic year 2022–2023
Master [60] in Computer Science

Acknowledgements

Before we delve into this master's thesis, we would like to thank our advisor, Professor Ramin Sadre, for his support, advice, and meetings throughout the year, without which this work would have been much more challenging.

We would also like to thank the household in which we were able to conduct our studies, without which our dissertation would not have been possible.

Finally, we would like to thank our families for their technical, psychological and logistical support.

Contents

Chapter 1

Introduction

Internet of Things (IoT) has emerged as a transformative force in recent years, impacting various aspects of our daily life. Among its many applications, IoT has been pivotal in shaping the evolution and enhancement of smart homes, which strive to elevate home comfort, security, and energy efficiency. Energy use monitoring and management represent a critical dimension of this technological revolution, contributing to the global effort towards reducing our environmental footprint and promoting sustainable living. Thus, leveraging IoT technologies, smart homes can provide homeowners with real-time data on energy usage, facilitating the identification and rectification of inefficiencies, and ultimately fostering more sustainable consumption habits.

This study contribute to the expanding body of knowledge on energy monitoring technology to promote sustainable living. This research aims to encourage the creation of more energy-efficient households and pave the way for a greener, more sustainable future by identifying effective techniques and best practices.

In addition to monitoring, this study also explores the integration of machine learning in energy management systems. By utilizing historical and real-time data, machine learning algorithms can predict future energy needs, thereby enabling more effective energy consumption and production scheduling. The predictive power of machine learning combined with automated controls can drive significant improvements in energy efficiency and cost savings.

1.1 Objectives

This study comprises two main parts: Data Acquisition and Integration, and Data Analysis and Prediction.

The first part, **Data Acquisition and Integration**, focuses on the technological framework and methodologies used to collect, process, and integrate energy consumption data from various Internet of Things devices within a smart home ecosystem.

The objective of data acquisition is to capture accurate and reliable energy consumption data from various smart appliances and devices in the home in real-time. The study will also address the challenges encountered during data acquisition, including ensuring data accuracy, managing the high volume of data generated by IoT devices, and maintaining the security and privacy of collected data.

After data acquisition, the study explores the critical phase of data integration. In a smart home setting, energy consumption data are often fragmented across various devices and systems, making it difficult to gain a comprehensive view of the home's overall energy usage. Therefore, data integration strategies are essential to aggregate and harmonize these disparate data streams into a unified dataset that can be readily analyzed and used for making informed energy management decisions.

This first component can be summarised in the following points:

- Examine the smart meter landscape in Belgium, including the types of smart meters deployed, their functionality, and the laws and regulations applicable to their use.
- Assess the compatibility of the various communication protocols used for transmitting smart meter data with home automation platforms such as Home Assistant.
- Identify the strengths and weaknesses of existing smart meter integration solutions available in Belgium, specifically Wallonia, in order to inform the development of a new system.
- Analyze data security measures and best practices for securing the communication between smart meters and home automation systems, protecting the privacy and confidentiality of metering data.
- Design a system for reading smart meter data that is compatible with Wallonia's smart meters and can be easily integrated with home automation

platforms.

- Create a prototype of the proposed smart meter integration system that demonstrates its functionality and Home Assistant compatibility.
- Evaluate the performance and dependability of the developed prototype and collect user feedback in order to enhance the system and determine potential future development areas.

Data Analysis and Prediction, the second primary component of this study, focuses on utilizing the collected and integrated energy consumption data to generate valuable insights and predictive models for better energy management.

In the analysis phase, the study applies various statistical and data analysis techniques to explore the integrated energy data comprehensively. The objective is to uncover the underlying patterns and trends in the energy consumption data, including time-of-day variations, seasonal changes, and specific appliance usage profiles. By understanding these patterns, homeowners can become more aware of their energy consumption habits and identify potential areas for improvement. This phase will also involve the use of visualization tools to present the analysis results in an intuitive and easily understandable manner.

The prediction phase is where machine learning comes into play. Leveraging the patterns and relationships identified in the analysis phase, machine learning algorithms will be used to develop predictive models for future energy consumption and production. These models can provide homeowners with an understanding of their expected energy usage and solar energy production based on historical data and real-time conditions, helping them plan their energy usage more effectively.

The study will also investigate the application of these predictive models in automation scenarios. For instance, they could be used to automatically schedule energy-intensive appliances during periods of predicted high solar energy production, thereby maximizing the use of renewable energy and minimizing reliance on grid power.

This second component can be summarised in the following points:

- Analyze patterns and trends in the energy consumption and production data obtained from the integrated smart meter.
- Develop a machine learning model for predicting energy consumption and production based on historical and real-time data.

- Assess the performance of the predictive model using appropriate metrics such as Mean Squared Error (MSE), R-squared score (R2), and Mean Absolute Error (MAE).
- Implement energy-efficient scheduling of household appliances based on the predicted energy consumption and production.
- Demonstrate the potential energy and cost savings achieved through the machine learning-guided energy management system.
- Discuss the broader implications and future applications of integrating smart meter data with home automation for energy-efficient living.

In order to ensure a proper interpretation of the objectives, it is essential to clearly state the scope of this study. Indeed, it is important to note thats smart meters deployed in Wallonia may have different requirements and specifications than those in other regions of Belgium. Due to these differences, it is possible that the findings and recommendations presented in this study are not directly applicable in other regions of Belgium or in other countries. In addition, not every possible communication protocol or data security measure applicable to smart meter integration projects is covered by this research. Instead, it focuses on the most prevalent and pertinent technologies within our context.

Part I

Data Acquisition and Integration

Chapter 2

Literature Review

This section presents an overview of the extant literature discussing the integration of smart meters with home automation systems, particularly in Wallonia. The literature study seeks to present an overview of the current state of the art, encompassing communication protocols, data security measures and current integration possibilities. By examining the body of knowledge in this area, we can identify gaps and opportunities for research and development.

2.1 Smart Meters and Their Functionality

Smart meters are digital, high-tech substitutes for traditional analog meters used to measure electricity, gas, and water. They were developed in recent years and offer numerous advantages to both utility companies and consumers. Here are several advantages of smart meters:

- Smart meters provide extremely detailed information about electricity usage and costs, enabling customers to make informed decisions regarding how to optimize their electricity consumption and lower their expenditures [1].
- Smart meters enable utilities to detect malfunctions and restore service more rapidly, resulting in less disruption to a customer's residence or place of business [1] [2].
- By keeping note of peak usage times, smart meters can assist in enhancing energy efficiency. If a customer elects to partake in the utility's time-based rates, they have the opportunity to reduce their electricity consumption during "peak" hours and save money on their monthly electric bill [3].
- Smart meters permit and encourage consumers to participate in analyzing their personal consumption patterns, which can assist them in gaining control

of their household energy use [4].

- Smart meters provide more precise meter readings, allowing for more accurate invoicing and improved pricing schemes [3] [5].
- Smart meters enable utilities to analyze usage patterns for each individual customer in real-time, enabling them to predict and model monthly cash flow scenarios based on real-time usage [5].

In brief, they are an essential component of modern energy management techniques due to their function in facilitating a more efficient, sustainable, and cost-effective energy grid.

Because of the Belgian government's dedication to meeting the objectives of the European Union regarding the improvement of energy efficiency and the reduction of greenhouse gas emissions, there has been a consistent increase in the number of households using smart meters. The Belgian electricity market is segmented into three regions: Flanders, Wallonia, and Brussels. Each of these regions has its own regulatory framework as well as its own schedule for the installation of smart meters.

Distribution system operators (DSOs) in Flanders kicked off the widespread installation of smart meters in 2019, with the initial emphasis placed on new connections, major renovations, and customers whose annual power consumption was greater than a certain threshold. According to the requirements of the Flemish Energy Law, the installation of smart meters in all homes and small enterprises is scheduled to be completed by the year 2034 [6].

According to the European Commission, the Wallonian government has approved a plan to install smart meters in phases beginning in 2023, with the goal of achieving complete coverage by 2034. At least 80% of smart meters will be installed by December 31, 2029, for consumers with a consumption of at least 6,000 kWh, prosumers with a net developable electrical capacity of at least 5 kWe, and public charging points [6].

In the Brussels region, smart meters are deployed when meters need to be changed or for new connections to the network. In May 2022, 32,170 meters were installed on the distribution network by the distribution network operator [7].

Regulations in Belgium pertaining to smart meters address a variety of topics, including data privacy and security, consumer rights, and grid management, among others. The General Data Protection Regulation (GDPR) of the European Union is a key piece of law that applies to smart meters. Its purpose is to ensure that

consumer data is handled in a secure and transparent manner. In addition, special regulations and standards for the correct installation, operation, and maintenance of smart meters have been created by the federal government of Belgium as well as by some of the country's regional governments [8] [6] [9] .

2.2 Home Assistant and Its Applications

Home Assistant is a versatile and robust platform that simplifies the control of smart home devices and provides comprehensive customization and automation options. Its applications include centralized control, device interoperability, energy management, and security, making it a popular choice for both novice and advanced users seeking to create a customized smart home experience.

Among the primary applications of Home Assistant are [10]:

- **Centralized control**: Home Assistant enables centralized control for a variety of smart devices, including lights, thermostats, security cameras, and entertainment systems. Its centralization facilitates the creation of custom scenes and automations and simplifies the maintenance of smart home devices.
- **Device interoperability**: Home Assistant supports a wide variety of smart devices and communication protocols, allowing for the seamless integration of products from diverse manufacturers. This interoperability enables consumers to mix and match devices based on their preferences and needs, without being limited to a single brand or ecosystem.
- **Customization and automation**: With Home Assistant, users may design automation rules that trigger particular actions or device behaviors based on factors such as time, location, or sensor data. This degree of customisation enables consumers to personalize their smart home experience to their specific needs and way of life.
- **Energy management**: Home Assistant can be used to monitor and control energy consumption in real-time, enabling customers to uncover inefficiencies, decrease energy waste, and ultimately save money on utility bills. By integrating with smart meters and other energy monitoring devices, Home Assistant may deliver energy-saving insights and automations.
- **Security and privacy**: As an open-source platform, Home Assistant provides a high level of security and confidentiality. Customers have the option to run Home Assistant locally on their own hardware, ensuring that their data stays private and secure without relying on cloud services provided by a third party.

- **Community-driven development**: Home Assistant is supported by an active community of developers and enthusiasts who contribute to its continuous development, share custom integrations, and provide help and guidance to other users. This community-driven approach ensures that Home Assistant remains compatible with the most recent smart home technologies and is always up-to-date.

Advantages and Disadvantages of Using Home Assistant Over openHAB

Home Assistant and openHAB are both open-source home automation platforms that provide a variety of features and smart device compatibility. When deciding between these two systems, a number of factors must be considered. Here are some benefits and drawbacks of using Home Assistant instead of openHAB:

Advantages:

- **Ease of use:** Home Assistant is frequently seen as more user-friendly than openHAB, particularly for novices. It has a more intuitive graphical user interface (GUI), and the setup process is generally simpler. Home Assistant also offers an official mobile app for Android and iOS, providing users with further convenience [11].
- **Community and support:** Home Assistant has a huge and active community that continually contributes to its development and provides help through online forums, manuals, and other resources. This robust community support might be advantageous for individuals seeking assistance or ideas for home automation projects [11].
- **Add-on ecosystem:** The number of Home Assistant add-ons, which are simply installable modules that increase the platform's functionality, is growing. This ecosystem for add-ons streamlines the addition of additional features and services, such as databases, video servers, and voice assistants [12].
- **Automation engine:** The automation engine of Home Assistant supports a variety of triggers, conditions, and actions, enabling users to design complicated automations with relative ease. The YAML-based configuration format is also more human-readable than openHAB's DSL (Domain Specific Language) and allows users to design and manage automations more easily [12].

Disadvantages:

- **Learning curve:** Even though Home Assistant's user interface is more intuitive than openHAB's, there is still a learning curve associated with installing and configuring the platform. Users may need to invest time in knowing YAML and the structure of the platform in order to construct and manage automations efficiently [11].
- **Performance and resource usage:** Home Assistant is constructed with Python, which may be less resource-efficient than openHAB, which is constructed with Java. This variation in resource use may not be significant for most users, but those running their home automation system on resource-constrained devices should take it into account [12].
- **Integration support:** Home Assistant supports a significant variety of integrations, however openHAB may give support for devices or services that Home Assistant does not currently support. Users must examine the supported integrations for each platform to ensure interoperability with their preferred smart devices [11].

The choice between Home Assistant and openHAB ultimately depends on the user's choices, needs, and experience with the respective systems. Home Assistant provides a more intuitive interface, a larger ecosystem of add-ons, and an easier-to-use automation engine, yet openHAB has a longer history and may be seen as more stable and mature. Each platform has advantages and disadvantages, and consumers should examine these elements carefully when determining which home automation platform best meets their needs.

2.3 Existing Smart Meter Integration Solutions

Several solutions have been developed in Belgium to combine smart meters with home automation systems, allowing customers to more efficiently monitor and regulate their energy consumption. Typically, these solutions require a combination of hardware and software components that enable communication between the smart meter and home automation platforms such as Home Assistant. Among the existing integration solutions for smart meters in Belgium are:

1. **P1 Port Readers**: Numerous smart meters in Belgium are equipped with a P1 port, which enables access to energy usage data in real time. Some commercially available P1 port readers, such as the homewizard module, can be directly connected to the smart meter and transfer data to a home automation system over Wi-Fi or Ethernet [13].

2. **DSO Web APIs**: In certain instances, Distribution System Operators in Belgium provide web APIs that enable remote access to smart meter data. These APIs can be used to create custom interfaces for home automation platforms like Home Assistant. However, the availability and functionality of these APIs may differ across DSOs and regions, and users may be required to seek access and comply with specific terms and restrictions [14].

3. **LoRaWAN-based solutions**: Several companies have developed LoRaWAN-based integration solutions for smart meters, enabling long-range and low-power communication between smart meters and home automation systems [15]. Typically, these solutions require a LoRaWAN gateway, and a LoRaWAN-compatible smart meter.

4. **DIY**: Technically-inclined customers may elect to design their own unique solutions for smart meter integration utilizing microcontrollers such as Arduino or Raspberry Pi, as well as the proper communication modules and sensors. These do-it-yourself solutions frequently necessitate a deeper understanding of hardware, software, and communication protocols, but they also provide greater flexibility and customization choices.

While there are various existing smart meter integration solutions in Belgium, it is essential to select the most appropriate one depending on variables such as the smart meter model, connection protocol, and level of customization requested.

2.4 Communication Protocols and Data Security

The integration of smart meters with home automation systems in Wallonia requires efficient communication protocols and robust data security measures to ensure seamless data exchange between devices, protect user privacy, and prevent unauthorized access. In this section, we delve into the primary data transmission protocols and the necessary security measures.

2.4.1 Data Transmission Protocols and Security

To facilitate data exchange between the P1 smart meter sensor and home automation systems in Wallonia, efficient communication protocols and stringent data security measures are required. Here are the primary data transmission protocols and security mechanisms for smart metering in Wallonia.

Data Transmission Protocols:

- LoRaWAN: LoRaWAN is a standardized wireless technology that uses existing mobile networks to connect devices. Its high throughput and extensive range make it ideal for intelligent utilities.
- Zigbee: Zigbee is a low-power wireless communication protocol designed for applications with modest data rates. It is utilized frequently in home automation systems.
- WiFi: Wi-Fi is a wireless networking technology that enables devices to communicate wirelessly to the internet and other networks.

The security and privacy of intelligent metering systems are ensured by a number of crucial safeguards. These measures include the following:

Data Protection Impact Assessment (DPIA) is an essential procedure for ensuring that there are no security or privacy concerns. It identifies potential hazards and implements the necessary countermeasures.

Smart metering systems are protected from security threats and data confidentiality is maintained through the use of techniques that preserve privacy. These methods are designed to prevent any disclosure or misappropriation of the data collected by smart meters.

Another important factor is the effectiveness of smart meter security testing. The purpose of these evaluations is to evaluate the security of smart meters in order to identify vulnerabilities and take corrective action.

Encryption is essential for safeguarding smart meter data. It is utilized to prevent unauthorized data access and modification. The transmission of data between the smart meter and the home automation system should be encrypted using methods such as AES with 128-bit, 192-bit, or 256-bit keys.

To verify the identities of devices and users, stringent authentication and authorization measures are implemented. This includes password protection, digital certificate use, and Public Key Infrastructure (PKI). These safeguards guarantee that only authorized parties have access to the metering data.

Also essential is the use of secure communication channels. Using HTTPS for web-based APIs or TLS with comprehensive encryption suites and perfect forward secrecy (PFS) can provide additional protection against data interception and man-in-the-middle attacks, for instance.

Lastly, it is crucial to adhere to applicable data protection laws and regulations, such as the General Data Protection Regulation (GDPR) of the European Union and national data protection laws. This includes not only reporting data breaches, but also obtaining consent when required and adhering to best practices when handling, preserving, and processing smart meter data.

These measures are essential for safeguarding user privacy and preventing unauthorized access to metering information. By considering the appropriate protocols and assuring data security, we can create a Wallonia smart meter ecosystem that is dependable and secure [16] [17] [18].

Chapter 3

System Design and Architecture

In this section, we delve into the design and architecture of the solution. We discuss the numerous components that make up the system, including hardware, software, communication protocols, and data processing algorithms. By explaining the system's design and architectural choices, we attempt to provide a full knowledge of the solution's structure, functionality, and underlying concepts.

3.1 Overview of the Proposed System

The proposed smart meter integration system seeks to provide a seamless and efficient method for reading and integrating energy consumption data into Home Assistant. The system consists of three essential components: an ESP8266 microcontroller, a Raspberry Pi 3 B+, and Home Assistant. Moreover, the system utilize smart outlets to enable energy monitoring for individual devices connected to them, thereby providing the user with a more comprehensive and understandable view of their energy consumption. The following is a summary of the function of each system component.

1. **The ESP8266 microcontroller** is an efficient and flexible microcontroller that connects directly to the P1 port of the smart meter to capture energy consumption data in real-time. It supports several communication protocols, including Wi-Fi, allowing it to transmit the processed data to the Raspberry Pi. Its low power consumption and compact size make it ideal for this application. The ESP8266 microcontroller has been used in various DIY projects to build smart energy monitoring devices or smart electricity meters from scratch [19].

2. **The Raspberry Pi 3 B+** receive the parsed data transmitted by the ESP8266 as the system's central server. He manages to receive data through

Home Assistant, stores it for analysis, and integrates it with the Home Assistant Energy platform. In addition to managing smart meter data, the Raspberry Pi will also handle the smart plugs.

3. **Home Assistant** is an open-source home automation platform, that will be deployed on the Raspberry Pi, allowing customers to monitor and control their energy utilization in conjunction with other smart home devices [20], such as smart plugs. The processed data from smart meters will be integrated into the Home Assistant platform, making it readily accessible via its user interface. Users can utilize Home Assistant's vast array of features, including automation rules, notifications, and data visualization tools, to better comprehend and manage their energy consumption, including the energy consumption of specific devices connected to smart plugs.

3.1.1 Additional integration

To further enhance the utility and adaptability of our proposed system, we have considered additional integrations, focusing on energy monitoring, security, and sustainability measures. These additional integrations include the use of smart outlets for granular energy consumption data, Cloudflare for cybersecurity protections, and solar panels for promoting renewable energy use.

Cloudflare Tunnel

We have kept the Home Assistant configuration straightforward and basic in this configuration. The primary objective is to enable access to Home Assistant from anywhere via a Cloudflare-created tunnel.

Cloudflare is an online utility that provides websites and applications with security and performance features. The Cloudflare tunnel enables secure and quicker access to Home Assistant via Cloudflare's servers, resulting in a connection that is both faster and more secure [21].

Instead of configuring external access to your Home Assistant server directly, the Cloudflare tunnel functions as an intermediary between the user and the Home Assistant server. This means that traffic between the user and Home Assistant is routed through Cloudflare's servers, adding an additional layer of security and improving performance.

To set up this configuration, a Cloudflared community plugin was deployed. Then, the Cloudflare route was configured to redirect traffic to the Home Assistant

server via the configured domain. Using the domain address, we can access Home Assistant from anywhere.

This configuration of Home Assistant with Cloudflare Tunnel enables secure and efficient access to Home Assistant from anywhere. By using this combination of components, the proposed system offers a comprehensive and user-friendly utilization. Moreover, the modular design makes it easy to customize and upgrade, and allows integration with other home automation platforms and services.

Energy Panel

In terms of sustainability, we have considered integrating solar panels into our proposed system. By integrating solar panels into our smart meter reader system, we aim to reduce dependency on the electricity grid and thus promote sustainable energy use. This allows homeowners to better plan their energy consumption, possibly scheduling energy-intensive activities during periods of high solar energy production.

With this configuration, the Home Assistant energy panel provides a straightforward overview of energy consumption and production, thereby facilitating the management of energy resources and contributing to a more responsible and efficient energy consumption. As shown in the Figure 11.1 from the Appendix B, the control interface is very straightforward and easy to use. The user can view and compare his consumption across days, weeks, months, and even years. The configuration panel depicted in the Figure 11.2 from the Appendix B demonstrates how the user can add entities based on categories and can even add a price gauge to get an accurate estimate of his statement. Users can also see the price they are paying in real time and the number of KwH detailed, as shown in Figure 11.3.

Closer view

In addition to the energy panel, we have created a custom page for users who require additional information regarding their energy consumption. This page provides a comprehensive view of the data and detailed graphs, enabling for additional energy consumption analysis.

While the energy panel provides a simplified view of incoming and outgoing energy, the comprehensive consumption page provides additional details, including:

- **P1 Actual Tariff Group**: Indicates the actual tariff group based on the present time of day (night tariff or day tariff).
- **P1 L1/L2/L3 Instant Power Current**: Instantaneous electric current for phase 1/2/3.
- **P1 L1/L2/L3 Instant Power Usage**: Instantaneous electrical power consumed for phase 1/2/3.
- **P1 L1/L2/L3 Voltage**: Electrical voltage for phase 1/2/3.
- **P1 Long Power Outages**: Number of long power outages.
- **P1 Short Power Drops**: Number of short duration power drops.
- **P1 Short Power Outages**: Number of short duration power outages.
- **P1 Short Power Peaks**: Number of short duration power peaks.

This information enables users to scrutinize their energy consumption more precisely and gain a deeper understanding of the various factors that influence it. This page is especially helpful for individuals who wish to optimize their energy consumption and reduce their energy footprint.

3.2 Hardware Components

This section describes the hardware components of our system, i.e. the features, functions and tasks of the different components of the system, and their importance for the operation and connectivity of the system.

3.2.1 Smart Meter Specifications

Sibelga is presently installing two types of smart meters of the most recent generation. The single-phase S211 meter and the three-phase T211 meter. We utilized the T211 (as shown in Figure 3.1) due to the three-phase installation in the residence where our evaluations were conducted. The same meters are also being installed in Flanders (Fluvius) and Wallonia (Ores) [22].

In this section, we describe the Siconia T211 smart meter's essential features [23]:

- **P1 Port**: The P1 port of the Siconia T211 is a standard data interface that provides access to real-time energy consumption statistics. For data transfer, the P1 port leverages the Dutch Smart Meter Requirements (DSMR) protocol, which uses serial communication protocols such as UART (Universal Asynchronous Receiver/Transmitter) or RS-232.
- **Power Supply**: With a voltage range of 230V +/- 15% and a frequency range of 50 Hz +/- 1 Hz, the smart meter is compatible with Belgium's standard power system.
- **Measurement Accuracy**: The Siconia T211 offers a high degree of measurement precision, assuring users of trustworthy energy consumption statistics.
- **Data Storage**: The smart meter is capable of recording energy consumption data for a predetermined amount of time, often up to one month, allowing customers to track their energy use trends over time.
- **Communication Protocols**: Several communication protocols are supported by the Siconia T211, including DSMR5 for real-time energy consumption data access and DLMS/COSEM for exchanging data with utility companies and other smart grid devices. This facilitates the integration of smart grid infrastructures with home automation systems.

Figure 3.1: Sagecom T211 smart meter

3.2.2 Microcontroller

The Wemos D1 Mini V4, an ESP8266-based microcontroller with a CH340 USB-serial chip, serves as the primary communication gateway between the smart meter and the Raspberry Pi running Home Assistant in our proposed solution. The Wemos D1 Mini V4 was selected due to its adaptable features and reasonable price. The following are among the most notable characteristics [24]:

- **CPU:** A 32-bit Tensilica Xtensa LX106 microprocessor with an 80 MHz maximum clock speed.
- **Memory:** To store data and applications, 80 KB of user-accessible SRAM and 4 MB of Flash memory are provided.
- **Connectivity:** Wi-Fi 802.11 b/g/n only.
- **Interfaces:** There are available UART, SPI, I2C, and GPIO interfaces enabling communication and interaction with a variety of peripherals and sensors.
- **Power:** The device can be powered by micro-USB or an external power supply with an input voltage range of 3.3V to 5V.

There are some major changes between the Wemos D1 Mini V4 and the classic ESP32. Both the WeMos D1 Mini V4 and the ESP32 are suitable for Internet of Things applications, but their strengths differ.

The WeMos D1 Mini V4 (Figure 3.2) is a powerful and compact development device based on the ESP8266 WiFi chip. It features a 32-bit processor, 11 digital

input/output pins, and a USB-C interface for programming convenience. With its compact size and low power consumption, the D1 mini V4 is ideal for Internet of Things projects, home automation, and other applications in which space and energy efficiency are crucial. It also has built-in support for the Arduino IDE, making programming and interfacing with the board simple for both novice and experienced developers. [24]

The ESP32, on the other hand, is a more powerful chip than the ESP8266, with greater memory and computational power. It is suited for IoT applications that are more complex and require more processing capacity and memory. It has more GPIO pins with features like hall effect and temperature sensor, faster Wi-Fi, supports Bluetooth 4.2 and Bluetooth low energy, and comes with touch-sensitive pins that can be used to wake up the ESP32 from deep sleep. However, the ESP32 is more expensive than the ESP8266 [25] [26].

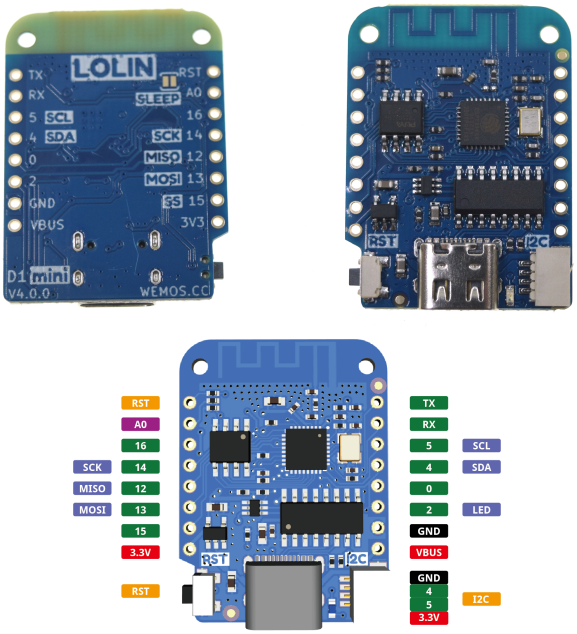

Figure 3.2: D1 Mini V4 front, back and pins

3.2.3 Raspberry the central Unit

The Raspberry Pi 3 Model B+ (Figure 3.3) is equipped with 1 GB of RAM, a BCM2837B0 wireless LAN, Bluetooth Low Energy (BLE), Gigabit Ethernet over USB, and 40-pin extended GPIO. It also has four USB 2.0 ports, stereo output

with four poles, and a composite video input. The Raspberry Pi 3 Model B+ is powered by a 2.5A adapter, which is recommended for maximizing its enhanced power management and supporting even more potent USB devices [27].

Home Assistant requires at least a Raspberry Pi 3 Model B [28], as earlier models, such as the Raspberry Pi 2, are incompatible. This incompatibility is predominantly the result of Home Assistant's increased processing power and memory requirements. The Raspberry Pi 3's are adequate for operating Home Assistant, whereas earlier Raspberry Pi models lacked the necessary processing power and memory capacity.

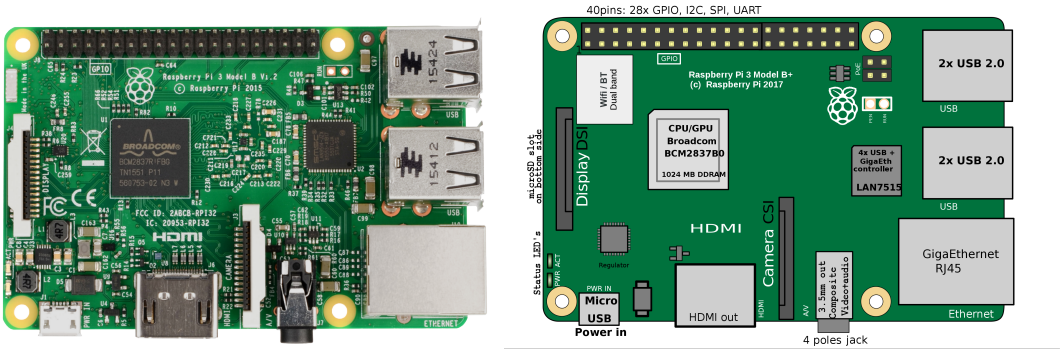

Figure 3.3: Raspberry Pi 3B+ top view and schematic

3.2.4 Smart Plugs

Smart outlets provide device-specific energy usage data in addition to smart meter integration system hardware. TP-Link's Tapo P110 Wi-Fi smart socket is used (Figure 3.4). This smart plug enables customers to monitor and regulate the energy consumption of connected equipment. The Tapo mobile app and Home Assistant integration offer users the ability to control the voltage of devices remotely. This feature enables users to turn off devices when they are not in use. Furthermore, the Tapo P110 includes scheduling and automation features, enabling users to set specific on/off times for connected devices or automate their power status based on external events like sunrise/sunset or presence detection [29] .

Figure 3.4: Smart Plug Tapo P110

3.3 Software Components and protocols

The proposed smart meter integration system relies on multiple software components and protocols to provide flawless data processing, communication, and integration with the Home Assistant platform. This section provides an overview of the system's most important software components and protocols.

3.3.1 MQTT Broker

MQTT (Message Queuing Telemetry Transport) is a lightweight publish-subscribe messaging protocol created for low-bandwidth, high-latency, or unreliable networks. It is the optimal solution for Internet of Things applications in which devices must communicate efficiently and dependably [30].

MQTT's architecture is client-server. In this instance, the Mosquitto MQTT broker which is an open source broker is installed on the Home Assistant server, functioning as the network's central communication hub. The ESP8266 D1 Mini and other smart home devices function as MQTT clients, publishing and receiving messages from the MQTT broker.

MQTT protocol's main concepts and components

The MQTT protocol ensures the expeditious and accurate transmission of energy consumption data in the smart home system by providing efficient and dependable device-to-device communication. It is well-suited for IoT applications and smart home systems due to its lightweight design and publish-subscribe architecture.

The main concepts of MQTT are [31]:

Broker: The MQTT broker is the central server responsible for receiving and distributing messages to MQTT consumers. In our case, the broker deployed on the Home Assistant server is Mosquitto MQTT.

Client: MQTT clients are devices that communicate to the MQTT broker in order to publish or receive messages from other clients. In our system, the ESP8266 D1 Mini and other smart home devices serve as MQTT clients.

Topics: In the MQTT system, topics are hierarchical identifiers used to classify and organize messages. Each message published to the broker has a topic, and clients subscribe to topics in order to receive messages. Multiple levels of topics, separated by forward slashes (/), permit the structured organization of messages.

Publish: When an MQTT client wants to send a message, it "publishes" the message to the broker with a specific topic. The broker then distributes the message to all clients subscribed to that topic.

Suscribe: MQTT clients can "subscribe" to specific topics, indicating their interest in receiving messages with those topics. When the broker receives a message with a topic that a client has subscribed to, it forwards the message to that client.

Using a specific topic, the ESP8266 D1 Mini publishes energy consumption data to the Mosquitto MQTT broker as part of our smart meter integration system. The Home Assistant server, which is also a MQTT client, subscribes to this topic and receives the ESP8266's energy consumption data. This information is then processed and incorporated into the Home Assistant platform, enabling users to monitor and analyze their energy consumption.

3.3.2 Firmware for Data Acquisition and Transmission

The custom firmware on the ESP8266 D1 Mini is designed to utilize the DSMR5 (Dutch Smart Meter Requirements Version 5) protocol to receive smart meter data from the P1 port. In the Netherlands and Belgium, the DSMR5 protocol is a widely adopted standard for smart meter communication. It is founded on a series of telegrams, as can be seen in the Figure 3.7, that the smart meter transmits at regular intervals. Each telegram comprises a collection of data objects representing various aspects of energy consumption, including values for power, voltage, and electrical current [32].

Firmware is responsible for processing raw data, parsing telegrams, and extracting relevant data from data objects. This requires identifying the beginning and

conclusion of each telegram and interpreting the data according to the DSMR5 standard. After extracting the pertinent data, the firmware converts it to a suitable transmission format, such as JSON.

Decoding DSMR5 Telegrams

The entire message, emanating from the P1 port in DSMR, appears as follows:

/	X	X	X	5	Identification	CR	LF	CR	LF	Data	!	CRC	CR	LF

Figure 3.5: Message example transmitted by the P1 port

And the data itself, contained within the telegram, appears as follows:

OBIS code				Values (multiple are possibe)						
0/1	Channel	:	Type	(	Value	*	Unit	)	CR	LF

Figure 3.6: Telegram data format

To decode DSMR5 telegrams, it is necessary to fully understand their framework and layout. A telegram is a message consisting of multiple lines, each of which represents a data object [33]. The data objects adhere to a particular syntax, which includes an OBIS code (Object Identification System) that designates the type of data contained within the object.

This example demonstrates how to decode a data object whose OBIS code is 2.8.1. This code represents the off-peak energy consumption in kWh.

A dispatch data object with an OBIS code of 2.8.1 might appear as follows:

```
1-0:2.8.1(001234.567*kWh)
```

A breakdown of the data object follows:

- `1-0`: This section specifies the device's logical address and channel. Typically, the logical device address for electricity meters is 1-0.
- `2.8.1`: This is the OBIS code representing the energy consumed during the off-peak tariff period in kWh.
- `( )`: The parentheses enclose the actual data value.

- `001234.567`: This is the actual energy usage in kilowatt-hours. In this instance, the value is 1,234,567 kWh.
- `*kWh`: This section specifies the unit of measurement, which in this case is kilowatt-hours (kWh).

To decode the telegram and derive the information regarding off-peak energy consumption, the line containing the OBIS code 2.8.1 must be located first. Once located, the line can be parsed to obtain the energy consumption value and its unit of measure. The decoded value in this example is 1,234,567 kWh, which represents the amount of energy consumed during the off-peak tariff period.

By following a similar procedure, it is possible to decode and extract additional data objects from the telegram, thereby obtaining detailed information regarding energy consumption and other smart meter parameters.

In specific documents, including the eMUCS - P1 (extended Muli-Utility Companinon Specification for the Consumer Interface P1) [33], which was designed by Fluvius, Ores, Resa, and Sibelga, you can find a list of OBIS codes and their specifics. The most recent version is 1.7.1, which was released on September 5, 2022.

```
0-0:96.1.4(xxxxx)
0-0:96.1.1(xxxxxxxxxxxxxxxxxxxxxxxxxxxx)
0-0:1.0.0(210204163628W)
1-0:1.8.1(000439.094kWh)
1-0:1.8.2(000435.292kWh)
1-0:2.8.1(000035.805kWh)
1-0:2.8.2(000012.156kWh)
0-0:96.14.0(0001)
1-0:1.7.0(00.233kW)
1-0:2.7.0(00.000kW)
1-0:21.7.0(00.233kW)
1-0:22.7.0(00.000kW)
1-0:32.7.0(236.2V)
1-0:31.7.0(002.04A)
0-0:96.3.10(1)
0-0:17.0.0(999.9kW)
1-0:31.4.0(999A)
0-0:96.13.0()
0-1:24.1.0(003)
0-1:96.1.1(xxxxxxxxxxxxxxxxxxxxxxxxxxxx)
0-1:24.4.0(1)
0-1:24.2.3(210204163500W)(00343.925*m3)
!1374
```

Figure 3.7: Exemple of a telegram

Example in our context

```
- name: P1 Actual Power Consumption
  unique_id: 'sensor.p1_actual_power_consumption'
  device_class: power
  state_class: measurement
  unit_of_measurement: 'kW'
  state_topic: "sensors/power/p1meter/actual_consumption"
  value_template: "{{ value|float / 1000 }}"
```

In this example, we will study a configuration entry from the MQTT broker installed on Home Assistant that represents the actual electricity consumption of the smart meter. Here is a description of the MQTT topic and the corresponding data:

- `name`: A human-readable designation for the data point.
- `unique_id`: A unique identifier for the sensor.
- `device_class`: The category of the device or sensor.
- `state_class`: This defines the type of state represented by the sensor.

- `unit_of_measurement`: The unit in which the power consumption data is measured.
- `state_topic`: This specifies the MQTT topic used for the data point.
- `value_template`: This is a template for processing the received MQTT message's value.

3.3.3 Home Assistant Integration

Combining smart meter data with Home Assistant is a crucial component of the proposed system, as it offers customers with a unified platform to monitor and manage their energy use alongside other smart home devices. As the central hub, the Raspberry Pi 3 running Home Assistant receives the smart meter data from the ESP8266 microcontroller and processes it for smooth integration into the Home Assistant platform.

To achieve this integration, the following steps should be taken:

Data reception and processing: The Raspberry Pi 3 must be set to receive the smart meter data supplied through the MQTT protocol by the ESP8266 microcontroller. On the Raspberry Pi 3, a MQTT broker, such as Mosquitto, to be installed to process incoming messages. The smart meter data must be analyzed and processed to obtain pertinent information for further study and Home Assistant integration.

Home Assistant integration: Home Assistant must be set up and configured for smart meter data processing and presentation. Creating custom sensors, entities, and templates inside the configuration files of Home Assistant enables the platform to detect and analyze incoming data.

Visualization and automation: The user interface of Home Assistant may be modified to show smart meter data in many ways, including graphs, charts, and gauges. Users may establish automation rules and triggers based on smart meter data, enabling them to get alerts, operate smart devices, and optimize energy use.

Connected smart plugs integration: Connected smart plugs are included into the suggested system to give a more complete and comprehensible energy monitoring solution. These smart plugs allow customers to monitor the energy use of each connected appliance and gadget. The data from smart plugs may be

included into Home Assistant, allowing users to pinpoint the origins of energy consumption spikes in the overall consumption graph.

By integrating smart meter data with Home Assistant, consumers get access to a variety of energy use statistics and management choices. The combination of smart meter data with data from linked smart plugs provides a comprehensive view of energy usage trends, allowing customers to make educated decisions on how to optimize their energy consumption and save expenses.

3.3.4 Customization and Automation Features

The integration of smart meters with home automation systems, such as Home Assistant, gives options for customization and automation capabilities that enable customers to more efficiently optimize their energy consumption. The usage of smart plugs to monitor and regulate energy consumption spikes is one such function.

Users can determine the origins of energy consumption peaks and design ways to reduce energy consumption with this data. The following are examples of customization and automation capabilities [34] that can be achieved using Home Assistant and smart plugs:

1. **Individual device monitoring:** Create custom dashboards in Home Assistant to monitor the energy consumption of each smart plug-enabled item. This enables customers to simply monitor the energy use of certain equipment and detect energy wasters.

2. **Automated device control:** By creating Home Assistant automations, it is possible to reduce the energy consumption of certain appliances during periods of high overall energy consumption. This can be achieved by switching off non-essential appliances during peak hours.

3. **Energy consumption alerts:** Configure Home Assistant notifications to warn users when the energy consumption of a certain device exceeds a predetermined threshold. This can assist customers detect devices that consume more energy than intended, allowing them to replace or repair the item as necessary.

4. **Automated device control:** Analyze the obtained energy usage data from smart plugs and the smart meter to determine trends and patterns. This data can be utilized to design more effective energy consumption plans and automations, as well as to advise users of energy-saving opportunities.

By utilizing Home Assistant, smart plugs, and smart meter data, customers may acquire a deeper insight of their energy consumption habits and optimize their energy usage using automation capabilities, ultimately leading to a more sustainable and cost-effective lifestyle [35].

3.3.5 Overview Diagram

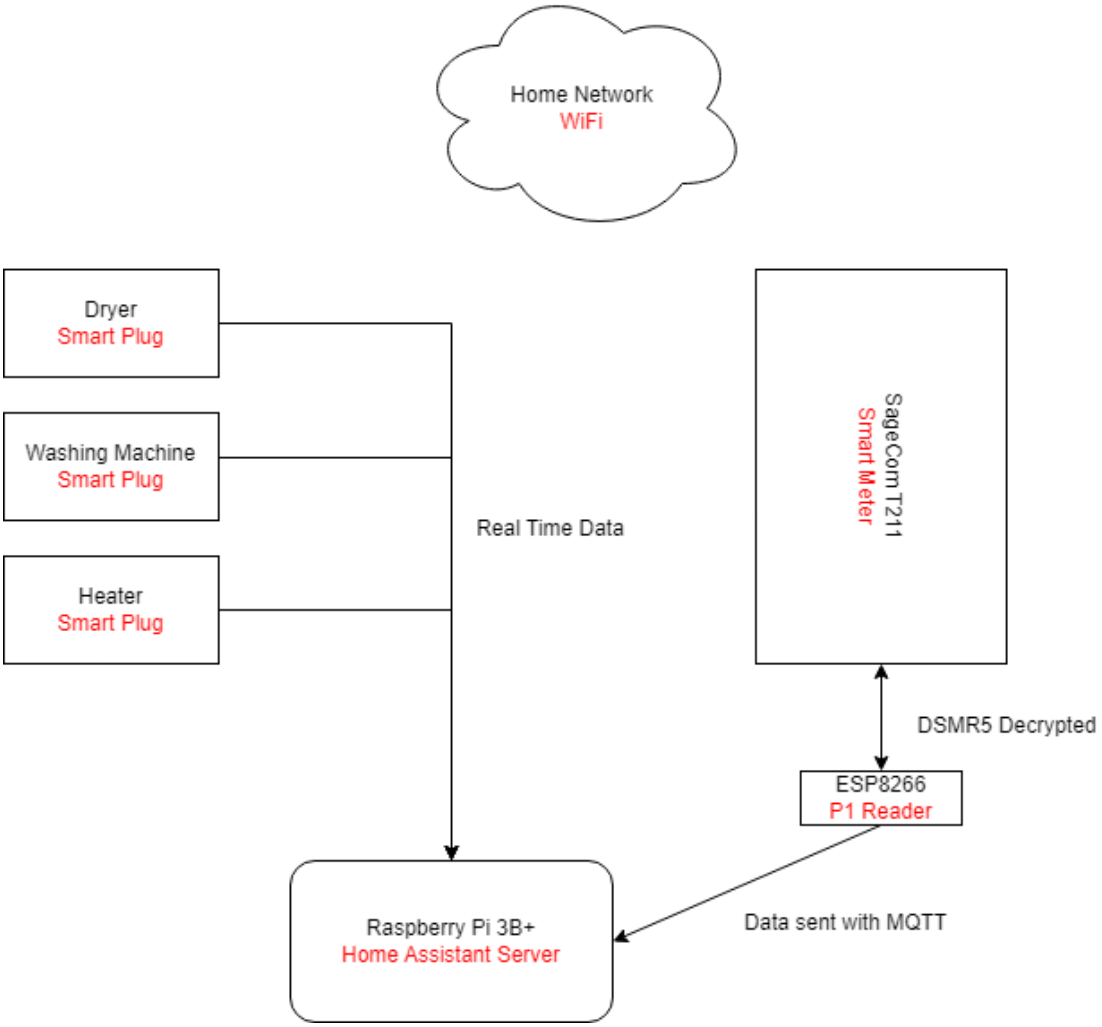

Figure 3.8: Overview diagram of our system

Chapter 4

Implementation and Testing

4.1 Hardware Assembly and Connections

An IoT-based smart meter monitoring system relies heavily on the assembly and connecting of hardware components to run without interruption. Appropriate device connections allow for the effective transmission of data while preserving the stability and dependability of the system. In this section, we describe the hardware assembly method and connections required to construct a fully functional smart meter monitoring system, focusing on the integration of the smart meter, ESP8266 D1 Mini microcontroller, and Raspberry Pi 3B +.

This hardware concept and the code that goes with it is based on open-source projects [36] that have been updated for recent smart meters (with DSMR5).

4.1.1 Smart Meter Connection

The connection between the smart meter and ESP8266 D1 Mini is an essential component of the proposed smart meter integration solution. To establish this connection, an RJ12 to 6-pin Dupont adapter cable is utilized.

This cable is selected due to its compatibility with the RJ12-connector-equipped P1 port on the smart meter. The opposite end of the cable contains six Dupont jumper wires that can be readily connected to the ESP8266 D1 Mini's GPIO pins. Without the need for additional adapters or soldering, this cable enables a simple and secure connection between the smart meter and the ESP8266.

In addition, the ESP8266 D1 Mini does not require an additional power source thanks to this connection method [36], as it is powered directly through one of the RJ12 cable's ports as shown in Figure 4.1. In particular, the 5V pin on the P1 port of the smart meter provides power to the ESP8266.

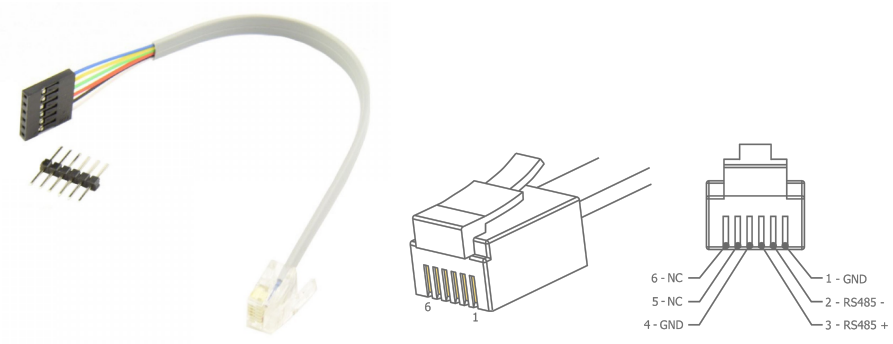

Figure 4.1: RJ12 overview

4.1.2 ESP8266 D1 Mini Setup

The ESP8266 D1 Mini is a crucial component of the proposed smart meter integration solution, as it serves as the primary data acquisition and processing unit.

Hardware Assembly

Connecting the ESP8266 D1 Mini to the smart meter via an RJ12 to 6-pin Dupont connector adapter provides a secure and reliable connection for data acquisition. Figure 10.1 in Appendix A shows our system assembled and plugged into port P1 of the meter, Figure 10.2 shows our system disconnected. Figure 4.2 shows the electronic diagram.

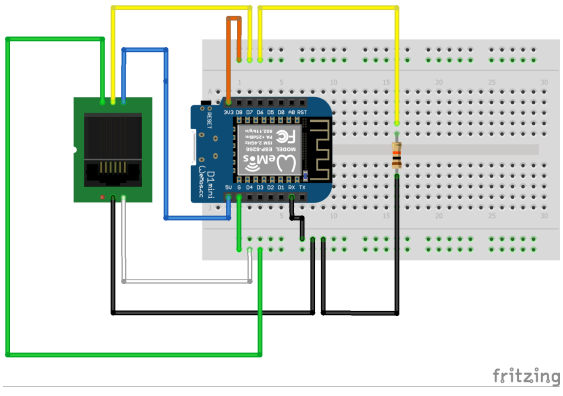

Figure 4.2: P1 port reader electronic diagram

Firmware and Configuration

Using the DSMR5 protocol, the ESP8266 D1 Mini's custom firmware is designed to receive and process smart meter data from the P1 port. This firmware is responsible for parsing raw data, extracting pertinent information, and converting it to a transmission-friendly format. Once programmed and configured, the ESP8266 D1 Mini uses the MQTT protocol to transmit processed data to the Raspberry Pi, ensuring efficient and low-latency communication.

4.1.3 Raspberry Pi Setup

The Raspberry Pi 3 B+ functions as the energy monitoring system's central processing unit and home automation server. The following procedures were carried out to configure the Raspberry Pi with Home Assistant and Mosquitto Broker:

1. **Home Assistant OS Preparation:** At the time of writing, the *haos_rpi3-64-9.5* version of Home Assistant OS for the Raspberry Pi 3 64-bit was downloaded. To ensure compatibility with the Raspberry Pi 3 B+, it is crucial to choose the correct version.

2. **Flashing Home Assistant OS:** Using BalenaEtcher, the Home Assistant OS image was flashed onto a microSD card. This procedure required selecting the downloaded Home Assistant OS image (haos_rpi3-64-9.5.img) as the source file, selecting the microSD card as the target, and executing the flashing procedure.

3. **Initial Home Assistant Setup:** Home Assistant was initially configured by configuring the Wi-Fi network, creating a user account, and adding the location and time zone information after booting.

4. **Mosquitto Broker Installation:** As an add-on to Home Assistant, Mosquitto Broker, a MQTT server implementation, was installed to facilitate MQTT communication between the ESP8266 D1 Mini and the Raspberry Pi.

Following these instructions, the Raspberry Pi 3 B+ was effectively configured to host the Home Assistant platform and Mosquitto Broker, enabling efficient and reliable energy monitoring communication with the ESP8266 D1 Mini.

4.1.4 Smart Plug Configuration

The Tapo P110 smart sockets enable homeowners to monitor the energy consumption of individual appliances. Following are the steps taken to configure the smart devices for use with the Home Assistant platform:

1. **Initial Smart Plug Setup:** The Tapo P110 smart plug was inserted into an electrical outlet, and the monitored device was then connected to the smart plug.

2. **Official Tapo App:** The official Tapo mobile app was downloaded to a smartphone, which was then used to configure the smart outlet for the first time. This included attaching the smart plug to the Wi-Fi network and assigning it a unique name for identification purposes.

3. **Home Assistant Community Store (HACS) Integration:** The Home Assistant Community Store (HACS) was installed in Home Assistant. HACS is a repository of custom components and integrations that can be added to the Home Assistant platform with minimal effort.

4. **Tapo P100 Integration:** HACS was used to install the Tapo P100 integration. This integration enables Home Assistant to seamlessly interact with Tapo P110 smart plugs.

5. **Home Assistant Configuration:** After deploying the Tapo P100 integration, Home Assistant detected the smart plugs automatically. The smart outlets were then configured in the Home Assistant user interface, allowing users to monitor energy consumption, control devices, and create automation rules as necessary.

   We decided to connect three electronic appliances with high energy consumption to these sockets: an electric heater, a washing machine and a tumble dryer. As the dishwasher is a built-in appliance, we were unable to check its performance, even though this would have been very interesting. The fridge and freezer were not checked because their constant energy consumption was not intriguing.

4.1.5 Wireless Connectivity

The proposed smart meter integration system relies heavily on wireless connectivity, which enables seamless communication between various components such as the ESP8266 microcontroller, Raspberry Pi, and Tapo P110 smart outlets. Using wireless connectivity within the system reduces installation complexity, improves adaptability, and facilitates scalability. The following protocols are used within the system:

- **Wi-Fi** is the primary wireless communication protocol used within the system, providing a quick and dependable link between the ESP8266 microcontroller, Raspberry Pi, and the local network.

- **MQTT** is the main communication protocol between the ESP8266 microcontroller and the Raspberry Pi is MQTT. On the Home Assistant platform, the Mosquitto MQTT broker is used to facilitate this communication. MQTT's publish/subscribe communication patterns, as well as its ability to operate efficiently with minimal bandwidth requirements, make it an ideal protocol for Internet of Things applications.

When utilizing wireless connectivity protocols such as Wi-Fi and MQTT, the proposed system guarantees a robust and dependable connection between its various components.

4.1.6 System Integration

System integration is essential to the proposed smart meter integration solution because it enables the seamless combination of multiple hardware and software components to form a unified and functional system. This section explains how to integrate the ESP8266 microcontroller, Raspberry Pi, Home Assistant platform, and Tapo P110 smart plugs to construct our system.

Data Acquisition and Transmission

Connected to the smart meter's P1 interface, the ESP8266 microcontroller acquires real-time energy consumption data using the DSMR5 protocol. The ESP8266 analyses the raw data and extracts pertinent information before transmitting it to the Raspberry Pi via the MQTT broker installed in Home Assistant.

Data Processing and Integration

The Raspberry Pi processes and stores the transmitted data from the ESP8266 in preparation for further analysis. The Raspberry Pi-hosted Home Assistant platform integrates smart meter data with other home automation devices and services, allowing users to monitor and control their energy consumption through a unified interface. The Tapo P110 smart outlets, when configured and connected to the Home Assistant platform, offer users a more detailed view of their energy consumption patterns via individual device energy monitoring.

4.2 Software Development and Configuration

The smart meter integration system relies on a combination of software components, tools, and settings to facilitate efficient communication, data processing, and seamless integration with the Home Assistant platform. This section provides

an overview of the software used and configuration processes involved in creating and maintaining the proposed system. It will cover the custom firmware for the ESP8266 D1 Mini, Home Assistant setup on the Raspberry Pi, MQTT and Mosquitto Broker configuration, smart plug integration, and data processing and visualization techniques.

4.2.1 ESP8266 D1 Mini Firmware

The logic of the firmware for the ESP8266 D1 Mini is based on several steps in order to manage the communication with the P1 meter, the connection to the WiFi network, and the communication with the MQTT server in an efficient manner.

The program begins by initializing the required libraries and declaring the variables and objects used to administer the various functionalities. Then, it establishes a connection to the specified WiFi network with the necessary credentials, for communication with the MQTT server. Simultaneously, it receives and processes data from the P1 meter, decoding the various parameter values, such as current, voltage, and power consumption. Once the data from the P1 meter is decoded, it is transformed into formatted MQTT messages and transmitted to the MQTT server. This process operates continuously, ensuring constant and seamless communication between the P1 meter and the MQTT server.

4.2.2 Energy Panel

We have incorporated the data retrieved by the MQTT server, including that of the P1 meter, into the corresponding sections of the energy panel in our configuration.

4.3 Transition to Data Analysis and Prediction

We have completed the first part of our study, Data Acquisition and Integration, focusing on collecting and integrating energy data from IoT devices in smart homes. This lays the foundation for the next part of our study, Data Analysis and Prediction. Here, we aim to analyze this integrated data to identify energy consumption patterns and devise machine learning models for predicting future energy needs. These insights and predictive models can aid homeowners in managing their energy use more effectively and contribute to energy efficiency and cost savings. We then delve deeply into Data Analysis and Prediction, exploring the methods, results, and their implications for energy management in smart homes.

Part II

Data Analysis and Prediction

Chapter 5

Literature Review

5.1 Data Analysis and Machine Learning for Energy Consumption Prediction

Predicting energy consumption is essential for optimizing energy use, reducing costs, and promoting sustainable energy consumption practices. In recent years, the combination of data analysis and machine learning has emerged as an effective method for predicting energy consumption, allowing for the creation of more accurate and trustworthy models.

Traditional Approaches to Energy Consumption Prediction

In the past, energy consumption has been predicted using time series analysis, regression models, and statistical techniques such as Autoregressive Integrated Moving Average (ARIMA), Exponential Smoothing State Space Model (ETS), and Multiple Linear Regression (MLR). In spite of the fact that these techniques have proven effective in particular situations, they are limited by their reliance on linear relationships and the underlying data distribution assumptions.

Numerous studies have utilized time series analysis to forecast energy consumption. Using time series analysis, a project aimed at predicting power consumption in 10-minute windows for the Moroccan city of Tétouan found that the 15-day SMA and the hour of that specific measurement were the two most significant factors, while temperature was the only external factor that was relatively significant in predicting energy consumption [37].

Another study suggested using ARIMA models to forecast future electricity consumption. In a separate paper, kernel principal component analysis (KPCA)

and Support Vector Machine (SVM) were used to forecast time series for electricity consumption [38].

However, these techniques have limitations as a result of their reliance on linear relationships and the underlying data distribution assumptions. Several studies have proposed alternative methods, such as kernel principal component analysis (KPCA) and support vector machine [39] or data analysis-based time series forecast [40], to resolve these limitations.

Machine Learning and Boosting Techniques for Energy Prediction

Energy consumption prediction is made more flexible and adaptable through the use of machine learning. Various energy prediction tasks have benefited from the application of Artificial Neural Networks (ANN), Support Vector Machines, Decision Trees, Random Forests, and boosting algorithms, as compared to conventional methods. These algorithms can capture non-linear relationships and complex patterns within the data, making them better suited for dealing with diverse and dynamic energy consumption scenarios.

In the field of energy consumption forecasting, boosting techniques, especially Gradient Boosting, have acquired popularity. Gradient Boosting is an ensemble learning technique that combines several weak learners (typically decision trees) to produce a strong learner. The Gradient Boosting Regressor (GBR) is a popular variant that minimizes residual errors by fitting weak learners iteratively to the negative gradients of the loss function. This method permits the GBR to effectively capture non-linear relationships and interactions between features, resulting in enhanced prediction accuracy and generalization performance [41] [42].

Feature Engineering and Selection

Engineering and selecting features are essential components of energy consumption prediction using machine learning. Identifying pertinent input variables (features) that can be used to train the model and make predictions constitutes the process. Typically, historical consumption data, weather variables, time-related variables (such as day of the week and hour), and appliance usage data are used to predict energy consumption. Feature selection techniques, such as Recursive Feature Elimination (RFE) and LASSO regularization, help identify the model's most informative features, thereby reducing noise and improving model performance [43] [44] [45]. The development and demonstration of a Python framework for feature engineering using an energy demand prediction use case. The results of the case study indicate an increase in the accuracy of predictions as a result of the applied feature engineering processes [45].

Model Evaluation and Performance Metrics

To evaluate the efficacy of machine learning models for predicting energy consumption, suitable performance metrics are required. Mean Absolute Error, Mean Squared Error, Root Mean Squared Error, and Coefficient of Determination are common metrics used in this context [46]. These metrics shed light on the accuracy and dependability of the model's predictions and inform the model selection and hyperparameter optimization processes.

5.2 Energy Optimization and Management Strategies

Increasing environmental consciousness and rising energy costs have sparked a growing interest in devising energy optimization and management strategies. Effective energy management seeks to reduce energy consumption, minimize waste, and enhance overall performance. Various energy optimization and management strategies that have been proposed and implemented in the literature are discussed in this section.

Energy Audits and Monitoring

Energy audits involve systematic evaluations of a building's or facility's energy consumption patterns to identify areas of inefficiency and potential energy-saving opportunities. Typically, these audits consist of monitoring energy consumption with sensors, analyzing historical energy usage data, and conducting on-site inspections to assess the efficiency of apparatus and systems. On the basis of the results of an energy audit, recommendations can be made for increasing energy efficiency through equipment upgrades, operational changes, or behavioral adjustments.

Demand-Side Management

Demand-side management (DSM) refers to strategies designed to influence end-users' energy consumption patterns in order to optimize the supply and demand balance by using strategies:

- Programs that incentivize consumers to reduce or transfer energy consumption during periods of peak demand [47] [48].
- Time-of-use pricing schemes that reduce energy prices during off-peak hours to encourage energy consumption [47] [48].

- Energy efficiency measures designed to reduce energy consumption by improving the energy efficiency of appliances, systems, and processes [48] [49].

DSM aims to reduce the disparity between power supply and demand by increasing energy efficiency, conserving energy, and modifying consumer energy demand [49] [50].

In this study, we integrate the principles into our smart home energy system to optimize energy usage. Our system provide homeowners with detailed energy consumption data, promoting an understanding of their energy habits and inspiring changes to reduce usage during peak demand. By leveraging machine learning, our system predict future energy needs and suggest optimal usage schedules, supporting DSM's intent. Energy-intensive appliances can be scheduled for off-peak hours, reducing energy costs. For synthesized, our system help identify inefficient appliances or habits, encouraging energy conservation. Our aim is to equip homeowners with tools to manage their energy consumption effectively, contributing to a sustainable energy future.

Renewable Energy Integration

Incorporating renewable energy sources, such as solar and wind power, into the energy balance can aid in decreasing reliance on fossil fuels and greenhouse gas emissions. Energy storage systems, smart grid technologies, and microgrids can be implemented to maximize the benefits of renewable energy. Batteries and pumped hydro storage are examples of energy storage systems that can store excess renewable energy for use during periods of low generation [51] [52] [53] [54]. Smart grid technologies permit real-time monitoring and control of energy generation, distribution, and consumption, thereby facilitating the integration of renewable energy sources [55]. Microgrids are decentralized energy networks that are capable of operating independently from the main grid and can be designed to optimize the use of renewable energy sources within a community or facility [56].

Energy Management Systems

Energy Management Systems (EMS) are software-based instruments that aid in monitoring, controlling, and optimizing the energy consumption of a building or facility. Typically, EMS collect and analyze real-time energy consumption data from sensors and smart meters, allowing facility administrators to identify inefficiencies and implement energy-saving measures. Advanced EMS may also include machine learning algorithms to predict energy consumption patterns and provide automated appliance and system control to optimize energy use [57].

In our study, we aim to implement our own Energy Management System (EMS) using the combination of Internet of Things devices for data acquisition and machine learning for data analysis and prediction. This custom EMS is tailored to fit the requirements of a residential setting, allowing homeowners to have real-time visibility of their energy consumption and enabling them to make informed decisions about their energy use.

The key components of our EMS include smart meters and connected devices that provide real-time data on energy consumption, and a data processing and analytics platform, in this case, Home Assistant, which aggregates and analyses this data to identify usage patterns and inefficiencies.

In the next sections, we present our own machine learning algorithms that are used to analyze this data, allowing us to predict future energy consumption patterns and provide recommendations for optimizing energy usage schedules.

Chapter 6

Data Analysis

The emergence of smart meters and the Internet of Things has made it possible to capture vast quantities of data pertaining to energy consumption, production, and usage patterns.

In this chapter, we concentrate on the methods and techniques used to extract meaningful information from the energy data collected by our smart meter integration system. We investigate several facets of energy consumption, including patterns and trends, seasonal and time-of-day variations, and profiles of household and appliance utilization. This analysis then inform the development of energy optimization strategies and automation features for more efficiently managing energy consumption.

6.1 Retrieve the data

To initiate the data analysis process, we began by accessing the stored smart meter data in Home Assistant. This involves retrieving the necessary information from the entire Home Assistant database via sql query (SQLite Studio is a robust and user-friendly utility that enables us to interact with the Home Assistant database).

Once the desired data has been retrieved, it is exported to a CSV (Comma Separated Values) file. This widely-used file format facilitates further data processing and analysis because it can be readily imported into a variety of data analysis tools and programming languages.

6.2 Preprocessing

After retrieving the relevant data from the Home Assistant database, it is necessary to preprocess it to ensure that it is clean, well-structured, and suitable for further analysis. In our case, the preprocessing step entails organizing the data, aggregating it by hour, and removing unnecessary information.

First, the data pertaining to the energy consumption of the three connected appliances (dryer, washing machine, and electric heater), the overall energy consumption, and the excess energy returned to the grid (produced by solar panels) were analyzed. These data elements provided the most relevant information for analyzing consumption patterns and identifying optimization opportunities.

To facilitate analysis and improve visualization of consumption patterns, we aggregate the data hourly. By doing so, we can gain a deeper comprehension of the energy consumption trends and identify the periods of highest and lowest energy consumption. This information is essential for determining the optimal periods to operate energy-hungry appliances and optimizing the use of available solar energy.

In addition, preprocessing may entail dealing with missing or incorrect data points, ensuring that the data is complete and accurate for subsequent analysis steps. By meticulously preprocessing the data, we can ensure the reliability and significance of our findings, which will ultimately result in more effective energy management strategies.

6.3 Analysing

After preprocessing the data, the next stage is to analyze it in order to gain insights into energy consumption patterns, identify opportunities for optimization, and gain a deeper understanding of the factors that influence household energy consumption. Combining exploratory data analysis, visualizations, and statistical methods, the analyzing phase identifies trends, correlations, and anomalies in the data.

Home Assistant users can also view these graphs (in a different format, such as a stick graph) directly on the application using the graphs we have created. The graphs we have reproduced are merely for illustrative purposes and have been manually created from the retrieved data in order to give a professional view of habits.

6.3.1 Patterns and Trends in Energy Consumption

This analyzing phase is divided into two parts. The first subsection is entitled "Return Delivery and Consumption", which includes two figures. Figure 6.1 presents a trend heat map for return delivery to the grid, while Figure 6.2 showcases a trend heat map for consumption. The second subsection focused on "Smart Plugs" and features Figure 6.3, which displays a trend heat map specifically for smart plugs.

Return Delivery and Consumption

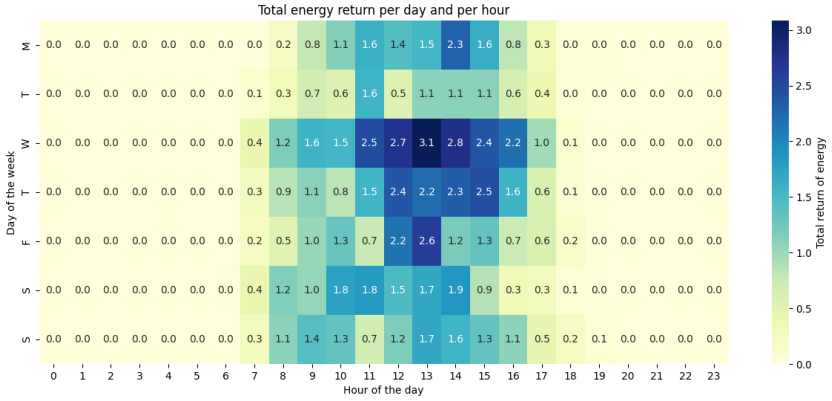

Figure 6.1: Trend heat map for return delievy to the grid

Based on the interpretation of the data presented in Figure 6.1, we can observe that the increase in consumption can be attributed to specific activities or an increased usage of certain devices.

By analyzing the data shown in Figure 6.2, we can clearly observe that the highest amount of energy feedback to the grid occurs at 1 p.m. This is a common occurrence when energy generated from renewable sources, in this case solar panels, exceeds consumption. The surplus energy is then fed back into the grid. Midday is expected to be the time when solar energy production reaches its zenith, as this is when the sun's rays are at their strongest.

In addition, the data reveal that the peak energy consumption occurs at 6 a.m. This increase in energy consumption could be attributed to the use of energy-intensive appliances at this time. From the information available, it seems that heating has a high energy consumption.

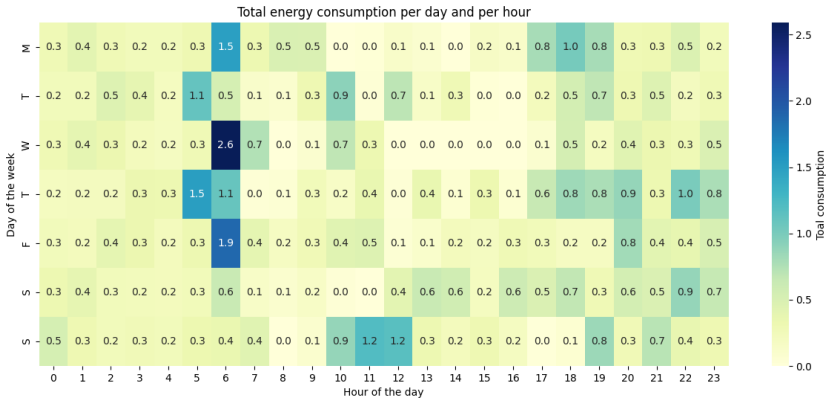

Figure 6.2: Trend heat map for the consumption

Smart Plugs

In the following figure 6.3, we have started by analysing the hourly consumption of each connected outlet:

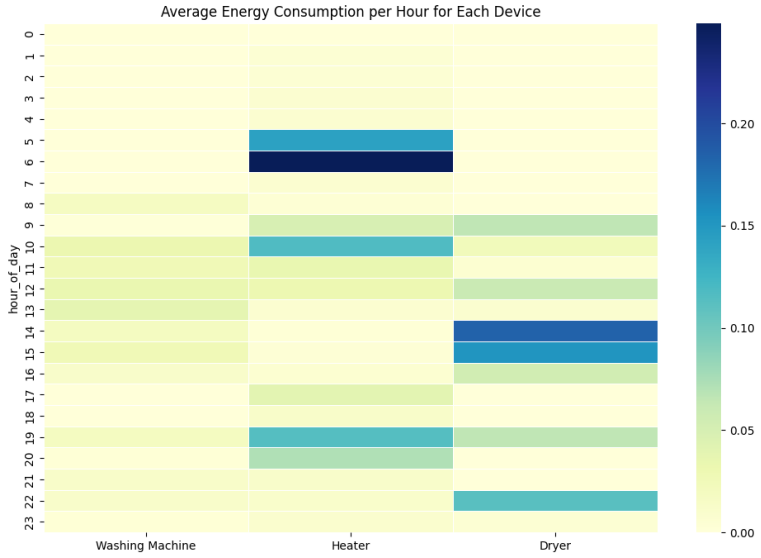

Figure 6.3: Trend heat map for smart plugs

Based on an analysis of the data presented in Figure 6.3, the washing machine is the appliance with the lowest energy consumption. The dryer uses the second-most energy on average, while the radiator utilizes the most energy between 5 a.m. and 6 a.m., which may seem odd but here is a rational explanation for this; the radiator is merely programmed to operate daily at these times. However, because the programming is obsolete, this consumption is unnecessary. Due to this, the family was able to recognize and rectify the issue. On the page in Appendix B, a bar chart (Figure 11.4) shows the monthly consumption of our 3 connected sockets as seen by the user.

Energy-consuming appliances

Several important observations can be made by analyzing the energy consumption of the various appliances connected to the three monitored outlets in our study over one month.

Initially, the radiator was determined to be the largest energy consumer, utilizing **21.32** kWh per month. Given that this is the only high-consumption appliance being monitored, this result highlights the potential importance of correctly programming such appliances. Secondly, the clothes drier was the second highest energy consumer, consuming **18,21** kWh during the same time period. This is a significant quantity of energy, but it is slightly less than the radiator. With **5.95** kWh, the laundry machine consumed the least energy among the three appliances. Despite this lower consumption, managing the operation of washing machines can still result in significant energy and cost savings.

Chapter 7

Machine Learning Model for Prediction

The application of machine learning techniques to energy consumption data enables the development of predictive models that can predict future energy utilization patterns, identify potential areas for improvement, and facilitate the implementation of effective energy management strategies. In this section, we will describe the process of developing a machine learning model for predicting energy consumption, as well as the benefits it contributes to the management of residential energy.

Using the preprocessed and analyzed data, we will train a machine learning model capable of predicting the amount of energy returned to the grid, taking into account factors such as appliance consumption, solar energy production returned to the grid, and weather.

Since we are using historical data, the weather data is accurate, but it may not be suitable for real-time forecasting. In this chapter, we presume that the forecast is made on the same day and attempt to make predictions for all hours of the day in order to obtain the most accurate overview possible.

7.1 Feature Selection and Preprocessing

7.1.1 Feeding the dataset

The primary objective of our predictive model is to forecast the amount of solar energy produced by a household which is not utilized per hour, which is then returned to the grid. In order to make accurate predictions, it is crucial to identify the most pertinent features from our dataset. In this instance, our target variable

is the energy returned to the infrastructure, and the hour is one of the essential features.

As the energy returned to the grid is indirectly related to solar energy production, which is influenced by meteorological conditions, our dataset must include weather-related characteristics. Home Assistant's categorical weather information (e.g., cloudy, sunny, smog) is insufficient for our model. Therefore, we turned to external sources for more precise and comprehensive weather information.

We retrieved about a hundred additional weather-related variables from the Open Meteo API (Application Programming Interface) [58], including UV index, atmospheric pressure, cloud cover percentage by type (high, medium, and low), wind speed, etc. These variables may contribute to our model and assist in predicting the amount of energy returned to the grid.

To generate a new dataset, we gathered historical weather data from the same location as the household and combined it with the data we already had from Home Assistant. This augmented dataset, which includes both energy consumption and meteorological data, serves as the basis for our machine learning model, enabling it to make more accurate and meaningful predictions.

7.1.2 Feature Selection

With approximately about one hundred variables in our dataset, it is essential to identify the most pertinent characteristics that will result in a more accurate predictive model. To accomplish this, multiple feature selection methods were utilized, including the SelectKBest method from the sklearn.feature_selection library.

SelectKBest is a method for feature selection that selects the top K features based on their univariate statistical relationship with the target variable [59]. By eradicating irrelevant or redundant features, the method helps reduce the dimensionality of the dataset and enhance the performance of the model.

To determine the optimal value of K, we developed a script to iterate through various K values and evaluate the efficacy of the model using the R-squared metric. R-squared represents the proportion of the variance in the dependent variable that can be predicted by the independent variables. In our case, it assesses the extent to which the selected features can explain the variation in energy returned to the grid.

The relationship between the R-squared value and the number of features (K) can be represented graphically to help determine the optimal K value. This graph

will illustrate how the R-squared value varies with the number of selected features, allowing us to choose an appropriate value of K that maximizes the R-squared value while minimizing the number of features used in the model.

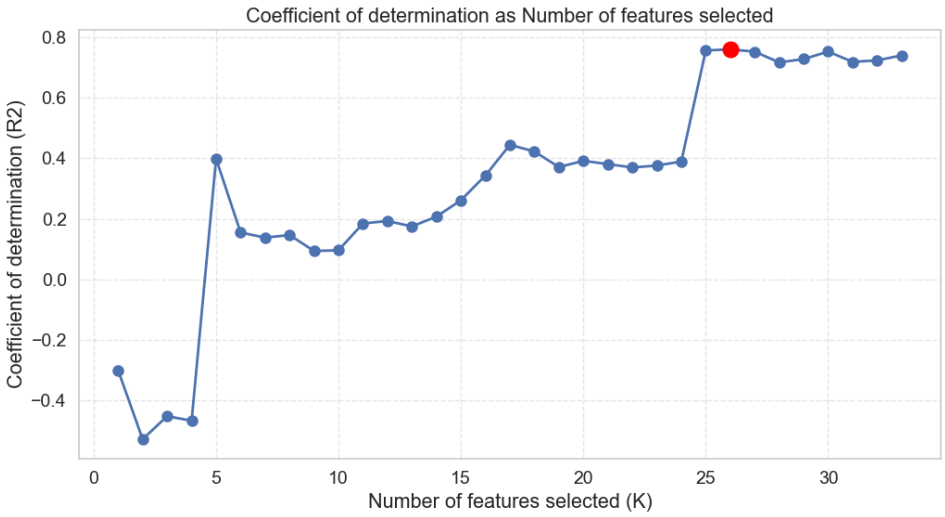

Figure 7.1: Plot of the SelectKBest function

The optimal value for K, as indicated by our graph in Figure 7.1, is 26, which corresponds to an R-squared value of 0.76. This result indicates that the 26 selected features can explain 76% of the variance in the energy returned to the grid, which is a respectable predictive model performance. A higher R-squared value indicates a better fit between the model and the data, whereas a lesser R-squared value indicates a poorer fit.

The selected variables, as shown in Figure 7.2, are those considered to be the best for prediction.

```
Best K: 26
Best R2: 0.76
features : Index(['temperature_2m', 'apparent_temperature', 'precipitation_probability',
       'rain', 'weathercode', 'pressure_msl', 'surface_pressure',
       'cloudcover_low', 'cloudcover_mid', 'cloudcover_high', 'visibility',
       'windspeed_10m', 'windspeed_80m', 'windspeed_120m', 'windspeed_180m',
       'winddirection_10m', 'winddirection_80m', 'winddirection_120m',
       'winddirection_180m', 'windgusts_10m', 'temperature_80m',
       'temperature_120m', 'temperature_180m', 'soil_temperature_0cm',
       'soil_temperature_6cm', 'soil_temperature_18cm',
       'soil_temperature_54cm', 'uv_index', 'uv_index_clear_sky', 'is_day',
       'hour', 'day', 'month'],
      dtype='object')
```

Figure 7.2: Features selected for our model

We also calculated the Mean Squared Error and Mean Absolute Error for our model in addition to the R-squared value. These metrics aid in evaluating the performance of the model and its predictive accuracy.

The definition of Mean Squared Error is as follows [60]:

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$$

Where y_i is the actual value, $\hat{y}_i$ is the predicted value, and $\hat{y}_i$ and n is the number of observations. MSE measures the average squared difference between the actual and predicted values. A lower MSE indicates better model performance. In our case, the MSE is 0.20.

The Mean Absolute Error is defined as [60]:

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i|$$

The MAE quantifies the average absolute deviation between actual and predicted values. Similar to MSE, a smaller MAE signifies improved model performance. The MAE in our case is 0.22.

The obtained R-squared, MSE, and MAE values indicate that our model predicts the energy returned to the grid reasonably well. The selected features considerably contribute to the performance of the model, and the model's accuracy is adequate for this type of prediction.

7.1.3 Preprocessing

We did not have to perform extensive data cleansing in our preprocessing phase because our dataset consists primarily of numeric variables and contains no missing data. To extract useful features and prepare the data for our machine learning model, we did conduct some preprocessing steps.

First, we converted the 'time' column to a datetime format and extracted time-related characteristics including hour, day, and month. This was essential to our analysis because it permitted us to investigate the relationship between time and energy production.

After converting and extracting the time-related characteristics, the input columns (X) and the target column (y) were selected. We removed the 'time' and 'total_production' columns from the input dataset, as the 'time' column had already been parsed into distinct features and 'total_production' was the variable of interest. The resulting dataset was suitable for training and evaluating our machine learning model.

7.2 Model Selection and Training

7.2.1 Choosing model for our task

To predict the amount of energy returned to the grid, we had to select an appropriate machine learning model. Since our goal was to forecast a continuous numeric value (energy returned to the grid), a regression model was the most suitable choice. The purpose of regression models is to predict and analyze the relationship between a dependent variable and one or more independent variables.

Initially, we considered two ensemble regression models, the Gradient Boosting Regressor and the AdaBoost Regressor, for our task. Both models are ensemble learning techniques, which integrate multiple weak prediction models, typically decision trees, into a robust overall model. While Gradient Boosting works by adding weaker learners to the ensemble iteratively, with each succeeding learner rectifying the mistakes made by its predecessors, AdaBoost works in a slightly different manner. AdaBoost modifies the weights of the weak learners based on the errors they committed in the previous iteration, focusing particularly on the more challenging instances. The final model in both cases is a weighted sum of the predictions of these weak learners.

To determine which model would perform better for our task, we conducted a cross-validation, comparing their performance on our chosen metrics (R2, MAE,

and MSE). The comparison results revealed that the Gradient Boosting Regressor was superior, demonstrating a higher coefficient of determination, lower mean absolute error, and lower mean squared error. A detailed explanation of these metrics will be provided in section 7.3.1. The comparison results are depicted in the Figure 7.3.

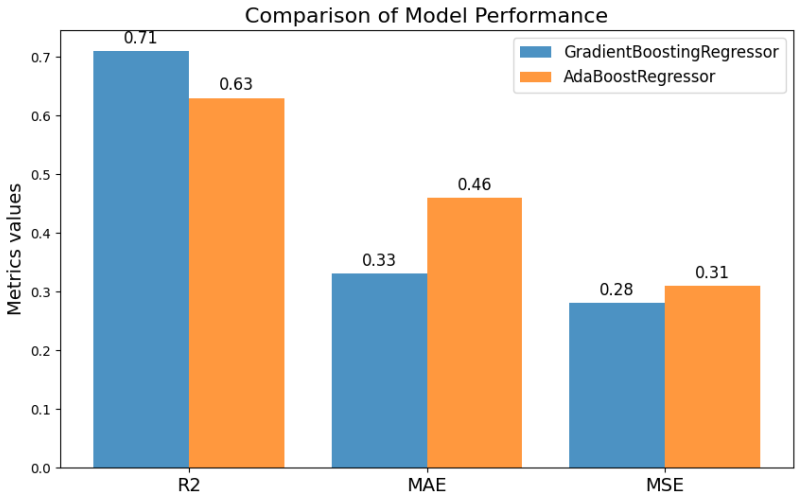

Figure 7.3: Adaboost Regressor versus GradientBoost Regressor diagram comparison

7.2.2 Hyperparameter Tunning

After selecting the Gradient Boosting Regressor as the model, hyperparameter tuning was undertaken to optimize the model's performance. The process of hyperparameter optimisation involves locating the optimal combination of hyperparameters that yields the most accurate predictions for the chosen metrics, which in our case were R2, MAE, and MSE. Grid Search was utilized for hyperparameter estimation. Grid Search is a technique for exhaustive exploration that examines every conceivable combination of hyperparameters provided in a parameter grid.

Our final model had the following parameters:

```
GradientBoostingRegressor(random_state=i, learning_rate=0.2, max_depth=20,
min_impurity_decrease=0.01, subsample=0.7,
n_estimators=100, max_features=0.8)
```

Here is an explanation of the Gradient Boosting Regressor model's parameters we choose:

Learning Rate: This is the learning rate that determines how much each tree contributes to the ultimate prediction. In general, a smaller value results in greater generalisation, but necessitates more trees in the set.

Max Depth: This parameter represents the utmost depth of the regression trees. A larger value enables the model to capture complex interactions in the data, but if it is too large, it can lead to overfitting.

Min Impurety Decrease: This is the minimal value by which the impurity must decrease for a node to be split during tree construction. This controls the growth of the tree by preventing divisions that do not sufficiently reduce impurity.

Sub Sample: This is the percentage of training samples that will be used to match each tree. A value less than 1.0 permits stochastic subsampling, which can aid in reducing the model's variance.

N Estimators: This is the total quantity of trees that will be constructed for the ensemble. The greater the number of estimators, the greater the model's ability to capture complex data patterns, but the longer the training time.

Max Features: The maximum number of features to evaluate when looking for the optimal split. A lesser value reduces tree redundancy and can enhance model performance.

7.2.3 Training our model

For the model, four months of monitored data are available and these data were retrieved by our system presented in the first part. We have chosen to retain only a portion of these four months: March, April, and the first half of May. This decision was made because we wanted to avoid seasonal behaviors. The seasonal habits of a household vary. For example, in the winter there is more heating and in the summer there may be air conditioning systems. By retaining the majority of spring data, our data accurately reflect the seasonal patterns and are not skewed by the winter months. Given that the data is collected per hour, this represents a dataset of approximately 1820 lines.

In the majority of machine learning initiatives, training and testing datasets are kept separate. However, we only had access to our real data for training purposes. We used a method known as "train-test split" to train and evaluate the efficacy of our model.

The train-test split is a method for separating the dataset into two distinct sets: one for training the model (training set) and one for evaluating its performance (testing set). This method aids in evaluating the model's ability to generalize to new, unobserved data and prevents overfitting. Typically, the training set consists of the majority of the data (e.g., 70-80%), while the testing set contains the remainder [62]. We chose a training size of 80% (1460 lines of data) and a test size of 20% (364 lines of data).

This procedure allowed us to evaluate the performance of the model and ensure that it could accurately predict future energy consumption, even when applied to new, unobserved data.

We also developed a pipeline to expedite the model training and feature selection processes. A pipeline is a method for sequentially applying a list of data transformations and a final estimator, making it simpler to manage complex workflows and ensure consistent preprocessing throughout the various modeling process stages. In our pipeline, we included a Gradient Boosting Regressor with optimized hyperparameters and a feature selection phase utilizing a f_regressor, which selects the twenty-six most crucial features as determined by our analysis.

By incorporating feature selection and the Gradient Boosting Regressor into a pipeline, we were able to train our model efficiently and generate accurate predictions for energy returned to the grid.

7.3 Model Evaluation and Performance Metrics

An integral part of the predictive modeling procedure in this study is evaluating the model's performance and accuracy with the aid of suitable evaluation metrics. The precision and dependability of the predictive models are crucial because they directly influence the efficacy of the energy management strategies that are based on these models.

In the following sections, we delve deeper into the evaluation process, discuss the metrics used to evaluate model performance, and present the results acquired using both the standalone model and the model in conjunction with the Bagging Regressor.

7.3.1 First model

Several performance metrics, including R-Squared score, Mean Squared Error, and Mean Absolute Error, were employed to assess the effectiveness of the Gradient

Boosting Regressor model we selected. The performance of our model in predicting the quantity of energy returned to the grid is evaluated from multiple angles by these metrics.

The R2 score, also referred to as the coefficient of determination, assesses the proportion of the variance in the dependent variable that can be predicted from the independent variables. It ranges from 0 to 1, with 1 indicating that the model explains all of the variation in the target variable and 0 indicating that no variation is explained [8]. Our R2 score was **0.71**, indicating that our model can account for 71% of the variance in the quantity of energy returned to the grid.

MSE measures the average squared difference between actual and predicted values. The objective is to reduce this value, as lower MSE values indicate superior performance [8]. In our case, the MSE was **0.28**, which indicates that our model's predictions are reasonably near to the actual values.

The MAE quantifies the average absolute deviation between actual and predicted values. Similar to MSE, the objective is to minimize this value. MAE has the advantage of being less sensitive to deviations than MSE because it does not square the differences [8]. In our case, the MAE was **0.33**, indicating that our model's predictions are reasonably accurate.

Overall, these performance metrics indicate that our Gradient Boosting Regressor model predicts the quantity of energy returned to the grid quite accurately. It is essential to consider all of these metrics concurrently, as they provide complementary information regarding the performance of the model. A low MSE and MAE, for instance, indicate that our model's predictions are near to the actual values, whereas a high R2 score indicates that our model can explain a substantial proportion of the variance in the target variable.

Prediction

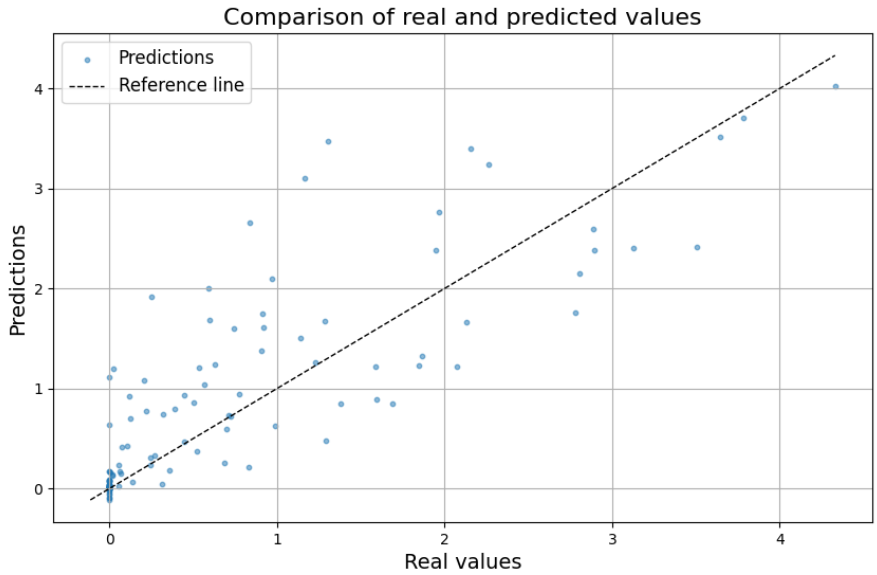

Figure 7.4: Plot of our first model prediction

As can be seen from the graph of predictions in Figure 7.4, from the middle onwards, the points are further apart and the accuracy decreases. For the three large values above 3.5 KwH, however, the predictions remain accurate.

7.3.2 BaggingRegressor

We utilized a BaggingRegressor to enhance the performance of our Gradient Boosting Regressor model. Bagging, short for bootstrap aggregating, is a technique in ensemble machine learning that increases stability and accuracy by creating multiple subsets of the original dataset, training a model on each subset, and then averaging the prediction results from each subset. This method reduces variance, prevents overfitting, and ultimately improves our model's efficacy.

By incorporating a BaggingRegressor into our primary model, we were able to attain better results. Our MSE decreased to **0.27**, our R2 score rose to **0.73**, and our MAE remained relatively modest at **0.30**. These results are superior to those obtained without the BaggingRegressor, indicating that this technique improved the predictive ability of our baseline model.

Prediction

Figure 7.5: Plot of our model prediction with BaggingRegressor

Intriguingly, when visualizing our predictions via a graph from the Figure 7.5, we observed that the predictions for mid-range values were closer to the reference line than those from the straightforward model, while the predictions for larger values were further away. However, given the rarity of superior production values, this is an acceptable trade-off. Therefore, it makes sense for us to prioritize a more accurate prediction of average production values.

These results demonstrate the efficacy of BaggingRegressor in enhancing the performance of our baseline model. It provides a more accurate and dependable estimate of the quantity of energy returned to the grid, validating our approach.

Chapter 8

Energy Optimization and Management

This chapter, delves into the essential concepts, strategies, and tools necessary to achieve optimal energy use within the context of a smart home.

The objective of this chapter is to underscore the importance of intelligent energy management and illustrate how our proposed system can contribute to its realization. As we proceed through the chapter, we discuss how the system uses real-time energy consumption data, combined with machine learning algorithms, to predict future energy needs, facilitate energy-efficient scheduling, and ultimately optimize overall energy consumption.

8.1 Reducing Standby Power Consumption

Often disregarded, standby power consumption contributes to a household's energy consumption. Devices that consume energy even when they are not in use can significantly increase a household's total energy consumption. The electric heater was identified as a device that consumed an average of **2.5** W continuously, even in inactive mode, and even more when in use.

To address this issue, we created a routine that regulates the electric heater using Home Assistant. This routine is programmed to switch on the heater at specific times when it is required and disconnect it from the power supply when it is not in use. This automation not only reduces superfluous energy consumption, but also increases the heater's longevity by reducing its operating hours.

Our analysis by code ((appendix C 12.2) revealed that the electric heater's standby power consumption wasted approximately 0.06 kWh per day, or 1.61 kWh

per month. Given that this household's electricity costs **0.1747** euros per kWh, this waste amounts to approximately 20 kWh per year, or approximately 3.5 euros per year. Despite the fact that this may not seem like a significant quantity, it is important to note that this is the energy and cost wasted by a single device.

The cumulative effect of standby power consumption across multiple household devices can result in a significant quantity of wasted energy and unnecessary costs. By identifying and addressing these sources of squandered energy, households can increase the efficiency and cost-effectiveness of their energy consumption.

The approach we took with the electric heater can be applied to other devices in the home, potentially resulting in further energy and cost savings.

8.2 Energy-Efficient Scheduling and Load Balancing

The implementation of machine learning and data-driven insights to common household activities can result in significant energy and cost savings. In our endeavor, we examined the accuracy of our predictive model with historical data thanks to the weather forecast.

Now that the predictions have been made, the next step was to record the predicted and actual energy data returned to the network by our test model in a new file and combine it with the energy consumption data monitored by our system in order to be able to simulate our method. The information was recorded as follows: time, washing, heater, dryer, total_consumption, total_return_predicted, and total_return_real. This data is therefore used to simulate the predictions made on the historical data. They are only useful as a guide to see if our model actually delivers energy savings.

8.2.1 Optimal Scheduling of Energy-Intensive Appliances

The objective of our study was to identify the time of day when the most significant energy return is anticipated. Using this information, households can strive to operate their energy-intensive appliances during these times in order to maximize energy efficiency. We acknowledge, however, that it may not always be possible to rigorously adhere to this schedule due to practical considerations.

In our case, only two of the three electrical outlets could be used for this purpose: the washing machine and the dryer. It was not possible to program the electric

heater's operation based on solar energy production because it must be operated based on actual heating requirements.

By shifting with code (Appendix C 12.1) the operation of these appliances to the predicted high-energy-return periods, we can simulate a pattern of energy consumption that reduces grid energy consumption and minimises surplus energy return to the grid.

8.2.2 Energy and Cost Savings

Noting that the washing machine and dryer do not account for the the share of the largest household's energy consumption is important. In our analysis, they accounted for 9.87% of the total energy consumption over the duration of one week.

An ADEME study [61] revealed that a washing machine consumes an average of 191 kWh annually, which is roughly equivalent to the consumption of a household in our study, whereas a drier consumes approximately 350 kWh annually, which is nearly double the consumption of the washing machine. Given the household's energy cost of 0.1747 euros per kWh, the annual cost of operating these two appliances is approximately 80 euros.

Our optimization strategy resulted in a weekly energy savings of 1.32 kWh, equivalent to a savings of 0.23 euros. This implies that the weekly cost of energy to operate these appliances decreased from 0.78 euros to 0.57 euros. This month-long energy savings corresponds to approximately 20 kWh, or approximately 3.50 euros. After optimization, the savings amount to approximately 12 kWh, or approximately 2.10 euros, bringing the monthly energy expenditure down to 1.40 euros.

While it may be tempting to extrapolate these savings over an entire year, which would imply an annual savings of approximately 20 euros or 25 percent of the total energy cost for these appliances, it is essential to remember that solar energy production fluctuates throughout the year. Consequently, the most significant reductions are likely to be realized during the warmer months. Despite this limitation, our approach demonstrates that data-driven energy scheduling has the potential to make a substantial contribution to household energy efficiency and cost reductions.

8.3 Automated Energy Recommendations

In the current digital era, the integration of data-driven insights with home automation technologies can yield significant benefits, particularly in terms of energy

conservation and cost reductions. Home Assistant, an open-source home automation platform that prioritizes local control and privacy, is one such powerful utility.

8.3.1 Home Assistant and Energy Scheduling

Home Assistant is a platform that enables users to automate the operation of household appliances based on custom settings or real-time data inputs. By utilizing the predictive capabilities of our machine learning model, households could automate their appliances to operate during periods of high solar energy production, maximizing their use of renewable energy and minimizing their reliance on utility energy.

In our case, we were able to configure Home Assistant to autonomously configure the electric heater's operation at the desired hours and disconnect it from the power supply when it was not in use. Similarly, the washing machine and dryer could be scheduled to operate at times of the day with the maximum predicted solar energy return. This energy-efficient schedule not only reduced energy waste but also improved the household's ability to utilize its own energy production.

8.3.2 Broadening the Scope of Automation

Electric radiators, washing machines, and dryers are by no means the only devices that can benefit from this strategy. Other household appliances can also be incorporated into this energy-efficient schedule in order to maximize energy savings.

Dishwashers, for instance, are an integral part of contemporary kitchens and are notorious for their high energy consumption. Scheduling dishwasher use during periods of high solar energy production could result in significant energy savings. To maximize its efficiency, this could be improved by only running the dishwasher when it is filled.

Electric vehicle charging can be a significant energy demand for households with electric vehicles. Scheduling EV charging during periods of high solar energy production can maximize energy efficiency and reduce energy costs. In addition, it contributes to a more sustainable use of renewable energy, reducing the carbon footprint of electric vehicles even further.

Based on the predictive model, charging and discharging cycles could be optimized for homes with a battery energy storage system. Charging the battery during periods of high solar energy production and discharging it during periods of low

production or high consumption would optimize system operation and maximize solar energy usage.

Smart plugs can be used to automate and control smaller appliances and devices, including coffee makers, televisions, and even individual lamps. Scheduling the operation of these devices during periods of peak solar energy production can contribute to overall energy savings.

In conclusion, the combination of machine learning and home automation technologies such as Home Assistant creates numerous opportunities for energy-efficient appliance scheduling, resulting in substantial energy and cost reductions. More than ever, households have the ability to actively manage their energy consumption, thereby contributing to sustainability and reducing energy costs.

Chapter 9

Conclusion

In this final section, we aim to reflect upon the journey we have undertaken and to consolidate the main findings and contributions of our work. We will organize our thoughts around three key aspects: "What has been achieved?", "Areas of improvements" and "Closing thoughts".

9.1 What has been achieved?

In our work, we have made significant progress in enhancing the energy efficiency of smart homes by leveraging the power of machine learning. The methodology and outcomes of our research can be categorized into two major categories: Data Acquisition and Integration, and Machine Learning and Prediction.

Data Acquisition and Integration

The first phase of our work involved the collection, processing, and integration of energy consumption data from a variety of IoT devices within a smart home environment.

Our data acquisition process was designed to collect accurate energy consumption information from various smart appliances and devices. We addressed several challenges in this phase, including ensuring data accuracy, managing the high volume of data produced by devices, and maintaining the privacy and security of the collected data.

Following the collection of data, we focused on the stage of data integration. We devised strategies to consolidate and harmonize the disparate energy data from various devices and systems, thereby producing a unified dataset. This exhaustive

data view enabled us to effectively analyze energy management and make informed decisions.

Machine Learning and Data Analysis

The second core component of our study involved using the collected and integrated energy consumption data to generate valuable insights and predictive models.

In the phase of analysis, numerous statistical and data analysis techniques were utilized to examine the integrated energy data. The objective was to identify underlying patterns and trends in the data on energy consumption. We used visualization tools to present the analysis results in a manner that was intuitive and easy to comprehend.

In the prediction phase, machine learning was utilized. We developed machine learning algorithms to forecast future energy consumption based on the patterns and relationships identified during the analysis phase. These predictions enabled homeowners to comprehend their anticipated energy consumption, enabling them to plan their energy consumption more efficiently.

The application of these predictive models to automation scenarios was also investigated. These models could be used, for instance, to automatically schedule energy-intensive appliances during predicted periods of high solar energy production. This method maximized the use of renewable energy while minimizing dependence on grid power.

Through a combination of data integration techniques and machine learning, our work offers a promising strategy for making homes more energy-efficient.

9.2 Areas of improvements

While our study primarily focused on the application of smart meter data integration and machine learning for energy consumption prediction in residential settings, the potential applications of these technologies are not limited to this context. In fact, they may be even more impactful when applied to business environments.

Due to their magnitude and scope of operations, businesses typically have a significantly larger energy consumption footprint than homes. In addition, their energy utilization patterns may be more complex, with numerous types of appliances and systems contributing to their total energy consumption. This makes the

potential for energy efficiency enhancements and cost savings setting even greater.

By implementing our methodology in a commercial setting, it would be possible to monitor and analyze energy consumption data from a wide variety of sources, from lighting and HVAC systems to manufacturing equipment and data centers. This comprehensive view of an business' energy usage may disclose new opportunities for efficiency improvements that may not be apparent in a residential context.

For instance, our machine learning models could predict periods of high and low energy consumption, allowing businesses to better schedule their operations to capitalize on off-peak energy rates. Similarly, the integration of energy consumption data with automation systems could enable dynamic adjustments of building systems to optimize energy efficiency.

Many businesses now prioritize sustainability as a central component of their corporate strategy, and energy efficiency improvement is a key component of this. By leveraging our system, businesses could not only realize cost savings but also reduce their environmental impact and demonstrate their commitment to sustainability.

While our study has demonstrated promising results in a residential context, it would be necessary to undertake a series of modifications to fully adapt it to an enterprise setting. Additionally, it would be intriguing to explore the integration of our project as a seamless extension within Home Assistant.

9.3 Closing thoughts

In retrospect, this work demonstrates the revolutionary potential of machine learning and data integration for smart home energy management. Through our research and development, we have demonstrated how these technologies can be utilized to provide real-time insights into energy consumption and to make accurate predictions that can guide more sustainable energy use.

We believe that the most important contribution of our work is its ability to influence energy consumption patterns. Our system enables homeowners to make more informed decisions and implement more energy-efficient behaviors by providing them with a clearer understanding of their energy usage patterns. Not only does this benefit have implications for individual households, but it can also contribute to broader societal energy conservation objectives. Indeed, we firmly believe that extending our methodology beyond the boundaries of individual homes

to larger establishments such as businesses or industrial environments can further amplify the impact of our work.

While we are pleased with the progress we have made in integrating machine learning into energy management, it would have been advantageous to have more information on connected devices and to incorporate additional data sources in order to facilitate widespread adoption.

In conclusion, the voyage we have undertaken has been a fascinating one, replete with enlightening discoveries, difficult obstacles, and noteworthy achievements. As we reflect on our work, we are more convinced than ever about the transformative potential of data and machine learning to promote sustainable living.

Bibliography

[1] Auth, T. (n.d.). The Benefits of Smart Meters. https://www.cpuc.ca.gov/industries-and-topics/electrical-energy/infrastructure/the-benefits-of-smart-meters.

[2] Sims, J. (2022). 5 Benefits of Using Smart Meters. Critter Guard. https://www.critterguard.org/blogs/articles/5-benefits-of-using-smart-meters.

[3] Contributor, S. N. (2012, July 19). 5 smart meter benefits. Utility Dive. https://www.utilitydive.com/news/5-smart-meter-benefits/41787/ .

[4] What are the benefits of smart meters? | YES Energy Solutions. (n.d.). https://www.yesenergysolutions.co.uk/advice/benefits-of-smart-meters.

[5] Smart Meter Benefits for Utilities. (2018, September 17). NEMA. https://www.nema.org/standards/technical/meters-and-sockets/smart-meter-benefits-for-utilities.

[6] Opinion of the European Commission pursuant to Article 20(5) of Regulation (EC) No 2019/943 on the implementation plan of Belgium, C(2020) 2654 final, 30 april 2020.

[7] Belgian electricity market : Implementation plan. https://economie.fgov.be/sites/default/files/Files/Energy/Belgian-electricity-market-Implementation-plan.pdf.

[8] Sekerogiu, B. (2022). Comparative Evaluation and Comprehensive Analysis of Machine Learning Models for Regression Problems. https://www.semanticscholar.org/paper/Comparative-Evaluation-and-Comprehensive-Analysis-Sekerogiu-Ever/b745b0262c91f33235ec9ec2db9f049c1950e08e.

[9] Energy market regulation and deployment of smart meters and flexibility – Policies - IEA. (n.d.). IEA. https://www.iea.org/policies/13735-energy-market-regulation-and-deployment-of-smart-meters-and-flexibility.

[10] Assistant, H. (n.d.). Home Assistant. Home Assistant. https://www.home-assistant.io/.

[11] Danny. (2023). Home Assistant Vs OpenHAB: What Is Better in 2023? What Smart Home. https://whatsmarthome.com/home-assistant-vs-openhab/.

[12] WunderTech. (2023, April 6). openHAB vs. Home Assistant: Comparison in 2023 - WunderTech. WunderTech. https://www.wundertech.net/openhab-vs-home-assistant/.

[13] Assistant, H. (n.d.). DSMR Slimme Meter. Home Assistant. https://www.home-assistant.io/integrations/dsmr/.

[14] Belgium - Energy. (n.d.). International Trade Administration | Trade.gov. https://www.trade.gov/country-commercial-guides/belgium-energy.

[15] Semtech, R. D. (2020). Implementing LoRa-based Solutions for Smart Metering. Industry Articles. https://www.allaboutcircuits.com/industry-articles/implemting-lora-based-solution-for-smart-metering-utilities/.

[16] Tupe, S. (2023). Mitigating Smart Meter Security Risk: A Privacy-preserving Approach. Technical Articles. https://eepower.com/technical-articles/mitigating-smart-meter-security-risk-a-privacy-preserving-approach/.

[17] Kohout, D., Lieskovan, T., Mlýnek, P. (2023). Smart Metering Cybersecurity—Requirements, Methodology, and Testing. Sensors, 23(8), 4043. https://doi.org/10.3390/s23084043 .

[18] How do you update and maintain smart meter data transmission protocols? (n.d.). https://www.linkedin.com/advice/0/how-do-you-update-maintain-smart-meter-data-transmission .

[19] IoT based Smart Energy Meter using NodeMCU ESP8266. (n.d.). https://iotdesignpro.com/projects/iot-based-smart-energy-meter-using-nodemcu-esp8266.

[20] Schoutsen, P. (2021, August 4). Energy Management in Home Assistant. Home Assistant. https://www.home-assistant.io/blog/2021/08/04/home-energy-management/.

[21] Cloudflare Tunnel · Cloudflare Zero Trust docs. (n.d.). Cloudflare Tunnel · Cloudflare Zero Trust Docs. https://developers.cloudflare.com/cloudflare-one/connections/connect-apps/.

[22] Les détails du compteur intelligent - Sibelga. (n.d.). https://www.sibelga.be/fr/raccordements-compteurs/compteurs-intelligents/les-details-du-compteur-intelligent.

[23] Les détails du compteur intelligent - Sibelga. (n.d.). https://www.sibelga.be/fr/raccordements-compteurs/compteurs-intelligents/les-details-du-compteur-intelligent.

[24] LOLIN D1 mini — WEMOS documentation. (n.d.). https://www.wemos.cc/en/latest/d1/d1_mini.html.

[25] Santos, S. (2022). ESP32 vs ESP8266 – Pros and Cons. Maker Advisor. https://makeradvisor.com/esp32-vs-esp8266/.

[26] Chaitanya. (2023). ESP32 vs ESP8266 – Which One To Choose? ElectronicsHub. https://www.electronicshub.org/esp32-vs-esp8266/.

[27] Ltd, R. P. (n.d.). Buy a Raspberry Pi 3 Model B+ – Raspberry Pi. Raspberry Pi. https://www.raspberrypi.com/products/raspberry-pi-3-model-b-plus/.

[28] Will a Raspberry Pi 3 do? (2022, December 11). Home Assistant Community. https://community.home-assistant.io/t/will-a-raspberry-pi-3-do/501333.

[29] Tapo P110 | Mini Smart Wi-Fi Socket, Energy Monitoring | Tapo. (n.d.). https://www.tapo.com/en/product/smart-plug/tapo-p110/.

[30] MQTT - The Standard for IoT Messaging. (n.d.). https://mqtt.org/.

[31] Technologies, E. (2021, December 12). MQTT Broker Server - EMQ Technologies - Medium. Medium. https://emqx.medium.com/mqtt-broker-server-9b17cb8eaccb.

[32] Technical information | Connected Digital Energy Meter. (n.d.). https://www.cdem.be/13_technical/what-information-is-provided-with-the-p1-port.

[33] Extended Multi-Utility Companion Specification for the Consumer Interface P1 V1.7.1 .

[34] Schoutsen, P. (2021, August 4). Energy Management in Home Assistant. Home Assistant. https://www.home-assistant.io/blog/2021/08/04/home-energy-management/.

[35] Assistant, H. (n.d.). Automating Home Assistant. Home Assistant. https://www.home-assistant.io/docs/automation/.

[36] Daniel-Jong. (n.d.). esp826_p1meter/telegram at master · daniel-jong/esp8266_p1meter. GitHub. https://github.com/daniel-jong/esp8266_p1meter.

[37] Valdata, G. (2023, January 5). Time Series Forecasting on Power Consumption - Towards Data Science. Medium. https://towardsdatascience.com/time-series-forecasting-on-power-consumption-273d56768b99.

[38] Time Series Analysis of Electricity Consumption Forecasting Using ARIMA Model. (2021, April 1). IEEE Conference Publication | IEEE Xplore. https://ieeexplore.ieee.org/abstract/document/9458543.

[39] Puspita, V., Ermatita. (2019). Time Series Forecasting for Electricity Consumption using Kernel Principal Component Analysis (kPCA) and Support Vector Machine (SVM). Journal of Physics, 1196, 012073. https://doi.org/10.1088/1742-6596/1196/1/012073.

[40] Bezzar, N. E., Laimeche, L., Meraoumia, A., Houam, L. (2022). Data analysis-based time series forecast for managing household electricity consumption. Demonstratio Mathematica, 55(1), 900–921. https://doi.org/10.1515/dema-2022-0176.

[41] Bourhnane, S., Abid, M., Lghoul, R., Zine-Dine, K., Elkamoun, N., Benhaddou, D. (2020). Machine learning for energy consumption prediction and scheduling in smart buildings. SN Applied Sciences, 2(2). https://doi.org/10.1007/s42452-020-2024-9.

[42] Yan, B. (n.d.). Energy Demand Forecasting using Machine Learning. https://cs109-energy.github.io/.

[43] Qiao, Q., Yunusa-Kaltungo, A., Edwards, R. (2022). Feature selection strategy for machine learning methods in building energy consumption prediction. Energy Reports, 8, 13621–13654. https://doi.org/10.1016/j.egyr.2022.10.125.

[44] Hai-xiang Z, F. MAGOULÈS (2012). Feature Selection for Predicting Building Energy Consumption Based on Statistical Learning Method. https://journals.sagepub.com/doi/pdf/10.1260/1748-3018.6.1.59.

[45] Wilfling, S. (2023). Augmenting data-driven models for energy systems through feature engineering: A Python framework for feature engineering. https://arxiv.org/pdf/2301.01720.pdf.

[46] Modeling energy consumption using machine learning. (n.d.). Frontiers. https://www.frontiersin.org/articles/10.3389/fmtec.2022.855208/full.

[47] Demand Side Management - METCO Engineering. (2021, January 17). METCO Engineering. https://www.metcoengineering.com/energy-infrastructure/demand-side-management/.

[48] Preparing Distribution Utilities for the Future - NREL. (2021). Retrieved from https://www.nrel.gov/docs/fy21osti/79375.pdf.

[49] Saini, S. (2004). Conservation v. generation. Refocus. https://doi.org/10.1016/s1471-0846(04)00146-5.

[50] Wikipedia contributors. (2023). Energy demand management. Wikipedia. https://en.wikipedia.org/wiki/Energy_demand_management.

[51] Pavlov, D. (2017). Invention and Development of the Lead–Acid Battery. Elsevier eBooks, 3–32. https://doi.org/10.1016/b978-0-444-59552-2.00001-8.

[52] Energy Storage Systems for Renewable Energy - Duke Energy Sustainable Solutions. (2023, January 17). Duke Energy Sustainable Solutions. https://sustainablesolutions.duke-energy.com/resources/energy-storage-systems-for-renewable-energy/.

[53] Declining Renewable Costs Drive Focus on Energy Storage. (n.d.). News | NREL. https://www.nrel.gov/news/features/2020/declining-renewable-costs-drive-focus-on-energy-storage.html.

[54] Battery storage: your questions answered. (n.d.). https://www.nationalgrid.com/stories/energy-explained/what-is-battery-storage.

[55] Energy storage is key to unlocking renewable power's full potential. (n.d.). World Business Council for Sustainable Development .(WBCSD). https://www.wbcsd.org/Overview/News-Insights/WBCSD-insights/Energy-storage-is-key-to-unlocking-renewable-power-s-full-potential

[56] Hutson, M. (2022, April 18). The Renewable-Energy Revolution Will Need Renewable Storage. The New Yorker. https://www.newyorker.com/magazine/2022/04/25/the-renewable-energy-revolution-will-need-renewable-storage.

[57] Segatto, M. E. V., De Oliveira Rocha, H. R., Silva, J. a. L., Paiva, M. H. M., Cruz, M. a. D. R. S. (2018). Telecommunication Technologies for Smart Grids: Total Cost Optimization. Elsevier eBooks, 451–478. https://doi.org/10.1016/b978-0-12-813185-5.00007-3.

[58] Free Open-Source Weather API | Open-Meteo.com. (n.d.). https://open-meteo.com/.

[59] Zhao, Z. (2020). Feature Selection Methods for Uplift Modeling. https://www.semanticscholar.org/paper/Feature-Selection-Methods-for-Uplift-Modeling-Zhao-Zhang/1d0bcfd230730835911adebae87b0a0153936889.

[60] Trevisan, V. (2022, March 25). Comparing Robustness of MAE, MSE and RMSE - Towards Data Science. Medium. https://towardsdatascience.com/comparing-robustness-of-mae-mse-and-rmse-6d69da870828.

[61] ADEME - Etude Usage Lavage Domestique - RAPPORT - DEF. Retrieved from https://librairie.ademe.fr/cadic/2039/ademe_-_etude-usage-lavage-domestique_-_rapport_-_def.pdf?modal=false.

[62] Tan, J. (2021, June 8). A critical look at the current train/test split in machine learning. arXiv.org. https://arxiv.org/abs/2106.04525.

Chapter 10

Appendix A

10.1 Prototype

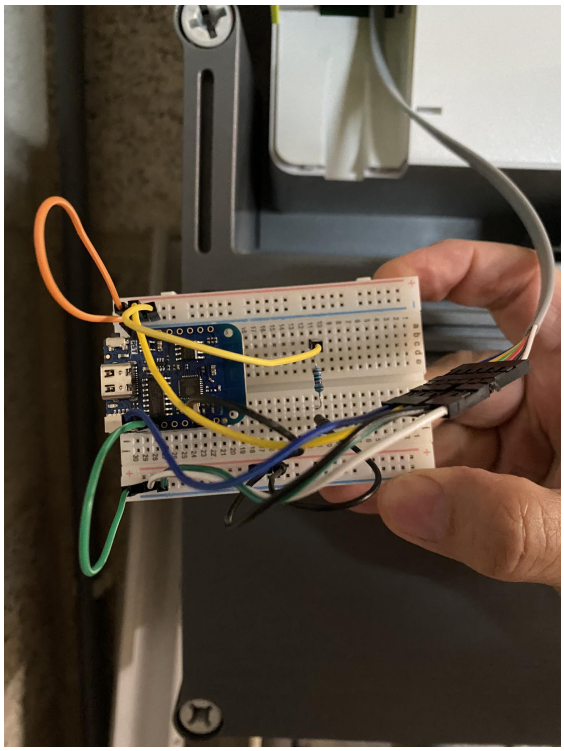

Figure 10.1: Our prototype connected to the smart meter

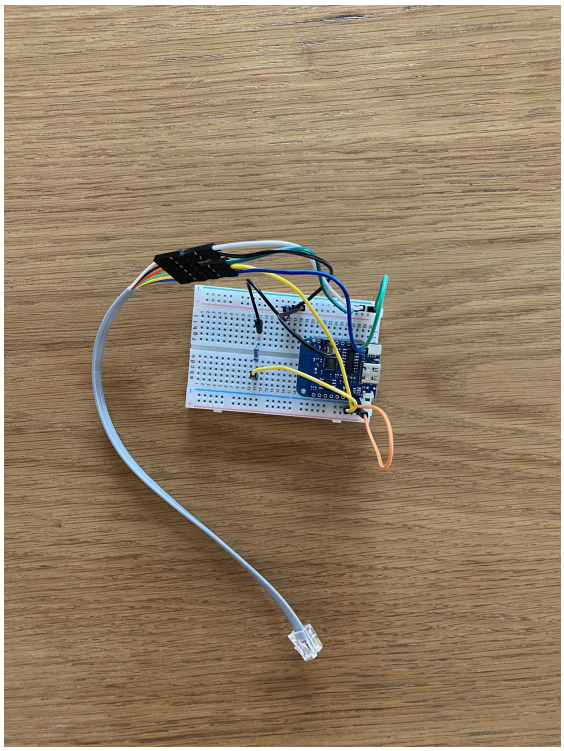

Figure 10.2: Our prototype disconnected to the smart meter

Chapter 11

Appendix B

11.1 Home Assistant Overview

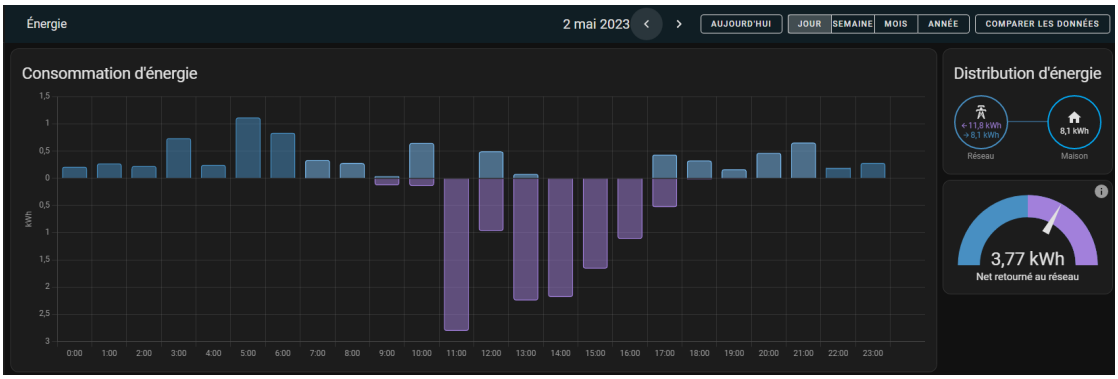

Figure 11.1: Energy Panel simplified view

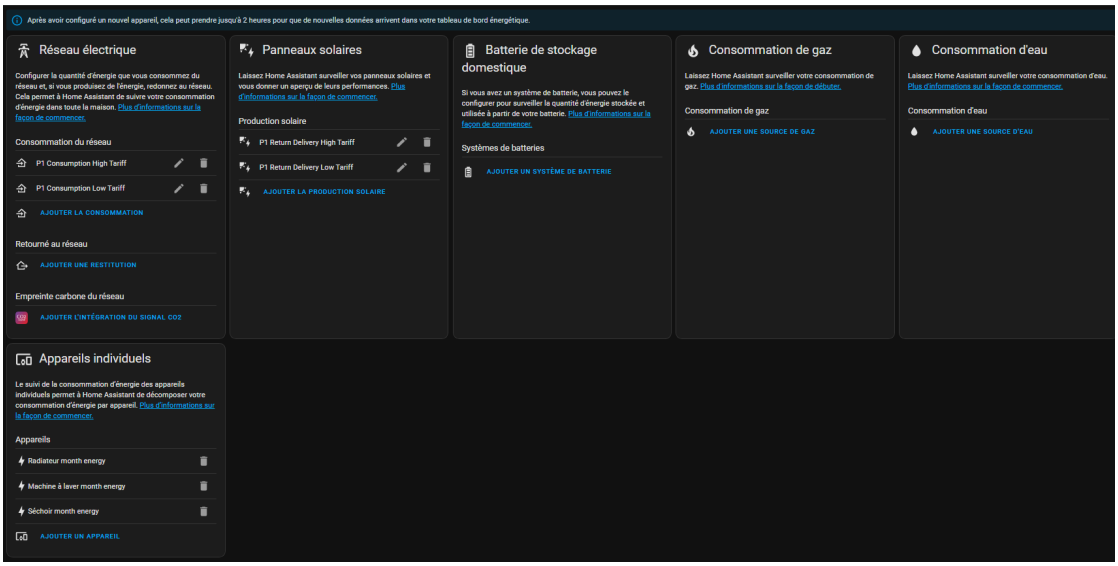

Figure 11.2: Energy Panel entities configuration

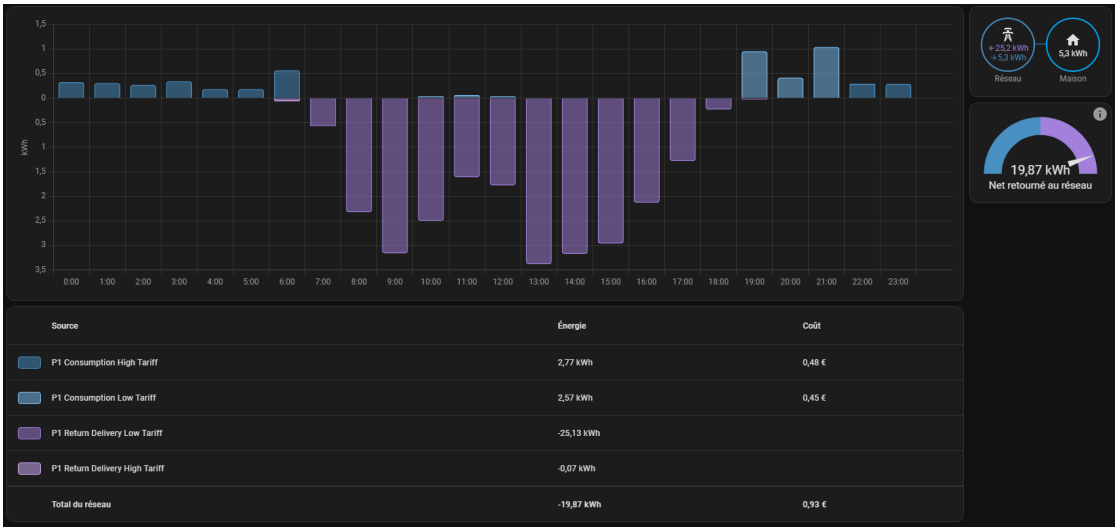

Figure 11.3: Basic Energy Overview with detailled informations

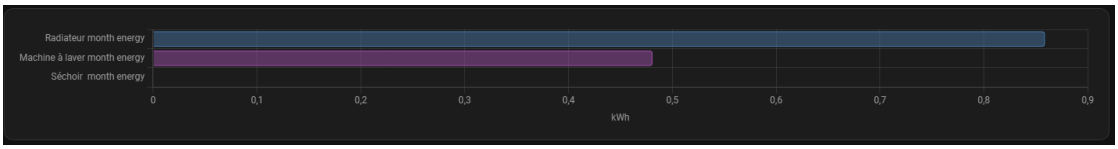

Figure 11.4: Monthly consumption of the 3 outlets

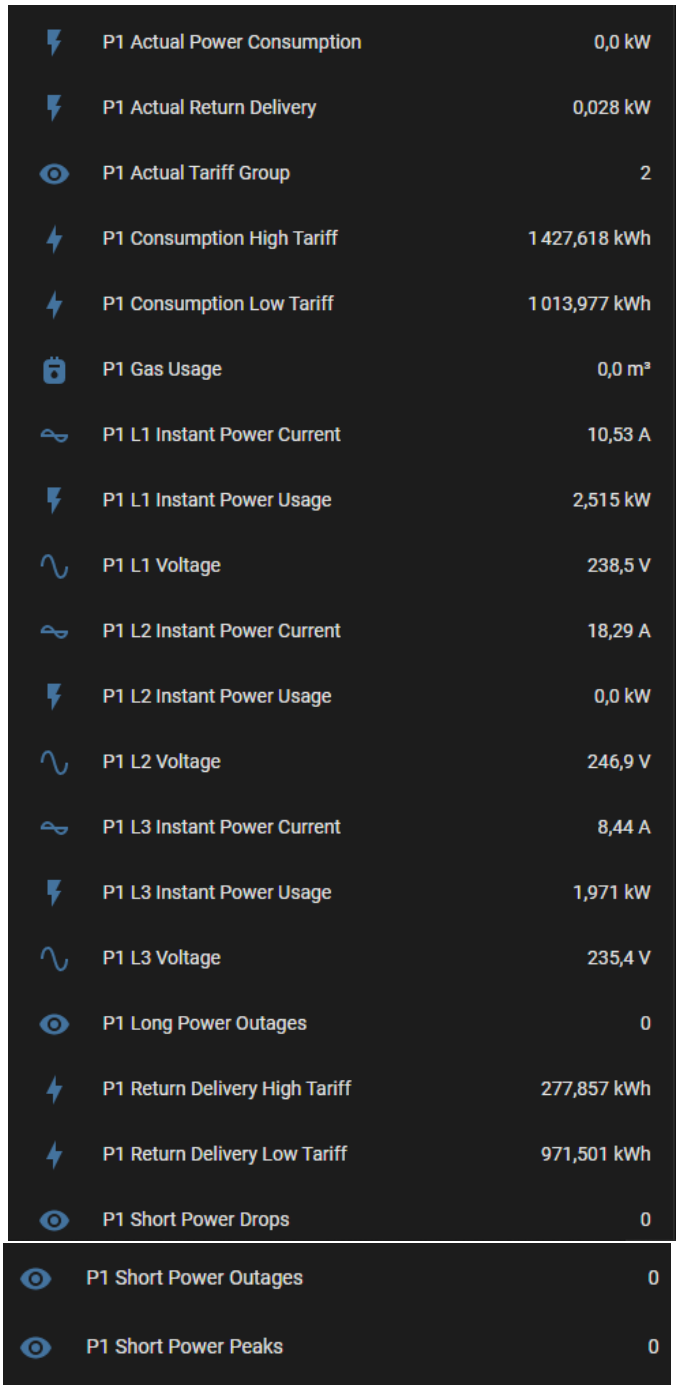

Figure 11.5: All the data captured by our system

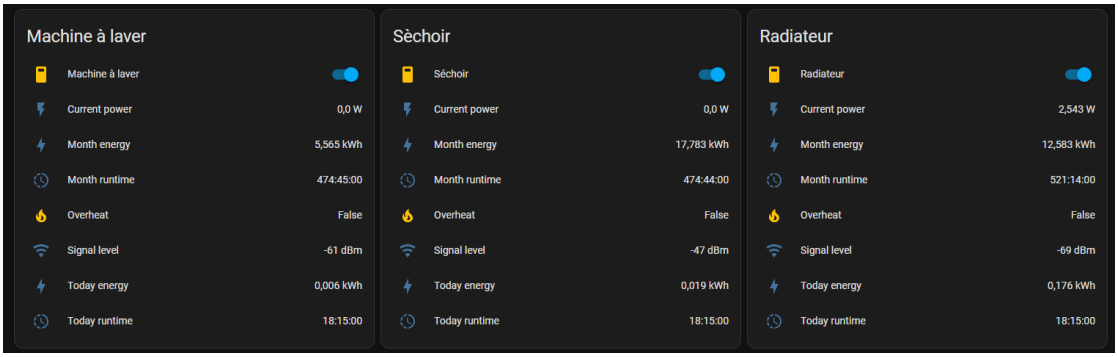

Figure 11.6: data from our connected sockets

Figure 11.7: Real Time Return and Consumption Energy

Chapter 12

Appendix C

12.1 Machine Learning code

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.metrics import mean_squared_error, r2_score
from sklearn.preprocessing import StandardScaler
from sklearn.ensemble import GradientBoostingRegressor,
    BaggingRegressor
import numpy as np

seed = 0

data = pd.read_csv("data-HA-plot.csv")
new_cols = [col.split(' (')[0] for col in data.columns]
data.columns = new_cols

X = data.drop(columns=['time', 'total_production'])
y = data['total_production']

mse = []
r2 = []
mae = []

for i in range(100):
    X_train, X_test, y_train, y_test = train_test_split(X, y,
    test_size=0.4, random_state=i, shuffle=False)

    X_test_index = X_test.index

    scaler = StandardScaler()
    X_train = scaler.fit_transform(X_train)
    X_test = scaler.transform(X_test)
```

```
29
30      reg = GradientBoostingRegressor(random_state=i, learning_rate
        =0.2, max_depth=20, min_impurity_decrease=0.01, subsample=0.7,
        n_estimators=100, max_features=0.8)
31      bag = BaggingRegressor(reg, random_state=i)
32
33      bag.fit(X_train, y_train)
34
35      y_pred = bag.predict(X_test)
36      mse.append(mean_squared_error(y_test, y_pred))
37      r2.append(r2_score(y_test, y_pred))
38      mae.append(np.mean(np.abs(y_test - y_pred)))
39
40  estimated_bcr = np.mean(mse)
41  print("MSE: %.2f" % estimated_bcr)
42
43  estimated_r2 = np.mean(r2)
44  print("R2: %.2f" % estimated_r2)
45
46  mae = np.mean(mae)
47  print("MAE: %.2f" % mae)
48
49  X_test_df = pd.DataFrame(scaler.inverse_transform(X_test), columns
        =X.columns)
50
51  X_test_df['time'] = data.loc[X_test_index, 'time']
52
53  y_test_reset = y_test.reset_index(drop=True)
54
55  X_test_df['total_production_pred'] = y_pred
56  X_test_df['total_production_real'] = y_test_reset
57
58  time_col = data.loc[X_test_index, 'time'].reset_index(drop=True)
59
60  X_test_df['time'] = time_col
61
62  data_b = pd.read_csv('data-HA-b.csv')
63
64  data_b['time'] = pd.to_datetime(data_b['time'], format="%Y-%m-%d %
        H:%M:%S")
65
66  merged_data = pd.merge(data_b, X_test_df[['time', '
        total_production_pred', 'total_production_real']], on='time',
        how='left')
67
68  merged_data = merged_data.dropna(subset=['total_production_pred'])
69
70  merged_data.drop(columns=['total_production', 'total_kwh'],
        inplace=True)
```

```
71
72 merged_data.loc[merged_data['total_production_pred'] < 0.1, '
   total_production_pred'] = 0
73
74 merged_data.to_csv('data-HA-bs.csv', index=False)
75
76 df = pd.read_csv('data-HA-bs.csv', parse_dates=['time'])
77
78 df['date'] = df['time'].dt.date
79 max_prod = df.groupby('date')['total_production_pred'].idxmax()
80
81 df['simulated_consumption'] = df['total_consumption']
82
83 df['devices_consumption'] = df['45'] + df['92']
84
85 for date, idx in max_prod.items():
86     day_indices = df[df['time'].dt.date == date].index
87
88     if idx in day_indices:
89         df.loc[day_indices, 'simulated_consumption'] -= df.loc[
   day_indices, 'devices_consumption']
90
91         df.loc[idx, 'simulated_consumption'] += df.loc[idx, '
   devices_consumption']
92
93         df.loc[idx, 'total_production_pred'] = max(0, df.loc[idx,
   'total_production_pred'] - df.loc[idx, 'devices_consumption'])
94         if df.loc[idx, 'total_production_pred'] == 0:
95             df.loc[idx, 'simulated_consumption'] += df.loc[idx, '
   devices_consumption']
96
97 print(df)
98 savings = (df['total_consumption'] - df['simulated_consumption']).
   sum()
99
100 print(df['devices_consumption'].sum())
101
102 print(f"Les economies d'energie simulees sont de {savings} kWh.")
103 print(f"Les economies d'energie simulees sont de {savings *
   0.1747} euros.")
104
105 print(f"La facture d'electricite des appareils avant la simulation
    est de {df['devices_consumption'].sum() * 0.1747} euros.")
106 print(f"La facture d'electricite des appareils apres la simulation
    est de {df['devices_consumption'].sum() * 0.1747 - (savings *
   0.1747)} euros.")
107
108 print(df['devices_consumption'].sum() / df['total_consumption'].
   sum() * 100, "% de la consommation totale est due aux appareils
```

```
    .")
```

Listing 12.1:

12.2 Heater wasted energy code

```
    import pandas as pd


data = pd.read_csv('data-HA-R.csv')

data = data[data['state'] != 'unavailable']

data['last_updated_ts'] = pd.to_datetime(data['last_updated_ts'],
    unit='s')
data['last_updated_ts'] = data['last_updated_ts'].dt.tz_localize('
    UTC').dt.tz_convert('Europe/Paris')

data['state'] = data['state'].astype(float)

data = data.sort_values(by='last_updated_ts')

data['duration'] = (data['last_updated_ts'].shift(-1) - data['
    last_updated_ts']).dt.total_seconds() / 3600

data_veille = data[data['sensor_name'] == 'sensor.
    radiateur_current_power']
data_veille = data_veille[data_veille['state'] < 3]

data_veille['wasted_energy'] = data_veille['state'] * data_veille[
    'duration']

total_wasted_energy_kwh = data_veille['wasted_energy'].sum() /
    1000

average_wasted_energy_per_day = total_wasted_energy_kwh / data['
    last_updated_ts'].dt.date.nunique()

print(f"L'energie totale gaspillee est de {total_wasted_energy_kwh
    :.2f} kWh.")
print(f"L'energie gaspillee moyenne par jour est de {
    average_wasted_energy_per_day:.2f} kWh.")
```

Listing 12.2:

UNIVERSITÉ CATHOLIQUE DE LOUVAIN
École polytechnique de Louvain
Rue Archimède, 1 bte L6.11.01, 1348 Louvain-la-Neuve, Belgique | **www.uclouvain.be/epl**