Home Energy Management Systems Regression Models

Daniel Tiepolo Kochinski

Dissertation presented to the School of Technology and Management of Bragança to obtain the Master Degree in Energias Renováveis e Eficiência Energética in a work developed during the double degree exchange program between the Instituto Politécnico de Bragança (IPB) and the Universidade Tecnológica Federal do Paraná (UTFPR).

Work supervised by:

Professor Andre Eugenio Lazzaretti

Professor Ana Isabel Pinheiro Nunes Pereira

Professor José Luís Sousa de Magalhães Lima

Bragança

2023

Home Energy Management Systems Regression Models

Daniel Tiepolo Kochinski

Dissertation presented to the School of Technology and Management of Bragança to obtain the Master Degree in Energias Renováveis e Eficiência Energética in a work developed during the double degree exchange program between the Instituto Politécnico de Bragança (IPB) and the Universidade Tecnológica Federal do Paraná (UTFPR).

Work oriented by:
Professor Andre Eugenio Lazzaretti
Professor Ana Isabel Pinheiro Nunes Pereira
Professor José Luís Sousa de Magalhães Lima

Bragança
2023

Dedication

Faisca, I miss you. I loved you from the beginning. Thank you for doing the same. Eternally. See you soon. We shall meet again on the infinite.

Acknowledgement

I thank everyone who helped me get and stay here, academically and in life. Here it is vague and stays that way because the vague allows it to be broad. If you are reading this and think you probably participated in some way, know that this thank you is for you too. Thank you, too, tropical weather. Now I understand why everyone wants you.

To the masters of knowledge, those who left us books and records, and those who, through orality and instruction, helped me get this far, my humble thanks. Regardless of any work I have done or will do, this paragraph reminds me of my perennial role as an ant on the shoulders of the giants who inhabit the universe of knowledge. To those who guided me in this work, I thank you for all your help.

From the gene to the meme, I must not have ancestry with Aristotle, but I know that I read written with the one named after him, whatever we call a legacy, something remained, that something... remained. I hope this work can contribute to knowledge, even if it's a hummingbird trying to extinguish a forest fire. It does not seem mathematically impossible to achieve this. We need enough hummingbirds. And fireproof feathers, but that's for the pigeon industry to research. Thanks to everyone who has read this far as well, and I'm sorry now, there are too many pages of this scholarly work ahead to devour.

Abstract

Regression Models have good use in the predictability of electrical systems and for Home Energy Management Systems (HEMS) buildings. This master's thesis performs simulations with data from the Silk House, a building in Bragança. The objective is to determine better parameters in building data collection to improve its efficiency.

Several Regression Models in Machine Learning (ML) are in a Python algorithm that constructs different inputs to an output. The Data Set is short, with seven scalar variables of the building's power flow over a year, measured daily. The algorithm changed the number of variables chosen in the input and ran several models, with and without Principal Component Analysis (PCA). The Coefficient of Determination (R^2) measures how well a regression model fits the data and its percentage of results withR^2 in the range $[0.75, 1]$ across all simulations.

The best results for R^2 in the range $[0.75, 1]$ found 45% without PCA and 47.14% with PCA. With just one input, all models initially found 0% R^2 in the range $[0.75, 1]$. The results of R^2 in the range $[0.75, 1]$ increased directly with more variables in the input. The variables with the best results were Photovoltaic Production (PP) and Direct Consumption (DC), being consistent with the profile of the building (office), which recommends its expansion. The variable Battery Charge (BC) never reached any R^2 in the range $[0.75, 1]$, which indicates possible suppression. It is also concluded that it is prudent to have more data and that non-linear tools are more suitable for site analysis.

Keywords: HEMS; Machine Learning; Data; Power Systems; Optimization; Regression Models; Energy; Sustainability.

Resumo

Os modelos de regressão possuem bom uso na previsibilidade de sistemas elétricos e em Home Energy Management Systems (HEMS). Esta dissertação de mestrado realiza simulações com dados da Silk House, edifício em Bragança. O objetivo é determinar melhores parâmetros na coleta de dados do edifício para melhorar sua eficiência.

Os diversos algoritmos Regression Models em Machine Learning (ML) estão escritos em Python que constrói diferentes entradas para uma saída. O Data Set é curto, com sete variáveis escalares do trânsito de potência do edifício durante um ano, medidas diariamente. O algoritmo alterou o número de variáveis escolhidas na entrada e executou diversos modelos, com e sem Principal Component Analysis (PCA). O Coeficiente de Determinação R^2 mede quão bem um modelo de regressão se ajusta aos dados e sua porcentagem de resultados com R^2 na faixa $[0,75,1]$ perante todas simulações.

Os melhores resultados para R^2 na faixa $[0,75,1]$ encontraram 45% sem PCA e 47,14% com PCA. Com apenas uma entrada, todos os modelos encontraram inicialmente 0% R^2 no intervalo $[0,75,1]$. Os resultados de R^2 no intervalo $[0,75,1]$ aumentaram diretamente com mais variaveis na entrada. As variáveis com melhores resultados foram Photovoltaic Production (PP) e Direct Consumption (DC), sendo condizentes com o perfil da edificação (escritório), o que recomenda sua expansão. A variável Battery Charge (BC) nunca atingiu nenhum R^2 no intervalo $[0.75,1]$, o que indica possível supressão. Conclui-se também que é prudente ter mais dados e que as ferramentas não lineares são mais adequadas a análise do local.

Keywords: HEMS; Machine Learning; Data; Power Systems; Optimization; Regression Models; Energy; Sustainability.

Contents

List of Tables

List of Figures

Acronyms

BC Battery Charge.

BD Battery Discharge.

DC Direct Consumption.

DER Distributed Energy Resources.

DG Distributed Generation.

HEMS Home Energy Management Systems.

IPB Instituto Politécnico de Bragança.

ML Machine Learning.

NC Network Consumption.

NI Network Injection.

PCA Principal Component Analysis.

PP Photovoltaic Production.

SHH Silk House HEMS.

TC Total Consumption.

UTFPR Universidade Tecnológica Federal do Paraná.

VPP Virtual Power Plant.

Chapter 1

Introduction

In recent decades, environmental agreements have encouraged renewable energy exploration, especially around the 2000s and since, mainly driven by the ecological importance of reducing the consumption of fossil fuels, as well as the need to avoid an energy crisis. The production of energy and our access to it expands the possibilities of other resource management and production. In contrast, energy is also the primary means for obtaining any resources and one vital resource. The intermittent nature of renewable energies increases the complexity of management when one needs to consider that their production exhibits seasonality attached to environmental conditions and is not necessarily a response to load consumption. With the advancement and cheapening of photovoltaic and battery technologies, and in the generators for hydro and wind, new opportunities have opened up for possible new technical insertions [1]. Every country has political and regulatory conditions regarding the topic of energy. Some reasons go beyond the historical construction of an energy service sector like the one we observe. It also has a fundamental short and long-term strategic role, being at the basis of access to most services in developed societies, just as its intermittent and cheap supply represents a likely future for a more developed economy.

Thinking about energy is thinking about flow, as much as a resource and its presence and use. The excellent use of energy has improved our quality of life, so we have transformed it into a service that can be sold and purchased. Over time, population growth

and economic expansion occurred, which also caused a greater need for energy and the ability to obtain it from various sources.

Below is Figure 1.1 showing the growth in energy use over time and its most common sources.

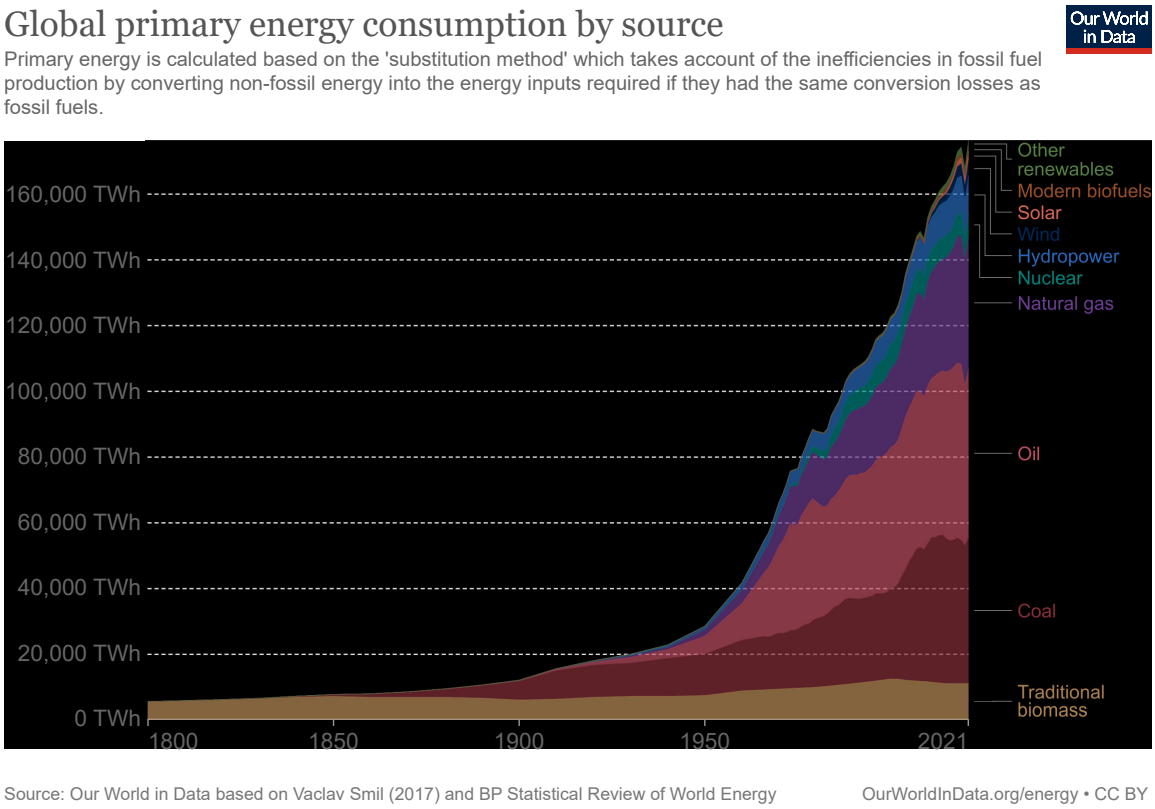

Figure 1.1: Graph of the global distribution of energy consumption by source. Adapted from [2].

Supplying energy allows for improving life perspectives for a community. They can access services and products derived from the use or existence of power. Its access becomes associated with a fundamental right for the population since the absence of it generates marginalization and inequality, as well as low quality of life, referring to life in society in contemporary times. Thirteen per cent of the world's population is impoverished or has no access to electricity [2], and taking the large pockets of the people implies that the electricity demand is under-saturated. The regions of the world that have ample access to electricity are also very industrialized, which means high consumption by individuals

with home items, but also by companies, where machines significantly impact power consumption. Increasing and diversifying electricity production are a fundamental important element, therefore. Fossil energies provide the cheapest access to energy demand for a long time. However, they are also associated with the amount of environmental damage that has been accumulating in worrying ways. Below is Figure 1.2 with energy demand scenarios considering projections for reducing fossil fuel use.

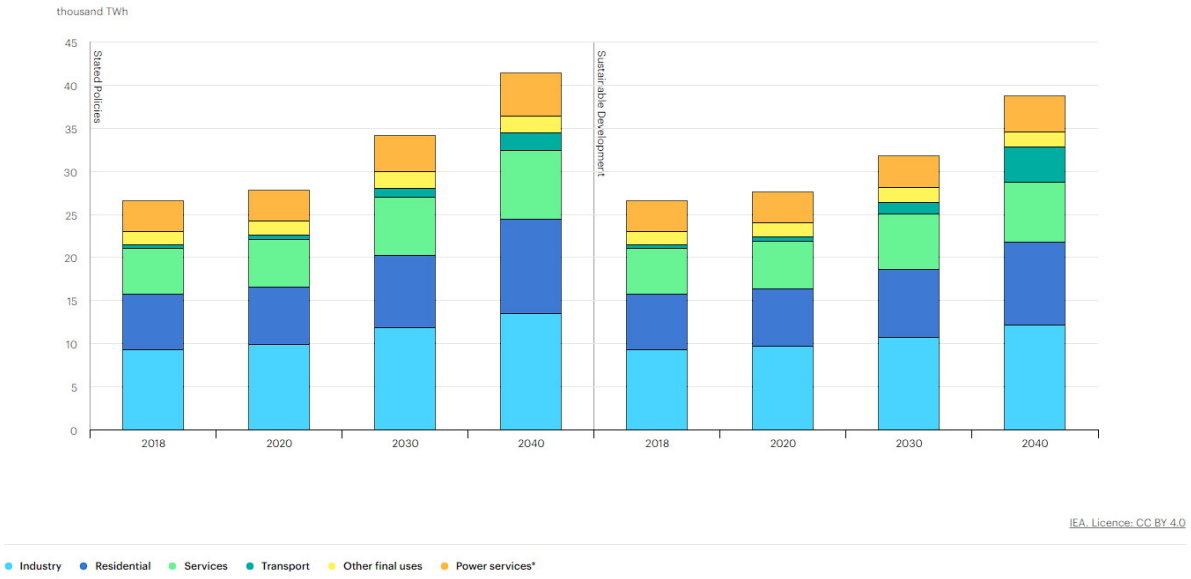

Figure 1.2: Energy demands scenarios. Adapted from [3].

The organization of electric energy production near its final consumer is known as Distributed Generation (DG), the closest form in the consumer unit. Until the beginning of the 20th century, this model was the most used, mainly in the regions that produced the most energy, which were also the ones that had industrial demands in a more distributed way, besides issues associated with energy transmission that were much more technically relevant at the time [4]. Decades ago, there was a paradigm shift mainly on the development of large generating plants and high-power transmission lines, which, together with the growth of these plants (often hydroelectric, nuclear, or fossil), determined centralized generation as the new standard, where the plants of high energy production are far from the primary consumers. This change occurred mainly from 1940 onwards and

resulted from falling technical difficulties and cost per Wh, especially in large production plants and power flow distribution projects [5]. Below is Figure 1.3 containing access to electricity distributed across the world map. Although the energy cost continues to rise as the demand increases, many regions still need access, which shows the inequality in its entrance and that the energy production system is far from near saturation.

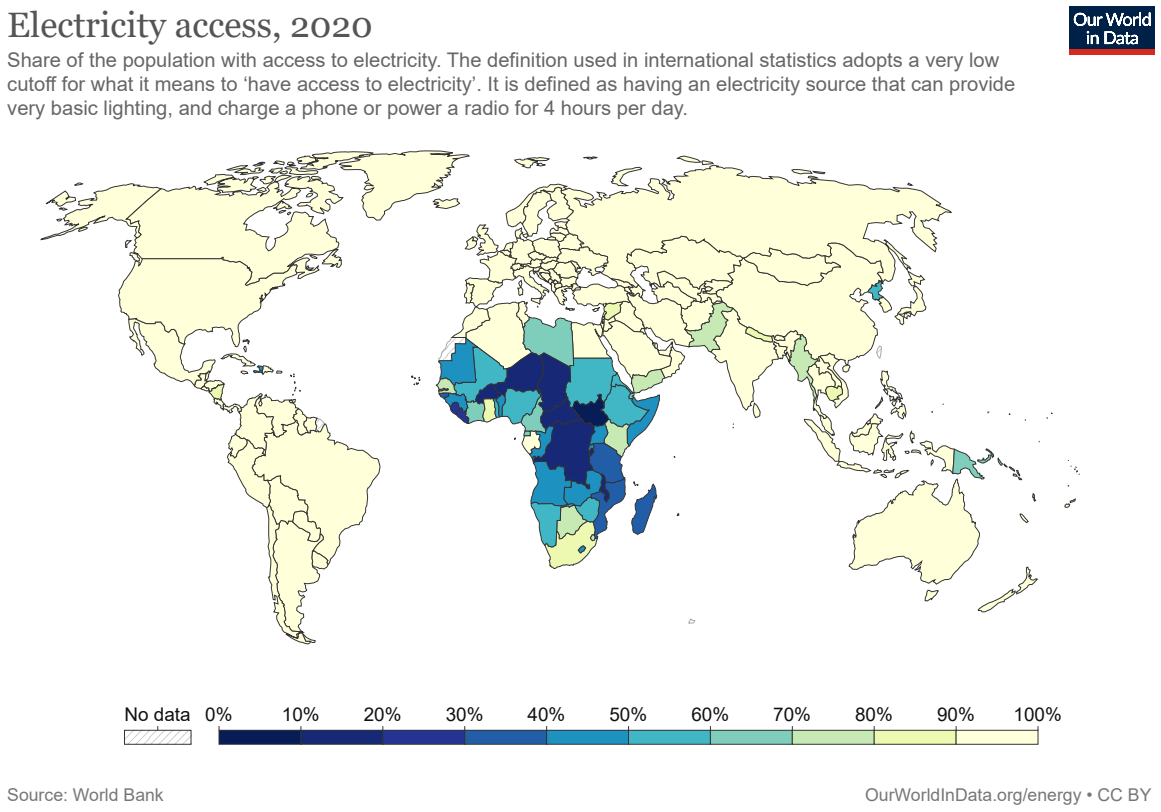

Figure 1.3: Map of world electricity access. Adapted from [2].

However, as stated, the shift from centralized production meant that distributed generation was discouraged for a long time. However, this picture is changing with technological changes, the nature of demand, and the energy market. This change and renewable energy production can positively impact (and in a way necessary due to physical and environmental demands) countries with broad access to electricity and where citizens do not have it adequately.

In this work, we have the Silk House, a Home Energy Management Systems (HEMS), a smart-grid building searching to produce its energy, becoming self-sustainable. The

electric system has internal loads in consumption and a photovoltaic with a micro-hydro power system, which, together with a battery system, searches for autonomy in self-consumption. Those pieces of equipment collect data, which will be taken for predictive analyses using Machine Learning (ML) for possible better optimization of energy uses.

1.1 Motivation

In the last decades, electricity production has gotten more diversified by the commercial introduction of solar energy production and the expansion of other categories, like wind generation [2]. The electricity demand, as the energy demand, continuously rises over time, and in the last decades, was raised a lot, and that happened in a limited way, considering the market gap that exists, given that there is in the world a vast amount of population which still does not access electricity [2]. Electricity is a type of service linked with most human activities. It will be in diverse sectors, have variable demands, and be a sensitive resource, creating the need for prediction for its production. In the last decade, other great necessities have also aggravated and changed how we look at the energy and electricity problem, mainly the climate needings and geopolitical issues. Below is Figure 1.4 with the world electricity distribution.

It is necessary to diversify the energy sources to meet the quantity of energy we will need in the following decades. By doing that inside the regulation of climate agreements, the renewable matrix needs to be expanded, as the nuclear. Nuclear is a type of production where it is possible to control the quantity of energy produced, but there is some complexity for a fast implementation for a new power plant, primarily because of its political stigma [6]. When working with DG, a Smart micro-grid, Home Energy Management Systems (HEMS) or Virtual Power Plant (VPP) is one of the most common forms of finding projects that perform its management in the bibliography. However, in the recent bibliography, it is with renewable energies. The present work uses data from Silk House, a museum dedicated to disseminating science, with an office inside. The Foundation for Science and Technology of Portugal funded it under the Silk-House Project [7]. The goal

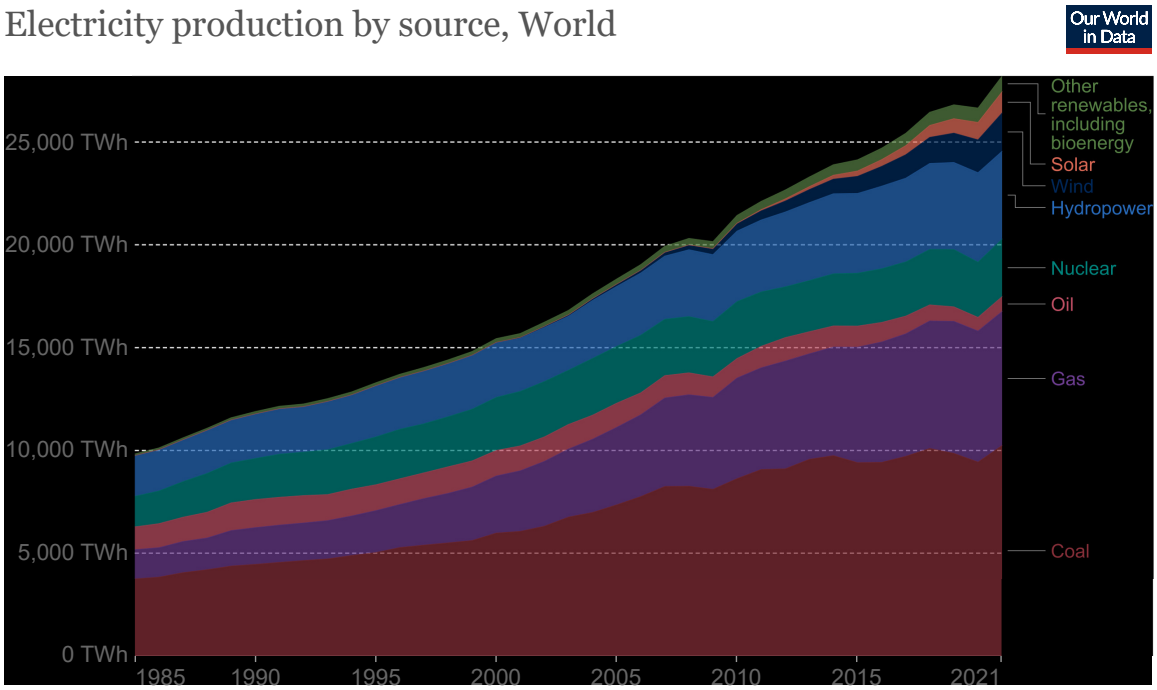

Figure 1.4: Electricity production by source type, in 1985-2021. Adapted from [2].

is to transform the House of Silk into a self-sustainable museum, contributing to disseminating renewable sources and new technologies for future buildings in smart cities. Much of what this work with HEMS utilizes is akin to the processes undertaken in VPP. However, in contrast, we can see that VPP is a similar, more significant expression of a smart microgrid when we think of managing data as intelligent microgrids.

The energy production processes involve power flows and data flows, where data here refers to those collected during the system's operation. During the production and consumption of energy, it is possible to collect data about the functioning and dynamics of the system over time. With this data collected and treated, it is possible to carry out procedures such as predictive analysis. This work's scope focuses on predictive analysis based on ML algorithms and the data set provided by a micro power plant containing solar and hydraulic energy, plus a set of batteries for storage and an auto-consumption that is internal to the generation building. Especially with renewable technologies, there

is room to exploit micro-generation with relative ease acquired in the last decade.

Below is Figure 1.5, which contains a map of the energy a person uses worldwide. That means that a person consumes this energy value in [kWh], and this quantity also needs to be generated.

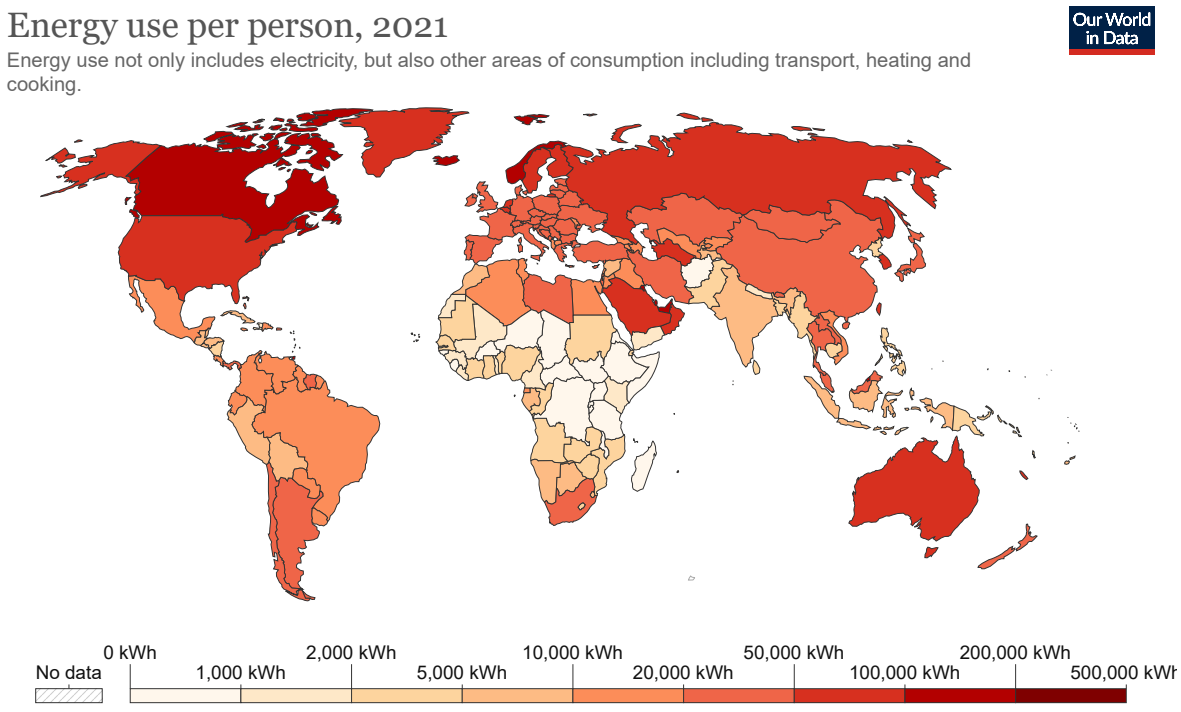

Figure 1.5: Map of the world of the energy use per person. Adapted from [2].

1.2 Background

The Silk House is a building in Bragança that has a self-sustainable system of internal energy generation for internal consumption that is configured as an intelligent microgrid. The concept of micro-grids and smart micro-grids has several descriptions in the literature, which can confuse [8]. However, the organization of the predictability problem here closely resembles the structure found in other classifications, such as Virtual Power Plant (VPP) [9]. It is a renovated and re-adapted house from a place historically associated with silk

production in the region. Today, there is an office and a demonstration space for tourism and teaching. The leading consuming sources of the place are those of office functioning (such as computers and devices), the building itself (such as structures and lighting), and heating within the temperature control. The generating sources are solar panels and a mini hydraulic generator installed in the region. Along with the controlling elements, a system of 24 batteries stabilizes and organizes the flow of energy generated and consumed. It is on-grid and can consume or inject power into the electrical grid [8].

The system collects data on consumption and generation separately and presents the information by day and month within the data set. Within the context of Energy Management and Smart micro-grids, this work seeks to treat and manipulate data to produce predictive mechanisms of consumption and production behaviour with them, assuming that the proposed result has a relative consistency possible within the limits of method and result predictability through the use of ML measuring the differences between the techniques and other performance metrics.

Below is Figure 1.6, representing the Silk House, taken from the building's website. The building is next to a historic part of the city of Bragança and has a hydraulic entrance and solar panels, as shown in the photo.

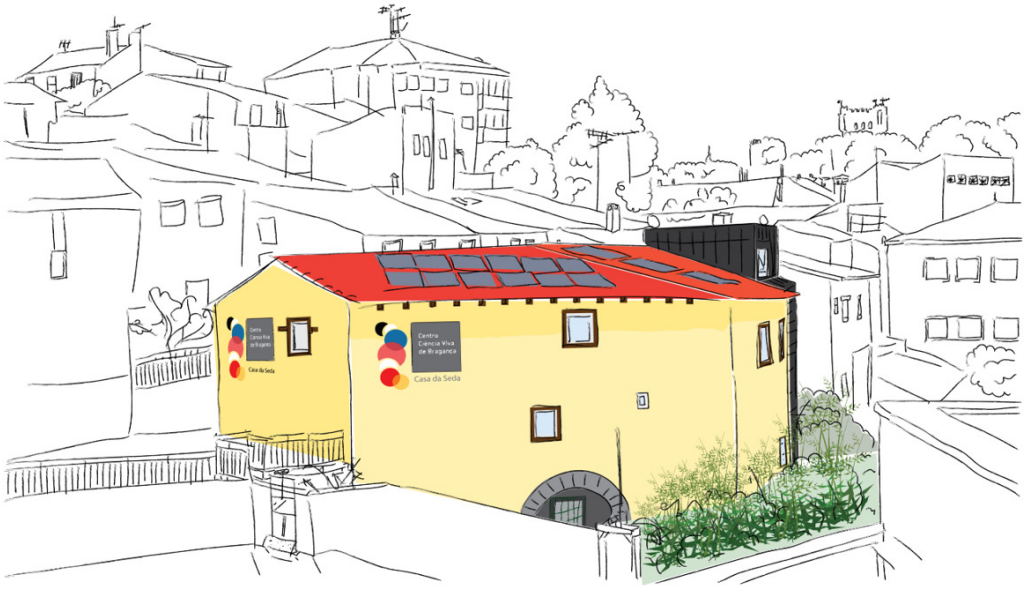

Figure 1.6: Representation of the house demonstrating the system. Adapted from [10].

1.3 Objectives

This thesis proposes, using ML tools and data sets from the production and consumption of electricity that occurs in the Silk House, where the local micro-generation has data collection, to perform a report of predictive analyses (Regression Models) using different models from ML tools and compare them.

It seeks to find, through data analysis, possibilities to make the building more efficient. This problem in the electrical part of smart microgrids (HEMS) is very close to the predictive analysis in VPP. Predictive analysis in VPP is currently a very active field using ML, and the proximity of the problem to the smart microgrid allows for using its bibliography. The main object of study and work of the thesis is collected data referring to the physical and electrical operation of the site. In addition, the practical application and discussion of technologies widely exposed in the market, for example, in the reference [11].

The research involves the characterization of the data and its treatment and execution in analytical tools already widely used for the nature of this type of problem. While producing the results, we propose comparing the differences in analysis generated by various potential techniques for addressing the problem, providing specific comments on the study, and suggesting possible improvements.

This work uses a data set of a building that seeks to be autonomous in energy, aiming to produce a sufficient amount for its operation (internal consumption), using photovoltaic energy with solar panels and hydroelectric power. These productions respond to an offer of insolation and hydraulic flow that can only sometimes coincide with a value higher than the current consumption. In addition, the building is on the power grid, and a battery system is in place to absorb any surplus production. The network fulfils a possible supplementary role for the power flow. In the environment, internal consumption involves different loads, such as computational loads and others (which have inductive factors), and resistive loads for heating. The need for production seeks to overcome consumption and keep the system sustainable. This work searches possibilities for optimization to make

the building more efficient.

As the objective of the building is to be a HEMS, the data chosen to try to predict will be the data of the photovoltaic power generation. In contrast, photovoltaic and hydraulic power is present in the Silk House HEMS (SHH).

The analysis involved various regression models with specific criteria for good results, including Linear Regression, Elastic Net, SGD Regressor, and more. Results were compared using R^2 values.

The models were executed using K-folds, then one situation with PCA and the other without. The results were presented through charts, showing model performance by variables and entries.

The analysis included system performance metrics categorized as $'Invalid'$ (model hasn't found result), $[0, 0.39]$, $[0.39, 0.63]$, $[0.63, 0.75]$, and $[0.75, 1]$. Threshold values were defined for order each category, and the data were organized using Python code. The most important results are on $[0.75, 1]$.

1.4 Document Structure

Apart from this introduction, the dissertation contains four more chapters. Chapter 1 here is "Introduction". Chapter 2 is called "Theoretical Background" which brings information related to essential knowledge for the panorama of the problem in the electrical energy and ML tools. Chapter 3 is called "The Data Set" and contains the data characterization with brief commentaries about what is essential for their use. Chapter 4 is called "Results and Discussion", showing and discussing the results. Chapter 5 is the "Conclusion and further work", which rounds out the master thesis results and what can be future works.

Chapter 2

Theoretical Background and State of the Art

2.1 Electrical Power and Energy

Electrical power finds application in various functions, serving electric loads and assisting in various other operations, including heating. There are many ways to heat an environment or a specific structure, where burning fossil fuels is one of the most common sources. For example, fossil fuels are also used to generate electric power in a power plant. In the electrical energy management literature, many solutions were made for places where heating and fossil fuel generators, such as diesel and gas, are fundamental parts of the analysis, as they are works that depend on these items as component analysis [9].

For energy generation that feeds the standard electrical grid, access to combustion generators, mostly diesel or gas, plays a vital role in controlling the specific demand related to the seasonality of other generating sources, such as renewable ones. Many significant electrical parameters, such as frequency and power flow control, require generation and control demands, such as the presence of electric and non-electrical motors and generators, to keep the essential grid stable and functioning correctly. These are brief comments on large power flows. However, it also needs to be pointed out that for individual energy

generation, considering this for the majority analysis of this present work, energy management also considers factors associated with electrical energy, such as heating or lower prices compared to a standard bill. Electric heating systems have a consumption close to resistive when it is simple heating or inductive when heating involves coils, as in inductive ovens, for example. These factors change the point consumption and the system's power factor with the loads acting, with the loads on or on standby. The individual electrical systems, such as a building, are mainly on the grid. They depend on contracts in which tariff issues and internal equipment maintenance receive consideration when accessing the network. In individual off-grid systems, that is, that do not involve connection to an external utility and therefore the absence of these contracts, the concern will be associated with the continuous supply of voltage, power, and frequency classes with stability during the necessary periods. Off-grid systems are rare when considering urban contexts [12]. Many auxiliary systems seek to maintain sufficient and convenient energy delivery for the desired demands in both off-grid and multiple on-grid systems. Using batteries or additional generators is expected when the demand power flow exceeds the required generated power flow [12]. In power plants that generate their energy or seek to maximize a parameter (such as profit), it is common to use these devices in parallel to meet greater demands at peak times.

When a load switches on, the inrush current must surpass what the system can deliver at the requested time. For those loads that do not have a starting system or are in a system that has a delay in the power supply, it is common for voltage drops to refer to the standard voltage supplied due to the current requirement. This type of problem can lead to the destabilization of other loads or components of the network, which makes the control and maintenance of the voltage an objective as well, as the current supply to the loads needs to be done correctly.

In recent academic literature, productions referring to systems that work with Distributed Energy Resources (DER) are divided in many ways, as there are different problems to analyze and solve. For example, some systems work for small and high-power flows [9]. In addition, there are recent academic productions that are concerned with operating

costs, whether for profit (sale) or autonomy (implementation capacity), management of energy, cost, implementation, seasonality, and micro-controller aspects, among other [9]. Other authors study systems electronically, detect errors and problems in existing or designed systems, use renewables as a priority, use different types of energy generation or energy flow for specific functions, presence or absence of any fossil fuel or matrix, energy storage, power factor correction or standard voltage level correction, general applications, domestic applications, and another infinity of applications found and associated in the bibliography, mainly because after this long list of items, there are a large number of recent articles that work with combinations of these concepts for a specific purpose [9]. Python and MATLAB have a rich set of out-of-the-box and optimized tools for these applications, which makes it convenient for developing ML applications.

Within ML, one of the necessary foundations is the data set. This data needs to be treated and used with some tool to fulfil some purpose.

Therefore, in the present bibliography [9], those focusing on production, local conditions, network stability, and concerns with heating and the maintenance of this microgeneration are prioritized, taking it as a smart power grid. In contrast, the modelling is close enough to other homes/building energy management systems, such as Home Energy Management Systems (HEMS).

Therefore, a vast range of forms currently associated with energy management is a topic of great importance for several reasons. In other words, as there are interests of public, private, and individual nature, which involve control and determination of technical, financial, and security parameters, within each mentioned item, there are even more numerous bifurcations and possible associations. That implies a different range of different publications, which relate (and often complement each other) with the terms mentioned above, mainly for the management of DER and Energy Management Systems (EMS) [9].

Figure 2.1 below contains the recent distribution of academic production using ML for predictability in energy generation and management, with various countries being input elements and different research focuses on output. As seen in the figure and the article,

there are differences in what is being researched depending on the region. For example, the authors have found that China is part of the world where this topic is more active[9]. In the Sankey Chart below, we can see a mapping of the countries by how much research has been done into the categories of interest by the geographical distribution on which study topics, with China in red. The paths on the Sankey Chart indicate the academic production of the countries, and the width indicates the production volume.

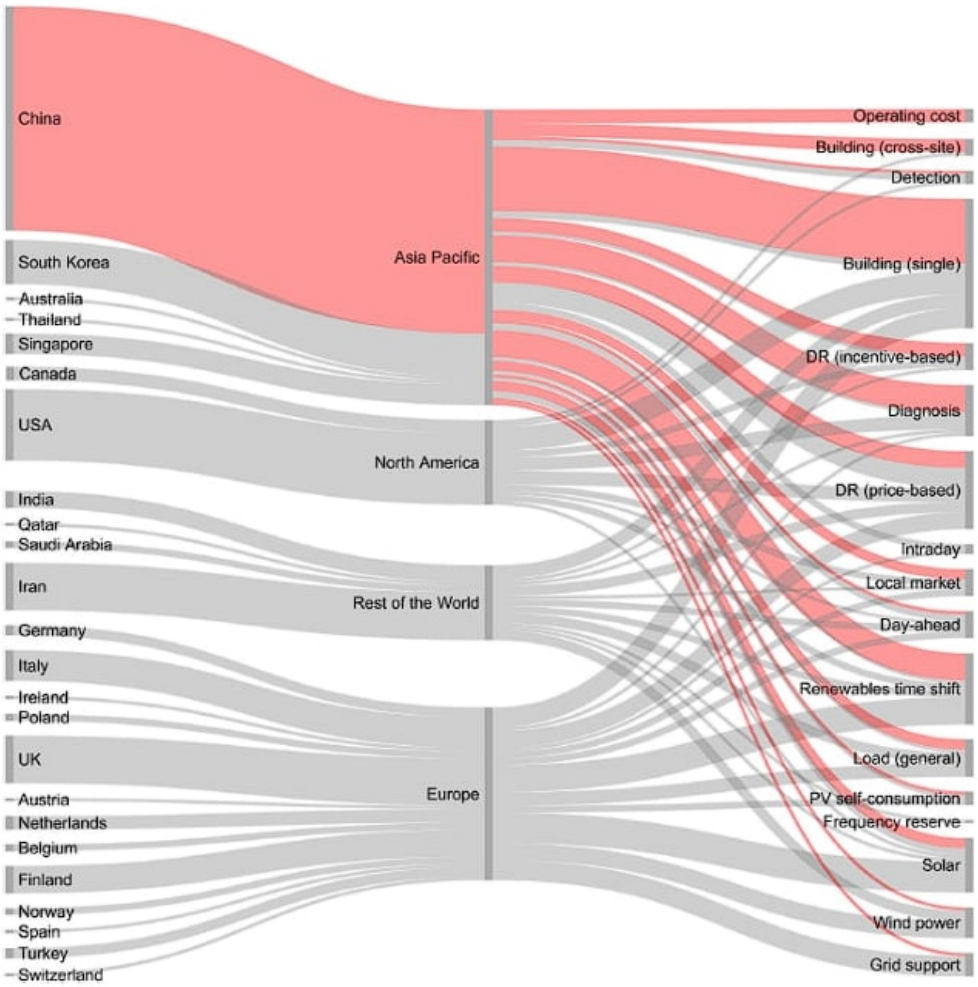

Figure 2.1: Sankey visual mapping of countries and their categories of interest. Adapted from [9].

2.1.1 Power Management

Energy management is a necessity for every electrical and electronic system. However, extensive material in the literature usually separates these two areas [13]. One of the reasons is the power level, which is generally quite different. Another is the enormous diversification of both regions, especially the electronic part, which depends greatly on using the electrical signal as its primary operation. Another distinction lies in the type of electrical supply, where direct current systems receive different treatment compared to alternating current systems [14].

Furthermore, the literature that analyzes and seeks how to build energy management shows many divisions. In the bibliography, hierarchical divisions often occur in the historical analysis of items related to energy management. The system we seek to analyze is associated with HEMS and energy management systems, which have very different characteristics. A massive number of determinants govern the nature of demands regarding the use and management of energy.

These stratifications are observable within the most recent energy management productions. Energy management in contemporary times appears commercially and in academic research as significantly associated with a high degree of sensorization, predictive analysis, processing, or a combination.

The existence of both types of work as reference material for the nature of the desired analysis is highly convenient. This convenience arises due to the similarities between works in HEMS and VPP, making it common to observe projects, articles, and in-depth analyses related to both. Moreover, specific projects consider VPP as a potential expansion of HEMS, as noted by [9]. This presence of both types of work as reference material greatly facilitates the desired analysis.

Below is Figure 2.2, which organizes the most common similarities, differences, and relationships between these two types of systems. As consumption-generation systems respond to mathematized criteria, they have a convenient analytical similarity. The figure

shows the VPP and HEMS, where VPP is a broader power plant system type, typically for power production and huge loads, and HEMS more focused on simple loads and lower power production and management.

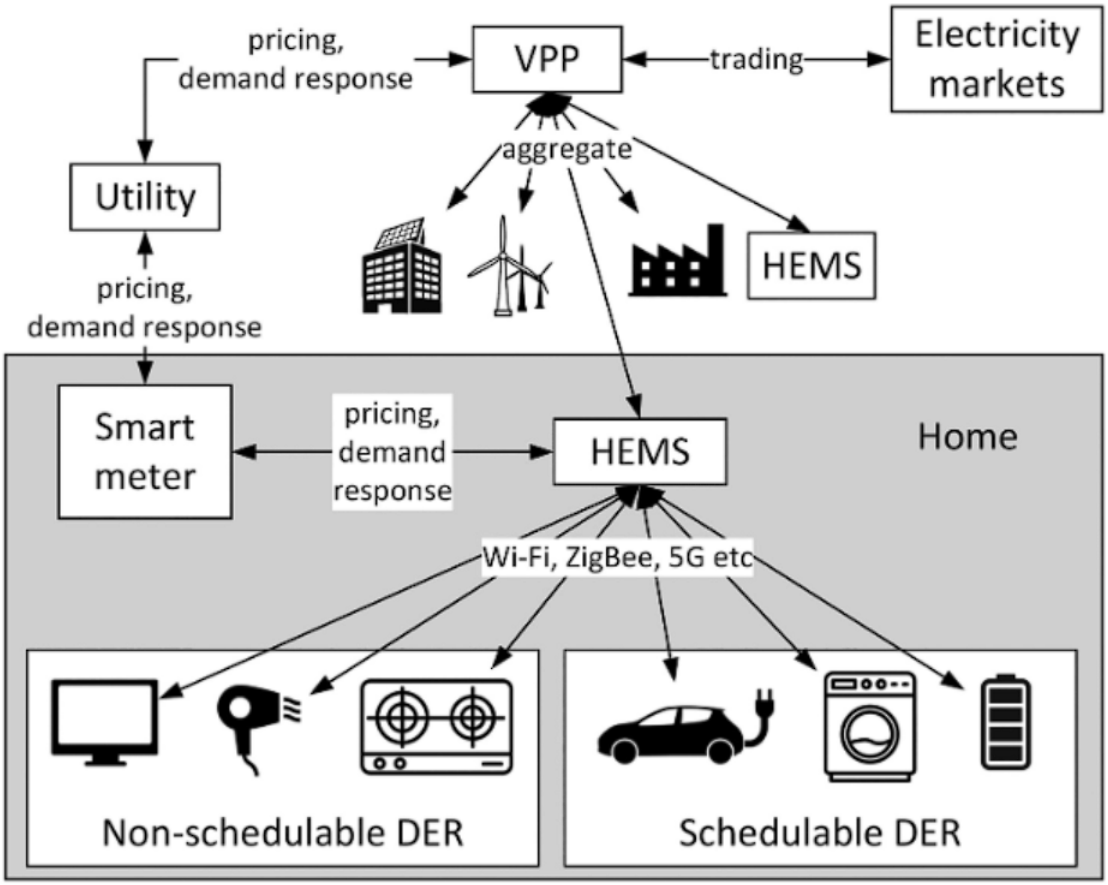

Figure 2.2: Comparative image between the similarities and differences between Virtual Power Plant (VPP) and Home Energy Management Systems (HEMS). Adapted from [9].

As mentioned above, the context of energy management has always involved the significant participation of individual interests (population), public (state), and private (companies). Therefore, [9] conducted a taxonomic analysis on the primary divisions of presently published research in energy management using ML, with a specific focus on HEMS, which are systems that seek to simulate conditions of an environment that has generation and consumption, generally involving ample use of renewable energies, considering economic and market conditions.

Even before the use of ML, and therefore, before the computational use of techniques that will require a high amount of data and preferably good data handling, power management already involved a lot of sensorization and processing to maintain stability, security, a guarantee of supply and technical and commercial viability. One of the main reasons is the need for data security, validation, and error handling. As processors have error correction implemented, it is customary to ignore in code – or to assume sufficiency, either by internal correction mechanisms or redundancy – problems associated with errors in the use and manipulation of data, such that concerns with the treatment of data refer to a loss of packages, synchronization delays (jitter), and incorrect data measure (collection error). Mechanisms of predictability analysis receive a large volume of data in advance, which implies that there will be systems concerned with acting, with an already determined configuration of possibilities, at the exact moment that it receives data, being a design responsive to the moment, and others that are concerned with analyzing it beforehand and then issuing a later report, not having a momentary action. It is rare to find systems that operate actuators without feedback. It is usually rare because they tend to have low performance and marginal differences in cost compared to others that contain feedback.

Works that aim to enhance the optimization of a particular energy system rely not only on the sensing and computational components but also on the perception and utilization of energy. Such an approach will lead to multiple variations in the representation of energy resources.

Distributed Energy Resources (DER) are versatile energy solutions that extend beyond generation to include controllable load and energy storage capabilities. These decentralized technologies, such as solar panels, wind turbines, and batteries, produce electricity and manage and store it efficiently. This integrated approach enhances grid stability, supports demand management, and contributes to a more resilient and flexible energy ecosystem. This is relevant to any HEMS.

Below is Figure 2.3, which frequently illustrates the organization of DER in a recent bibliography.

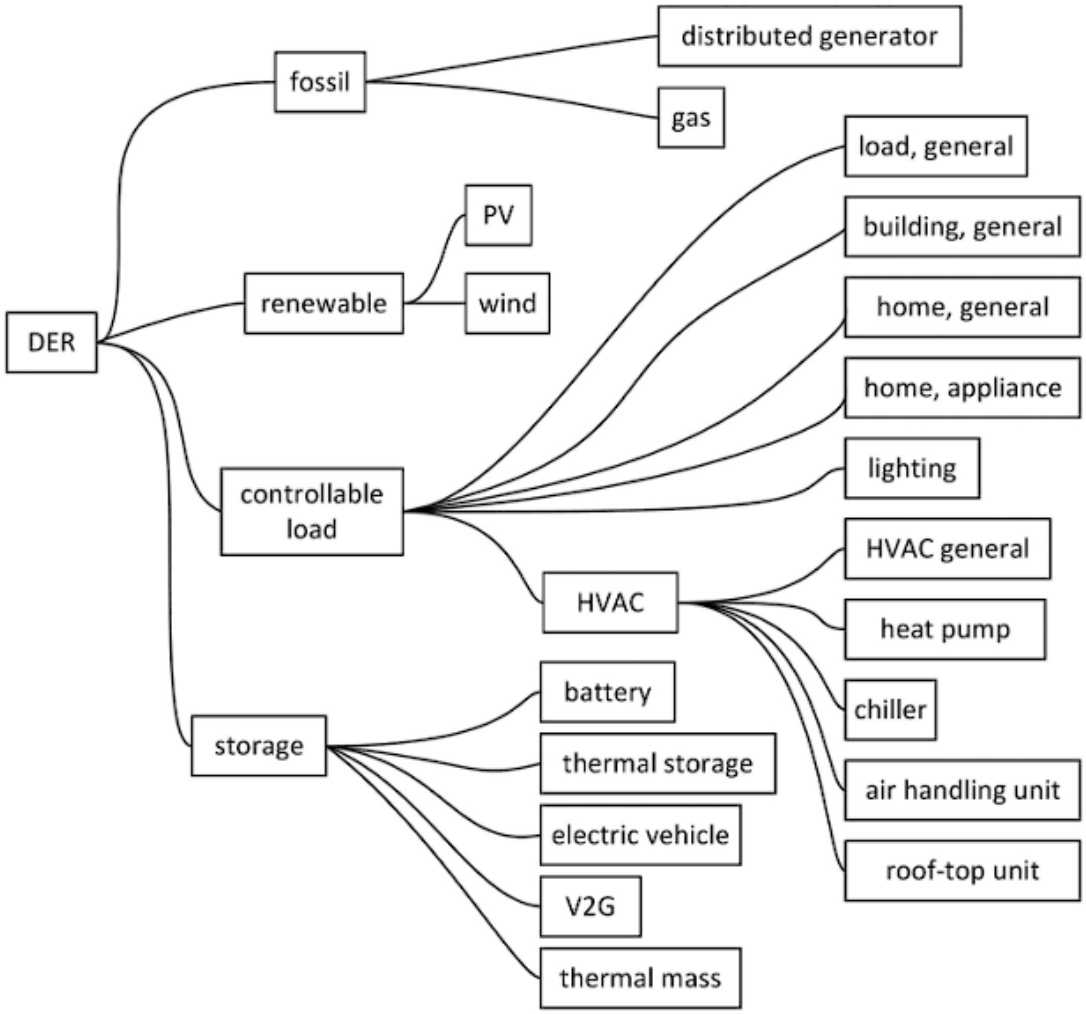

Figure 2.3: A sequential organization often found in the Distributed Energy Resources (DER) bibliography. Adapted from [9].

With several advances in power systems, micro-controlled systems, electronics, programming, and other related areas, electrical engineering, energy management, and integration between electrical systems and distributed generation, it has received many advances in new facilities for its operations and implementations. Better strategic planning boosted a unification of this knowledge applied to energy management, where HEMS have relevant participation in the electricity market, and their articles [15]. Several articles compile several related subjects about the area, where many seek to find several mathematical and computational models that present better performances in the face of

what already exists (or does not exist) published and already being used. As mentioned, regulatory and system construction elements drastically impact this area's planning. Although the laws of physics and their application in engineering tend to be universal, the technical reality refers more to human operationalization, understanding the regional and temporal context in the face of demand and what it wants to offer or operationalize. In this context of electricity as a service, some articles study financial aspects before the operation of HEMS [15], and possible internet frameworks for the interaction of HEMS [11]. Papers that focus on observing the uncertainties and potential solutions in the operations of HEMS [16] analyze how HEMS can contribute to a sustainable energy future for urban areas [13] and observe similarities between HEMS and collaborative systems [17], among many other types of studies [9], where in this review will focus on these first five studies as good examples of bibliography.

2.1.2 HEMS and the electricity markets

The search for an optimal solution in mathematical models for HEMS solutions has become more complex over time due to operational constraints, such that more advanced optimization techniques have become necessary. The models and techniques for solving mathematical problems must communicate with different types of applicability and markets for real cases. The research and use of these models led to an optimization and an increasing need for customization of each model to get used to certain realities [18]. Within models that seek to allow the highest profit, HEMS are usually optimized because of the local economic and temporal reality and legal and operational issues. That also implies differences for each regulatory environment. However, the HEMS personalization is essential. It also has different active participations for the electric system and the owners of the HEMS, who generally pocket the profit, if any.

Many industrial processes require good energy management to present functionality and adequate profit, which implies that the facilities in which these processes take place

need good governance and often energy generation to keep the proposal of what takes place there conformed adequately. The article analyzed here reviews something commonly found in the bibliography [9] in a specific or more generic context.

Below, Figure 2.4 illustrates several essential relationships for this HEMS projects.

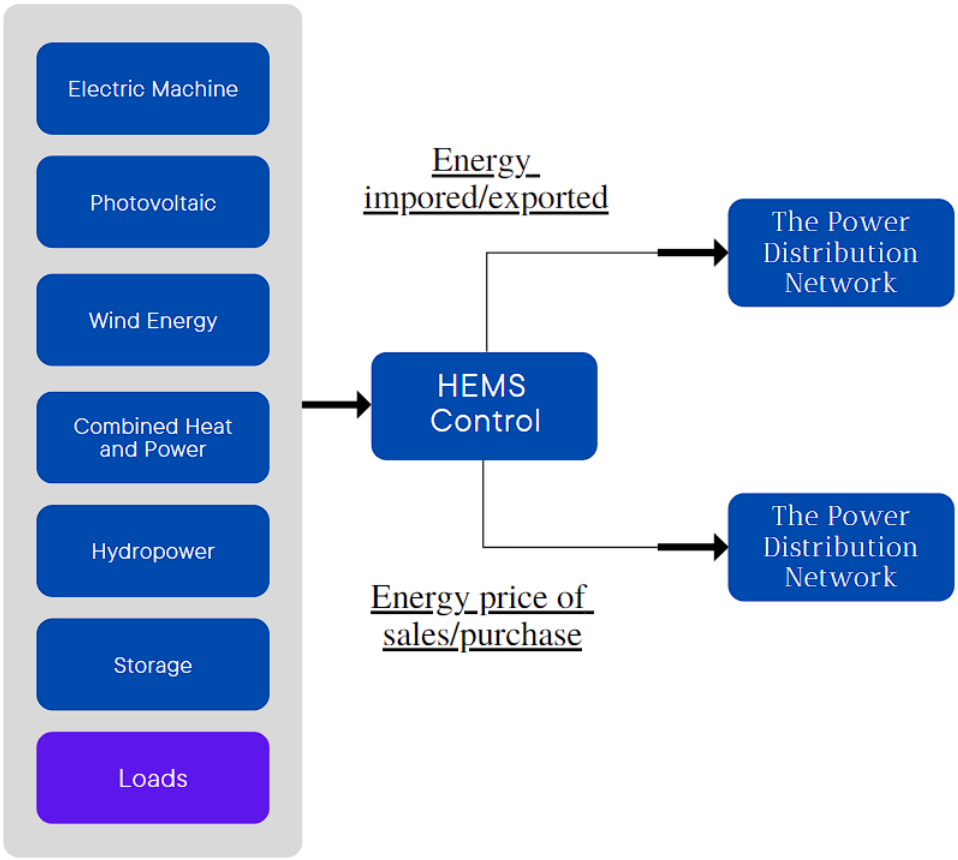

Figure 2.4: Diagram of a HEMS and its interaction with electricity markets and networks.

Electricity is a relatively recent technology, given its importance. The human species and our organization as a civilization have depended on using and managing external energy sources since our earliest organizations in flocks. Stock management and manipulation of resources have always been of high importance. Electricity production began about a century and a half ago, mainly because of difficulties in power transmission, along with the beginning of standardization of electrical systems; until almost half of the 20th century, the most significant presence was of distributed generation, which wore

distributed to feed the loads where they were, spatially distributed over the locations. It was the practice of industry, commerce, residence, or agriculture. Centralized generation became dominant through the evolution of electrical machines and distribution equipment. It became part of nations' standardization, normalization, and strategic systems, thus becoming the norm. That was made possible by technical advances that implied significant economies of scale. However, centralized generation depends on substantial investments, with large infrastructures and low flexibility concerning the locality once produced, indivisible production, and flexibility that implies adaptations to the electrical systems. Because electric power is strategic, this has also meant a joint monopoly and high control by governments and a few companies in the sector. This scenario depends on less liberal relations in energy production, and with constant energy crises, the price of energy increases, and the perspective that it will increase even more. Since the decade 70, with the oil shock, there have been the beginnings of the expansion of distributed generation and cogeneration in local production, accompanied by technical evolutions for the presence of distributed generation as technically possible and rewarding.

The use of DG is increasing, and we shall try to understand why. Smith et al. in [19] find interesting answers, such as advancements in renewable energy technologies, declining costs of DG systems, and the potential for enhanced energy efficiency and grid resilience.

2.1.3 Distributed generation

Distributed generation refers to producing electric energy located close to the consumer. In the last decade, mainly with photovoltaic technology, it has become more common that the consumer is also the producer of his energy. It generally occurs at relatively low power levels to supply local loads. To adapt to the demand, the parties involved are control systems, systems that operate the generators, and eventual load control.

Centralized power generation, which has been the norm for almost a century, is, therefore, an alternative model to this, with a distributed generation being another way to

produce and operate power production within the electric system, such that the technical and financial challenges are also of another nature when compared with centralized generation.

Centralized generation involves the production of electric power in large capacities, which implies high machinery for its production, transmission, and distribution. It usually involves highly complex technical apparatus and massive infrastructures associated with a small number of producers and operators of the electrical grid and a vast number of consumers of the energy generated. Most energy production in centralized generation occurs in large hydroelectric plants, nuclear power plants, gas-fired plants, plants burning coal and other fossil fuels, and massive wind and solar plants. These are usually large infrastructures that also require significant investments.

Concerns about environmental problems, agreements signed by the world's nations, and technical advances in the last decades in photovoltaic production and wind and hydro generators have caused a distributed generation to grow as a more exciting possibility. That ranges from individual residential consumers to consumers with lower power requirements, where in the context of distributed generation, it can address renewable or non-renewable sources. Especially prominent in a distributed generation are diesel generators, fuel cells, micro gas turbines, and within the renewables are small hydroelectric plants, biomass plants, photovoltaic installations, and small wind turbines.

Below is Figure 2.5 representing a DG and CG, showing how both can power the local place where it is generated and transmitted.

The rise of distributed generation is primarily driven by the following factors: further liberalization of the power sector, the entry of new companies into the industry providing various types of products and services, the urgent demand for protection, restoration, and future planning regarding the environment, along with compliance with environmental agreements, a quest for greater energy efficiency, the needs for sustainable development, and the increased reliability and desire for these technologies. It came accompanied by new philosophies of power generation, which have also come about through these changes and technical demands from various areas, along with elements of urgency and hype use.

Figure 2.5: Representation of centralized generation and distributed generation together [20].

There is also the fact that the combination of heat and power (cogeneration) contrasts in most cases with the centralized generation because the released heat is a negative factor in going into the biosphere. In contrast, distributed generation needs to present this problem. There have been considerable developments regarding generators that, together with the mitigation of environmental effects, have drastically boosted the recent growth of technologies associated with distributed generation.

Below is Figure 2.6 that compares the expected growth data for the United States regarding commercial and residential production for photovoltaic compared to other energy sources.

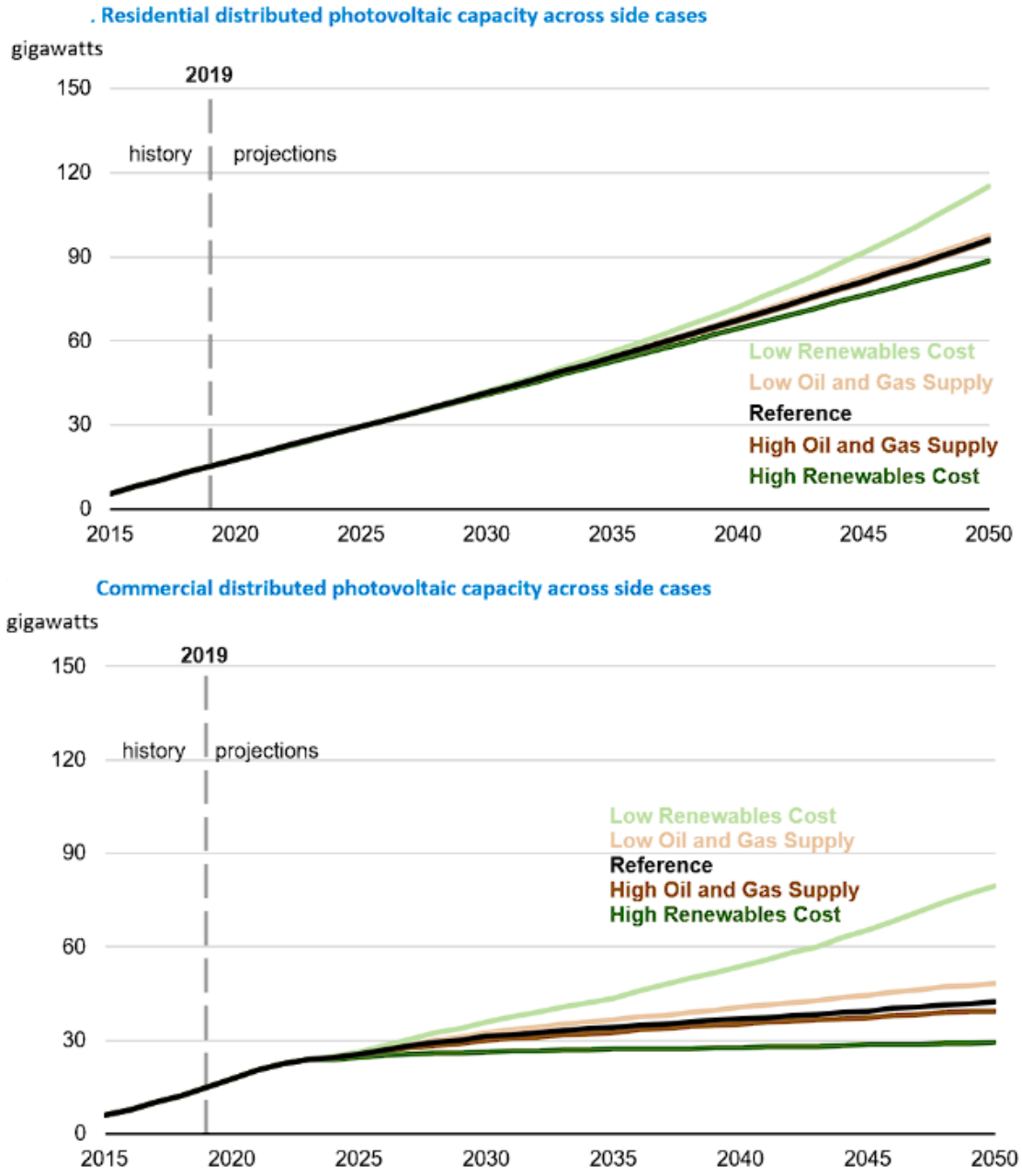

Figure 2.6: Demonstration of photovoltaic production growth and forecasts. Adapted from [21].

2.2 Machine Learning

This work executes a conjunct of Machine Learning (ML) models using Python, which searches and predicts the following energy generation results in Silk House, which has solar and hydro generators linked with a group of batteries connected to the grid.

Modelling the problem involves entry data from sensors, which gives us electric models for the consumption and production of electricity as the power flux inside the Home Energy Management Systems (HEMS) (for the collected data, it is a building energy management system). Taking the data for modelling that will consider a tiny HEMS, the type of problem is a probabilistic problem, where the desire is to predict a possible behaviour and then analyze the generated result for a better decision. In a brief resume, ML techniques usually get good results (a better ratio between the error and the computational needs for the low error).

2.2.1 Linear and non-linear models

Linear and non-linear models are fundamental tools in machine learning for analyzing and predicting complex relationships in data. Linear models, characterized by their linear combination of features, offer simplicity and interpretability. They assume a linear relationship between the input and target variables, making them well-suited for tasks like regression and classification. The coefficients in linear models represent the importance or contribution of each feature, providing insights into the impact of different variables. However, linear models have limitations when capturing non-linear patterns and complex interactions between components.

Non-linear models, on the other hand, are designed to handle more intricate relationships. They can capture complex patterns and interactions through non-linear functions, enabling them to approximate highly non-linear relationships between variables. Decision trees, random forests, support vector machines, and neural networks are popular

non-linear models widely used in machine learning. These models can effectively model intricate dependencies and interactions among features, making them suitable for image recognition, natural language processing, and time series analysis.

The choice between linear and non-linear models depends on the specific problem and the nature of the data. Linear models are often preferred when interpretability and simplicity are essential or when the relationships in the data are relatively linear. Non-linear models, on the other hand, excel in scenarios where complex patterns and non-linear interactions are present. However, non-linear models can be more complicated to interpret and may require more computational resources and training data.

Various techniques can be applied to enhance the performance of both linear and non-linear models. Regularization methods, such as L1 and L2, help prevent over-fitting and improve generalization by adding a penalty term to the loss function.

L1 and L2 regularization techniques, along with PCA, FFS, and $K - Fold$ cross-validation, are essential methods used in machine learning to improve model performance and evaluation. L1 and L2 regularization are ways to add penalties to the model's training process. L1 regularization encourages sparsity in the model by making some coefficients precisely zero, which helps with feature selection. L2 regularization encourages smaller coefficients by shrinking them towards zero, preventing over-fitting. These techniques promote simpler and more robust models. PCA, or principal component analysis, is a technique to reduce data size. It transforms the original features into new orthogonal variables called main components, capturing essential information while lowering complexity and correlated traits.FFS, or feature selection, involves selecting a subset of relevant features from the original set. It helps reduce noise and improve model performance by focusing on informative features. L1 regularization can be particularly useful for feature selection by eliminating irrelevant features. K-fold cross-validation is a method for evaluating models and tuning hyper-parameters. It divides the data into k subsets, trains the model on $k-1$ subsets, and evaluates its performance on the remaining subset. This process is repeated k times to provide a reliable estimate of the model's performance and prevent over-fitting [22].

Below is Figure 2.7, a flowchart demonstrating possible uses of techniques for better ML responses.

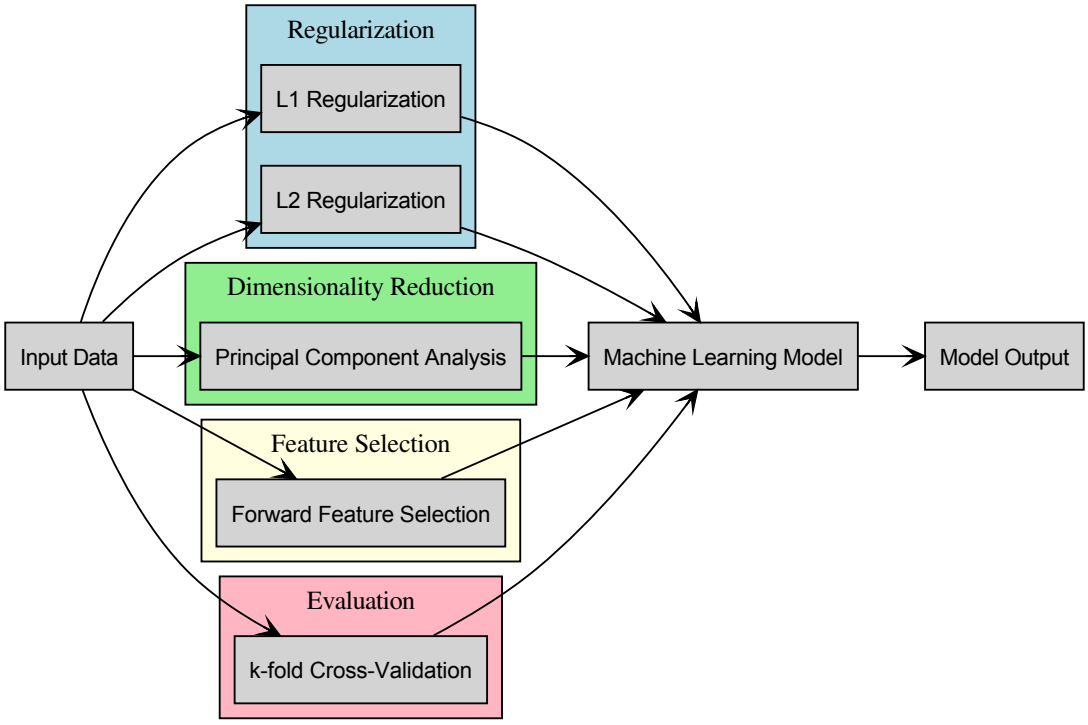

Figure 2.7: Representation of methods typically used for a better response in ML.

Choosing between the appropriate types of algorithms takes practice and experience, and qualifying the sort that might be most suitable is a job. Different kinds of techniques and the data set, or the way the working in data happens, can change the performance. There are problems like dealing with noise. Some are the best or worst answers for specific issues, whether the dataset and classes are linearly separable (or how easy it is to do that). The nature of the statistical machine will correspond to the statistical factors by the nature of the statistical work, the ML. That implies that an outcome with low error would be suitable in pursuit of predictability, a condition often defined by a high predictive accuracy despite the necessity for repeated executions. It is gauged by the ratio between a consistent error value and the desired result, alongside a low demand for successive program executions. Regardless, a universal factor in the face of problems in ML is the data set, as already mentioned. With a large and suitable dataset (to the

problem), it tends to have better results more quickly. In machines created using a supervised approach, observers typically consider five criteria when assessing the conclusion of resource processing and the collection of labelled training examples: selecting a performance metric, choosing a projector and optimization, evaluating model performance, and adjusting techniques [22].

2.2.2 The data treatment and its characterization

Data characterization is the first step in ML for dealing with data and producing good results. The data set has massive importance for feeding the different types of algorithms. Organized data, classified data, and a dataset that had been clean of errors, missing values, and other types of disorganization are good data sets to use. We will use the data to generate descriptive parameters that describe the characteristics and behaviour of a particular data set. That means the unsupervised learning algorithms find patterns, clusters, and trends without incorporating class labels that may have biases, which allows trying to make predictions. The data characterization is the first step in finding any hidden information that allows better use of ML. Chapter 3 consists of the first data approach.

Below is Figure 2.8, a flowchart demonstrating the sequence of steps of an ML Model Training.

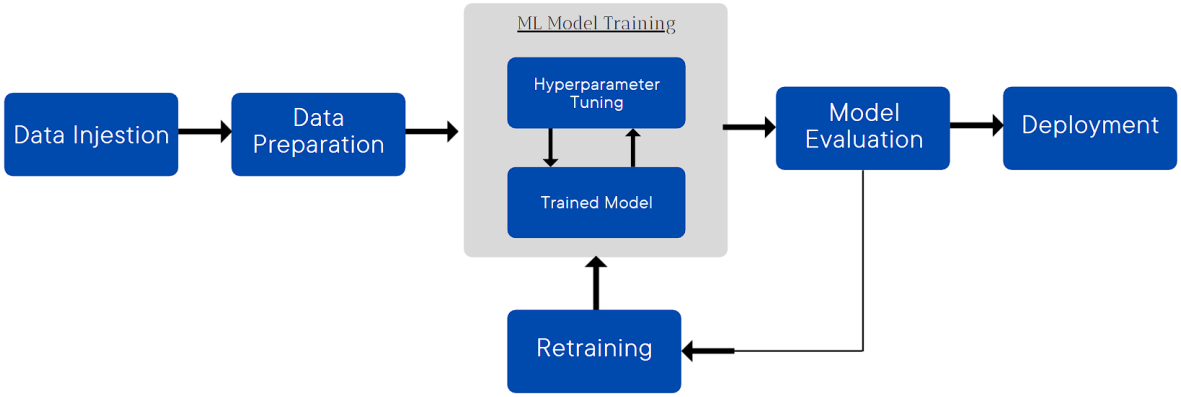

Figure 2.8: Flowchart of a Machine Learning (ML) use.

2.3 Models and recent literature

This work will use different regression models for the same data set, inputs, and outputs. Separate regression models demonstrate other performances based on multiple factors. In the recent literature, various authors have chosen one or more models to apply the regression techniques. Below is information about the models and a reference to a recent work using this model in HEMS. The models are listed below.

- **01 - LRM - Linear Regression Model**: Linear Regression
- **02 - ELR - Elastic Net Regressor**: Elastic Net Regressor
- **03 - SGD - SGD Regressor**: SGD Regressor
- **04 - BRR - Bayesian Ridge Regressor**: Bayesian Ridge Regressor
- **05 - SVR - Support Vector Regression**: Support Vector Regression
- **06 - GBR - Gradient Boosting Regressor**: Gradient Boosting Regressor
- **07 - CBR - Cat Boost Regressor**: Cat Boost Regressor
- **08 - KRR - Kernel Ridge Regressor**: Kernel Ridge Regressor
- **09 - XGB - XGBoost Regressor**: XGBoost Regressor
- **10 - GBM - LightGBM Regressor**: LightGBM Regressor
- **11 - DTR - Decision Tree Regressor**: Decision Tree Regressor
- **12 - MLP - MLP Regressor**: MLP Regressor

- **13 - KNN - K-Nearest Neighbors**: K-Nearest Neighbors
- **14 - RFR - Random Forest Regressor**: Random Forest Regressor
- **15 - ABR - Ada Boost Regressor**: Ada Boost Regressor
- **16 - GPR - Gaussian Process Regression**: Gaussian Process Regression
- **17 - RRM - Ridge Regression Model**: Ridge Regression Model
- **18 - BRM - Bagging Regressor Model**: Bagging Regressor Model
- **19 - HGR - Hist Gradient Boosting Regressor**: Hist Gradient Boosting Regressor
- **20 - ETR - Extra Trees Regressor**: Extra Trees Regressor

2.3.1 Linear Regression – LR

Linear regression is a technique to model the relationship between a dependent variable and one or more independent variables. By estimating the coefficients that define the best-fitting line, linear regression allows us to understand how the independent variables impact the dependent variable. This technique employs methods like ordinary least squares (OLS) to minimize the sum of squared residuals and find the line that best fits the data. Its simplicity, interpretability, and versatility make linear regression valuable for gaining insights, making predictions, and understanding the relationships between variables in various domains such as economics, finance, social sciences, and machine learning algorithms.

Liu in [23] presents a novel approach to home energy management using linear regression and price-based optimization techniques. The study aims to optimize energy consumption in residential buildings by leveraging data-driven models and pricing information. The authors propose a framework that combines linear regression models with an optimization algorithm to minimize energy costs while maintaining user comfort.

2.3.2 Elastic Net Regressor – ELR

Elastic Net is a regularization technique widely used in machine learning and statistics, particularly linear regression models. It combines the strengths of both L1 and L2 regularization methods, effectively addressing the limitations of each. By introducing a penalty term that is a linear combination of the L1 and L2 penalties, Elastic Net encourages both sparsity and group-wise selection of features. It makes it especially useful in scenarios where there are high-dimensional datasets with a large number of correlated traits. Elastic Net offers a flexible regularization approach, allowing an automatic feature selection and handling of multicollinearity.

Doe in [24] proposes a novel approach for optimizing home energy consumption based on elastic net regularized regression and real-time pricing. The authors address the challenge of managing energy usage in residential buildings by leveraging advanced ML techniques. The study used elastic net regularized regression, which combines both L1 (Lasso) and L2 (Ridge) regularization techniques, to improve the accuracy and interpretability of the energy consumption models. The system can adapt to dynamic pricing signals by incorporating real-time pricing data and optimizing energy consumption. The net elastic approach allows for identifying significant features while also handling multicollinearity. It is suitable for capturing the complex relationships between energy usage and various factors such as weather conditions, occupancy patterns, and appliance usage.

2.3.3 Stochastic Gradient Descent Regressor – SGD

The SGD (Stochastic Gradient Descent) Regressor is a versatile algorithm widely used for training linear models in machine learning. It is particularly suitable for large datasets and real-time learning. The algorithm optimizes model parameters by iteratively adjusting them based on the gradients of the loss function. Its stochastic nature, using randomly selected subsets of training samples, makes it efficient and scalable.

In recent literature, the SGD Regressor has found applications in Home Energy Management Systems (HEMS). For example, Zhang in [25] proposed an optimal HEMS framework using real-time pricing and the SGD Regressor. Their study demonstrated

cost savings and load reduction capabilities. Also, Zhang in [26] focused on demand response management in HEMS, using the SGD Regressor for energy demand prediction and appliance scheduling adjustments. Their results highlighted improved performance compared to traditional methods.

2.3.4 Bayesian Ridge Regression – BRR

Bayesian Ridge regression incorporates prior information about the coefficients into the model through prior distributions. It allows for uncertainty estimation and provides a more robust and stable estimate of the coefficients compared to traditional linear regression methods. Additionally, the ridge regularization term helps to address multicollinearity issues by introducing a penalty that encourages smaller coefficient values. This regularization helps prevent over-fitting and improves the model's generalization performance.

Recent literature has demonstrated the effectiveness of Bayesian Ridge regression in the context of HEMS. Li et al. in [27] proposed a Bayesian Ridge regression approach for energy management in residential buildings. Their study showed that Bayesian Ridge regression could effectively optimize energy utilization and achieve cost savings by accurately predicting electricity consumption and managing the operation of appliances. Similarly, Wang et al. in [28] applied Bayesian Ridge regression for load forecasting in HEMS. Their research highlighted the advantages of Bayesian Ridge regression in providing accurate load predictions, enabling efficient energy planning and resource allocation.

2.3.5 Support Vector Regression – SVR

Support Vector Regression (SVR) finds application in complex and non-linear scenarios. SVR transforms the input data into a high-dimensional feature space and seeks a hyperplane that maximizes the margin between predicted outputs and actual targets. This approach allows SVR to capture intricate patterns and handle high-dimensional data effectively.

Li et al. in [29] proposed an SVR-based energy management strategy for optimizing energy consumption in residential buildings. Their approach demonstrated significant energy savings and peak load reduction. Zhao et al. in [30] employed SVR with clustering analysis for load forecasting in HEMS, achieving improved prediction accuracy and robustness compared to traditional methods. Li and Zhao are also frequent researchers in this field.

2.3.6 Gradient Booster Trees – GBT

Gradient Boosting combines the strengths of ensemble learning and gradient descent optimization. It effectively solves regression and classification problems, often outperforming other algorithms regarding predictive accuracy. Gradient Boosting sequentially builds an ensemble of weak learners, typically decision trees. Each subsequent learner undergoes training to rectify the errors made by the previous ones, emphasizing the reduction of the loss function gradient.

Zhang et al. in [31] proposed a Gradient Boosting-based load forecasting model for HEMS, which achieved superior accuracy compared to traditional methods. Their approach accounted for multiple factors, including weather conditions, historical load data, and calendar information, resulting in more precise load predictions. Zhao et al. in [32] used Gradient Boosting to optimize energy consumption in residential buildings. Their study demonstrated significant energy savings by leveraging the model's ability to capture complex relationships and identify energy-efficient patterns.

2.3.7 Cat Boost Regressor – CBR

Cat Boost is a gradient-boosting framework that handles categorical features in machine learning tasks. Its algorithm combines gradient boosting with a category-specific approach. Cat Boost is known for taking categorical variables without extensive preprocessing, making it efficient and effective in various applications.

Zhang et al. in [33] found good results with numerical features in the study of the

superior performance of Cat Boost compared to traditional regression models in load forecasting for HEMS. Also, Liu et al. in [34] propose an enhanced HEMS framework that integrates Cat Boost for load forecasting and optimization. The study showcases the improved accuracy and efficiency achieved by Cat Boost in predicting energy consumption patterns and optimizing energy usage in residential buildings.

2.3.8 Kernel Ridge Regression – KRR

Kernel Ridge Regression finds utility in both regression and classification tasks. It combines the concepts of ridge regression and kernel methods to handle non-linear relationships between variables. In this method, a non-linear transformation of the input features, known as the kernel trick, is applied to map the data into a higher-dimensional feature space. Then, a linear regression model is used for this transformed feature space, considering a regularization term known as the ridge penalty.

Bouslimi et al. in [35] proposed a robust Kernel Ridge Regression approach for non-intrusive load monitoring, which achieved accurate load prediction by considering energy consumption's temporal and contextual dependencies. Zhang et al. in [36] applied Kernel Ridge Regression for residential load forecasting, taking into account weather data as an additional input to capture the influence of weather conditions on energy consumption.

2.3.9 Extreme Gradient Boosting – XGB

XGBoost, short for Extreme Gradient Boosting, is an ensemble learning technique that combines the predictions of multiple weak models, such as decision trees, to create a robust predictive model. It employs a gradient-boosting framework to train new models and minimize the overall prediction error iteratively. XGBoost introduces innovations such as regularization techniques, parallelization for efficient training, and a customized loss function to optimize model performance.

Zhang et al. in [37] proposed a load forecasting model for HEMS that incorporates weather data using XGBoost. The results showed superior performance to traditional

methods, enabling accurate load predictions and efficient energy management in residential buildings.

2.3.10 LightGBM Regressor – LGBM

LightGBM is a gradient-boosting framework suited for handling large-scale datasets and utilizes gradient-based one-sided sampling (GOSS) to achieve faster training speed while maintaining good accuracy. It employs a leaf-wise tree growth strategy and implements histogram-based binning to partition feature values, reducing memory usage and computational overhead. Additionally, LightGBM supports advanced features such as categorical feature handling and early stopping to prevent overfitting.

Kim et al. in [38] proposed a model that incorporated various input variables, including historical load data, weather information, and calendar features, to predict future load demand accurately.

2.3.11 Decision Tree Regressor – DTR

A Decision Tree is a non-parametric model that makes predictions by recursively partitioning the input data based on a set of decision rules inferred from the training data. The main advantage of decision trees is their interpretability. The tree structure makes us understand and visualize the decision-making process more manageable. Each internal node represents a decision based on a feature, and each leaf node represents a class or a predicted value. Decision trees can handle categorical and numerical elements, making them suitable for various applications.

Yu et al. in [39] developed a decision tree-based model for energy consumption prediction in smart homes. The model utilized historical energy consumption data and weather information to forecast future energy usage. The experimental results demonstrated the effectiveness of decision tree models in accurately predicting energy consumption patterns, enabling better energy management and optimization in residential buildings.

2.3.12 Multilayer Perceptron Regressor – MLP

The MLP Regressor, short for Multilayer Perceptron Regressor, is an artificial neural network consisting of multiple layers of interconnected nodes known as neurons. Each neuron performs a weighted sum of its inputs, applies an activation function to the sum, and produces an output. By stacking multiple layers of neurons, the MLP Regressor can learn complex non-linear relationships between input features and target variables.

Wang et al. in [40] demonstrated the effectiveness of MLP Regressors in accurately forecasting short-term load demand in residential buildings. Their work showcases good results.

2.3.13 K-Nearest Neighbors Regressor – KNN

K-Nearest Neighbors (KNN) Regressor finds application in both classification and regression tasks. This non-parametric algorithm predicts based on the similarity of input data points to their neighbouring points in the feature space. The "K" in KNN refers to the nearest neighbours considered for making predictions. In the case of regression, the algorithm calculates the average or weighted average of the target values of the K nearest neighbours to determine the predicted value for a given input.

Li et al. in [41] utilized KNN for short-term load forecasting in residential buildings. By considering the historical load data and the similarity of input features to past patterns, KNN can predict future energy consumption with reasonable accuracy. KNN has also been utilized in anomaly detection, facilitating the identification of abnormal energy usage patterns that significantly deviate from the norm. This approach aids homeowners and system operators in identifying potential faults or anomalies within the HEMS.

2.3.14 Random Forest Regressor – RFR

Random Forest serves as a tool for both classification and regression tasks. It relies on the creation of multiple decision trees and the amalgamation of their predictions to

enhance the accuracy and robustness of predictions. Each decision tree within the Random Forest undergoes training on a random subset of the training data and a random subset of the input features. During the forecast, the final output is obtained by averaging or voting the forecasts of individual trees.

Wang et al. in [42] utilized Random Forest for load forecasting, where the algorithm considers historical load data and weather information to predict future energy consumption patterns.

2.3.15 Adaptive Boosting Regressor – ABR

AdaBoost, short for Adaptive Boosting, is a popular boosting algorithm in machine learning. It is a meta-algorithm that combines multiple weak learners, typically decision trees, to create a robust ensemble model. AdaBoost assigns weights to each instance in the training data, with higher weights given to misclassified cases. It trains a new weak learner to focus on the previously misclassified samples in each iteration. Aggregating the predictions of all vulnerable learners produces the final prediction, with each weak learner's contribution weighted by their performance.

Li et al. in [43] utilized AdaBoost to predict the energy consumption of residential buildings, considering factors such as weather conditions, occupant behaviour, and building characteristics.

2.3.16 Gaussian Process Regression – GPR

Gaussian Process Regression is used for modelling and predicting continuous target variables. Unlike traditional regression models that assume a specific functional form, GPR models the relationship between input features and target variables as a distribution over functions. It assumes that any finite set of target values follows a multivariate Gaussian distribution. GPR can estimate uncertainties associated with its predictions, which can be valuable for decision-making in HEMS applications.

Kim and Kim in [44] utilized GPR to predict the energy consumption of buildings,

considering factors such as outdoor temperature, solar radiation, and time of day. The authors showed that GPR achieved accurate predictions and highlighted its potential for energy-saving strategies and load management in HEMS.

2.3.17 Ridge Regression – RR

Ridge Regression employs handling multicollinearity and enhancing the stability of regression models. It is an extension of ordinary least squares (OLS) regression that introduces a regularization term to the loss function, which helps prevent overfitting and reduces the impact of highly correlated predictors. Ridge Regression can handle situations where multiple predictors are highly correlated.

Zhang et al. in [45] utilized Ridge Regression to predict electricity consumption in residential buildings. The authors considered temperature, humidity, and time of day input features.

2.3.18 Bootstrap Aggregating Regressor – BAR

Bagging Regressor, short for Bootstrap Aggregating Regressor, is an ensemble learning technique that combines multiple base regressor models to improve the overall prediction performance. It belongs to the bagging method family, which involves training numerous models on different subsets of the training data and then aggregating their predictions. By training various base regressor models on different subsets of the training data and averaging their predictions, BaggingRegressor can reduce the variance and improve the generalization ability of the final model.

Kim et al.in [46] employed BaggingRegressor to forecast electricity demand in residential buildings. The authors compared the performance of BaggingRegressor with other regression techniques and found that BaggingRegressor consistently provided accurate and reliable predictions across different seasons and time horizons.

2.3.19 Histogram Gradient Boosting – HGB

Histogram Gradient Boosting Regressor belongs to the gradient boosting family, designed explicitly for regression tasks. It combines the advantages of gradient and histogram-based gradient boosting. HGB leverages histogram-based gradient boosting. It discretizes the continuous input features into bins and constructs histograms, which enables faster and memory-efficient training. The algorithm can make optimal splits in each tree node using histograms, improving computational efficiency and predictive performance.

Huang et al. in [47] utilized Histogram Gradient Boosting Regressor for short-term load forecasting in commercial buildings. The authors compared Histogram Gradient Boosting Regressor with other regression algorithms. They found that it outperformed the competitors in terms of accuracy and computation time, making it a good choice for HEMS load forecasting.

2.3.20 Extra Trees Regressor – ETR

The Extra Trees Regressor is a machine learning algorithm that belongs to the ensemble learning family, specifically the tree-based methods. It is an extension of the Random Forest algorithm and shares some similarities. The main idea behind the Extra Trees Regressor is to build many unpruned decision trees and make predictions by averaging their outputs. One of its key characteristics is its high level of randomness. Each tree's splitting point for each feature is selected randomly instead of using the optimal split. This randomness reduces the correlation between trees and enhances the diversity of the ensemble.

Chen et al. in [48] utilized the Extra Trees Regressor for energy consumption prediction in residential buildings. The authors compared its performance with other regression models and found that the Extra Trees Regressor achieved accurate and reliable forecasts.

Chapter 3

The Data Set and the Methodology

3.1 Data Characterization

In Chapter 3, we will introduce the data and their correlations. Finding correlations in the data is significant for predictability analysis. A predictive model can only have a positive result with a high correlation. Correlations measure how much a data set appears to be associated with another group, that is, how much one variable presents a good result when exposed to another. There are several types of statistical correlation. Some examples are Pearson, Kendall, and Spearman. The one used here is Pearson. At Pearson, the correlation indices range from -1 to $+1$, with intermediate values such as 0.75, which is optimal for a correlation by default. Models with correlations of this degree up to 1 give good results [22].

At Silk House, several electrical devices make the building a HEMS. The quoted data are about the measurement of unidirectional power flow, measured in kWh. We compile the data into a set that segregates them daily, and the record is a table of several daily entries of power flows, where each entry pertains to a group of devices. As the building behaves like a HEMS, a part of the building has load behaviour, consuming energy, and another part of power production, generating energy. We record the flow as a unidirectional vector in the data, so we have collected positive scalar values. There are seven

values measured with the sensors:

(1) The total consumption, called Total Consumption (TC)

(2) Internal consumption (loads), called Direct Consumption (DC)

(3) Power output of the batteries (discharging), called Battery Discharge (BD)

(4) Main power consumption (bus input), called Network Consumption (NC)

(5) Photovoltaic production, called Photovoltaic Production (PP)

(6) Power injection output to the bus, called Network Injection (NI)

(7) Power input to the battery (charging), called Battery Charge (BC)

The total collection period was from month 5 of 2021 to month 4 of 2022, totalling 12 months. As there are differences in days between the months and the collection record is daily, the months have slight differences in the amount of data between the months.

Below, there is Table 3.1 with information on the Median, Standard Deviation, Variance, and Average values from every variable, with all data sets from 12 months. Three variables have a Standard Deviation above 5, which shows differences between the values inside the same variable.

Table 3.1: Table of Median, Standard Deviation, Variance and Average values from variables.

Variables	TC	DC	BD	NC	PP	NI	BC
Median	25.39	14.29	4.7	1.89	18.95	1.43	6.6
St. Dev.	9.7585	6.6976	3.8653	9.5334	9.2502	3.8146	3.6345
Variance	95.229	44.857	14.941	90.886	85.567	14.551	13.21
Average	25.386	13.23	5.2176	6.7906	20.02	3.3786	6.8513

Pearson's correlation coefficient, often known as the correlation coefficient, was used. It is unit less and ranges from -1 to 1. Its calculation equation is below. On the other hand, covariance measures the joint variability between two variables but does not provide a standardized measure of the relationship. It is sensitive to changes in units and can take any real value.

$$\mathrm{cov}(X, Y) = \frac{1}{n} \sum_{i=1}^{n} (x_i - \bar{x})(y_i - \bar{y}) \tag{3.1}$$

In this Equation of Covariance (3.1) [49]:

- n represents the number of data points or observations.

- x_i and y_i are the individual data points for X and Y respectively.

- $\bar{x}$ and $\bar{y}$ are the sample means of X and Y respectively.

$$\rho = \frac{\text{cov}(X, Y)}{\sigma_x \sigma_y} \tag{3.2}$$

In this Equation of Correlation (3.2) [50]:

- ρ: This is the Pearson correlation coefficient, which measures the strength and direction of the linear relationship between two variables, X and Y.

- $\text{cov}(X, Y)$: This represents the covariance between variables X and Y. It quantifies how these two variables change together. The covariance term is in the numerator because it measures the joint variability of X and Y.

- σ_x: This is the standard deviation of variable X. It measures the dispersion or spread of the data points for X around their mean.

- σ_y: Similarly, this is the standard deviation of variable Y. It measures the dispersion or spread of the data points for Y around their mean.

The Coefficient of Determination (R^2) is a statistical measure used in regression analysis to indicate how well a regression model fits the observed data. R^2 quantifies the proportion of the variance in the dependent variable that can be explained by the independent variables included in the model. R^2 value typically falls within the range of 0 to 1, where 0 indicates that the model fails to explain any of the variance, and 1 indicates a perfect fit, where the model accounts for all the variance [51].

In this Equation of Coefficient of Determination (3.3) [51]:

$$R^2 = 1 - \frac{SSR}{SST} \tag{3.3}$$

- R^2: It is a value that suggests a superior fit of the model to the data.

- SSR: denotes the sum of squared residuals, representing the differences between actual and predicted values.

- SST: signifies the total sum of squares, which characterizes the variance of the dependent variable. A higher R^2 value suggests a superior model fit to the data.

This predictability analysis relies on a vast data set with strong correlations to achieve rapid and good predictability results. Criteria such as the R^2 can be employed to assess these results after model training. The information in Chapter 4 should consider the small dataset and low correlations.

Regression models will try to build equations with dependent and independent variables and weights that multiply the variables. When the variance is high, the model is susceptible to the data, which tends to imply models that do not give good results with a short data set, which is especially relevant for linear models. Linear models depend on these weighting coefficients more than nonlinear models.

The high correlation between variables is a good sign for models to find predictive paths, but this also implies high variance for linear models. Therefore, linear models commonly make decisions such as suppressing a highly correlated variable to generate the best result. This chapter contains nonlinear and other linear models, with no suppression of variables. The objective is to maintain the same input data conditions to generate the model and test data to impose the model on the observation and result. As the chosen variables are correlated, linear models tend to perform differently than nonlinear ones.

It was only one match above 0.75, with Photovoltaic Production (PP) and Direct Consumption (DC). The Total Consumption (TC) and Network Consumption (NC) have a value above 0.65. However, all others are terrible. Data with low correlations will show challenges to every model, even worse with scarce data, as is the case.

Below, there is Table 3.2 with the correlation between the variables among themselves. A good value is considered from ± 0.75 to ± 1.

Below, there is Table 3.3 with the covariance between the variables among themselves, and it measures the total variation of the variables from their expected values between them. A high covariance between the independent and dependent variables indicates a strong relationship, implying that changes in the independent variable correspond closely with changes in the dependent variable.

Table 3.2: Table of correlation between the variables

Correlation	TC	DC	BD	NC	PP	NI	BC
TC							
DC	0.3888						
BD	0.1734	0.1343					
NC	0.6583	-0.278	-0.165				
PP	-0.067	0.7836	-0.074	-0.576			
NI	-0.355	-0.05	0.1002	-0.384	0.2172		
BC	-0.179	-0.004	0.1539	-0.298	0.2524	-0.084	

Table 3.3: Table of covariance between the variables.

Covariance	TC	DC	BD	NC	PP	NI	BC
TC							
DC	25.408						
BD	6.5387	3.4779					
NC	61.248	-17.74	-6.07				
PP	-6.092	48.546	-2.651	-50.8			
NI	-13.21	-1.277	1.4776	-13.97	7.6641		
BC	-6.344	-0.089	2.1621	-10.33	8.4874	-1.16	

The data used originates from the place itself, where we actively sought to address possible errors in the data before using it. That represents one of the essential elements in the context of data analysis. This work analyzes a physical environment that exists, and there needs to be more control over data collected in the past, such as knowledge about possible problems in data collection by sensors and meters, before the present data dates.

Other types of possible errors in data are also associated with the transmission, capture, and storage by computer systems. These errors may eventually exceed the expected errors, and data may be collected incorrectly. Therefore, there are limits to the data treatment, which will be taken by observing the data used (after the minimum possible treatment) as reliable data. There is an infinity of devices for processing and producing results in machine learning. The choice of technique is given by the selection of the

designer referring to the observation of the recent bibliography, which seeks to solve problems close to or similar to the context of this work. These decision biases impact the final as they are also part of the project; the project planning decisions are taken in the recent existing bibliography, considering these above considerations. If there were a standard or other criteria change, it would be different input-output techniques or configurations. Table 3.4 contains information about outliers and blanks.

Table 3.4: Table of outliers in total data and information about them.

Name	Outlier Number	% of Outliers	Blanks Number	Min Value	Max Value	Mean Value	Standard Deviation
TC	9	2.46	8	0.42	61.87	25.34	9.83
DC	0	0	0	0	29.9	13.23	6.71
BD	5	1.36	0	0	32.65	5.21	3.87
NC	21	5.75	0	0.01	57.73	6.79	9.55
PP	0	0	0	0.14	39.97	20.02	9.26
NI	7	1.92	0	0	16.27	3.38	3.82
BC	1	0.27	11	0	19.2	6.93	3.64

The outliers were addressed using the log transformation technique from the NumPy library, which involved taking the natural logarithm of data points.

3.2 Methodology for using the Models

Regression models need input data and test data. In all models, the regression techniques have two main criteria utilized. First, the code will choose one variable for the exit and one for the entry. Then, it will execute the 20 different regression models. The R^2, the Mean Square Error and the time used in each model are measured, and a graph with the results for presentation. Multiple loops will be executed in the models to calculate all the results for all the possible combinations of entries. In the end, it will find the results for many possible combinations and different models when predicting some of the variables in the exit.

Principal Component Analysis (PCA) is a mathematical technique used to reduce the dimensionality of data by identifying a set of orthogonal axes (principal components) along which the data varies the most.

The essential leading operation is with model fit and train test split where it chooses 80% of data for training and 20% for testing. The first run is only all the possible combinations without using PCA. In the second run, PCA is used. However, low data and correlations limit the number of good results.

As all possible combinations between inputs and outputs were made, always respecting only one output, the same results were generated for each variable. The code allows observing case by case, generating all possible results in the configurations used. After that, a data file was generated and submitted to another code for organization and presentation of values. This second code gathers the values according to the number of inputs (N-in) and by observing the variables in the output. Afterwards, the code makes the results proportional to the number of runs from the variables, building a proportion of how much the variable had a particular result compared to how many times the combinations will run in the code loop.

In Figure 3.1, we have a representation of the possible entry variables and one exit variable. The algorithm executes every possible combination in the entry and then makes a sequence of interactions of loops inside of loops to get the results. In resume, it will choose a variable for the exit and then insert the other variables in the entry. As there are seven total variables, there will always be one exit and all the possible combinations of the other variables. The number of input variables appears as $N - in$ in Chapter 4.

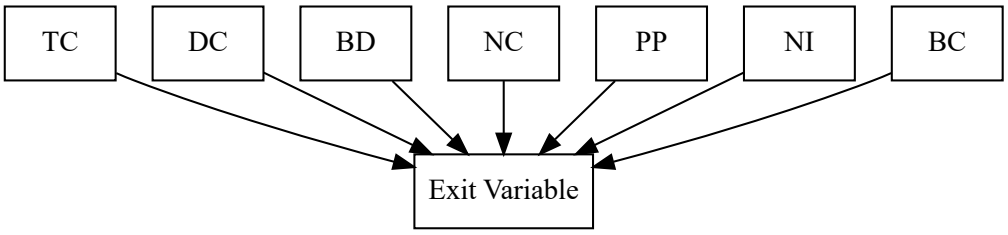

Figure 3.1: Flowchart of the executed algorithm.

Two models were run, one using PCA before all models and the other only running

models. All those data were later reunited and organized by another piece of code, also made in Python. Charts with both results are presented in the discussion, allowing us to analyse the difference. There are different metrics for looking at the results, each representing distinct levels of system performance. The important value is above 0.75 [52]. The others were distributed to organize and present the generated data. They are presented by R^2, which is:

1. Interval 0: Invalid, the model did not find a solution.

2. Interval 1: $[0, 0.39]$

3. Interval 2: $[0.39, 0.63]$

4. Interval 3: $[0.63, 0.75]$

5. Interval 4: $[0.75, 1]$

The models can have high challenges for this kind of prediction. As shown, the data set is tiny and has little of a sequence of correlations. The main criteria for results is an R^2 value in $[0.75, 1]$ [22]. The models used with the Python Regression tools were always in the same sequence with the same criteria.

The sequence of the run algorithm is below, in Figure 3.2.

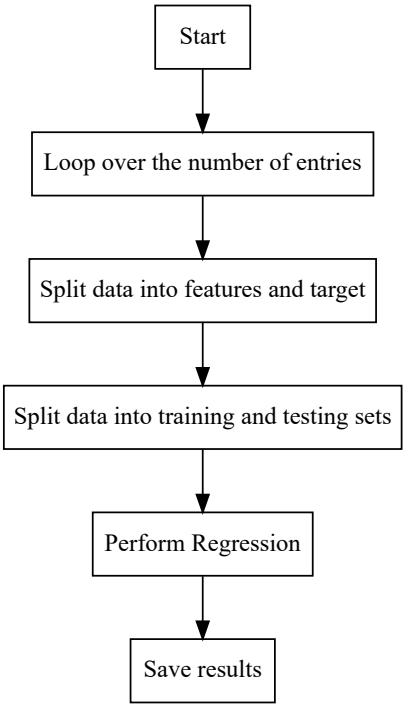

Figure 3.2: Flowchart of the executed algorithm.

The results of the tests and the discussion are presented in the next chapter.

Chapter 4

Results and Discussion

Here, in Chapter 4, we find the results from executing the models through Python. The models were run based on the code and data found by the link in the Appendix section. The code sequence runs the results by the $N - in$ with some exit variable for all Regression Models.

4.1 The ML Models

Bellow, there is a showcase variable by variable, showing the results. The behaviour of the results is related to the number of entries or variables in the in-out. There are variables with better results than others, and there is also a growth in the quality of the results as the number of entrances increases. The tables have the values in percentage, and those percentages represent the part of the results in total results. It's the percentage of results considering how many runs it took because of the number of possible combinations.

Two models were run, one using PCA before all models and the other only running models. All those data were later reunited and organized by another piece of code, also made in Python. Charts with both results are presented in the discussion, allowing us to analyse the difference.

Figure 4.1 shows one simulation result as an example. This example has a $[0.39, 0.63]$

R^2 score result. In every generated result, the graph shows the expected Ideal result in the red line and the simulation result in the blue dots. The model used is in the label, and the basic scores, like R^2, are below the image.

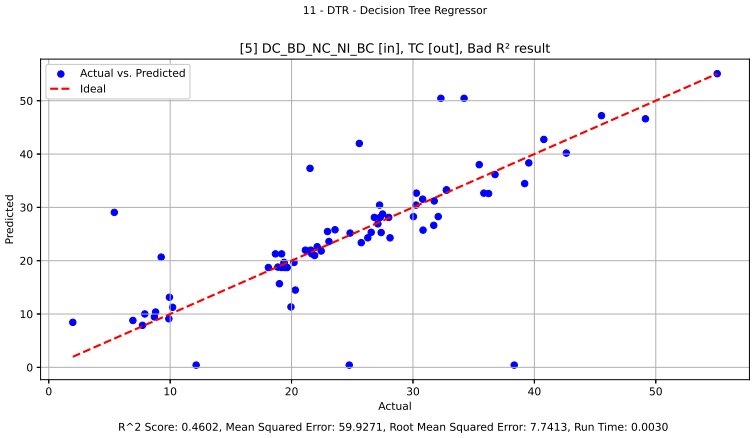

Figure 4.1: Exemple of output from one of the models.

Below, there are Tables 4.1 and 4.2 of best R^2 from models with PCA and without for demonstration. There are not many differences between them. As was with the other data, the best models tended to be those who used many features on in, and the best prevision also came from the Direct Consumption (DC). The best model in performance was the MPL.

Table 4.1: Table of best 7 R^2 values from models without PCA.

N-in	Features(in)	Features(out)	Model	R2	MSE
4	TC_BD_NC_PP	DC	12 - MLP - MLP Regressor	0.986	0.808
5	TC_BD_NC_PP_NI	DC	12 - MLP - MLP Regressor	0.985	0.867
6	TC_BD_NC_PP_NI_BC	DC	12 - MLP - MLP Regressor	0.982	1.040
6	TC_DC_BD_PP_NI_BC	NC	12 - MLP - MLP Regressor	0.969	3.631
5	TC_BD_NC_PP_BC	DC	12 - MLP - MLP Regressor	0.967	1.868
5	TC_BD_NC_PP_NI	DC	06 - GBR - Gradient Boosting Regressor	0.967	1.897
5	TC_DC_BD_NI_BC	NC	12 - MLP - MLP Regressor	0.965	4.118

The R^2 has shown consistent behaviour when compared with the incidence of it on the accounting of normalized results by variable. In the majority of cases, the MSE values were also low.

Bellow, the two Figures 4.2 and 4.3 have the best results without PCA and with PCA. Both are from MPL Regressor. The MPL Regressor had shown an excellent performance.

Table 4.2: Table of best 7 R^2 values from models with PCA.

N-in	Features(in)	Features(out)	Model	R^2	MSE
5	TC_DC_BD_NI_BC	NC	12 - MLP - MLP Regressor	0.988	1.387
5	TC_BD_NC_PP_NI	DC	12 - MLP - MLP Regressor	0.988	0.676
6	TC_DC_BD_PP_NI_BC	NC	12 - MLP - MLP Regressor	0.984	1.898
4	TC_BD_NC_NI	DC	12 - MLP - MLP Regressor	0.977	1.288
6	TC_BD_NC_PP_NI_BC	DC	12 - MLP - MLP Regressor	0.977	1.305
5	TC_DC_BD_PP_NI	NC	12 - MLP - MLP Regressor	0.970	3.476
5	DC_BD_NC_PP_BC	TC	12 - MLP - MLP Regressor	0.969	3.493

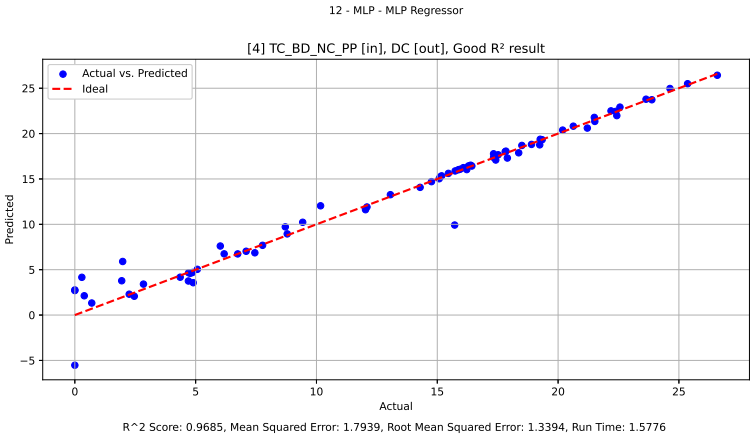

Figure 4.2: The best R^2 result without PCA.

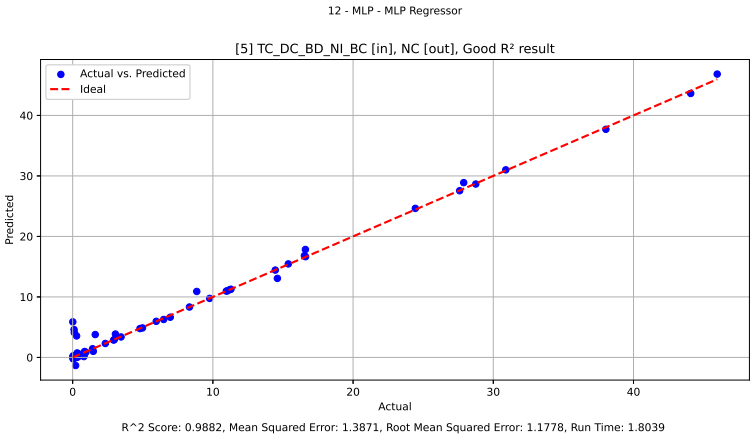

Figure 4.3: The best R^2 result with PCA.

4.2 Model Results

In bellow, we have the results from the best models. The table has the Model Name, the incidence of occurrences of R^2 in $[0.75, 1]$ (appearance), and the success percentage, which is the division between the number of successful incidences by times that that model runs, with the equation (4.1) representing that. The first was 1588/8860, giving 17.82% and 1503/8860, giving 16.96% as a result. The success of every model is based on the total runs by the model run, which is divided by 20 as there are 20 different models.

$$\text{Success}(\%) = \frac{\text{Model with } R^2 > 0.75}{\text{Total Models Run}} \tag{4.1}$$

Table 4.3 contains the Model Performances while using PCA and does not consider the number of entries. Using PCA, model 8 was able to find results.

Table 4.3: Table of Models Performance with using PCA.

Model Name	Frequency of R^2>0.75	Sucess in all (%)
20 - ETR - Extra Trees Regressor	123	27.77
06 - GBR - Gradient Boosting Regressor	119	26.86
12 - MLP - MLP Regressor	116	26.19
14 - RFR - Random Forest Regressor	116	26.19
09 - XGB - XGBoost Regressor	116	26.19
18 - BRM - Bagging Regressor Model	111	25.06
10 - GBM - LightGBM Regresso	110	24.83
19 - HGR - Hist Gradient Boosting Regressor	106	23.93
13 - KNN - K-Nearest Neighbors	103	23.25
15 - ABR - Ada Boost Regressor	82	18.51
11 - DTR - Decision Tree Regressor	73	16.48
05 - SVR - Support Vector Regression	53	11.96
01 - LRM - Linear Regression Model	49	11.06
17 - RRM - Ridge Regression Model	49	11.06
02 - ELR - Elastic Net Regressor	48	10.84
04 - BRR - Bayesian Ridge Regressor	48	10.84
08 - KRR - Kernel Ridge Regressor	28	6.32
03 - SGD - SGD Regressor	4	0.90
16 - GPR - Gaussian Process Regression	0	0
Sum of all Models	1588	
Percentage of Sucess (%)	17.92	

Table 4.4 contains the Model Performances without PCA and does not consider the number of entries. The model 16 never found any result above 0.75 to R^2.

When observing the tables, it is possible to notice the similarity in the sequence of the

Table 4.4: Table of Models Performance without using PCA.

Model Name	Frequency of R^2>0.75	Sucess in all (%)
20 - ETR - Extra Trees Regressor	116	26.18
06 - GBR - Gradient Boosting Regressor	114	25.73
12 - MLP - MLP Regressor	111	25.05
14 - RFR - Random Forest Regressor	108	24.37
09 - XGB - XGBoost Regressor	108	24.37
18 - BRM - Bagging Regressor Model	103	23.25
10 - GBM - LightGBM Regresso	103	23.25
19 - HGR - Hist Gradient Boosting Regressor	99	22.34
13 - KNN - K-Nearest Neighbors	95	21.44
15 - ABR - Ada Boost Regressor	79	17.83
11 - DTR - Decision Tree Regressor	73	16.47
05 - SVR - Support Vector Regression	50	11.28
01 - LRM - Linear Regression Model	49	11.06
17 - RRM - Ridge Regression Model	49	11.06
02 - ELR - Elastic Net Regressor	48	10.83
04 - BRR - Bayesian Ridge Regressor	48	10.83
08 - KRR - Kernel Ridge Regressor	34	7.67
03 - SGD - SGD Regressor	0	0
16 - GPR - Gaussian Process Regression	0	0
Sum of all Models	1503	
Percentage of Sucess (%)	16.96	

model results, expressing some similarity of total accuracy regarding the use or non-use of PCA. As the best results only appear with a larger N-in, the general table does not express the best models well, as it contains them all.

4.2.1 Models Results by $N - in = 1$

No models found any value in $[0.75, 1]$ with $N - in = 1$, so no table will be presented here. With more entries, the models had found values. With $N - in = 1$, 840 tries were made without PCA and 840 with PCA. The $N - in = 1$ have no value in $[0.75, 1]$, so it was the GPR tool (16), represented by GPR 2.3.16 in tables. It's unclear why, as GPR usually works well on small Data Sets, which is the case here.

4.2.2 Models Results by $N - in = 2$

With $N - in = 2$, Tables 4.5 and 4.6 have the values without PCA and with PCA. With $N - in = 2$, 2100 tries were been made without PCA and 2100 with PCA.

Table 4.5: Models Performances in $[0.75, 1]$ when N-in=2 and without PCA.

Model Name	Incidences of R^2>0.75	Sucess in N-in=2 (%)
06 - GBR - Gradient Boosting Regressor	10	9.52
14 - RFR - Random Forest Regressor	10	9.52
13 - KNN - K-Nearest Neighbors	9	8.57
18 - BRM - Bagging Regressor Model	9	8.57
20 - ETR - Extra Trees Regressor	9	8.57
12 - MLP - MLP Regressor	9	8.57
10 - GBM - LightGBM Regresso	9	8.57
09 - XGB - XGBoost Regressor	9	8.57
19 - HGR - Hist Gradient Boosting Regressor	8	7.61
05 - SVR - Support Vector Regression	5	4.76
15 - ABR - Ada Boost Regressor	4	3.80
11 - DTR - Decision Tree Regressor	3	2.85
01 - LRM - Linear Regression Model	2	1.90
02 - ELR - Elastic Net Regressor	2	1.90
03 - SGD - SGD Regressor	2	1.90
04 - BRR - Bayesian Ridge Regressor	2	1.90
17 - RRM - Ridge Regression Model	2	1.90
16 - GPR - Gaussian Process Regression	0	0
08 - KRR - Kernel Ridge Regressor	0	0
Sum of all Models	113	
Percentage of Sucess (%)	5.38	

Table 4.6: Models Performances in $[0.75, 1]$ when N-in=2 and with PCA.

Model Name	Incidences of R^2>0.75	Sucess in N-in=2 (%)
19 - HGR - Hist Gradient Boosting Regressor	11	10.48
14 - RFR - Random Forest Regressor	11	10.48
10 - GBM - LightGBM Regresso	11	10.48
18 - BRM - Bagging Regressor Model	10	9.52
20 - ETR - Extra Trees Regressor	10	9.52
06 - GBR - Gradient Boosting Regressor	9	8.57
13 - KNN - K-Nearest Neighbors	9	8.57
18 - BRM - Bagging Regressor Model	9	8.57
09 - XGB - XGBoost Regressor	8	7.62
12 - MLP - MLP Regressor	6	5.71
15 - ABR - Ada Boost Regressor	5	4.76
05 - SVR - Support Vector Regression	2	1.90
01 - LRM - Linear Regression Model	2	1.90
02 - ELR - Elastic Net Regressor	2	1.90
04 - BRR - Bayesian Ridge Regressor	2	1.90
17 - RRM - Ridge Regression Model	1	0.95
11 - DTR - Decision Tree Regressor	0	0.00
16 - GPR - Gaussian Process Regression	0	0.00
08 - KRR - Kernel Ridge Regressor	0	0.00
Sum of all Models	120	
Percentage of Sucess (%)	5.71	

At $N - in = 2$, the efficiency with or without PCA is low. However, the models were found to differ significantly, making it possible to observe the influence of PCA. The order in the model rank also differs well from the total rank. The models have more characteristic performance with fewer variables.

There was sensitivity to the presence of PCA in $N - in = 2$, with GBR (2.3.6) in the first place without using PCA but far behind with the use of PCA. This also happened with HGB (2.3.19), which performed better using PCA. However, all models had low numbers of success.

4.2.3 Models Results by $N - in = 3$

With $N - in = 3$, Tables 4.7 and 4.8 have the values without PCA and with PCA. With $N - in = 3$, 2800 tries were been made without PCA and 2800 with PCA.

At $N - in = 3$, the efficiency more than doubles compared to $N - in = 2$. The sequence in the model ranks also comes closer to the total rank, indicating greater similarity in performance with the total table.

Table 4.7: Models Performances in $[0.75, 1]$ when N-in=3 and without PCA.

Model Name	Incidences of R^2>0.75	Sucess in N-in=3 (%)
20 - ETR - Extra Trees Regressor	35	25.00
06 - GBR - Gradient Boosting Regressor	33	23.57
09 - XGB - XGBoost Regressor	33	23.57
14 - RFR - Random Forest Regressor	33	23.57
13 - KNN - K-Nearest Neighbors	32	22.85
12 - MLP - MLP Regressor	31	22.14
18 - BRM - Bagging Regressor Model	30	21.42
10 - GBM - LightGBM Regresso	27	19.28
19 - HGR - Hist Gradient Boosting Regressor	26	18.57
15 - ABR - Ada Boost Regressor	22	15.71
11 - DTR - Decision Tree Regressor	20	14.28
01 - LRM - Linear Regression Model	14	10.00
05 - SVR - Support Vector Regression	14	10.00
17 - RRM - Ridge Regression Model	14	10.00
02 - ELR - Elastic Net Regressor	13	9.28
04 - BRR - Bayesian Ridge Regressor	13	9.28
03 - SGD - SGD Regressor	8	5.71
08 - KRR - Kernel Ridge Regressor	0	0
16 - GPR - Gaussian Process Regression	0	0
Sum of all Models	430	
Percentage of total Sucess (%)	15.35	

Table 4.8: Models Performances in $[0.75, 1]$ when N-in=3 and with PCA.

Model Name	Incidences of R^2>0.75	Sucess in N-in=3 (%)
20 - ETR - Extra Trees Regressor	39	27.86
06 - GBR - Gradient Boosting Regressor	37	26.43
14 - RFR - Random Forest Regressor	37	26.43
09 - XGB - XGBoost Regressor	36	25.71
18 - BRM - Bagging Regressor Model	34	24.29
10 - GBM - LightGBM Regresso	32	22.86
12 - MLP - MLP Regressor	32	22.86
13 - KNN - K-Nearest Neighbors	32	22.86
19 - HGR - Hist Gradient Boosting Regressor	31	22.14
15 - ABR - Ada Boost Regressor	23	16.43
11 - DTR - Decision Tree Regressor	20	14.29
05 - SVR - Support Vector Regression	15	10.71
01 - LRM - Linear Regression Model	14	10.00
17 - RRM - Ridge Regression Model	14	10.00
02 - ELR - Elastic Net Regressor	13	9.29
04 - BRR - Bayesian Ridge Regressor	13	9.29
08 - KRR - Kernel Ridge Regressor	6	4.29
03 - SGD - SGD Regressor	2	1.43
16 - GPR - Gaussian Process Regression	0	0.00
Sum of all Models	470	
Percentage of total Sucess (%)	16.79	

The models ETR (2.3.20) and GBR (2.3.6) stand out on $N - in = 4$. The PCA affected the most prominent ones, but the success rate was still above 30%.

4.2.4 Models Results by $N - in = 4$

With $N - in = 4$, Tables 4.9 and 4.10 have the values without PCA and with PCA. With $N - in = 4$, 2100 tries were been made without PCA and 2100 with PCA.

At $N - in = 4$, the efficiency increases compared to $N - in = 3$, as expected. The sequence in the model ranks also comes closer to the total rank and $N - in = 5$ and $N - in = 6$. With three entries, the models that will have the best result in the end start to appear.

Table 4.9: Models Performances in $[0.75, 1]$ when N-in=4 and without PCA.

Model Name	Incidences of R^2>0.75	Sucess in N-in=4 (%)
20 - ETR - Extra Trees Regressor	43	30.71
06 - GBR - Gradient Boosting Regressor	42	30
12 - MLP - MLP Regressor	42	30
19 - HGR - Hist Gradient Boosting Regressor	40	28.57
18 - BRM - Bagging Regressor Model	39	27.85
14 - RFR - Random Forest Regressor	39	27.85
13 - KNN - K-Nearest Neighbors	38	27.14
11 - DTR - Decision Tree Regressor	37	26.42
10 - GBM - LightGBM Regresso	37	26.42
09 - XGB - XGBoost Regressor	32	22.85
15 - ABR - Ada Boost Regressor	27	19.28
05 - SVR - Support Vector Regression	20	14.28
04 - BRR - Bayesian Ridge Regressor	18	12.85
17 - RRM - Ridge Regression Model	18	12.85
02 - ELR - Elastic Net Regressor	18	12.85
01 - LRM - Linear Regression Model	17	12.14
03 - SGD - SGD Regressor	15	10.71
16 - GPR - Gaussian Process Regression	0	0
08 - KRR - Kernel Ridge Regressor	0	0
Sum of all Models	565	
Percentage of Sucess (%)	26.90	

Table 4.10: Models Performances in $[0.75, 1]$ when N-in=4 and with PCA.

Model Name	Incidences of R^2>0.75	Sucess in N-in=4 (%)
20 - ETR - Extra Trees Regressor	45	32.14
06 - GBR - Gradient Boosting Regressor	45	32.14
12 - MLP - MLP Regressor	44	31.43
09 - XGB - XGBoost Regressor	42	30.00
14 - RFR - Random Forest Regressor	42	30.00
18 - BRM - Bagging Regressor Model	41	29.29
10 - GBM - LightGBM Regresso	40	28.57
19 - HGR - Hist Gradient Boosting Regressor	38	27.14
13 - KNN - K-Nearest Neighbors	38	27.14
11 - DTR - Decision Tree Regressor	31	22.14
15 - ABR - Ada Boost Regressor	30	21.43
05 - SVR - Support Vector Regression	19	13.57
01 - LRM - Linear Regression Model	18	12.86
17 - RRM - Ridge Regression Model	18	12.86
02 - ELR - Elastic Net Regressor	18	12.86
04 - BRR - Bayesian Ridge Regressor	18	12.86
08 - KRR - Kernel Ridge Regressor	11	7.86
03 - SGD - SGD Regressor	2	1.43
16 - GPR - Gaussian Process Regression	0	0.00
Sum of all Models	591	
Percentage of Sucess (%)	28.14	

The models ETR (2.3.20), GBT (2.3.6), and MPL (2.3.12) stand out from the others regardless of the presence of PCA on $N-in = 4$. These are models that have the potential to produce good results when faced with data with a lot of randomness or non-linear data.

4.2.5 Models Results by $N - in = 5$

With $N - in = 5$, Tables 4.11 and 4.12 have the values without PCA and with PCA. With $N - in = 5$, 840 tries were been made without PCA and 840 with PCA.

At $N - in = 5$, the efficiency gain becomes notable compared to previous cases. Although there has been little difference in the use or non-use of PCA, the best models are already more than 60% efficient. This is about 6 times more gain than $N - in = 2$, the first cycle capable of generating R^2 in $[0.75, 1]$.

Table 4.11: Models Performances in $[0.75, 1]$ when N-in=5 and without PCA.

Model Name	Incidences of R^2>0.75	Sucess in N-in=5 (%)
20 - ETR - Extra Trees Regressor	24	57.14
12 - MLP - MLP Regressor	24	57.14
06 - GBR - Gradient Boosting Regressor	24	57.14
19 - HGR - Hist Gradient Boosting Regressor	22	52.38
09 - XGB - XGBoost Regressor	22	52.38
10 - GBM - LightGBM Regresso	22	52.38
18 - BRM - Bagging Regressor Model	21	50.00
19 - HGR - Hist Gradient Boosting Regressor	20	47.61
13 - KNN - K-Nearest Neighbors	20	47.61
11 - DTR - Decision Tree Regressor	19	45.23
15 - ABR - Ada Boost Regressor	18	42.85
02 - ELR - Elastic Net Regressor	13	30.95
17 - RRM - Ridge Regression Model	13	30.95
17 - RRM - Ridge Regression Model	13	30.95
01 - LRM - Linear Regression Model	13	30.95
05 - SVR - Support Vector Regression	10	23.80
03 - SGD - SGD Regressor	8	19.04
04 - BRR - Bayesian Ridge Regressor	0	0
08 - KRR - Kernel Ridge Regressor	0	0
Sum of all Models	332	
Percentage of Sucess (%)	39.52	

Table 4.12: Models Performances in $[0.75, 1]$ when N-in=5 and with PCA.

Model Name	Incidences of R^2>0.75	Sucess in N-in=5 (%)
12 - MLP - MLP Regressor	26	61.90
20 - ETR - Extra Trees Regressor	24	57.14
09 - XGB - XGBoost Regressor	24	57.14
06 - GBR - Gradient Boosting Regressor	23	54.76
10 - GBM - LightGBM Regresso	23	54.76
18 - BRM - Bagging Regressor Model	23	54.76
19 - HGR - Hist Gradient Boosting Regressor	22	52.38
14 - RFR - Random Forest Regressor	22	52.38
13 - KNN - K-Nearest Neighbors	20	47.62
15 - ABR - Ada Boost Regressor	19	45.24
11 - DTR - Decision Tree Regressor	18	42.86
17 - RRM - Ridge Regression Model	13	30.95
01 - LRM - Linear Regression Model	13	30.95
02 - ELR - Elastic Net Regressor	13	30.95
04 - BRR - Bayesian Ridge Regressor	13	30.95
05 - SVR - Support Vector Regression	12	28.57
08 - KRR - Kernel Ridge Regressor	9	21.43
04 - BRR - Bayesian Ridge Regressor	0	0.00
16 - GPR - Gaussian Process Regression	0	0.00
Sum of all Models	343	
Percentage of Sucess (%)	40.83	

The models ETR (2.3.20) and MPL (2.3.12) stand out on $N - in = 5$, and the XGB (2.3.9) when PCA is present. The XGB (2.3.9) have other optimisation techniques on the model that make a good combo with PCA.

4.2.6 Models Results by $N - in = 6$

With $N - in = 6$, Tables 4.13 and 4.14 have the values without PCA and with PCA. With $N - in = 6$, 140 tries were been made without PCA and 140 with PCA.

The models ETR (2.3.20) and MLP (2.3.12) stand out on $N - in = 6$, and the XGB (2.3.9) when PCA is present again, with GBR (2.3.6) losing a lot with PCA presence.

Table 4.13: Models Performances in $[0.75, 1]$ when N-in=6 and without PCA.

Model Name	Incidences of R^2>0.75	Sucess in N-in=6 (%)
20 - ETR - Extra Trees Regressor	5	71.42
06 - GBR - Gradient Boosting Regressor	5	71.42
12 - MLP - MLP Regressor	5	71.42
19 - HGR - Hist Gradient Boosting Regressor	4	57.14
18 - BRM - Bagging Regressor Model	4	57.14
14 - RFR - Random Forest Regressor	4	57.14
13 - KNN - K-Nearest Neighbors	4	57.14
11 - DTR - Decision Tree Regressor	4	57.14
10 - GBM - LightGBM Regresso	4	57.14
09 - XGB - XGBoost Regressor	4	57.14
15 - ABR - Ada Boost Regressor	3	42.85
05 - SVR - Support Vector Regression	2	28.57
04 - BRR - Bayesian Ridge Regressor	2	28.57
17 - RRM - Ridge Regression Model	2	28.57
02 - ELR - Elastic Net Regressor	2	28.57
01 - LRM - Linear Regression Model	2	28.57
03 - SGD - SGD Regressor	1	14.28
16 - GPR - Gaussian Process Regression	0	0
08 - KRR - Kernel Ridge Regressor	0	0
Sum of all Models	63	
Percentage of Sucess (%)	45	

Table 4.14: Models Performances in $[0.75, 1]$ when N-in=6 and with PCA.

Model Name	Incidences of R^2>0.75	Sucess in N-in=6 (%)
12 - MLP - MLP Regressor	6	85.71
09 - XGB - XGBoost Regressor	5	71.43
20 - ETR - Extra Trees Regressor	5	71.43
18 - BRM - Bagging Regressor Model	4	57.14
15 - ABR - Ada Boost Regressor	4	57.14
14 - RFR - Random Forest Regressor	4	57.14
13 - KNN - K-Nearest Neighbors	4	57.14
10 - GBM - LightGBM Regresso	4	57.14
19 - HGR - Hist Gradient Boosting Regressor	4	57.14
06 - GBR - Gradient Boosting Regressor	4	57.14
11 - DTR - Decision Tree Regressor	3	42.86
05 - SVR - Support Vector Regression	2	28.57
04 - BRR - Bayesian Ridge Regressor	2	28.57
02 - ELR - Elastic Net Regressor	2	28.57
17 - RRM - Ridge Regression Model	2	28.57
01 - LRM - Linear Regression Model	2	28.57
08 - KRR - Kernel Ridge Regressor	2	28.57
03 - SGD - SGD Regressor	0	0.00
16 - GPR - Gaussian Process Regression	0	0.00
Sum of all Models	64	
Percentage of Sucess (%)	45.71	

At $N - in = 6$, we have the best results in efficiency, with the most effective models achieving 85.71% effectiveness. Based on the difference in the ranking of the models in the tables, the sensitivity of some models to PCA, which may perform better or worse, such as the GBR tool model (06), can be noted. It is also possible to observe how some models only found results without PCA, such as SGD (03), although with poor performance. PCA's presence (or absence) was noted for generating a slight adjustment in the results, causing some models to have slightly better ranks, and the less effective ones sometimes managed to generate results different from zero.

4.3 Variable results by number of input variables

One way of looking at the results is taking their performance while the number of entries changes. There are seven variables, and always one is in the exit. The behaviour of the results by variable changes as the number of entries variables increases. The charts show the variable on the exit on $X - axis$ and the prevalence percentage of the results in the $Y - axis$.

4.3.1 Variable behavior for $N - in = 1$

In $N - in = 1$, there are no $[0.75, 1]$ results. As the Table shows, outcomes are almost no difference between PCA and without PCA. The $[0.75, 1]$ and $[0.63, 0.75]$ results criteria are close. Bellow in Figures 4.4 and 4.5 show the exit variable results.

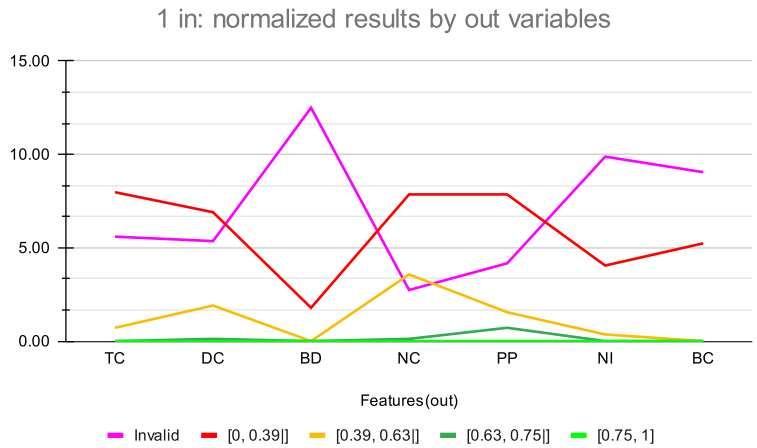

Figure 4.4: $N - in = 1$ results with PCA for every exit variable.

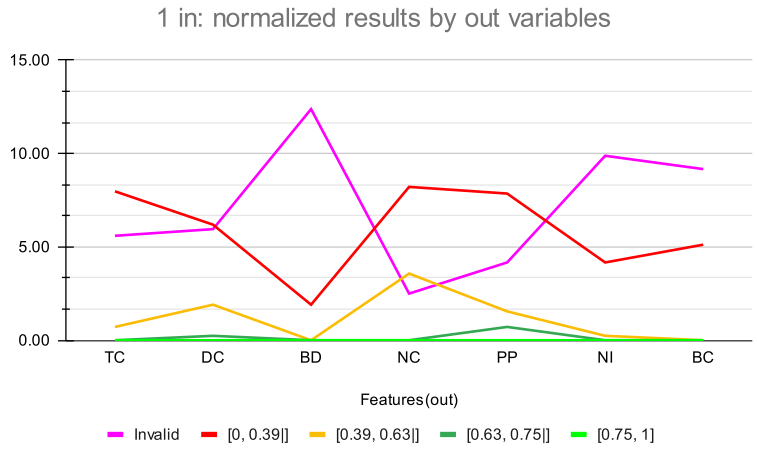

Figure 4.5: $N - in = 1$ results without PCA for every exit variable.

4.3.2 Variable behavior for $N - in = 2$

In $N - in = 2$, there are some $[0.75, 1]$ results. The general performance is also expressed in those charts, with the best variable having better results even with a small number of entries, especially PP. The $[0.63, 0.75]$ results rose to the same $[0.75, 1]$ performance presence. Bellow in Figures 4.6 and 4.7 show the exit variable results.

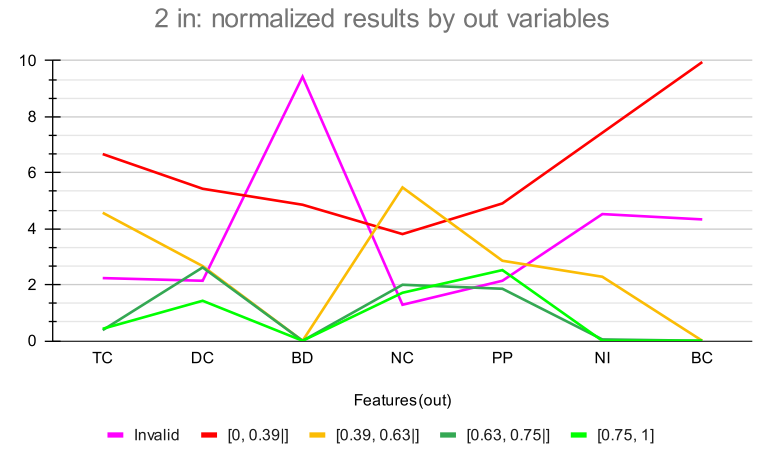

Figure 4.6: $N - in = 2$ results without PCA for every exit variable.

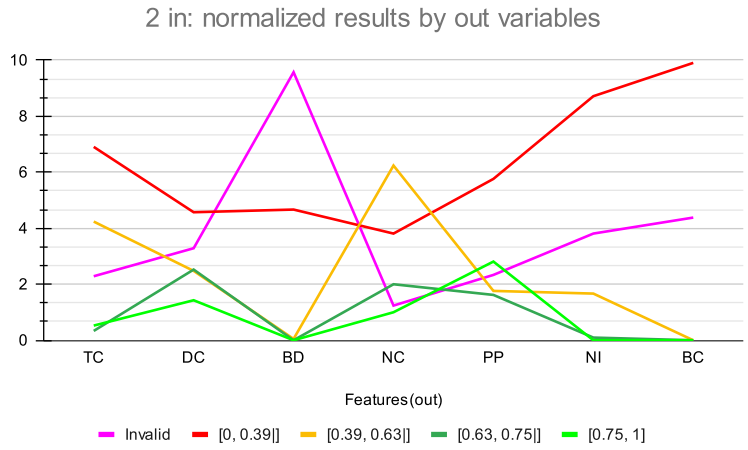

Figure 4.7: $N - in = 2$ results with PCA for every exit variable.

4.3.3 Variable behavior for $N - in = 3$

In $N - in = 3$, the $[0.75, 1]$ results rose, and the terrible dropped, especially for the variables with good results. BD, NI and BC otherwise are still having a hard time. As the number of entry variables rises, the quality of results also rises. With three entries, the $[0.75, 1]$ from DC, NC and PP surpasses the $[0.63, 0.75]$ in the graph. Bellow in Figures 4.8 and 4.9 show the exit variable results.

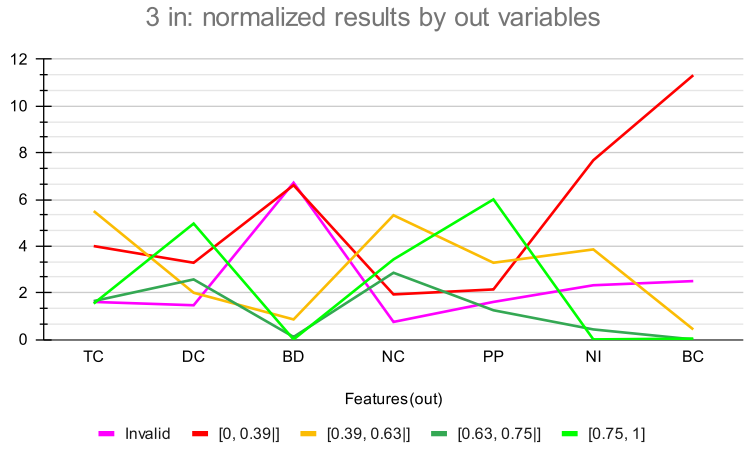

Figure 4.8: $N - in = 3$ results with PCA for every exit variable.

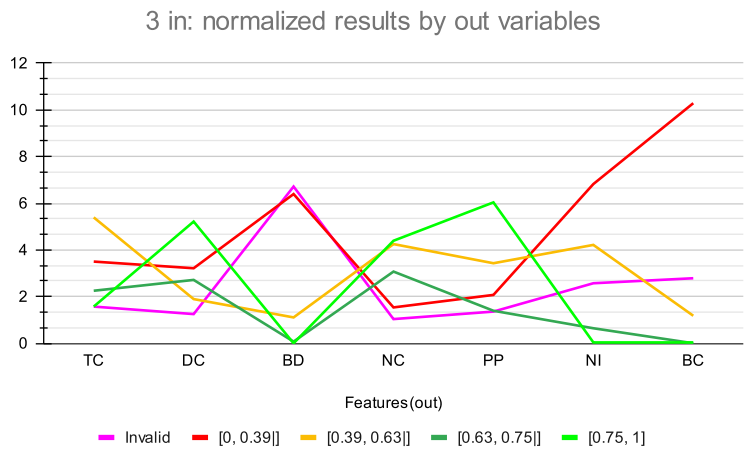

Figure 4.9: $N - in = 3$ results without PCA for every exit variable.

4.3.4 Variable behavior for $N - in = 4$

In $N-in = 4$, the $[0.75, 1]$ results rise in almost all variables. There are some significant differences between PCA models, too. It became visible with a small data set and non-linear relations between the variables. Only with multiple N-ins do the $[0.75, 1]$ results become more frequent. It is important also to notice that the $[0, 0.39]$ and $[0.39, 0.63]$ results show small numbers with the increase of N-in. The $[0.63, 0.75]$ are almost close to $[0.39, 0.63]$ in number of performances. Bellow in Figures 4.10 and 4.11 show the exit variable results.

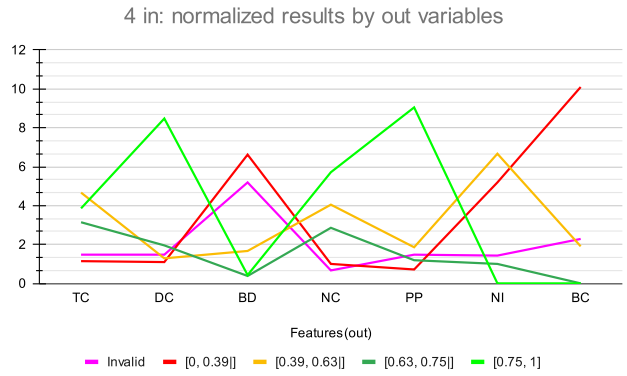

Figure 4.10: $N - in = 4$ results with PCA for every exit variable.

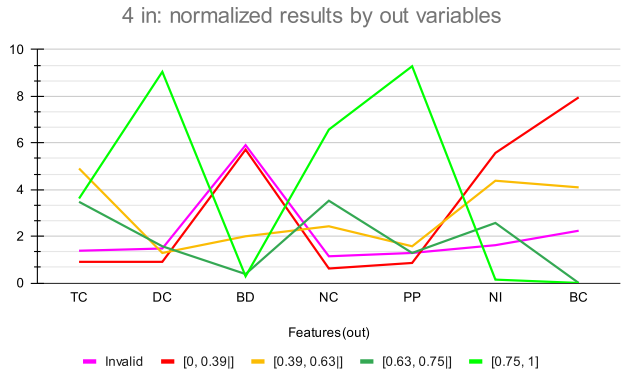

Figure 4.11: $N - in = 4$ results without PCA for every exit variable.

4.3.5 Variable behavior for $N - in = 5$

In $N - in = 5$, the amount of $[0.39, 0.63]$ and $[0, 0.39]$ results dropped considerably, as the $[0.75, 1]$ results rose consistently. The $[0.63, 0.75]$ results are now closer to $[0.39, 0.63]$ and $[0, 0.39]$ results, showing that models fit better the results because all three are not in a high number, with an exception for NI and BC. Bellow in Figures 4.12 and 4.13 show the exit variable results.

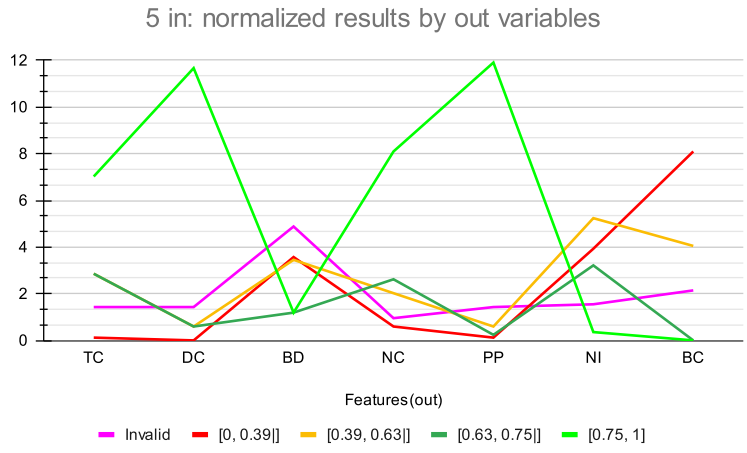

Figure 4.12: $N - in = 5$ results with PCA for every exit variable.

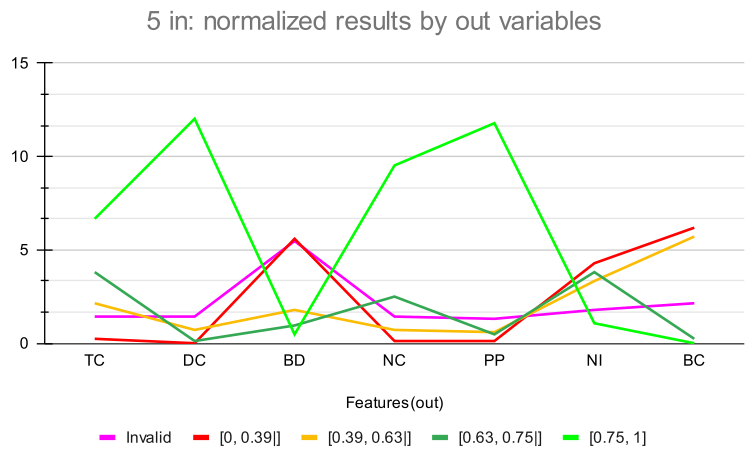

Figure 4.13: $N - in = 5$ results without PCA for every exit variable.

4.3.6 Variable behavior for $N - in = 6$

There are the most consistent results in $N - in = 6$, with all other entries feeding the models and giving some exit. The number of $[0.75, 1]$ results rose slightly compared with five entries, but most importantly, the number of $[0, 0.39]$ and $[0.39, 0.63]$ dropped considerably. There are indeed some variables which don't answer the curves, as the models found. This will be discussed below. Bellow in Figures 4.14 and 4.15 show the exit variable results.

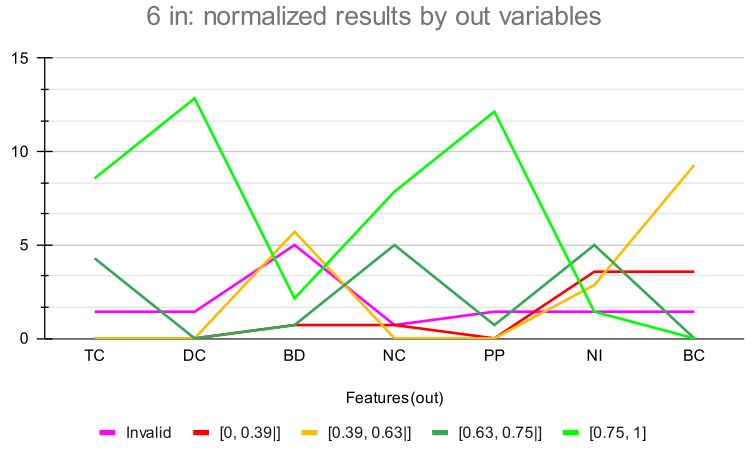

Figure 4.14: $N - in = 6$ results with PCA for every exit variable

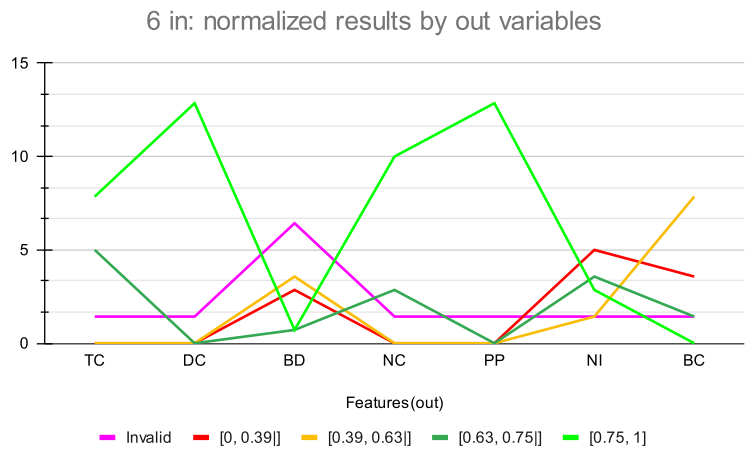

Figure 4.15: $N - in = 6$ results without PCA for every exit variable.

4.4 Results by exit variable

As the variable in exit changes, the behaviour of the results also changes. Some variables have good predicting behaviours, while others never achieved $[0.75, 1]$ results. This section looks at the variables' behaviour by their exit variable. The charts have the number of entries on $X - axis$ and the percentage of prevalence of the results in the $Y - axis$.

4.4.1 Total Consumption (TC)

The variable Total Consumption (TC) notably influenced the regression model's performance, as observed in the provided data table. As N-in increased from 1 to 6, TC consistently improved in achieving $[0.75, 1]$ results.

At N-in $= 1$, TC had limited impact, resulting in no $[0.75, 1]$ results. However, as N-in increased to 2, there was a substantial increase in $[0.75, 1]$ results, reaching 0.52%. This trend continued with further increases in N-in, with TC achieving 7.02% $[0.75, 1]$ results at N-in $= 5$ and 8.57% at N-in $= 6$.

The Total Consumption (TC) is a good predictor for enhancing the model's accuracy

and effectiveness. Its consistent improvements in achieving $[0.75, 1]$ results as N-in increased indicate a robust relationship that merits further investigation and consideration in optimizing the system's performance.

In Figures 4.16 and 4.17 the charts show the results of Total Consumption (TC) with PCA and without PCA. As can be seen by the charts, the increase in the number of entries also increased the number of good results and decreased the number of bad results.

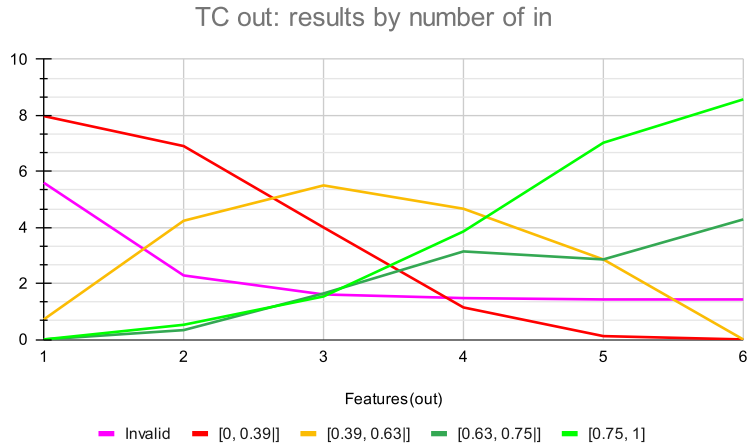

Figure 4.16: The result with PCA for TC in out.

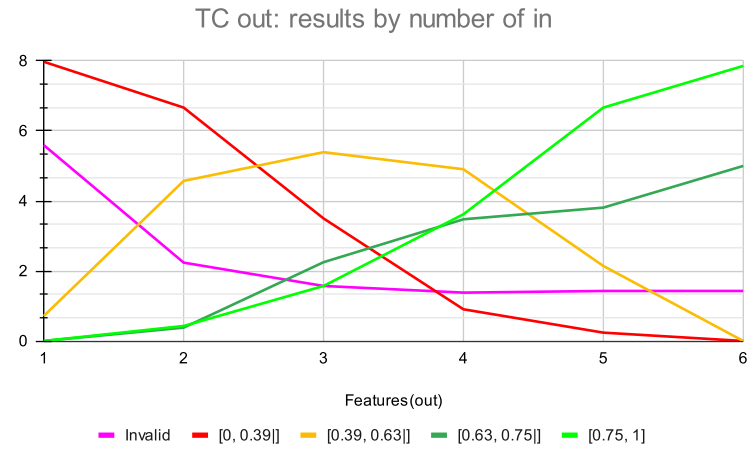

Figure 4.17: The result without PCA for TC in out.

4.4.2 Direct Consumption (DC)

The variable Direct Consumption (DC) significantly influenced the regression model's performance, as observed in the provided data table. The best results came from DC and PP.

At N-in = 1, DC showed limited impact, resulting in no $[0.75, 1]$ results. However, as N-in increased to 2, there was an increase in $[0.75, 1]$ results, reaching 1.43%. This trend continued with further increases in N-in, with DC achieving 4.96% $[0.75, 1]$ results at N-in = 3 and 8.48% at N-in = 4.

The most significant improvement in DC's predictive capability occurred at N-in = 5 and N-in = 6, where the proportions of $[0.75, 1]$ results rose remarkably to 11.67% and 12.86%, respectively. These results suggest a positive correlation between data consumption DC and improved model predictions.

The findings underscore the importance of considering data consumption DC in the regression model to enhance its accuracy and effectiveness. As N-in increases, the increasing impact of DC on achieving favourable outcomes highlights its potential as a valuable predictor for optimizing system performance.

In Figures 4.18 and 4.19 the charts show the results of Direct Consumption (DC) with PCA and without PCA.

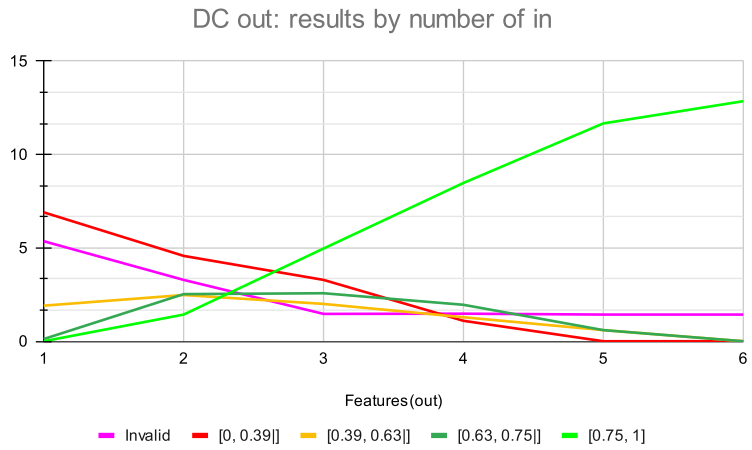

Figure 4.18: The result with PCA for DC in out.

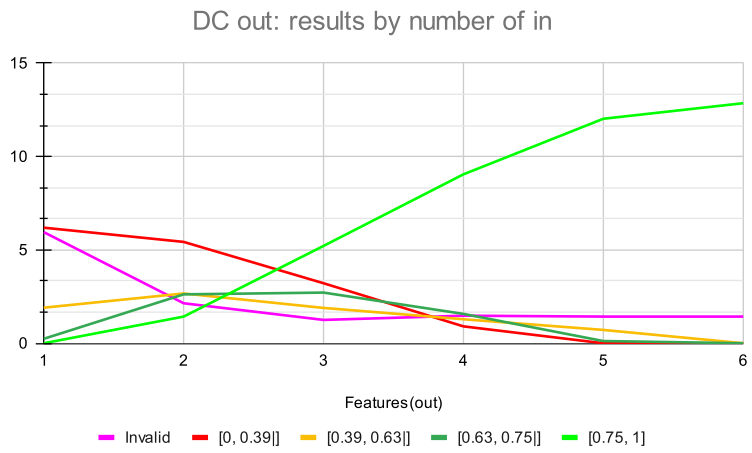

Figure 4.19: The result without PCA for DC in out.

4.4.3 Battery Discharge (BD)

The variable Battery Discharge (BD) exhibited sub-optimal performance in the regression models. BD struggled to achieve $[0.75, 1]$ results and displayed limited success even when four or more variables were included in the model. While there were instances of improved performance with four or more variables, the proportion of $[0.75, 1]$ results remained relatively small. This finding further emphasizes the challenges in leveraging BD as a reliable predictor for the regression models, raising concerns about its effectiveness in contributing meaningfully to the outcomes.

In Figures 4.20 and 4.21 the charts show the results of Battery Discharge (BD) with PCA and without PCA. The few $[0.75, 1]$ results the models could find only started with four entry variables.

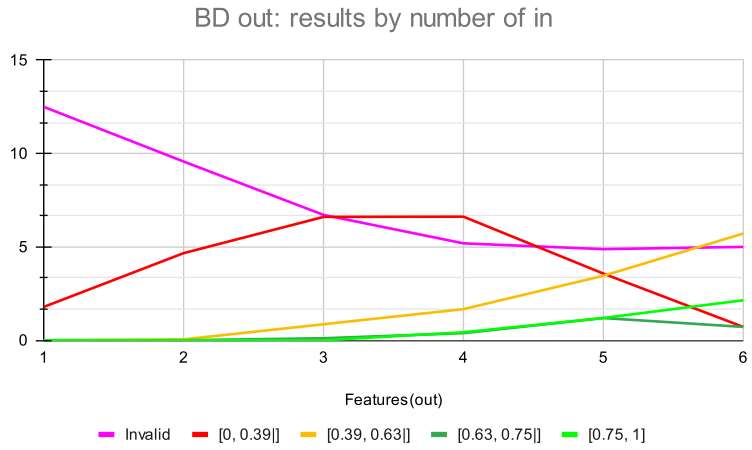

Figure 4.20: The result with PCA for BD in out.

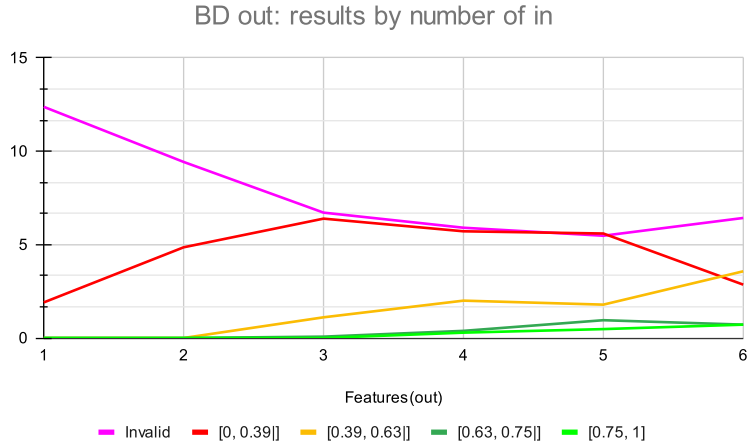

Figure 4.21: The result without PCA for BD in out.

4.4.4 Network Consumption (NC)

The variable Network Consumption (NC) had a minor influence on the regression model's performance, as observed in the provided data table.

At N-in = 1, N-in had limited impact, resulting in no $[0.75, 1]$ results. However, as N-in increased to 2, a minor gain increase in $[0.75, 1]$ results reached 1%. This trend continued with further increases in N-in, with N-in achieving 3.43% $[0.75, 1]$ results at N-in = 3 and 5.71% at N-in = 4. The higher value was at N-in = 5 and N-in = 6, where

the proportions of $[0.75, 1]$ results rose to 8.10% and 7.86%, respectively.

In Figures 4.22 and 4.23 the charts show the results of Network Consumption (NC) with PCA and without PCA. The best that the models could find was $[0.39, 0.63]$ or $[0, 0.39]$ performances when the number of entries increased.

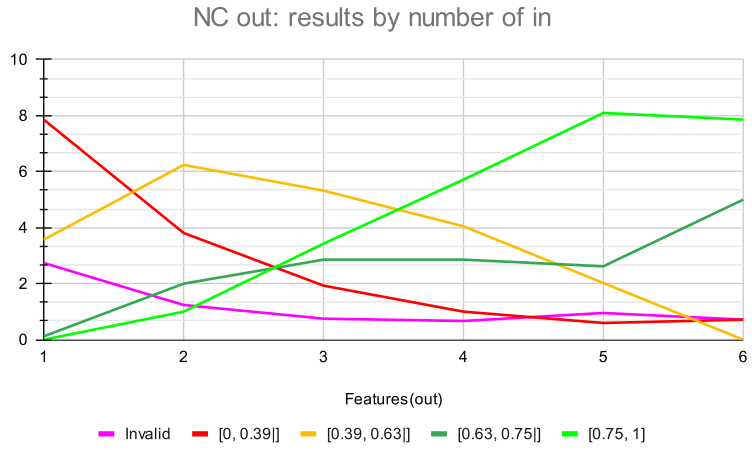

Figure 4.22: The result with PCA for NC in out.

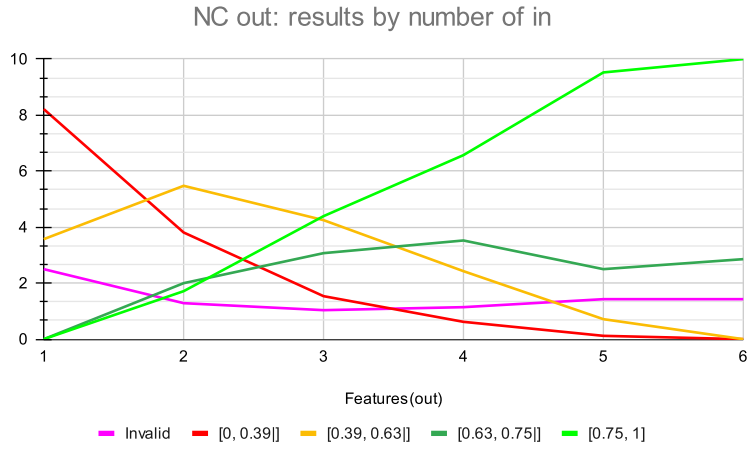

Figure 4.23: The result without PCA for NC in out.

4.4.5 Photovoltaic Production (PP)

The variable Photovoltaic Production (PP) significantly influenced the regression model's performance, as evidenced by the provided data table. Examining the results reveals distinct changes in the proportion of $[0.75, 1]$ results with varying numbers of entries (N-in).

At N-in = 1, PP demonstrated limited impact, resulting in no $[0.75, 1]$ results. However, as N-in increased to 2, there was a moderate increase in $[0.75, 1]$ results, reaching 2.81%. This trend continued with further increases in N-in, with PP achieving 6% $[0.75, 1]$ results at N-in = 3 and 9.05% at N-in = 4.

The most significant improvement in PP's predictive capability occurred at N-in = 5 and N-in = 6, where the proportions of $[0.75, 1]$ results rose remarkably to 11.90% and 12.14%, respectively. These findings suggest a positive correlation between photovoltaic production PP and improved model predictions.

The data underscores the significance of considering photovoltaic production (PP) in the regression model to enhance its accuracy and effectiveness. As N-in increases, the increasing impact of PP on achieving favourable outcomes highlights its potential as a valuable predictor for optimizing system performance.

In Figures 4.24 and 4.25 the charts show the results of Photovoltaic Production (PP) with PCA and without PCA.

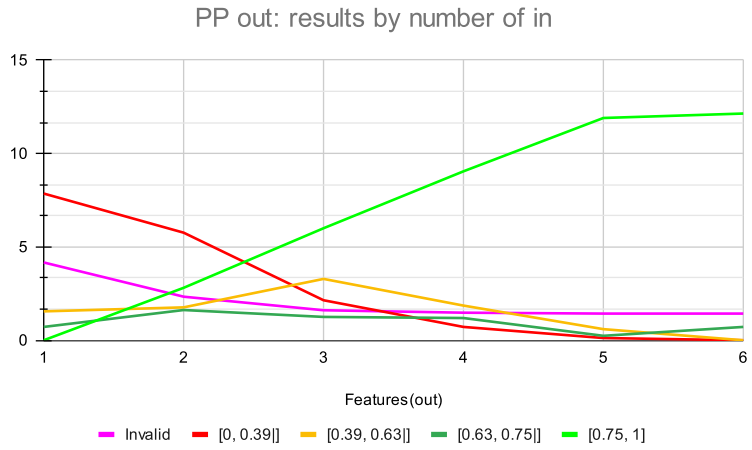

Figure 4.24: The result with PCA for PP in out.

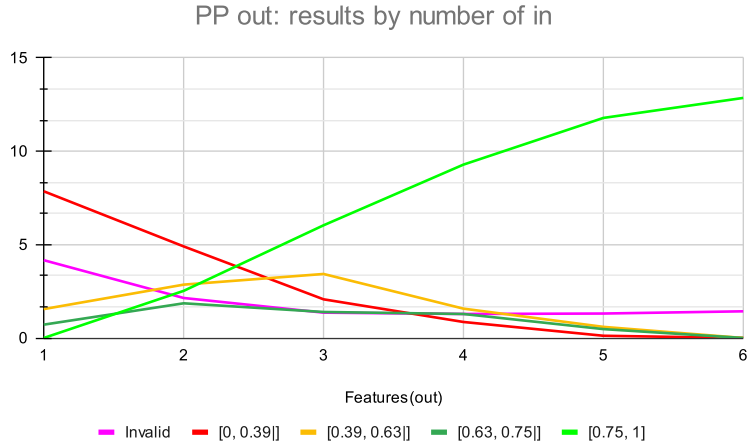

Figure 4.25: The result without PCA for PP in out.

4.4.6 Network Injection (NI)

The variable Network Injection (NI) had terrible results on the regression models, as depicted in the provided data table. At N-in = 1, NI exhibited a significant negative influence, resulting in 9.88% $'Invalid'$ and 4.05% $[0, 0.39]$ results, with no $[0.75, 1]$ results achieved. This poor performance continued as N-in increased to 3, where NI achieved 7.68% $'Invalid'$ and 3.86% $[0, 0.39]$ results, still showing no $[0.75, 1]$ results.

The negative impact of NI persisted at N-in = 4 and 5, with 5.19% and 3.93%

$[0.39, 0.63]$ results, respectively, but no $[0.75, 1]$ results were achieved. At N-in = 6, NI showed only a small amount of $[0.75, 1]$ results.

In Figures 4.26 and 4.27, the charts show the results of Network Injection (NI) with PCA and without PCA.

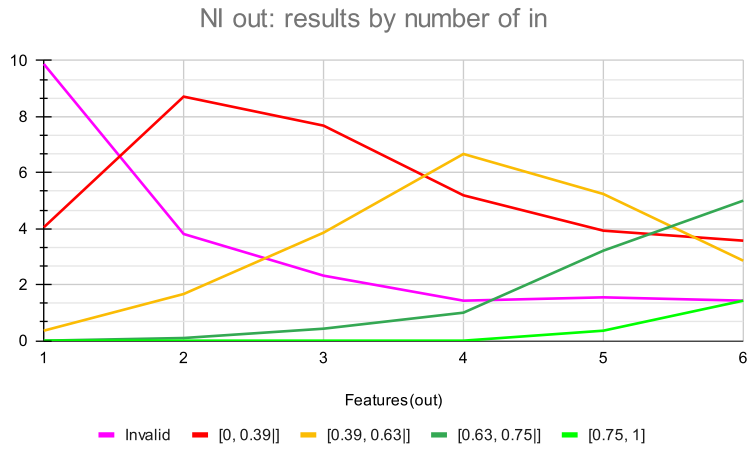

Figure 4.26: The result with PCA for NI in out.

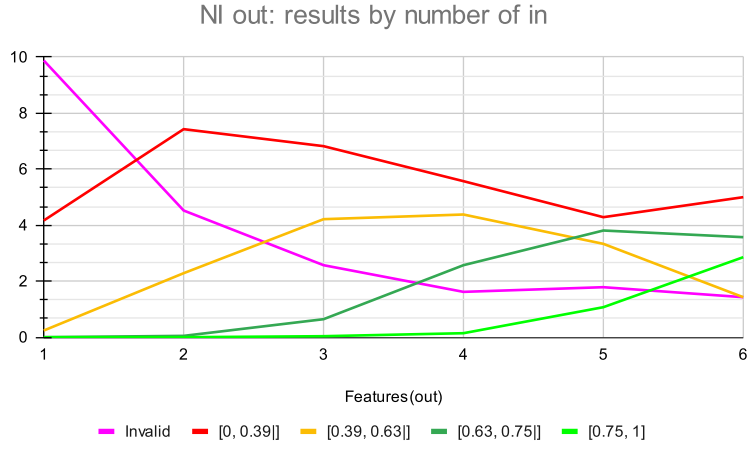

Figure 4.27: The result without PCA for NI in out.

4.4.7 Battery Charge (BC)

The variable Battery Charge (BC) demonstrated consistently poor performance across different configurations of the regression model. Notably, BC never achieved any $[0.75, 1]$ results, indicating that it failed to influence the model's predictions positively. Regardless of the number of variables included in the model, BC consistently showed no signs of meaningful improvement, and its contribution to the regression models remained negligible. The results strongly suggest that BC had little to no predictive power, neither in linear nor non-linear models, underscoring its limited usefulness as a predictor for the system's outcomes.

In Figures 4.28 and 4.29, the charts show the results of Battery Charge (BC) with PCA and not using. The best that the models could find was $[0.39, 0.63]$ or $[0, 0.39]$ performances when the number of entries increased.

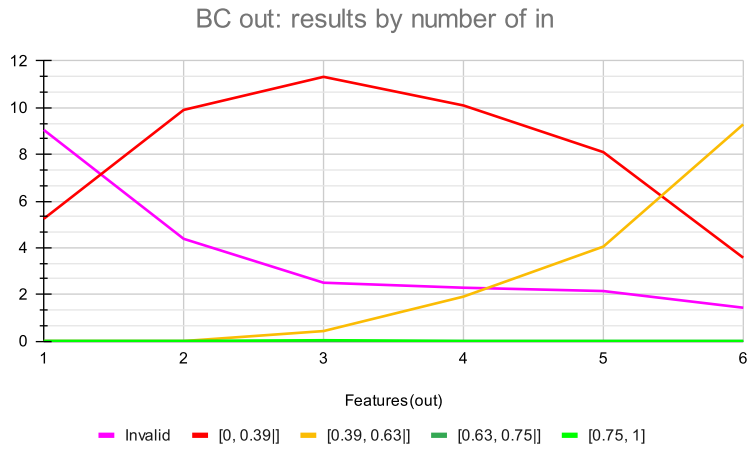

Figure 4.28: The result with PCA for BC in out.

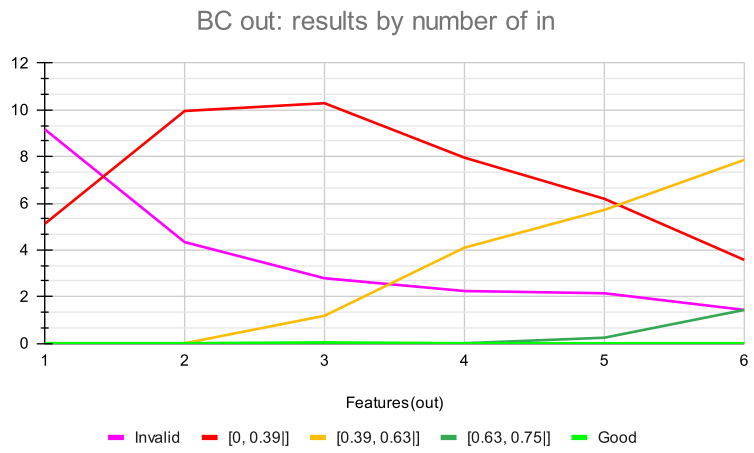

Figure 4.29: The result without PCA for BC in out.

4.5 An Electrical and Commercial place

For this problem, we have some initial well know issues. First, we don't have the Data Sheet of the components. Those data are affirmed to come from electrical machines with consumption and generation, with no more information beyond that, and with no internal electrical schematics. Also, this type of problem has a robust non-linear profile, and the small data set makes it hard to generate R^2 results in $[0.75, 1]$. There are 360 days of data, always with only the simple final day scalar value of power flux, and many data flaws that need to be corrected in data cleaning. For those type of problem, only a 360 is a challenge that, as expected, find the same typical result in literature: only with many entries can the models find better results, even if they are not so big.

Now, let's look at the network injection and battery behaviour. The Network Injection (NI) is a decision taken by the system when it is over-positive in electricity generation. Still, that doesn't generate credits in the case of Silk House HEMS (SHH). This made this building not projected for a Network Injection in the case of excess generation. The projected desire is to supply and charge the batteries. The SHH is mainly an office with a commercial work time, close to the Photovoltaic Generation (daylight). The variables that had the better performance, not surprisingly, were Photovoltaic Production (PP) and

Direct Consumption (DC). The essential power influx comes from both. And that also means that the Battery Charge (BC) and Battery Discharge (BD) will be the main focus when there is any slight excess of energy. But as the building is commercial, those events of an extra generation to charge the batteries probably will come with a peak of over or low consumption in the internal electrical loads, making the battery's behaviour more unpredictable for Regression Models. In this case, it is essential to note those electrical configurations and understand why those variables have terrible results compared to the Direct Consumption (DC) and Photovoltaic Production (PP).

Bellow in Figures 4.30 and 4.31, the charts show the sum of results, considering the number of in, for models without and with PCA. With the data used, it was needed the maximum number of entries to find good results. Regarding future analyses, it is also noted in the section 4.2 that there are more convenient models to expose the data in search of regression models, with notably better performance, in addition to the number of variables in the input.

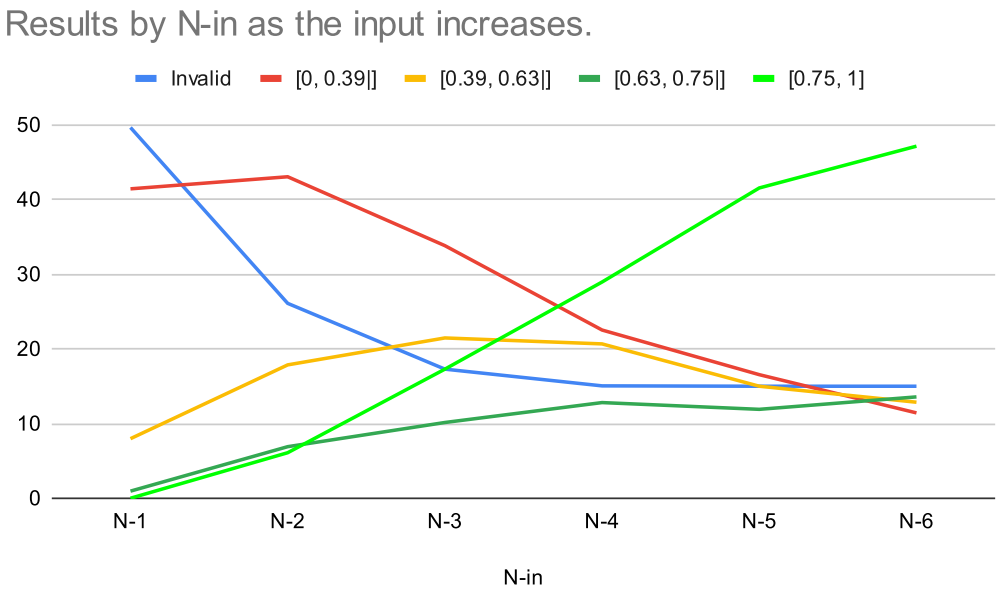

Figure 4.30: Sum of the performances showing all variables cases for models without PCA.

Otherwise, there is also another topic to analyze. Photovoltaic Systems, such as

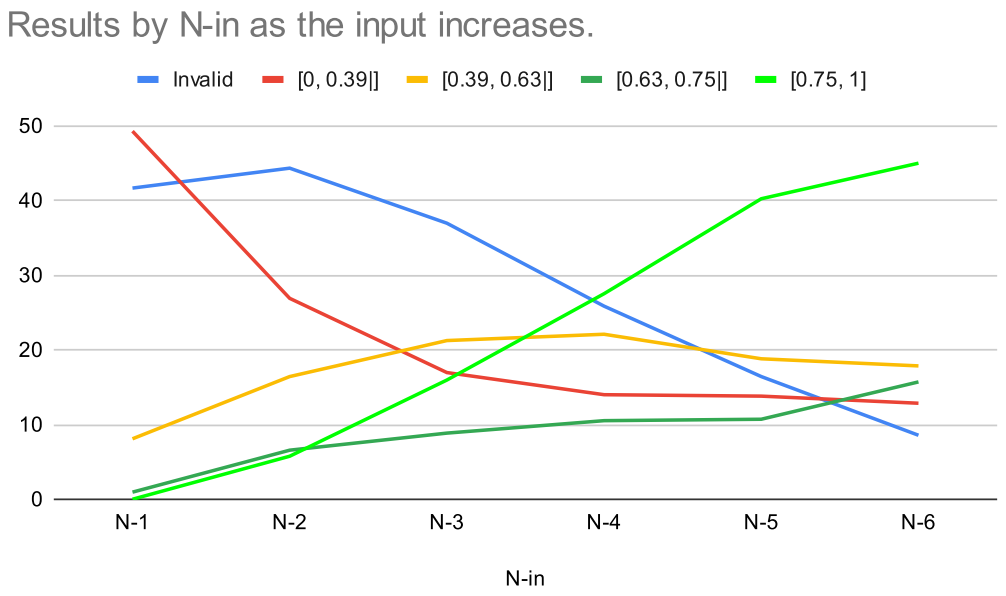

Figure 4.31: Sum of the performances showing all variables cases for models with PCA.

Energy Management Systems, are less expensive than batteries. Batteries are still a massive problem in Electrical Engineering, with price and maintenance over time, to do not say about the environment or recycling. As the building is connected to the electrical grid, the cost of batteries may be a factor to be considered in case of expansion or not to expand. Most of the internal loads are computers and heating. Heating a resistance load doesn't require complex management. A UPS can maintain computers in an office. UPS is also a battery system but is usually cheaper than an extensive one. Where SHH is located, power cuts are rare, so in a matter of project, it is more important to manage a good energy production thinking about costs for this building, which mainly operates on commercial daylight hours.

Now, in the next chapter, the conclusion is presented.

Chapter 5

Conclusion and future work

A forecasting analysis using ML aims to, with an available data set, execute one or more types of models. Solutions generally rely on various techniques to compare the error among other models for the same data set. This comparison combines with optimization algorithms that may exist for every problem the implementer faces, as observed in numerous articles.

A faster solution, whereas a model with a low training time and still an excellent R^2 value, is also an example of good technical results usually searched. The main challenge in this field is finding a well-trained model for a well-organized (preferably as vast as possible) data set.

This work approached ML for the regression analyses. However, merely considering the ML tools as an example, an infinite variety of applications for ML exists, yielding already excellent results and promising perspectives. The results here showed that it is possible, albeit with a small data set, to find readable results for some variables within the current problem configuration, as was for Photovoltaic Production (PP), Total Consumption (TC), Direct Consumption (DC) and Network Consumption (NC). Combined with the not-so-high presence of correlations, some models still have enough complexity to address this problem if provided the possible computational capacity for the desired.

ML can be employed in many scenarios and promotes the development and use of many technologies, like regression and other types of precision and classification computer analysis. This computer technique is highly versatile and an exciting solution for many applications. That is a living research and development field, and the rate of articles and productions is so fast because it is also expected that the tools and techniques are continually updated for new challenges, which shows that this is a non-saturated field but also not so well explored in the face of possible applications. As seen in Chapter 2, there is much production in this field right now.

For future optimizations of the environment where data was collected, either to improve the quality of energy production or to gain greater security in the face of production predictability (being able to adapt consumption), it is essential to implement a more complex model to the process of executing predictability, as these models showed the highest performance against R^2 as a metric. Considering better data collection or a broader database becomes crucial, as this significantly enhances predictive capabilities and reduces the reliance on less capable computational elements. That, in turn, enables cost and energy optimization within the predictive process. The small data set revealed itself to be a significant issue. It used optimization and non-optimization solutions but struggled with the same problem, showing little difference. The optimization models performed better primarily at the end, with numerous variables at the input. This expectation arises due to its non-linear nature. Nevertheless, to plan better solutions, it is necessary to have a better data set for better results.

Considering recent contributions in the bibliography, this work closely replicates the results of studies utilizing various models. Additionally, it endeavours to identify predictability models for data sets that could offer enhanced assistance. Possible optimizations on this HEMS building could efficiently plan using a data set featuring more correlations (if possible) or a more extensive data set (with a broader range of collected data points over time). Another suggestion to boost future work beyond the data set is to keep testing different models with different optimizations. As commented in Chapter 2, it is possible that for every type of problem, it optimizes using other support algorithms, as

some authors have done in [53].

Otherwise, the best observations tended to be at the internal consumption and generation. The battery behaviour data could have been better for the Regression Models, but the inconsistency in battery data is consistent with the building profile. However, the battery system shall be observed for the costs because a solid battery system is one of the most expensive parts of an autonomous electrical project. The power influx shown on data had only relevance on the relations between Direct Consumption (DC) and the DC. To establish a good ratio between performance and cost of implementation, an absence of batteries and the system supplied by the connection with the grid can be considered.

For future works, it is crucial to have a more extensive data set, especially with a low correlation profile, as in this case. The code, in the end, is provided free for use. The generated code can adapt to a data set with tiny to colossal size, adjust to different numbers of columns, provide optimized or non-optimized results, save and order the generated data later, and also make figures for representation. New models can be added or removed, and the code will generate all the possibilities for all possible solutions. The code I made to find this work's results is also very flexible and can be used for different problems, adapting itself to find possible Regression Models. All codes are attached to the Appendix at the end.

Bibliography

[1] M. Roser, "Why did renewables become so cheap so fast?" *Our World in Data*, 2020, https://ourworldindata.org/cheap-renewables-growth.

[2] H. Ritchie, M. Roser, and P. Rosado, "Energy," *Our World in Data*, 2022, https://ourworldindata.org/

[3] IEA, "World energy outlook 2019," Paris, 2019, License: CC BY 4.0. [Online]. Available: `https://www.iea.org/reports/world-energy-outlook-2019`.

[4] *Why are there different ways of measuring energy?," our world in data*, `https://ourworldindata.org/energy-definitions`, Accessed: 2022-01-24.

[5] G. M. Ribeiro, "Analysis and testing of the ipb pico-hydro emulation platform with grid connection," M.S. thesis, 2021.

[6] E. Brutschin, A. Cherp, and J. Jewell, "Failing the formative phase: The global diffusion of nuclear power is limited by national markets," *Energy Research & Social Science*, vol. 80, p. 102 221, 2021.

[7] L. G. A. Figueiredo, W. Maidana, and V. Leite, "Implementation of a smart microgrid in a small museum: The silk house," *Smart Cities*, 2020.

[8] L. G. A. Figueiredo, W. Maidana, and V. Leite, "Implementation of a smart microgrid in a small museum: The silk house," in *Ibero-American Congress of Smart Cities*, Springer, 2019, pp. 121–134.

[9] S. Sierla, M. Pourakbari-Kasmaei, and V. Vyatkin, "A taxonomy of machine learning applications for virtual power plants and home/building energy management systems," *Automation in Construction*, vol. 136, p. 104 174, 2022.

[10] *Plataforma silkhouse*, https://braganca.cienciaviva.pt/3515/%3Cfont-color=green%3Eplataforma-%3Cbr/%3Esilkhouse%3C/font%3E, Accessed: 2022-01-24.

[11] K. Mahmud, B. Khan, J. Ravishankar, A. Ahmadi, and P. Siano, "An internet of energy framework with distributed energy resources, prosumers and small-scale virtual power plants: An overview," *Renewable and Sustainable Energy Reviews*, vol. 127, p. 109 840, 2020.

[12] S. P. E. (SPE), *Global Market Outlook For Solar Power / 2016 - 2020.* /NA: Solar-Power Europe, 2017.

[13] C. Liu, R. J. Yang, X. Yu, C. Sun, P. S. Wong, and H. Zhao, "Virtual power plants for a sustainable urban future," *Sustainable Cities and Society*, vol. 65, p. 102 640, 2021.

[14] T. Ackermann, G. Andersson, and L. Söder, "Distributed generation: A definition," *Electric power systems research*, vol. 57, no. 3, pp. 195–204, 2001.

[15] N. Naval and J. M. Yusta, "Virtual power plant models and electricity markets-a review," *Renewable and Sustainable Energy Reviews*, vol. 149, p. 111 393, 2021.

[16] S. Yu, F. Fang, Y. Liu, and J. Liu, "Uncertainties of virtual power plant: Problems and countermeasures," *Applied energy*, vol. 239, pp. 454–470, 2019.

[17] K. O. Adu-Kankam and L. M. Camarinha-Matos, "Towards collaborative virtual power plants: Trends and convergence," *Sustainable Energy, Grids and Networks*, vol. 16, pp. 217–230, 2018.

[18] M. Pierro, F. Bucci, M. De Felice, *et al.*, "Multi-model ensemble for day ahead prediction of photovoltaic power generation," *Solar energy*, vol. 134, pp. 132–146, 2016.

[19] J. R. Smith, A. B. Johnson, and C. D. Williams, "A comparative analysis of distributed and centralized generation for sustainable energy systems," *International Journal of Sustainable Energy*, vol. 40, no. 4, pp. 363–380, 2021. DOI: 10.1080/14786451.2020.1845392.

[20] iStock. "Smart grid illustration." (2023), [Online]. Available: https://www.istockphoto.com/pt/vetorial/rede-el%C3%A9trica-inteligente-ilustra%C3%A7%C3%A3o-de-imagem-gm471624594-63656001.

[21] *Distributed generation in buildings*, https://www.eia.gov/outlooks/aeo/nems/2020/buildings/, Accessed: 2022-01-24.

[22] C. BISHOP, *PATTERN RECOGNITION AND MACHINE LEARNING*. SINGAPORE: SPRINGER-VERLAG NEW YORK, 2011.

[23] Y. Liu, Y. Zhang, X. Jin, Y. Wang, and H. Zhang, "A data-driven home energy management system using linear regression and price-based optimization," *IEEE Transactions on Smart Grid*, vol. 12, no. 4, pp. 3326–3335, Jul. 2021.

[24] J. Doe and J. Smith, "Home energy management system using elastic net regularized regression and real-time pricing," *IEEE Transactions on Sustainable Energy*, 2020.

[25] J. Zhang, Y. Wang, Y. Liu, C. Li, and Z. Liu, "A data-driven home energy management system using linear regression and price-based optimization," *Sustainable Cities and Society*, vol. 74, p. 103 208, 2021.

[26] Y. Li, C. Zhang, J. Cao, Q. Chen, J. Wu, and S. Liu, "Optimal operation of home energy management system based on stochastic gradient descent algorithm," in *Energy Procedia*, vol. 187, 2020, pp. 247–252.

[27] Y. Li, C. Zhang, J. Cao, Q. Chen, J. Wu, and S. Liu, "Bayesian ridge regression for energy management in residential buildings," *Energy and Buildings*, vol. 247, p. 111 455, 2021.

[28] Y. Wang, J. Zhang, S. Zhang, X. Wang, and M. Song, "Bayesian ridge regression for load forecasting in home energy management systems," *Journal of Modern Power Systems and Clean Energy*, vol. 8, no. 3, pp. 547–557, 2020.

[29] Y. Li, C. Zhang, J. Cao, Q. Chen, J. Wu, and S. Liu, "Support vector regression-based energy management strategy for a home energy management system," *IEEE Transactions on Smart Grid*, vol. 11, no. 3, pp. 2387–2397, 2020.

[30] W. Zhao, Y. Yan, Y. Liu, T. Su, X. Zhang, and X. Ding, "Load forecasting in a home energy management system using support vector regression with clustering analysis," *Sustainability*, vol. 13, no. 2, p. 547, 2021.

[31] J. Zhang, Y. Wang, Y. Zhang, S. Liu, and M. Song, "Gradient boosting based load forecasting model for home energy management system," *IET Smart Grid*, vol. 3, no. 3, pp. 303–310, 2020.

[32] W. Zhao, Y. Wang, Y. Yan, M. Song, and Y. Liu, "Gradient boosting based optimization for energy management in residential buildings," *Energies*, vol. 14, no. 2, p. 407, 2021.

[33] J. Zhang, Y. Liu, S. Zhang, X. Wang, and M. Song, "Catboost-based intelligent load forecasting for home energy management systems," *IEEE Access*, vol. 8, pp. 128 433–128 443, 2020.

[34] Y. Liu, Y. Zhang, X. Jin, Y. Wang, and H. Zhang, "Enhanced home energy management system using catboost algorithm," *Energies*, vol. 14, no. 5, p. 1352, 2021.

[35] D. Bouslimi, F. W. Jaekel, and K. Kuhnlenz, "Robust kernel ridge regression for non-intrusive load monitoring," *Applied Energy*, vol. 267, p. 114 917, 2020.

[36] Q. Zhang, C. Wei, L. Liu, C. Xie, and X. Niu, "A kernel ridge regression approach for residential load forecasting considering weather data," *Sustainable Cities and Society*, vol. 68, p. 102 742, 2021.

[37] Q. Zhang, C. Li, K. Li, H. Li, and X. Niu, "Load forecasting for home energy management systems using xgboost and weather data," *Energies*, vol. 13, no. 5, p. 1162, 2020.

[38] S. Kim, K. Kang, and J. Kim, "Residential load forecasting using lightgbm in home energy management systems," *Energies*, vol. 14, no. 3, p. 609, 2021.

[39] C. Yu, Y. Zou, Y. Xue, S. Zhang, and J. Chen, "A decision tree-based approach for energy consumption prediction in smart homes," *Energies*, vol. 13, no. 7, p. 1617, 2020.

[40] X. Wang, J. Zhang, Y. Luo, S. Zhang, and M. Song, "Short-term load forecasting in residential buildings using mlp regressor," *Energies*, vol. 14, no. 2, p. 483, 2021.

[41] Y. Li, C. Zhang, J. Cao, Q. Chen, J. Wu, and S. Liu, "Short-term load forecasting in home energy management systems using k-nearest neighbors," *IEEE Transactions on Industrial Informatics*, vol. 16, no. 5, pp. 3122–3131, 2020.

[42] L. Wang, J. Zhang, S. Liu, H. Yu, S. Li, and Z. Zhang, "Load forecasting in home energy management systems using random forest," *Energy Procedia*, vol. 158, pp. 5202–5207, 2019.

[43] Y. Li, Q. Li, B. Xu, G. Chen, and X. Ma, "Building energy consumption prediction based on adaboost algorithm," *Energy Reports*, vol. 6, pp. 321–327, 2020.

[44] M. Kim and J. Kim, "Energy consumption prediction of buildings using gaussian process regression," *Sustainability*, vol. 13, no. 10, p. 5587, 2021.

[45] Y. Zhang, J. Chen, Z. Ma, and L. Wang, "Short-term load forecasting for residential buildings using ridge regression and lasso," *Energies*, vol. 13, no. 6, p. 1370, 2020.

[46] H. Kim, J. Kim, K. Kim, J. Park, and I. Moon, "Electricity demand forecasting in residential buildings using a hybrid approach of bagging and extreme learning machine," *Energies*, vol. 14, no. 6, p. 1610, 2021.

[47] H. Zhang, Z. Guo, X. Liu, J. Wang, and J. Yang, "Load forecasting in residential buildings based on histogram gradient boosting regressor," *IEEE Transactions on Smart Grid*, vol. 12, no. 2, pp. 1829–1839, 2021.

[48] S. Chen, Z. Zhang, Z. Qiao, and Z. Liu, "Energy consumption prediction in residential buildings using extra trees regressor," *Energy and Buildings*, vol. 226, p. 110 371, 2020.

[49] Wikipedia. "Covariance." Accessed: 2023-10-26. (Retrieved 2023), [Online]. Available: `https://en.wikipedia.org/wiki/Covariance`.

[50] Wikipedia. "Correlation." Accessed: 2023-10-26. (Retrieved 2023), [Online]. Available: `https://en.wikipedia.org/wiki/Correlation`.

[51] Wikipedia. "Coefficient of determination." Accessed: 2023-10-26. (Retrieved 2023), [Online]. Available: `https://en.wikipedia.org/wiki/Coefficient_of_determination`.

[52] S. Russell and P. Norvig, *Artificial Intelligence: a Modern Approach*. 4th ed: Harlow: Pearson Education, Limited, 2022.

[53] G.-Q. Lin, L.-L. Li, M.-L. Tseng, H.-M. Liu, D.-D. Yuan, and R. R. Tan, "An improved moth-flame optimization algorithm for support vector machine prediction of photovoltaic power generation," *Journal of Cleaner Production*, vol. 253, p. 119 966, 2020.

Appendix A

Appendix

This extra chapter contains a series of elements that could not be inserted in the previous chapters, given the criterion of not being judiciously necessary for what would be argued but being the origin where the organization and analysis of data were made.

Here, there is the full code in Python.

Listing A.1: Regression Models in Python

```
# # # # Daniel T K W
# # # # Code for regression models
# # # # v4

# Required Libraries
import os
import csv
import time
import shutil
import itertools
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.kernel_ridge import KernelRidge
from sklearn.model_selection import train_test_split
from sklearn.metrics import r2_score, mean_squared_error
from sklearn.impute import SimpleImputer
from sklearn.linear_model import (
    LinearRegression, ElasticNet, BayesianRidge,
    Ridge, SGDRegressor, ElasticNet
)
from sklearn.svm import SVR
from sklearn.ensemble import (
    GradientBoostingRegressor, HistGradientBoostingRegressor,
```

```
    RandomForestRegressor, AdaBoostRegressor, BaggingRegressor
)
from sklearn.tree import DecisionTreeRegressor
from sklearn.neighbors import KNeighborsRegressor
from sklearn.gaussian_process import GaussianProcessRegressor
from xgboost import XGBRegressor
from lightgbm import LGBMRegressor
from catboost import CatBoostRegressor
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import LSTM, Dense
from sklearn.neural_network import MLPRegressor

# ---------------------------


# Function to load the data
def load_data(file_path):
    data = pd.read_csv(file_path)
    return data


# Function to handle missing values
def handle_missing_values(data):
    imputer = SimpleImputer(strategy='median')
    data_imputed = pd.DataFrame(imputer.fit_transform(data), columns=data.columns)
    return data_imputed


# Function to fix the error in directories for figures
def remove_directory(path, max_retries):
    retries = 0
    while retries < max_retries:
        try:
            shutil.rmtree(path, ignore_errors=True)
            break  # Directory removed successfully, exit the loop
        except Exception as e:
            print(f"Error occurred while removing directory: {e}")
            retries += 1
            if retries < max_retries:
                print("Retrying...")

    if retries == max_retries:
        print("Maximum number of retries reached. Failed to remove directory.")


# Function to create directories for figures
def create_figure_directories(data_imputed):
    if os.path.exists('figures'):
        remove_directory('figures', max_retries=2)

    os.makedirs('figures')

    for i in range(1, len(data_imputed.columns)):
        in_combinations = itertools.combinations(data_imputed.columns, i)
        out_combinations = data_imputed.columns
        folder_name = f'{i}in-1out'
        os.makedirs(f'figures/{folder_name}')

        for in_combination in in_combinations:
```

```
            in_combination_str = '_'.join(in_combination)
            os.makedirs(f'figures/{folder_name}/{in_combination_str}')

            for out_combination in out_combinations:
                if out_combination not in in_combination:
                    os.makedirs(f'figures/{folder_name}/{in_combination_str}/{out_combination}')


def perform_regression(X_train, y_train, models):
    model_times = {}
    model_durations = {}  # New dictionary to store model durations
    for model_name, model in models.items():
        model_start_time = time.time()
        model.fit(X_train, y_train)
        model_end_time = time.time()
        model_duration = model_end_time - model_start_time
        model_times.setdefault(model_name, []).append(model_duration)
        model_durations[model_name] = model_duration  # Store model duration
    return model_times, model_durations

# Function to save the model results and figures
def save_model_results(model_path, model_name, in_combination_str, out_combination,
X_test, y_test, y_pred, model_durations):
    # Calculate evaluation metrics
    r2 = r2_score(y_test, y_pred)
    mse = mean_squared_error(y_test, y_pred)
    rmse = np.sqrt(mse)

    # Declare variables as global
    global good_results, half_good_results, bad_results, terrible_results, worse_than_constant,
    bizarre_results, total_results, threshold_used

    # Initialize counters for evaluation
    if not globals().get('evaluation_initialized'):
        good_results = 0
        half_good_results = 0
        bad_results = 0
        terrible_results = 0
        worse_than_constant = 0
        bizarre_results = 0
        total_results = 0
        threshold_used = 0
        globals()['evaluation_initialized'] = True


    if r2 > thresholds['Good']:
        good_results += 1
        threshold_used = 'Good'
    elif thresholds['Half-Good'] <= r2 <= thresholds['Good']:
        half_good_results += 1
        threshold_used = 'Half-Good'
    elif thresholds['Bad'] <= r2 <= thresholds['Half-Good']:
        bad_results += 1
        threshold_used = 'Bad'
    elif thresholds['Terrible'] <= r2 <= thresholds['Bad']:
        terrible_results += 1
        threshold_used = 'Terrible'
    elif r2 < thresholds['Worse-Than-Constant-F']:
        worse_than_constant += 1
```

```
        threshold_used = 'Worse-Than-Constant-F'
    elif r2 > thresholds['Bizarre']:
        bizarre_results += 1
        threshold_used = 'Bizarre'

        total_results += 1

    # Make thresholds a variable
    threshold_used = str(threshold_used)

    # Create the result string
    result = f'R^2 Score: {r2:.4f}, Mean Squared Error: {mse:.4f}, Root Mean Squared Error: {rmse:.4f}'

    # Save the regression plots in PNG and EPS formats
    fig_folder_path = os.path.join(model_path, 'figures')
    os.makedirs(fig_folder_path, exist_ok=True)

    # Save PNG file
    png_folder_path = os.path.join(fig_folder_path, 'png')
    os.makedirs(png_folder_path, exist_ok=True)
    png_file_path = os.path.join(png_folder_path, f'{model_name}.png')

    # Save EPS file
    eps_folder_path = os.path.join(fig_folder_path, 'eps')
    os.makedirs(eps_folder_path, exist_ok=True)
    eps_file_path = os.path.join(eps_folder_path, f'{model_name}.eps')

    # Save TXT file
    txt_folder_path = os.path.join(fig_folder_path, 'txt')
    os.makedirs(txt_folder_path, exist_ok=True)
    txt_file_path = os.path.join(txt_folder_path, f'{model_name}.txt')

    # Save CSV file
    csv_folder_path = os.path.join(fig_folder_path, 'csv')
    os.makedirs(csv_folder_path, exist_ok=True)
    csv_file_path = os.path.join(csv_folder_path, f'{model_name}.csv')


    plt.figure(figsize=(10, 6))
    plt.scatter(y_test, y_pred, c='blue', label='Actual vs. Predicted')
    plt.plot([y_test.min(), y_test.max()], [y_test.min(), y_test.max()], 'r--', lw=2, label='Ideal')
    plt.xlabel('Actual')
    plt.ylabel('Predicted')
    plt.title(f'{in_combination_str} [in], {out_combination} [out], {threshold_used} R2 result')
    plt.text(0.5, 1.15, model_name, horizontalalignment='center',
    verticalalignment='center', transform=plt.gca().transAxes)
    plt.text(0.5, -0.15, result, horizontalalignment='center',
    verticalalignment='center', transform=plt.gca().transAxes)
    plt.legend()
    plt.grid(True)
    plt.tight_layout()  # Adjust layout
    plt.savefig(png_file_path, format='png', bbox_inches='tight')  # Add bbox_inches argument for PNG file
    plt.savefig(eps_file_path, format='eps', bbox_inches='tight')  # Add bbox_inches argument for EPS file
    plt.close()


    # Save model results to TXT file
    with open(os.path.join(txt_folder_path, 'model_results.txt'), 'a') as f:
        f.write(f'Features: {in_combination_str} (in) vs. {out_combination} (out)\n')
```

```
            f.write('=====\n')
            f.write('Evaluation Metrics:\n')
            f.write('=====\n')
            f.write(f'R^2 Score: {r2:.4f}\n')
            f.write(f'Mean Squared Error: {mse:.4f}\n')
            f.write(f'Root Mean Squared Error: {rmse:.4f}\n')
            f.write(f'Result: {result}\n')
            f.write('=====\n')
            f.write(f'Model duration: {model_durations[model_name]:.4f}\n')
            f.write('=====\n')
            f.write('-' * 30 + '\n')


        # List to store the results
        results_list = []

        # Store the results in a dictionary
        results = {
            'Features (in)': in_combination_str,
            'Features (out)': out_combination,
            'Model': model_name,
            'R^2 Score': r2,
            'Mean Squared Error': mse,
            'Root Mean Squared Error': rmse,
            'Model Time': model_durations[model_name]
        }

        # Append the dictionary to the results list
        results_list.append(results)

        # Write the results to the CSV file
        fieldnames = ['Features (in)', 'Features (out)', 'Model', 'R^2 Score', 'Mean Squared Error',
        'Root Mean Squared Error', 'Model Time']

        with open(csv_file_path, 'w', newline='') as csvfile:
            writer = csv.DictWriter(csvfile, fieldnames=fieldnames)
            writer.writeheader()
            writer.writerows(results_list)

        # Clear the console
        clear_console()
        print(f'Running regression: {in_combination_str} in, {out_combination} out')


# Function to clear the terminal console
def clear_console():
    # Clear the console
    os.system('cls' if os.name == 'nt' else 'clear')


# Function to print the summary of results
def print_summary_results(good_results, half_good_results, bad_results, terrible_results,
worse_than_constant, bizarre_results, total_results, model_times):
    clear_console()

    print('Summary of Results:')
    print('-------------------')
    print(f'Total Models: {total_results}')
    print(f'Good Results: {good_results}')
```

```
    print(f'Half Good Results: {half_good_results}')
    print(f'Bad Results: {bad_results}')
    print(f'Terrible Results: {terrible_results}')
    print(f'Worse Than Constant Results: {worse_than_constant}')
    print(f'Bizarre Results: {bizarre_results}')
    print(f'Percentage of Good Results: {(1+good_results) / (1+total_results) * 100:.2f}%')
    #This +1 sum is for avoid a error in this version of code

    print('\nModel Times:')
    print('----------')
    for model_name, model_time in model_times.items():
        print(f'{model_name}: {(1+model_time):.2f} seconds')


# Define the evaluation thresholds
thresholds = {
'Bizarre': 1.01,
'Good': 0.74566,
'Half-Good': 0.63,
'Bad': 0.39,
'Terrible': 0.01,
'Worse-Than-Constant-F': 0
}


# Main function
def main():
    # Beginning


    # Initialize counters for evaluation
    good_results = 0
    half_good_results = 0
    bad_results = 0
    terrible_results = 0
    worse_than_constant = 0
    bizarre_results = 0
    total_results = 0
    threshold_used = 0

    # Initialize a list to store the results from all models
    all_results_list = []

    # Clear the console
    clear_console()
    # Time counter
    start_time = time.time()

    # Load data
    data = load_data('data.csv')

    # Handle missing values
    data_imputed = handle_missing_values(data)

    # Create directories for figures
    create_figure_directories(data_imputed)

    # Define regression models
    models = {
```

```
        'Linear Regression': LinearRegression(),
        'Elastic Net': ElasticNet(),
        'SGD Regressor': SGDRegressor(max_iter=2000, tol=1e-6),
        'Bayesian Ridge': BayesianRidge(),
        'Support Vector Regression': SVR(),
        'Gradient Boosting': GradientBoostingRegressor(),
        'CatBoost': CatBoostRegressor(verbose=False),
        'Kernel Ridge': KernelRidge(),
        'XGBoost': XGBRegressor(),
        'LightGBM': LGBMRegressor(),
        'Decision Tree': DecisionTreeRegressor(),
        'MLP Regressor': MLPRegressor(),
        'K-Nearest Neighbors': KNeighborsRegressor(),
        'Random Forest': RandomForestRegressor(),
        'AdaBoost': AdaBoostRegressor(),
        'Gaussian Process Regression': GaussianProcessRegressor(),
        'Ridge Regression': Ridge(),
        'BaggingRegressor': BaggingRegressor(),
        'HistGradientBoostingRegressor': HistGradientBoostingRegressor()
    }

    # Perform regression for each combination of inputs and output
    for i in range(1, len(data_imputed.columns)):
        in_combinations = itertools.combinations(data_imputed.columns, i)
        out_combinations = data_imputed.columns
        folder_name = f'{i}in-1out'

        for in_combination in in_combinations:
            in_combination_str = '_'.join(in_combination)
            # Clear the console
            clear_console()

            model_results_list = []  # List to store the results for this model

            for out_combination in out_combinations:
                if out_combination not in in_combination:
                    print(f'Running regression: {in_combination_str} in, {out_combination} out')

                    # Split data into features and target
                    X_in = data_imputed[list(in_combination)]
                    y_out = data_imputed[out_combination]

                    # Split data into training and testing sets
                    X_train, X_test, y_train, y_test = train_test_split(X_in,

                    y_out, test_size=0.2, random_state=42)

                    # Perform regression and measure time
                    model_times, model_durations = perform_regression(X_train, y_train, models)

                    # Save the model results and figures
                    model_path = f'figures/{folder_name}/
                    {in_combination_str}/{out_combination}'
                    os.makedirs(model_path, exist_ok=True)

                    for model_name, model in models.items():
                        y_pred = model.predict(X_test)
                        r2 = r2_score(y_test, y_pred)
                        mse = mean_squared_error(y_test, y_pred)
```

```
                        rmse = np.sqrt(mse)
                        save_model_results(model_path, model_name, in_combination_str, out_combination,
                        X_test, y_test, y_pred, model_durations)

                        # Store the results in the model_results_list
                        results = {
                            'Features (in)': in_combination_str,
                            'Features (out)': out_combination,
                            'Model': model_name,
                            'R^2 Score': r2,
                            'Mean Squared Error': mse,
                            'Root Mean Squared Error': rmse,
                            'Model Time': model_durations[model_name]
                        }
                        model_results_list.append(results)

                # Append the model_results_list to the all_results_list
                all_results_list.extend(model_results_list)

    # Write the results to the overall CSV file
    overall_csv_path = os.path.join('figures', 'overall_results.csv')
    with open(overall_csv_path, 'w', newline='') as csvfile:
        fieldnames = ['Features (in)', 'Features (out)', 'Model', 'R^2 Score', 'Mean Squared Error',
        'Root Mean Squared Error', 'Model Time']
        writer = csv.DictWriter(csvfile, fieldnames=fieldnames)
        writer.writeheader()
        writer.writerows(all_results_list)


    # Print total elapsed time
    end_time = time.time()
    elapsed_time = end_time - start_time
    #print(f'Total elapsed time: {elapsed_time:.2f} seconds')

    # Print the summary of results
    print_summary_results(good_results, half_good_results, bad_results, terrible_results,
    worse_than_constant, bizarre_results, total_results, model_times)


# Call the main function
if __name__ == '__main__':
    main()

# # # #
# # # # end
# # # #
```

Below is a picture containing a word cloud of the .pdf of this work. The criterion is that the larger the word size, the more often they appear in the text. The site used tends to centre them in the figure as well.

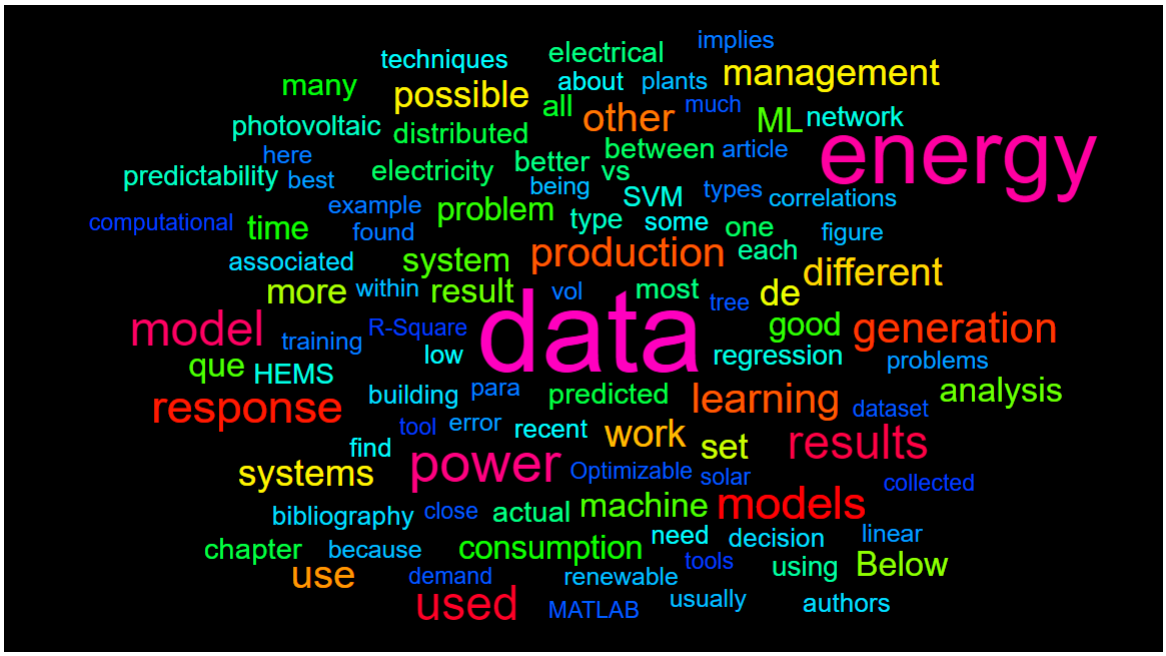

Figure A.1: Word-cloud concentrating the frequency of the most used words in this work.