FACULDADE DE ENGENHARIA DA UNIVERSIDADE DO PORTO

Explaining the optimization of dynamic tariffs for electric vehicles

Tiago Filipe da Costa Bessa

Mestrado em Engenharia Eletrotécnica e de Computadores

Orientador: Doutor Ricardo Jorge Bessa

October 21, 2025

Resumo

Esta dissertação aborda o desafio de aumentar a transparência nos modelos de otimização definidores das tarifas para o carregamento de veículos elétricos (EVs), motivada tanto pelo crescimento na sua adoção, quanto pelo crescente interesse nas diversas aplicações de inteligência artificial, particularmente métodos de Inteligência Artificial Explicável e modelos de Linguagem de Grande Escala (LLM).

Dois objetivos principais de investigação orientaram este trabalho. Em primeiro lugar, avaliar o potencial dos modelos de machine learning (ML) baseados em árvores para aproximar e esclarecer o mecanismo de definição de tarifas, definidas previamente por um modelo de otimização. Em segundo lugar, explorar a capacidade do LLM para gerar de forma independente modelos *surrogate* transparentes, reduzindo a necessidade de pipelines de ML tradicionais, que normalmente exigem conhecimentos técnicos especializados.

A metodologia desenvolvida combinou várias componentes. Primeiramente, uma série de modelos de ML baseados em árvores, incluindo árvores de decisão (DT) e métodos *ensemble*, como Random Forests e XGBoost, foram treinados para aproximar as tarifas geradas por um modelo de otimização previamente desenvolvido. A interpretabilidade do modelo foi avaliada usando métricas de esforço cognitivo, como Operações Mentais Necessárias (RMO) e Operações Mentais Totais (TMO), aplicadas para quantificar a complexidade do processo de tomada de decisão do modelo. Posteriormente, foi desenvolvida uma estratégia para avaliar a capacidade de um LLM gerar explicações acessíveis, em linguagem natural, das resultantes DTs. Por fim, foi proposto um método iterativo para geração de regras baseado em LLM, aproveitando-se do dataset criado através da otimização das tarifas, permitindo que utilizadores não especialistas criem automaticamente modelos preditivos transparentes através de refinamentos sucessivos guiados por mecanismos de feedback internos.

Os resultados mostram que, embora modelos como o XGBoost tenham alcançado a maior precisão preditiva, apresentando um erro médio absoluto (MAE) igual a 0,028797 €/kWh para o *dataset* utilizado, as DT simples ofereceram o melhor compromisso entre desempenho preditivo e interpretabilidade, obtendo-se um MAE de 0,029611 €/kWh. Além disso, o LLM Explainer produziu explicações coerentes e acessíveis para Árvores de Decisão, enquanto o LLM-based Rule Generator produziu conjuntos de regras executáveis a partir do zero, melhorando progressivamente o desempenho preditivo e incorporando justificações e raciocínios textuais, contribuindo assim para a explicabilidade destas regras. Entre várias execuções, a Sim1 alcançou o melhor desempenho, com um MAE igual a 0,03341 €/kWh na iteração 813 de 1000.

Ainda assim, forma identificadas limitações neste trabalho. No âmbito da ML, por exemplo, a seleção e criação de *features* foi limitada, tendo a exploração de técnicas para o efeito sido mínima. Por sua vez, tanto o LLM Explainer quanto o LLM Rule Generator estão sujeitos a alucinações, sendo que o último provou ser computacionalmente exigente, precisando de aproximadamente um dia completo para completar mil iterações.

Por fim, futuros trabalhos devem explorar modelos interpretáveis alternativos, focando-se

na possibilidade de preservar o desempenho preditivo de *ensembles boosting* sem sacrificar a transparência. Além disso, outras linhas de investigação futuras incluem o desenvolvimento de uma estrutura de feedback para explicações baseadas em LLM, a introdução de estratégias *ensemble* na geração de regras, a avaliação do efeito de variações de temperatura sobre as resposta de um LLM, o teste de diferentes LLMs e incorporação de avaliações centradas no utilizador para validar tanto a interpretabilidade quanto a usabilidade prática.

Palavras-chave Veículos elétricos, Tarifas, Transparência, Interpretabilidade, Aprendizagem Automática, Grandes Modelos de Linguagem

Abstract

This dissertation addresses the challenge of enhancing transparency in optimization models used for setting electric vehicle (EV) charging tariffs, motivated by both the growing adoption of EVs and the increasing interest in artificial intelligence techniques, particularly Explainable Artificial Intelligence methods and Large Language Models (LLMs).

Two main research objectives guided this work. Firstly, evaluating the potential of tree-based machine learning (ML) models for approximating and clarifying the tariff-setting mechanism, previously defined by an optimization framework, and secondly, exploring LLM's capability to independently generate transparent surrogate models, reducing the need for traditional ML pipelines, which typically require specialised technical expertise.

The methodology developed in this dissertation combined several components. First, a set of tree-based ML models, including decision trees (DT) and ensemble methods such as Random Forests and XGBoost, were trained to approximate the tariffs generated by a previously developed optimization model. The interpretability of these models was evaluated using cognitive effort metrics, specifically Required Mental Operations (RMO) and Total Mental Operations (TMO), which were applied to quantify the complexity of the models' decision-making processes. Subsequently, a strategy was developed to assess the ability of an LLM to generate accessible, natural language explanations of the resulting DT. Finally, a novel iterative rule generation method based on LLMs was proposed, leveraging the dataset created from tariff optimization to enable non-expert users to automatically create transparent predictive models through successive refinements guided by internal feedback mechanisms.

Results show that while ensemble models such as XGBoost achieved the highest predictive accuracy, presenting a mean absolute error (MAE) of 0.028797 €/kWh, across the utilized dataframe, while simple Decision Trees offered the best compromise between performance and interpretability, achieving an MAE of 0.029611 €/kWh. Furthermore, the LLM Explainer effectively produced coherent, accessible explanations for Decision Trees while the LLM-based Rule Generator produced executable rule-sets from scratch, progressively improving predictive performance while incorporating textual justifications and reasoning, thereby enhancing explainability. Among several executions, Sim1 achieved the best performance, with an MAE equal to 0.03341 €/kWh at iteration 813 out of 1000.

Nevertheless, these methodologies and results are not without limitations. Within the ML scope, for instance, feature selection and engineering remained limited, having the exploration of advanced techniques been minimal. In turn, both LLM Explainer and Rule Generator are subject to hallucination, and the framework applied to the latter proved to be computationally demanding, requiring approximately one full day to complete one thousand iterations.

Finally, future research should explore alternative interpretable models, focusing on leveraging the predictive performance of boosting ensembles without sacrificing interpretability. Additionally, directions for further work include the development of a feedback framework for LLM explanations, ensemble-based rule generation, assessing the effect upon the LLM's response of varying

temperature, testing different LLMs, and incorporating human-centred evaluations to validate both interpretability and practical usability.

Keywords Electric vehicles, Tariffs, Transparency, Interpretability, Machine Learning, Large Language Models

UN Sustainable Development Goals

In 2015, the United Nations developed, as part of the 2030 Agenda for Sustainable Development, seventeen interconnected objectives through a framework named Sustainable Development Goals. Such goals seek to end poverty, protect the environment, ensure peace and enhance prosperity for all, ultimately directing member states toward coordinated action and long-term policy commitments, focusing on the most pressing environmental, economic, and social challenges.

This dissertation supports key global priorities defined in the Sustainable Development Goals, particularly by contributing to areas such as the energy transition, sustainable mobility, and digital equity. Below is a more detailed description of the alignment of this work with specific Sustainable Development Goals.

SDG 7 Ensure access to affordable, reliable, sustainable and modern energy for all.

SDG 9 Build resilient infrastructure, promote inclusive and sustainable industrialisation, and foster innovation.

SDG 12 Ensure sustainable consumption and production patterns.

Table 1: Alignment of this dissertation with selected UN Sustainable Development Goals (SDGs).

SDG	2030 Target	Contribution of this dissertation
7	7.1 – Ensure universal access to affordable, reliable and modern energy services	Transparent EV-charging tariff formation provides clearer and potentially fairer pricing for all users.
9	9.5 – Enhance scientific research, upgrade technological capabilities and foster innovation	Integrates interpretable tree-based ML models and LLMs into energy-pricing systems, demonstrating methodological innovation.
12	12.8 – Ensure that people everywhere have the information and awareness needed for sustainable development	Interpretable models and natural-language explanations equip consumers to make informed, sustainable charging choices.

Acknowledgements

This master's thesis marks the end of yet another stage in my academic journey. At the culmination of five years of studies, I submit this work with the feeling that the mission has been accomplished. By this, I do not necessarily refer to the innovation or scientific objectives that this document may include, but rather to the broader change I experienced, both personally and professionally, as a student and as a worker.

It is on these levels that I wish to focus my acknowledgments. First, I want to thank my mother and father for the strength, patience, and care with which they shaped me as a person and for the way they encouraged me to face challenges. Even when I doubted myself, they always believed in me and reminded me that I was capable.

I also want to highlight the one who is a constant source of joy in my life: my brother. Despite the age gap, which sometimes surprises some and leads others to mistake what they see as a disadvantage for the extraordinary strength it truly is, without any notion of the bond we share. Thank you, Tomás, for showing me throughout this journey, often inexplicably difficult, the lightness and joy with which life should be embraced.

I extend my thanks to my friends from high school, from APA, and from my small village, but especially to my university friends. To these last ones, I can only hope that one day I'll be able to express, in more than words, the impact each of you has had on my life. Thank you all for the companionship, motivation, and every moment shared.

Finally, I would like to mention INESC TEC, the place that welcomed me since my curricular internship and where I now work as a researcher. A huge thank you to Dr. Ricardo Bessa, supervisor of this thesis, for introducing me to research and the professional world, and for placing his trust in me over the past years. I also want to highlight the support of Dr. José Villar and Engineer André Oliveira, who have daily invested in my professional development and fostered my passion for research. To Engineer Carlos Silva, I offer my deepest thanks for his outstanding guidance, patience, and the ease with which he supervised this dissertation, proving to be the professor both this work and I, as his student, needed.

Tiago Filipe da Costa Bessa

"O que as vitórias têm de mau é que não são definitivas. O que as derrotas têm de bom é que também não são definitivas."

José Samarago

Contents

List of Figures

List of Tables

List of Acronyms

AC	Alternating Current
AI	Artificial Intelligence
API	Application Programming Interface
BEV	Battery Electric Vehicle
CESS	Community of Energy Storage Systems
CSO	Charging Station Operators
CSV	Comma-Separated Values
DC	Direct Current
DT	Decision Tree
EF	Evolutionary Forest
EM	Electric Motor
EU	European Union
EV	Electric Vehicle
GBT	Gradient-Boosted Trees
GPT	Generative Pre-trained Transformer
GS	Grid Search
HEV	Hybrid Electric Vehicle
ICE	Internal Combustion Engine
LGB	Light Gradient Boosting
LIME	Local Interpretable Model-agnostic Explanations
LLM	Large Language Model
LLaMA	Large Language Model Meta AI
LR	Linear Regression
LSTM	Long Short-Term Memory
MAE	Mean Absolute Error
MAPE	Mean Absolute Percentage Error
ML	Machine Learning
MSE	Mean Squared Error
OPRO	Optimisation by Prompting
OR	Operations Research
PHEV	Plug-in Hybrid Electric Vehicle
PV	Photovoltaic
Q&A	Questions and Answers
RES	Renewable Energy Sources
RF	Random Forest
RMO	Required Mental Operations
$\overline{\text{RMO}}$	Average Required Mental Operations
RS	Randomized Search
RTP	Real-Time Pricing
SHAP	Shapley Additive Explanations
TCO	Total Cost of Ownership
TMO	Total Mental Operations
ToU	Time of Use

VSRS	Virtual Storage Rental Service
xAI	Explainable Artificial Intelligence
XGB	Extreme Gradient Boosting

Chapter 1

Introduction

1.1 Motivation and Problem Statement

In recent years, across the globe, society has become increasingly environmentally aware. Individuals worldwide have come to recognize the urgency of improving their environmental conduct in order to preserve the planet's natural resources. Consequently, individual-level efforts rose to change daily habits, while governments and institutions took measures to develop innovative projects to modernize technology through sustainable, eco-friendly approaches, seeking to mitigate climate change and foster environmental protection. Among several domains, the energy sector quickly emerged as a critical priority due to its central role in both emissions and potential solutions.

Reducing global dependence on fossil fuels, for instance, was crucial for preserving natural resources and lowering overall greenhouse gas emissions. As a result, innovation became imperative. A notable example is the electricity sub-sector, which underwent a major shift towards decarbonization and renewable energy integration, reshaping how power is generated worldwide.

As the electricity mix transitioned toward renewables, equal attention was devoted to applying clean energy solutions across other high-emission sectors. Mobility, in particular, stood out given its reliance on fossil fuels and significant contribution to air pollution. Therefore, electrification of transportation emerged as a promising solution, which pushed the development and adoption of electric vehicles. However, environmental benefits rely not only on the vehicles themselves, but also on the cleanliness of the electricity supply, hence proving that the decarbonization of mobility cannot be dissociated from the greening of the power grid.

While EVs are increasingly embraced as a sustainable alternative, their integration into existing frameworks presents a new set of challenges for both system operators and end-users. For instance, concentrated charging demand can overload networks. Moreover, public charging infrastructure often lags behind sales growth, as many regions still lack conveniently located chargers, deterring potential buyers [1–3]. Finally, charging-price schemes are usually opaque, creating uncertainty for drivers and eroding consumer confidence [4].

This opacity often originates from the complexity of the underlying pricing models themselves. Charging tariffs are usually determined using advanced optimisation techniques that consider variables such as electricity prices and demand forecasts. Even though these algorithms are effective in defining tariffs, these optimization models tend to function as black boxes, making it difficult to trace how inputs influence the final output. As a result, the general user fails to grasp the logic behind the offered prices, which could end up limiting their willingness to engage in grid-friendly programs [4].

Furthermore, beyond this specific challenge of understanding EV charging tariffs, there remains a broader issue: the general inaccessibility of predictive modelling tools for non-technical users. Forecasting from data typically requires programming and machine learning skills, which ends up creating barriers for many stakeholders across several domains. As interest grows in interpretable and user-centric artificial intelligence solutions, there is a necessity to make data-driven decision-making more accessible and explainable outside traditional expert circles.

1.2 Research Objectives and Approach

This dissertation addresses the opacity challenge as one of its two main areas of focus, concentrating mainly on the explainability of EV charging tariffs. It seeks to explain how such tariffs are determined by applying tree-based ML models and leveraging their inherent capacity to make the underlying decision-making process transparent. This work is an expansion of the framework proposed by Silva *et al.* which introduces an optimization model designed to compute carbon-aware dynamic tariffs for EV charging [4]. By approximating the behaviour of this model through interpretable decision trees, the aim is to clarify the decision-making process behind these Tariffs, making the structure and rationale behind price formation more straightforward. Despite the naturally interpretable structure of a decision tree, the resulting model was further processed using a large language model to convert its logic into natural language. This final step aims to generate a textual representation of the decision-making process, allowing the pricing rules to be expressed in a more accessible and human-readable format.

The expected increase in interpretability may lead to positive outcomes for both end users and charging station operators. On one hand, from an end-user perspective, a clear understanding of how charging tariffs are defined should help reduce perceived complexity and foster greater confidence in engaging with flexible charging schemes. In turn, operators should be able to identify key drivers behind pricing decisions, which may support the evaluation and refinement of existing tariff models, as well as facilitate more effective communication of pricing structures to users or regulators.

In light of the growing scientific interest in LLMs and their wide-ranging applications, the second component of this dissertation briefly explores how these models can assist non-expert users in performing predictive tasks. Since forecasting a target variable from data typically requires programming skills and familiarity with machine learning concepts, users without a technical

background struggle to perform such tasks effectively or rely heavily on data specialists to assist them.

To address this, a prototype framework was developed leveraging two LLMs working iteratively, where the first generates a rule-set to predict the target variable, and the second provides context-aware feedback, which is then used to refine the rule-set in the subsequent iterations. Situated within the broader spectrum of explainability, the first LLM not only creates decision rules, but also justifies them, resorting to natural language, ensuring a transparent and accessible rule set.

This approach should eliminate the need for coding or algorithmic expertise, therefore allowing users to engage with complex predictive problems in an accessible way. Furthermore, although this framework is tested and applied innovatively as a surrogate to the tariff-setting optimization model, the methodology is generalizable and extendable to a variety of use cases.

1.3 Dissertation Structure

The remainder of the document is structured as follows: Chapter 2 presents the literature review, covering fundamental aspects of electric mobility, including environmental, economic and technical aspects, as well as dynamic pricing strategies, explainable artificial intelligence, and the use of large language models in energy systems, optimization and explainability. In turn, Chapter 3 provides the necessary ML background, focusing on tree-based models and interpretability techniques. Chapter 4 outlines the methodology adopted during the project, detailing the procedures taken for both the ML approach and the LLM novel framework. Next, Chapter 5 presents the results obtained and provides a critical discussion regarding predictive performance, both explainability and interpretability and LLM-generated rule sets. Finally, this dissertation is concluded by Chapter 6, which summarizes the main findings, highlights limitations and proposes directions for future work.

Chapter 2

Literature Review

2.1 Review Methodology

The purpose of this review is to synthesize existing research on concepts related to electric mobility, demand response, dynamic pricing, explainability/interpretability of opaque optimization models, and the several applications LLMs present. To ensure a systematic and replicable process, the methodology was based on keyword searches across widely recognized platforms for scientific publications, particularly in the scope of electrical engineering (IEEE and Elsevier, for instance). Search platforms such as Google Scholar and Scopus were also resorted to. The keywords in question were:

1. Electric Mobility;
2. EV Benefits and Challenges;
3. Dynamic Pricing;
4. Demand Response;
5. EV Charging Pricing Strategies;
6. AI-explainability;
7. Interpretability for black-box models;
8. Large Language Models on Energy Systems;
9. Large Language Models and Optimization;
10. Large Language Models and xAI.

The following sections describe the main findings, starting with an exploratory section regarding Electric Mobility. In turn, the second section pertains to the state-of-the-art pertaining to dynamic pricing, followed by a third section which focuses on explainability or interpretability of black-box models. Finally, the last section covers recent advances in LLMs, highlighting their emerging role in energy systems, optimization and xAI.

2.2 Fundamental Aspects of Electric Mobility

In the 19th century, the development of the first internal combustion engine car marked a turning point in human history. From that point onwards, the grounds were formed for the foundation of the automobile industry, and the conceptualization of society's mobility started to cross minds [5]. Nowadays, and after countless modernizations, the ICE vehicle is still the standard used car.

However, as necessity is the driving force behind innovation, growing environmental concerns compelled efforts to facilitate the transition from internal combustion engines to electrical motors. Consequently, between the late 20th century and the early 21st century, significant progress was made in the development of EVs. This marked the rise of electric mobility, which is defined as the use of vehicles powered, either fully or partially, by electricity instead of conventional fossil fuels.

Electric Vehicles operate by converting electrical energy stored in a rechargeable battery into mechanical energy to propel themselves. As highlighted by Figure 2.1, the operation of an EV can be described through a simplified energy flow [6]. In this framework, the system begins with the battery, which serves as the primary energy reservoir, typically storing energy in the form of direct current. Since most EMs run on alternating current, the power converter plays a crucial role by turning DC into AC. Additionally, it regulates voltage and current to ensure the motor receives appropriate electrical inputs.

In turn, the motor is responsible for generating the rotational force. It is noted for its high efficiency, delivering an almost instant and uniform torque. This enables the EV to accelerate rapidly and operate smoothly across a range of speeds. The torque generated by the motor is then transferred to the wheels via the transmission system. Unlike ICE vehicles, which rely on complex multi-gear mechanisms, the EM can efficiently operate over a broad speed range. As a result, a single-speed gear system is sufficient for transmitting torque from the motor to the wheels. Finally, the wheels represent the final point of energy transfer.

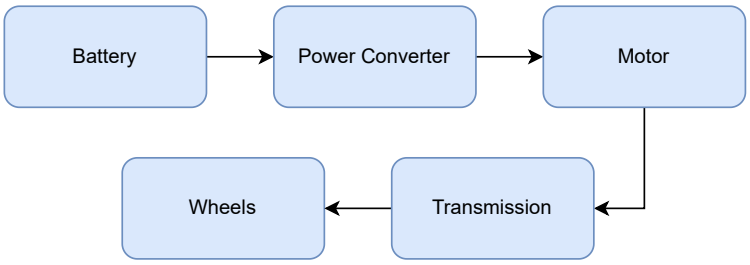

Figure 2.1: Simplified energy flow in an EV.

Furthermore, research and development of electric vehicle technology culminated in the creation of several types of EVs. Figure 2.2 provides a classification of EVs, broadly divided into three main categories.

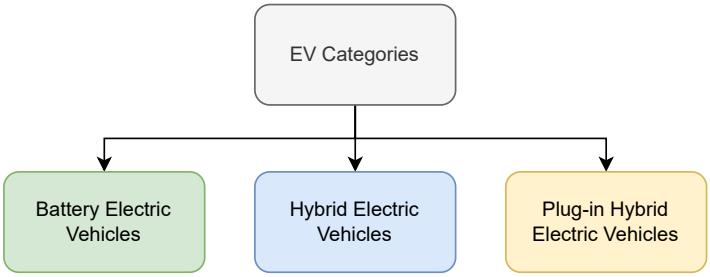

Figure 2.2: Main categories of EVs.

The first category is battery electric vehicles, which are solely powered by electricity stored in their rechargeable battery and usually recharged at home or public charging stations. These are known to be the most environmentally friendly, since they do not produce harmful carbon emissions during their operation. Although BEVs are limited by their battery capacity, or driving range, they are particularly suitable for those seeking a clean energy solution with complete independence from fossil fuels [7].

A second major category is Hybrid Electric Vehicles, which integrate both an EM and an ICE. Despite the option to be externally charged not being inherently available, unlike BEVs, HEVs' electric charge is replenished internally through the available ICE and a mechanism named regenerative braking, which resorts to kinetic energy from the brakes to recharge the battery. While HEVs are less environmentally friendly than BEVs due to their continued reliance on fossil fuels, they provide greater flexibility and longer driving ranges. Notably, HEVs can be further categorized based on the interaction between the EM and the ICE [5, 7]

1. Series Hybrid: The EM is responsible for driving the vehicle, while the ICE recharges the battery.
2. Parallel Hybrid: The EM and ICE both drive the vehicle. However, EM works at low speeds while ICE takes over at higher speeds.
3. Series-Parallel Hybrid: Allows operation in either one or the other mode, subject to driving circumstances.

The final category is represented by Plug-in Hybrid electric Vehicles, which, like HEVs, combine an EM with an ICE. However, these can be externally charged and feature larger batteries, allowing for extended electric-only driving. Once the battery is depleted, the vehicle automatically switches to gasoline operation. This dual-mode capacity helps address concerns such as range anxiety while achieving lower emissions compared to traditional ICE vehicles [5, 7]. Below, Table 2.1 summarizes this description.

Table 2.1: Comparison of Electric Vehicle Categories.

Category	BEVs	HEVs	PHEVs
Energy Source	Electricity stored in rechargeable batteries	Combination of EM and ICE	Combination of EM and ICE
Charging Method	External charging via public or home stations	No external charging; relies on regenerative braking and ICE	External charging (like BEVs) and regenerative braking
Tailpipe Emissions	None	Reduced (dependent on ICE usage)	Reduced compared to conventional vehicles
Driving Range	Limited by battery capacity	Greater range due to ICE integration	Electric mode for short distances; ICE for extended range (biggest of the three)
Environmental Impact	Zero emissions during operation	Lower emissions than ICE vehicles	Lower emissions than conventional vehicles when ICE is used
Special Features	Fully independent of fossil fuels	Regenerative braking to recharge battery	Combines BEV-like charging with ICE reliability

2.2.1 Environmental Considerations

Electric Mobility elucidates part of the scientific community's effort to transition society to a carbon-neutral point. Specifically, the projected mass adoption of EVs aims to reduce the major environmental impact that the transportation sector has on global emissions.

The traditional vehicle is widely recognized as a source of greenhouse gas emissions and air pollution. EVs, on the other hand, present themselves as a sustainable alternative. However, one must be truly aware of the benefits and challenges that their broad integration in society's mobility represents. During operation, ICE vehicles produce substantial amounts of carbon dioxide. BEVs, for instance, produce zero tailpipe emissions. Therefore, air quality is increased, and consequently, public health risks associated with vehicle circulation are decreased. An environmental circumstance usually left neglected is that the process of recharging an EV entails the use of electrical energy, which might have come from a variety of sources. Despite the fact that even when electricity comes from fossil fuels, EVs still create less pollution, it is possible to go even further and coordinate charging patterns to periods when the energy mix has a higher share of renewable energy [4, 7].

Another topic of general reflection in this area is battery management. As previously referred, EVs excel in reducing emissions during operation, however, their production phase might entail environmental challenges. The most common EV batteries are lithium-ion based, and their extraction is a complex process that can result in environmental concerns [1]. Literature is now trying to explore mechanisms to recycle valuable materials to minimize waste and promote a circular economy. Currently, researchers aim to innovate this field by developing battery technology that relies less on harmful materials.

2.2.2 Economic Aspects

An EV cannot solely benefit the environment, its user must obtain tangible benefits to justify their investment. Nowadays, EVs still have higher up-front costs than traditional ICE vehicles.

However, these vehicles hold fewer mechanical parts, meaning they require less maintenance and therefore have lower operating costs. Over the year, maintenance costs for EVs are typically 0.5% of their retail price, compared to 1.5% for ICE vehicles. Furthermore, EVs benefit from lower fuelling costs. For example, the average electricity cost for charging in the US is around $0.15/kWh, compared to the higher cost of gasoline, which averages around $1.00/liter [2].

The Global EV Outlook 2024 states that electric cars need to be made more affordable to reach mass-market adoption. To ease the following analysis, let us consider the Total Cost of Ownership. TCO allows the cost calculation regarding the ownership and operation of a vehicle over a given period of time. For that, it requires information on purchase costs, fuelling/charging costs, maintenance, financing over a vehicle's lifetime, and depreciation factors [2].

Earlier, it was stated that the upfront retail prices for EVs are commonly higher than those of their ICE counterparts. For example, in the US, purchasing an ICE SUV would cost around $40,121 compared to $52,131 for a BEV SUV. This increases EV's TCO, for instance. Nevertheless, this gap is shortened by the $7,500 tax break the US government grants [2].

Additionally, higher efficiency and lower maintenance costs would reduce this measure. The government, through financial support mechanisms, could also help accelerate TOC parity with ICE vehicles since these subsidies would lower the retail price. Alongside reduced operating costs, the report shows that EVs' TCO would achieve parity with ICE vehicles within 7 years across various markets, making them a more compelling option for consumers [2].

Finally, regarding European trends on the EV market, one should be made aware that 25% of accounted EV sales globally were accomplished in Europe, in 2023. In the same year, at a national level, countries such as Norway and Sweden stand out with EVs comprising 95% and 60%, respectively, of their overall vehicle market sales. By 2030, EV sales shares within Europe are expected to surpass 60% of the total car sales [2].

2.2.3 Key Challenges

Recognizing the urgent necessity to modernize mobility entails focusing on solving major challenges regarding technical, infrastructural, and economic areas. At a technical level, EVs still operate under a driving range significantly shorter than that of an ICE vehicle. Therefore, potential users develop range anxiety, especially in regions with inadequate charging infrastructure [7]. Another critical technical challenge involves battery degradation. As the battery's cycle count increases, its maximum available capacity progressively decreases, which may ultimately entail battery replacement, a process that is usually expensive. Therefore, advancements in energy density are essential to support the large-scale adoption of EVs [7].

With regard to charging infrastructure, the availability of charging stations remains insufficient, especially in rural and less densely populated areas [1]. Although the global number of charging points is increasing, the current scarcity is sufficient to deter potential users. This infrastructural deficit is exacerbated by the significant upfront investment and operational costs required for the deployment of charging stations, often discouraging investment [2]. Moreover, charging

times are longer compared to the refuelling process of ICE vehicles. However, research shows that faster charging technologies could introduce an additional strain on the electrical grid [7].

Moreover, Dynamic Pricing mechanisms, increasingly applied to EV Charging Tariffs, add significant complexity to the system, as providers adjust prices based on operating conditions. While this approach encourages users to shift consumption in response to price signals, acting as a form of demand response, computational demands increase [8]. Additionally, complex and often opaque tariff structures represent another layer of challenges [4]. However, applying explainability and interpretability techniques to dynamic pricing models helps to clarify how prices are determined, enhancing transparency and fostering consumer trust.

Finally, computational and operational challenges also emerge. Determining the optimal placement and scheduling of EV chargers, as well as performing load balancing, requires advanced computing models [9]. For instance, emerging vehicle-to-grid technologies contribute to addressing load variability by enabling bidirectional energy flow between EVs and the grid, thereby supporting grid stability during periods of congestion or peak demand [10].

2.2.4 Integration with Renewable Energy Sources

This widespread adoption of EVs will undoubtedly increase electricity consumption, which the current infrastructure is not ready to handle. The report [2] indicates that electric vehicles could represent 6-8% of electricity demand by 2035, compared to just 0.5% in 2023 for reference. Globally, EVs consumed 130 TWh of electricity, which is the same value as Norway's 2023 total electricity demand.

Therefore, addressing this challenge requires a dual approach: ensuring adequate charging infrastructure and optimizing energy sourcing. Subsequently, the charging infrastructure must be expanded and integrate dynamic and interpretative tariff models, aiming for transparency and efficient energy use. Further development on tariffs is pivotal to align user behaviour with energy availability, particularly during peak renewable generation periods. Dynamic Pricing strategies, which this document will expand later, and their time-varying pricing nature are a crucial factor to ease integration and synchronization of EV charging and renewable energy availability [8].

A sustainable solution to handle this escalating demand relies, consequently, on renewable energy usage. However, their inherent intermittency introduces challenges that require robust scheduling models. Their priority is to synchronize users' charging patterns with renewable energy availability, therefore reducing dependency on fossil fuels [11]. At a computational level, the development of such models entails a hard challenge that seeks to solve charging scheduling while balancing grid stability and considering cost-efficiency and user convenience.

Furthermore, charging stations could facilitate renewable energy integration by having that generation on-site [4], or by aligning charging processes with low carbon intensity periods from the electrical grid side. Therefore, self-consumption should be prioritized to reduce reliance on the grid during peak periods. This results in a greater resilience for the whole system.

2.2.5 Future Outlook and Recent Advancements

According to the Global EV Outlook, EV stocks are expected to grow significantly by 2030, rising from fewer than 45 million (2023) to at least 250 million by 2030, with projections of reaching a minimum of 525 million by 2035. Additionally, EV sales share would increase from 15% in 2023 to 40% in 2030 and over 50% in 2035. These projections take into account existing policies and enacted laws. These become even more optimistic statistics if targets and ambitions announced by government institutions respect their deadlines [2].

Additionally, the increasing adoption of EVs and simultaneous increase in the share of renewable sources in the global energy mix is expected to result in a reduction of oil production by at least 6 million barrels per day by 2035 [2]. On top of that, it is necessary to reduce battery cost to support mass EV production and adoption. Achieving a circular economy regarding the battery market while considering the environmental threat their exploration might represent is of extreme importance.

As discussed in Section 2.2.3, electric mobility faces key challenges that require innovation. Therefore, research work is under development across multiple fields. For instance, the current state of the art in battery technology is a contributing factor to issues like range anxiety. Article [3] highlights Solid State Batteries, a new battery technology which offers a greater energy density (meaning longer driving ranges), enables faster charging times, while also providing greater safety when compared to traditional lithium-ion batteries.

Moreover, research on alternative ways of charging is also taking place. The same article refers to inductive charging, commonly known as wireless charging. This recent technology provides, regarding vehicle charging, a high level of efficiency (up to 90%). In addition, users are increasingly resorting to strategies that aim to reduce range anxiety. An example of such is introduced in article [7], claiming that battery swapping stations are yet another viable option since batteries would be kept available, fully charged and ready to be used swiftly by EV users who cannot afford to spend their time at conventional charging stations. This idea is gaining popularity, especially in China, and studies suggest that swapping technology could represent as much as 30% of EV charging in the country by 2030.

Furthermore, advancements in optimizing charging times have undoubtedly increased. For instance, Fesli *et al.*, while highlighting the difference between different charging levels (according to a specific regional standard), they also refer to hardware advancements on DC Fast Chargers technology, which resulted in a high charging power level of up to 350 kW. This amount of power enables longer driving ranges and decreases charging times [3]. The innovation in this field is being propelled by government initiatives, as reported in [2]. For instance, the European Union issued a directive (AFIR) which requires that member-states, as of 2025, install DC Fast Charging facilities every 60 km along the Trans-European Transport Network, increasing the share of such chargers by 15% (of 2023 levels). It is worth mentioning that China holds one of the largest proportions of fast chargers in relation to its total public charging infrastructure, approximately 45%.

2.3 Demand Response and Dynamic Pricing

2.3.1 Introducing Dynamic Pricing

Since this dissertation focuses on the explainability of an optimization model that defines EV Charging Tariffs, understanding the mechanisms behind their definition becomes crucial. In particular, Dynamic Pricing represents a key strategy within the broader context of Demand Response, which allows CSOs to adjust tariffs dynamically in response to changing grid-side operating conditions. Such a concept is of vital importance, as previously mentioned, since it improves energy efficiency and facilitates the integration of EVs with RES [8]. Therefore, it is indeed a central subject for understanding how current pricing models operate and why they are often treated as complex or opaque.

Moreover, Limmer categorizes dynamic pricing strategies in two main types based on their user interaction and structure: Price-Profile-based and Session-based, their respective description is elaborated below [8]. These refer to how prices are structured and presented to EV users.

1. Price-Profile-based: sets dynamic prices per energy unit over different time periods.
   (a) Coarse-grained: sets a unique price for a longer scheduling period
   (b) Fine-grained: sets a unique price for each scheduling period
      i. Personalized: different consumers might be charged different prices in the same interval.
      ii. Non-personalized: the price is equal for every consumer.
2. Session-based: a total price is presented for the entire charging session, rather than for individual intervals.
   (a) Personalized: individual prices determined dependent on auction mechanisms.
   (b) Non-Personalized: the price applied is uniform if a user-requested amount of energy to be charged is the same over the same period of time.

The same article classifies dynamic pricing strategies by their respective implementation approach, i.e., how these prices are determined and adjusted, which can be either online or offline. The latter is characterized by planning prices for a longer time period and relies on forecasts for energy demand, EV arrival patterns and grid conditions to determine optimal pricing. In contrast, online approaches adjust prices dynamically in real time, which makes it more resilient to variable grid conditions and unexpected changes in EV charging demand, or renewable production.

Furthermore, Chen *et al.* focus on the application of AI in scheduling EV charging/discharging considering dynamic pricing models. The research goal is to explore state-of-the-art techniques in forecasting, scheduling, and pricing while identifying research gaps. Essentially, the authors study how Dynamic Pricing can function as a Demand Response mechanism, whose aim is to motivate EV users to take part in scheduling programs, therefore enhancing the coordination of

charging/discharging to ensure grid stability and maximization of social welfare. The study also classifies EV charging/discharging into four main strategies: uncontrolled, controlled, smart, and indirectly controlled charging, which were briefly described below [10]:

1. Uncontrolled charging: EVs charge at rated power upon connection to the grid. This strategy is convenient for EV owners, however it leads to significant stress on the grid-side.

2. Controlled Charging: flexible strategy towards demand-side management, given that this mechanism allows charging station operators to dictate when the EV will be charged/discharged. In spite of reducing user satisfaction, since the mechanism requires the relinquishment of control, this strategy benefits grid performance.

3. Smart Charging Techniques: based on dynamically adjusting charging scheduling considering real-time grid conditions. Despite its obvious benefits for the grid side, this technique often lacks transparency regarding economic benefits for EV owners, therefore reducing their adoption.

4. Indirectly Controlled Charging: resorts to more straightforward price signals to incentivize EV-user support in alleviating grid conditions. Basically, this strategy is characterized by Dynamic Price techniques used as a form of Demand Response. For instance, strategies such as Time of Use and Real-Time Pricing aim to shift EV charging patterns to accommodate grid demand.

Essentially, focusing on Indirectly Controlled Charging, the same article emphasizes Dynamic Pricing strategies as key to influence EV charging and discharging behaviours. Fundamentally, the authors describe pricing techniques as ToU and RTP. The first applies fixed rates for peak and off-peak hours, which are determined based on historical grid usage patterns, usually lacking flexibility to reflect real-time grid conditions. On the other hand, RTP dynamically adjusts prices based on real-time information from the grid-side. Despite providing more accurate price signals that align with the grid's operational needs, according to the authors, these dynamic pricing models often prioritize the grid operator's goals over user preferences. The study's conclusion states that the integration of EVs into the grid infrastructure requires further research in forecasting and scheduling accuracy, as well as developing user-centric dynamic pricing models, meaning pricing mechanisms that reflect real-time grid conditions that ensure fairness and transparency to motivate EV-user participation in grid-alleviating programs.

2.3.2 Dynamic Pricing effect on Energy Systems

In turn, a study developed by Dai *et al.* considers the significant impact that uncoordinated EV charging would have on the distribution system. The paper regards the development of a dynamic pricing scheme, based on a Stackelberg game, for self-consuming charging stations, more specifically charging stations with generation on-site, particularly using photovoltaic panels. Therefore,

the main goal is to coordinate charging while also aiming to stabilize the power grid and integrate renewable generation to reduce overall greenhouse gas emissions. The Stackelberg approach seeks to model EV-users - PV CSOs interactions. It aims to align the objectives of both parties by boosting CSO's financial benefits and minimizing EV-users' charging costs. This optimization model incorporates user satisfaction as a key metric to ensure that their preferences are prioritized. Furthermore, this approach addresses uncertainties in energy generation and consumption by transforming the problem into a deterministic framework, ensuring an efficient management of stochastic variables. Results showed the scheme's ability to motivate EV users to change their charging behaviours according to price signals, to increase grid stability by reducing peak loads and filling valleys across the demand profile and to increase CSO profitability by 48.4% while reducing the selling price (benefiting EV-users) by 20% [11].

RTP techniques are further explored by Yu *et al.*, within the broader context of electrical systems beyond EV-specific applications. This research captures the interactions between consumption and real-time pricing signals within smart grids, meaning it focuses on understanding how consumers adjust their demand based on changes to real-time prices. Therefore, the paper's main goal is to design price signals that encourage efficient electricity usage, reduce peak demand, and increase overall benefits for both consumers and utility companies. Perhaps the most important concept to grasp from this document is the definition of time-dependent utility. In essence, it is the satisfaction, or value, a consumer derives from completing an electricity-related task at a specific time, depending on the time when the task is requested, how much delay it can endure, and how much electricity costs at that time. Therefore, the paper models utility mathematically using a time-dependent utility function [12]. This framework allows the model to capture two different types of elasticity:

1. Self-elasticity: how demand is affected by the price at that same time. If the price increases and the utility does not justify the cost, demand decreases.

2. Cross-elasticity: refers to how prices in a different time slot influence demand in others. Suppose the price at t_1 increases significantly. For a deferrable electricity task, the utility may no longer justify the cost at time t_1, leading consumers to delay the task to time t_2. Cross-elasticity will quantify this relationship by computing how much demand increased at t_2 in response to the price increase at t_1.

These concepts allow the development of a statistical model of demand elasticity, and, afterwards, the design of an optimization model to determine the best fitting RTP signals. The present model was tested considering six buses and a period of 24 hours. The results demonstrated the effectiveness of this approach, that when compared to flat-pricing, was able to reduce by 23.68 % the peak-to-average ratio. Furthermore, social welfare increased by 5.26% and 95.65% of the demand was fulfilled (compared to 91% of the flat-pricing mechanism). More importantly, results showed that 15.21% of the demand was shifted to off-peak periods. Therefore, the already stated potential of RTP, a Dynamic Pricing technique, to balance grid demand while enhancing user satisfaction was proven successful [12].

2.4 Explainability and interpretability

2.4.1 Distinguishing Interpretability from Explainability

Literature defines a key distinction between interpretability and explainability. According to Garouani *et al.*, explainability is essentially a grouping of post-hoc tools that clarify a model's prediction without modifying the model itself. Typical methods include saliency or SHAP maps of feature influence, for instance [13]. Within this dissertation, it is proposed a narrative technique, where a rule-set of a previously-trained Decision Tree is passed to a LLM seeking a natural language explanation.

On the other hand, the article claims that interpretability aims for intrinsic transparency by ensuring the model's inner workings map onto human concepts. Strategies include model distillation, where a surrogate model is trained to mimic the decision-making process of a complex, non-transparent teacher model [13]. Such is accomplished through the ML stage of this work. Essentially, explainability describes why a prediction was made, while interpretability describes how it was made.

Both explainability and interpretability are of crucial importance within the EV scope. Indeed, Kalakanti *et al.* focus specifically on computational challenges related to EVs, identifying the black-box factor as one of the most significant issues within this domain [9]. Many EV-related models rely on variables or parameters whose internal influence and operation are not fully understood, rendering them non-explainable. Even though users are capable of identifying inputs and outputs, they remain unaware of the internal mechanisms that lead to those outcomes. These models are classified as black-box models, representing not only a computational challenge but also a social one, since they undermine user trust.

2.4.2 xAI Categories

Firstly, Vilone *et al.* analyse different methods of xAI aiming to help researchers select their most appropriate algorithm by establishing a hierarchical classification framework into five different types: numerical, rule-based, textual, visual and hybrid [14].

1. Numerical explanation: designed for optimization problems to evaluate each variable's contribution to the solution. To quantify that contribution, numerical methods such as SHAP are often utilized. For instance, both Silva *et al.* and Li *et al.* apply such an approach [4, 15].

2. Rule-based explanation: translate decision-making processes into logical, human-readable steps, making them ideal to achieve transparency and accountability. Often implemented using decision-trees [16], given their effectiveness in regulatory and safety-critical contexts, for instance. However, complex interdependencies can overwhelm this format.

3. Textual explanation: focuses on natural language to communicate a model's reasoning.

4. Visual explanation: facilitate the identification of relationships through intuitive graphical aids. Similar to textual, although these methods are able to grasp insights from complex

optimization models and translate to a non-technical point of view, they can sometimes lack the depth required for a detailed technical analysis.

5. Hybrid explanation: aiming to enhance clarity and accessibility, this type of explanation combines previous formats of explanation. It enables approaches to identify complex interdependencies by combining multiple visual aids with numerical or rule-based explanations. Additionally, this form of explanation is very valuable since it combines formats to present results in a user-friendly manner, for both technical and non-technical contexts.

This is yet another contribution to make opaque models clearer and comprehensive. Structuring different kinds of explanations is crucial to enhance further application of these methods to complex optimization models, ultimately aiming at fostering transparency and user trust.

2.4.3 Pricing Policies

Biggs *et al.* confirm the importance of interpretability of pricing policies, addressing this challenge alongside revenue optimization. According to the authors, these pricing mechanisms (often built using complex models) result in complex and opaque models, meaning the internal mechanisms through which their inputs lead to the observable solutions are incomprehensible. Therefore, their adoption is hindered due to a lack of user trust and practical validation [16].

To address this issue, the author's approach begins by training a highly accurate but opaque teacher model, typically designed to predict demand or optimize pricing decisions. While effective, this teacher model is inherently complex. Then, the knowledge contained within this high-capacity teacher model is distilled into a simpler, interpretable student model. The latter takes the form of a Decision Tree, generated using a customized recursive partitioning algorithm. The result is a transparent model capable of providing interpretable pricing decisions, while retaining much of the predictive performance of the original model.

The effectiveness of the framework was demonstrated through its ability to generate transparent and interpretable pricing policies. Therefore, complex pricing strategies were successfully distilled into a decision tree, whose rule-based structure enables the decision-making process to be easily understood by stakeholders. Therefore, a key barrier to ML pricing is addressed: the lack of transparency in black-box models.

Moreover, Li *et al.* develop an explainable pricing policy for Virtual Storage Rental Service, integrated in a Community of Energy Storage Systems. The main goal of this paper is to maximize CESS operator revenue from energy arbitrage and the VSRS while simultaneously reducing the cost of renting this virtual capacity. Therefore, this problem is modelled as bi-level optimization, solved by both deep reinforcement learning and mixed-integer linear programming techniques. The upper-level aims to maximize CESS's profit, considering revenues from the rental service, as well as from energy arbitrage, while considering battery deterioration cost. The lower-level minimizes the user's electricity bill, considering energy cost and virtual storage rent. The innovative aspect of this work regards the explainability of the model, since Shapley values are used to measure the contribution of different input variables to the model's decision-making process.

Results showed that variables such as consumers' net demand during peak periods and market prices have the greatest influence on pricing decisions [15]. Therefore, there are already available, in published literature, approaches regarding numerical explanations of energy-related opaque optimization problems.

A particularly recent and relevant addition to the literature is presented by Silva *et al.*, and is also examined in this review. This paper forms the foundation for this dissertation since it proposes a chance-constrained stochastic optimization that uses probabilistic forecasts regarding load, PV generation, and grid carbon intensity to form dynamic tariffs for EV charging stations. Notably, it imposes a daily CO_2 budget, which turns its framework carbon-aware. Results showed that the dynamic 75th-percentile budget yielded both the highest emissions reductions and the fewest infeasible days. Furthermore, it presents an xAI layer, an XGBoost surrogate interpreted with SHAP values, which revealed the hour of the day as the dominant feature to predict charging tariffs and increased the overall transparency of this pricing mechanism. Additionally, the stochastic approach lowered emissions 24% more per feasible day than an equivalent deterministic model while maintaining similar cost savings. At last, the presented pipeline: forecasting, optimisation and explainability provided a deployable template for carbon-aware pricing that could be generalized to other sites once elasticity parameters and joint forecast errors are recalibrated [4].

2.4.4 Power Systems and Energy Markets

In another example, Pütz *et al.* study explainability within electricity markets, exploring how xAI techniques could potentially explain the effect of regulatory changes in energy markets. To that end, two specific cases of regulatory interventions are analysed: the separation of the German-Austrian bidding zone and the reform applied to the pricing scheme for the German balancing energy market. The energy prices are modelled using ML techniques, and the influence of those regulatory interventions is explained using the already discussed technique SHAP. Results show that when Germany and Austria shared a common electricity market, the German residual load was the most influential factor in determining the shared electricity price. Once separated, the SHAP tool concludes that the Austrian market was still, although less, influenced by the neighbouring country's residual load. Furthermore, Austria's features gained much more relevance in defining its energy price [17].

Additionally, before 2018, bids for balancing energy were based solely on the capacity price (the cost associated with reserving energy availability without necessarily delivering it). However, a mixed priced system was introduced, which combined the capacity price with the energy price (the cost paid for actual energy delivered when balancing was required). During the mixed price period, SHAP analysis concluded that higher lignite (a type of coal) generation would lower prices because their plants could offer balancing energy at reduced costs. After the mixed price period was discontinued, lignite's importance decreased and gas generation became the most influential factor.

Hence, Explainable Artificial Intelligence not only permitted the evaluation of how these regulatory policies impacted the energy market, but also highlighted the specific features that most contributed to these impacts.

In addition, Kruse *et al.* explore the importance of power grid frequency to its stability and how ML methods could ease its control. Given the increasing size of data and information pool, AI-based optimization models, aiming to predict grid frequency, are becoming more and more achievable. However, recognizing its potential means understanding the black-box nature that complex ML models entail as well. Therefore, the study employs ML models based on Gradient-Boosted Trees, combined with SHAP values to identify the relative importance between variables. Results show that the primary factors that influence frequency stability are load and generation ramps, particularly in Continental Europe. However, in Nordic countries, the most important factor are fluctuations and forecast errors in generation. In turn, Great Britain, the main determinant were forecasting errors in renewable energy production [18].

2.5 Large Language Models

2.5.1 Introducing LLMs

Large Language Models are a class of deep learning models based on Transformer architecture, designed to process, understand, and generate human language at scale [19, 20]. Typically, LLMs contain hundreds of billions of parameters and are trained on large volumes of text data, performing tasks such as translations, summarization, question answering, and code generation. In sum, LLMs show enhanced capabilities in language comprehension, reasoning, and instruction following.

These models are now widely used across different areas, with popular examples including GPT-4, LLaMA and Claude. These are often used through prompting, where users give natural language instructions to have a specific task performed. However, according to both Singh *et al.* and He *et al.* this prompt should be designed carefully, since its quality significantly impacts LLM outputs [20, 21].

Techniques such as Chain-of-Thought prompting, applied both by Singh *et al.* and Bordt *et al.*, encouraging step-by-step reasoning, or few-shot prompting, a strategy employed again by Singth *et al.* and Hsieh *et al.*, where the model is given a few examples within the prompt to guide its behaviour, are popular strategies for improving prompts and thereby improving LLM performance on complex tasks without requiring additional training [19, 20, 22].

Despite their impressive capabilities, LLMs face several important limitations and risks. A central concern is the presence of social and cultural bias, which arises because these models are trained on massive datasets, which could reflect historical prejudices or other harmful patterns. As a result, LLMs can unintentionally generate biased, offensive, or discriminatory content. Another major issue is the risk of misuse, where the generative power of these models can be exploited to produce misleading or harmful content such as disinformation. In addition, LLMs are

resource-intensive, since both training and deployment require much computational power. Their increasing number of parameters and opacity further complicates matters, since it challenges user interpretation or capability to properly audit how decisions are made. Moreover, LLMs are prone to hallucination. Therefore, they could generate fluent but incorrect or unfounded information, making them unreliable in contexts where factual accuracy is critical. Consequently, their outputs should be viewed as suggestions rather than definitive answers, especially in sensitive applications [20, 23, 24].

Recently, LLMs have rapidly transitioned from experimental natural language processing constructs to versatile, knowledge-rich engines capable of reasoning across diverse technical domains. These advances enabled LLMs to become a crucial general-purpose cognitive component in scientific and engineering workflows. This section aims to highlight different applications of LLMs across several key domains, including energy systems, optimization, machine learning and explainable AI. The following examination of several academic contributions will illustrate how these models are being integrated into various complex workflows.

2.5.2 Energy Systems

Dong *et al.* question the readiness of LLMs for power-system engineering. The authors frame the challenge of turning grid measurements, imagery, and documents into fast, trustworthy insights. Therefore, to examine its capabilities a LLM model is tasked with multiple representative jobs: load and price forecasting, correlation analysis, equipment fault detection, wildfire-risk mapping, document Q&A via retrieval-augmented generation, on-site hazards recognition and power-flow related problems. Authors state that performance depends heavily on a carefully designed prompt, on fine-tuning the model (retraining it on specialized data) and delegating physics-heavy sub-problems to external solvers [25].

For instance, the models achieved roughly 2% of mean absolute percentage error in 24-hour load forecasts and classified defective equipment as such 70% of the time. Weaknesses surface when training data are sparse, when strict physical laws govern the answer, or when confidentiality and safety controls are missing. Crucially, the study shows that prompt design by itself breaks down on physics-heavy tasks, whereas acceptable answers only emerge when LLMs are allowed to call an external numerical solver that enforces physical constraints. Consequently, the study concludes that LLMs can already augment conventional analytics, but deploying them in critical grid settings will require curated datasets, a retrieval system that respects physical constraints, and governance aligned with emerging AI-risk frameworks.

Furthermore, Xue *et al.* tackle the persistent difficulty of forecasting electricity-price volatility in the highly liberalised New South Wales market, where pronounced non-stationarity and multi-seasonality factors undermine conventional models and even advanced deep-learning architectures. As per the authors, these approaches either fail to capture non-linear dependencies or introduce exogenous information without assessing its relevance, thereby inflating dimensionality and noise [26].

As such, this paper handles these issues by deriving two price-based volatility factors, the Continuity and Price-Jump indices, thereby capturing both continuous fluctuations and sudden spikes. Afterwards, the candidate feature set is enriched with five exogenous meteorological covariates: air temperature, wind speed, relative humidity, mean sea-level pressure and a 1-5 weather-rating score. All these variables are generated by a retrieval-augmental GPT 3.5 pipeline that retrieves those physical measurements from 10 years worth of weather reports and assigns a relevance score to each passage, resorting to LLaMA-Index, thereby mitigating noise. The feature space is subsequently thinned, leaving the 30 most important predictors. These feed a two-stage hybrid learner in which XGBoost models non-linear relationships, producing the initial predictions. Afterwards, an LSTM improves such results by incorporating temporal dependencies. Finally, both outputs are aggregated to yield the overall predictions.

This framework ended up reducing MAE, MSE and MAPE by approximately 49 %, relative to the best conventional benchmark, demonstrating the value of LLM-enhanced feature engineering within energy-market volatility forecasting.

2.5.3 Optimization

This review also explored the state-of-the-art regarding the applications of LLMs towards optimization. Zhang and Luo target a known issue in Operations Research, i.e., the dependence on domain experts for formulating models and programming problem-specific solvers, which ends up curtailing the large-scale deployment of optimization in industry. Therefore, the OR-LLM-Agent framework is introduced. Such work resorts to reasoning Large Language Models to transform natural-language problem statements into formal linear programming models, later autogenerating Python-Gurobi code [27].

Methodologically, the authors test the agent on a new benchmark of 83 textbook-based, real-world operational research problems. The framework, using GPT-o3-mini, attained a 100 % code-execution pass rate and 85 % solution accuracy, exceeding state-of-the-art reasoning LLMs by more than 8% and non-reasoning models by more than 23%. These results were obtained when considering an OR-CodeAgent designed to iteratively repair automatically generated code in a sandbox environment. Moreover, its absence caused the results to degrade. These findings demonstrate the practical feasibility of LLM-centric automation, turning these modelling and solving issues expert-free [27].

Furthermore, Yang *et al.* also addresses optimization, particularly the challenges related to derivative-free optimisation, where gradient information is unavailable or costly, by re-framing the optimiser itself as an LLM that operates purely through prompting. Therefore, a new framework named Optimization by PROmpting is proposed by the authors. This method treats an optimisation task written in natural language as the input, then uses the LLM to iteratively propose new candidate solutions, which are thereafter evaluated and appended to the prompt for the next step. This natural-language interface removes the need for task-specific solvers, promising rapid adaptation across different domains [28].

More specifically, OPRO encodes two key elements in a *meta-prompt*: the natural-language description of the objective and the problem constraints, and the optimisation trajectory consisting of previous solutions paired with the respective score, sorted by performance. Interestingly, the LLM balances exploration and exploitation of the search space by sampling multiple solutions per step under a tuned temperature, improving stability and enabling the LLM to discover the most promising direction to follow. A case study on the travelling salesman problem shows that this approach matches or exceeds classical heuristics while using markedly fewer evaluations. Furthermore, LLMs proved to be capable of resorting to previous information to gradually improve results [28].

2.5.4 Explainability

Chen *et al.* confront the black-box perception that hinders users from trusting and exploiting optimisation models developed by specialists. Drawing on recent advancements in LLMs, they established a method named *OptiChat*, a dialogue system that first translates Pyomo model scripts into plain-language narratives and then channels user questions through a layered, task-oriented agent hierarchy. A coordinator agent distinguishes descriptive queries from analytical queries. The latter flows to an *engineering team* comprising an operator, a programmer, an evaluator and a reminder agent. The operator addresses diagnosing, retrieval, sensitivity and what-if queries, whereas the programmer-evaluator loop generates test code for open-ended counterfactual why-not questions. The reminder, in turn, supplied hints to curb argument errors. This pipeline is designed so that each agent feeds their technical findings back to an explainer that crafts user-friendly answers. Even though this framework does not show perfect results, it returns human-readable model descriptions in under a minute and answers queries with high success rates, successfully tackling the lack of understanding the general user has over the optimization models, across different fields [29].

Moreover, Sivasothy *et al.* inquire if LLMs could be utilized as a shortcut for capturing domain rules that normally demand prolonged collaboration with subject-matter experts. Consequently, they introduce RuleFlex, a pipeline that converts a plain-language description into executable Python rules via GPT-3.5-Turbo or GPT-4, then lets clinicians edit the output before deployment. The evaluation that follows centres on a few-shot prompting setup and assesses interpretability, accuracy (variable and threshold overlap) and run-to-run consistency. Compared with 41 clinician-authored rules, the LLMs produced 2-4 rule sets, omitted key variables, and almost never supplied correct numeric thresholds. The authors therefore conclude that LLMs can bootstrap an initial rule scaffold but still require expert validation to fill threshold gaps and ensure reliability in safety-critical domains such as health [30].

Within xAI, Suh *et al.* interrogate the rising practice of using LLMs solely to renarrate technical explanation artefacts such as SHAP or LIME. The authors argue that this surface-level polishing risks hiding model weaknesses, fostering an illusion of transparency and encouraging user over-reliance. These concerns are especially acute in high-stakes domains. Therefore, the authors synthesise documented failure modes of existing xAI plus LLM pipelines and propose the LLM

Devil's Advocate framework, which essentially is a catalogue of adversarial prompting strategies, including uncertainty flagging, alternative explanations, bias checks and counterfactual probes. Such a framework aims to make the LLM interrogate SHAP/LIME explanations themselves rather than simply rephrase them. Deployed in this sceptical role, an LLM can expose missing variables, among other issues, thereby curbing misplaced trust. However, authors caution that empirical studies are still needed to weigh these transparency gains against the extra cognitive load such adversarial dialogue may impose [31].

2.6 Review Conclusions

Firstly, this review confirmed the rapid growth of EVs and associated technical, infrastructural, and economic challenges. Among these, the lack of transparency in tariff-setting models emerged as a critical barrier to user trust and adoption. This dissertation focuses specifically on addressing the opacity of a dynamic pricing model applied to EV charging, contributing to the development of a more transparent and interpretable tariff-setting optimization framework.

The review also identified and analysed the main categories and types of dynamic pricing strategies, including Price-Profile-based and Session-based models, as well as different implementation approaches such as offline scheduling and real-time adjustment. Furthermore, while the reviewed studies demonstrated the effectiveness of Dynamic Pricing in shifting demand, reducing peak loads and improving grid stability, they also hinted to the inherent complexity of such models. The variety of pricing structures, mechanisms, and real-time processing contributes to opaque decision-making, underscoring the need for explainable solutions in EV Charging Tariffs.

Additionally, the literature also shows that xAI methods such as rule-based models have been effectively used to explain complex pricing problems, while techniques such as SHAP are consistently used across the energy field. These approaches relied on post-hoc interpretation of black-box models, which fail to expose the actual decision-making process, offering abstract contributions rather than clear, rule-based conclusions. However, the application of explainability or interpretability frameworks directly to EV Charging Tariffs is still limited.

This gap highlights the relevance of the present dissertation, which advances the field by exploring inherently transparent models like Decision Trees, complemented by LLMs for automatic rule generation and natural language explanations. This approach not only simplifies the underlying decision-making process but also enhances practical explainability, contributing directly to user trust and transparent decision-making in EV charging contexts.

Finally, the review confirmed the growing incorporation of LLMs as tools across optimization, energy systems, and xAI. Applications include enhancing feature engineering to automating mathematical modelling and enabling natural-language optimization. However, their use remains constrained by issues as hallucination, incomplete domain understanding, and dependency on expert validation. Therefore, this dissertation aims to build this framework.

Chapter 3

Machine Learning Background

3.1 Review of Tree-Based Machine Learning Models

Due to its crucial importance to this dissertation methodology, reviewing tree-based ML models is essential to establish the theoretical foundation.

3.1.1 Decision Tree Regressor

The Decision Tree Regressor is a non-parametric, supervised learning model that builds a hierarchical set of if-then rules, mapping input features $x \in R^p$ to a target y. Such work is achieved by recursively partitioning the feature space into axis-aligned rectangles (Figure 3.1), where each of these regions is assigned a constant output value. In other words, a decision tree is an interpretable, label-guided rule hierarchy that adaptively segments the input space into homogeneous regions, each one associated with a target prediction [32–34].

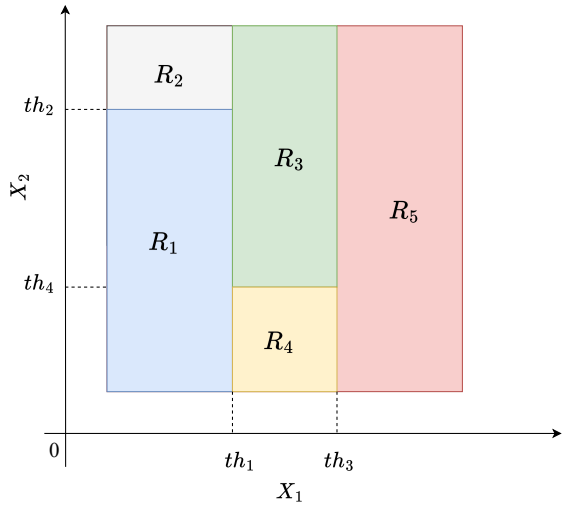

Figure 3.1: Example of a feature space (containing two features) recursively divided.

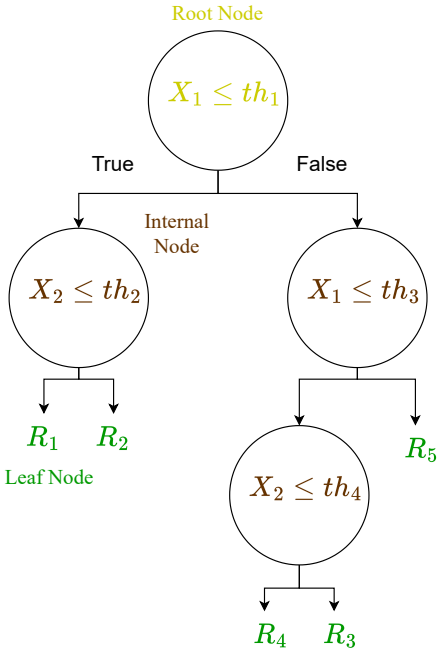

Figure 3.2: Example of a Decision Tree as an alternative representation of the feature space partition.

Consequently, Decision Trees are structured by a root node, an internal node, and a leaf or terminal node. The tree starts at the root node, thus representing the entire dataset. Therefore, the decision-making process occurs from the root to the leaf node (the latter corresponding to the target prediction), having navigated through internal nodes where split conditions were evaluated along the way [33, 34].

The schematics introduced by Figures 3.1 and 3.2 allow the visualization of the partitioning of the feature space and how this recursive process can be translated into a tree-based structure. Clearly, the example shows a two-dimensional feature space (X_1, X_2), and the thresholds that the Decision Tree algorithm found optimal to have as the split conditional variable ($X_1 \leq th_i : i \in \{1,3\}, X_2 \leq th_j : j \in \{2,4\}$). For simplicity, let's assign the values from Table 3.1 to these features and thresholds. According to the data and Figure 3.1, X_1 is greater than th_1, which eliminates regions R_1 and R_2 from consideration. The remaining candidate regions - R_3, R_4, R_5 - are then evaluated further. Since X_1 is also less than th_3, region R_5 is dismissed. As a result, only the central regions R_3 and R_4 remain. These two are separated based on the value of X_2, which is equal to 6, and compared against the threshold $th_4 = 7$. Since X_2 is less than the threshold $th_4 = 7$, and given that X_2 corresponds to the vertical axis, the lower region - namely R_4 - is selected as the final assignment.

However, despite seeming fairly easy to retrieve results from representations as Figure 3.1, the reader should be aware that as the dimensionality of the feature space increases, it becomes

somewhat challenging to represent it [32]. Therefore, the decision tree offers a much more suitable representation, as it is both visually appealing and significantly more interpretable in terms of tracking the decision-making path.

Table 3.1: Examples for Variables and Thresholds of a DT.

Variable	**Value**
X_1	3
X_2	6
Threshold	**Value**
th_1	2
th_2	4
th_3	6
th_4	7

It has been established so far that a Decision Tree is structured by recursively partitioning the feature space, and that its representation remains vital for those seeking an interpretable decision-making process. That said, some questions remain unanswered: How is the splitting decision made? What is the criterion for stopping the growth of the tree? What impacts the Decision Tree performance? How limited is a Decision Tree's performance?

3.1.1.1 How is the splitting decision made?

The algorithm must be able to autonomously determine both splitting variables and their corresponding thresholds. Its prediction equals the mean of the target values from the subset of the input dataset that, based on the splitting decisions, fell into a given region [32].

The decision to split a root or internal node is made using a greedy algorithm, since it selects the variable and corresponding threshold that produces the greatest immediate improvement in prediction accuracy instead of searching for a globally optimal tree structure. Such a procedure would require considering all possible sequences of splits across all variables and thresholds. Since this global optimization is computationally prohibitive [32], the algorithm only evaluates the immediate effect of each potential split at the current node, as demonstrated below.

To determine the best split, the algorithm systematically considers each feature and, for each one, examines all possible threshold values observed within that feature. Each possible combination of feature and threshold defines a candidate split, which effectively partitions the dataset at the current node into two distinct regions.

For each candidate split, the algorithm evaluates how effectively the resulting two regions separate the target variable. This is measured by calculating the prediction error within each region, where the lower error indicates that observations in each region are more similar to one another, achieving greater *purity*.

The split that minimizes the error across both regions is selected as the best split for that node. Conceptually, this process could be understood as the reduction of variance of the target values

within the newly created regions. Therefore, the algorithm selects both feature and threshold that lead to the greatest reduction in variance.

3.1.1.2 What is the criterion for stopping the growth of the Decision Tree?

This process is bound to continue until a stopping criterion is met, such as a minimum number of observations per node. Once that number is reached, there is no further splitting and the region becomes a terminal node. Another example of a criterion is, for instance, the definition of a maximum depth for tree growth.

These questions become particularly important since deep decision trees are prone to overfitting. In an ML pipeline, it is good practice to divide your dataset into a training and testing dataset, thus allowing for the evaluation of the model's ability to generalize (assess performance using unseen data by the model). Overfitting occurs if a model fits the training data too closely. As a result, the model achieves low error during training, but ends up failing to generalize when it is tested against the remaining dataset. In such cases, a simpler model with slightly worse training performance could actually generalize better. Therefore, overfitting is a challenge when using decision trees. However, strategies have been developed aiming to reduce this performance obstacle, such as pre-pruning and post-pruning [33, 35].

Pre-pruning strategies halt the growth of the tree at an early stage, meaning before it fully, or nearly, fits the training data. This approach is often applied given its simple integration into the ML pipeline. Furthermore, since it stops early, computational expenses are reduced. However, such proceedings must be taken with caution, as setting these stop conditions harshly might lead to underfitting. As mentioned earlier, criteria such as maximum depth, minimum observations per node, and minimum purity gain are examples of pre-pruning strategies [19, 33, 35].

On the other hand, post-pruning mechanisms first allow the tree to overfit the training data, so that afterwards, post-pruning techniques, such as M5 pruning, are applied. These mechanisms primarily involve removing branches of the tree that minimally enhance the prediction of the target variable. [19, 33, 35]

Moreover, limiting tree growth to a predefined depth is a delicate task. The programmer must be aware that, although deeper trees may improve accuracy, they are also more prone to overfitting as depth increases.

3.1.1.3 What impacts a Decision Tree Performance?

A Decision Tree strives to get the best prediction possible. It has been discussed how its structure is created and how its growth is halted. Furthermore, a common issue with this type of model, overfitting, was also reviewed.

However, there are many other variables that influence the predictive power of the model. Firstly, the user should look at data-related factors, for instance, feature relevance, feature scaling, data quantity and noise. When not overlooked, these may contribute to a better splitting decision from the model. Secondly, the model hyperparameters: maximum depth, minimum samples to

have a split, minimum samples to create a leaf, the maximum number of features used when looking for the best split, and how many leaf nodes the model can have [33].

3.1.1.4 How limited is a Decision Tree's performance?

Firstly, a single decision tree is, undoubtedly, a weak learner [36]. This work will later expand on developed strategies that sought to design strong tree-based models. Secondly, decision trees are by nature unstable. Such instability arises from the fact that a minor change within the input information leads to a completely different sequence of splits, which, depending on the dataset, might make the model's interpretation unreliable. Essentially, these models fall victim to high variance [32].

Therefore, DTs have a tendency to produce prediction functions that are not smooth. As these models partition the feature space into distinct, axis-aligned regions, their predictions change abruptly at the boundaries of these areas. This further reflects the issue of the variability (sudden jump in the output) given an input change. Regression is particularly affected by this issue, since the target function is expected to vary continuously with the input. In essence, these abrupt transitions end up degrading predictive performance [32].

Moreover, as previously addressed, decision trees are prone to overfitting. Therefore, hyperparameter optimization and pruning strategies may be required. Furthermore, an error made at an early split will propagate downward, affecting all subsequent decisions. Although using a more robust split criterion might help to mitigate this effect, the fundamental instability remains [35].

3.1.2 M5 Prime

The M5 Model, introduced in 1992 by Quinlan [37] and later improved by Wong *et al.* [38], is an advancement in DT methodology by integrating regression models at the leaves of the tree, diverging from the traditional approach of assigning constant values (which is, typically, the average value of the samples within that region). This methodology seeks to blend the interpretability and structure of Decision Trees with linear regression, thereby aiming to enhance predictive performance.

Firstly, the algorithm proceeds by growing a decision tree using standard deviation reduction to evaluate the splits. Then, once the tree is fully built, the pruning process starts. Essentially, the strategy entails the comparison of the expected error of a subtree to that of a linear model applied to that subtree's root. If the linear model yields a lower expected error, it replaces the entire subtree. Such an approach is key to balancing model complexity and generalization abilities, ensuring that the final model remains parsimonious, meaning it avoids overfitting, while retaining sufficient predictive accuracy [37, 38].

Additionally, M5 offers the possibility to mitigate potential discontinuities between adjacent linear models through smoothing. This issue arises when different leaves produce divergent predictions for similar input values. Therefore, the prediction of the leaf model is combined with that of its parent node, weighted by the number of training instances. The model seeks to ensure that

the transition between different regions from the input space does not result in abrupt changes in predicted values, therefore improving the model's continuity [38].

Therefore, it allows the model to approximate complex, non-linear mappings from inputs to outputs by combining local linear approximations, each valid in a specific region of the feature space.

3.1.3 Ensemble models

As mentioned earlier in Section 3.1.1, single trees are weak learners. Therefore, research brought a technique that seeks to improve predictive performance by combining single ML models, leading to an enhancement of their individual accuracy capabilities, namely Ensemble Learning. Therefore, multiple base learners are trained, and their predictions are fused so that the final model has greater predictive power and better generalization ability. More specifically, this technique deals with known issues from single Decision Trees such as their inherent high variance and low accuracy [32, 36].

It is possible to categorize Ensemble Learning into different categories: parallel and sequential ensembles. Firstly, parallel ensemble trains each base learner independently and, afterwards, combines their predictions. An example of such implementation is bagging algorithms, the most common being Random Forest (RF). This technique is characterized by the diversity that each of the base learners represents. Moreover, parallel ensembles are further categorized into homogeneous if the base learners that comprise it belong to the same model type, or heterogeneous if otherwise [36].

Secondly, a sequential ensemble adopts an iterative training process in which each subsequent model is designed to correct the errors of its predecessor, meaning that its base models are not trained independently. An example of such an implementation is the boosting algorithm [36].

Finally, the quality of the ensemble depends not only on the accuracy of each base learner but also on its diversity. In the following subsections, the reader will find detailed descriptions of each algorithm. Moreover, the trade-off between interpretability and predictive power leans towards the latter. Interpreting hundreds of Decision Trees within an ensemble might prove to be a difficult task, since looking individually at each estimator and tracing its decision-making path would not be faithful to global results from the ensemble [33].

3.1.3.1 Bagging Algorithm

As described earlier, the bagging algorithm is an example of a parallel ensemble. Bagging relies on the combination of independently trained base learners in order to predict an outcome. Therefore, it aims to increase performance accuracy using multiple decision trees from randomly generated datasets [36].

Its algorithm is characterized by the introduction of diversity between base learners seeking to reduce overfitting. This process involves generating bootstrapped replicas of the training data,

which in turn entails randomly sampling multiple subsets from the original dataset with replacement, meaning that each data point selected is put back into the pool, so that it may be selected once again. As a result, each base learner is trained on a distinct, though overlapping, subset of data, promoting diversity while using the same underlying machine learning algorithm [34, 36].

Therefore, bagging stands out for its particular effectiveness when applied to inherently unstable base learners. Unstable learners are those whose generalization performance can vary significantly in response to minor changes within the training data. Consequently, it is beneficial to use bagging across ML models characterized by high variance, such as the Decision Tree. Therefore, for most datasets, it is expected that bagged ensembles of DTs will outperform a single DT Model regarding predictive performance [36].

However, one must be aware of the issues regarding the interpretability of these models. Section 3.2 explores mechanisms to interpret an ensemble model, seeking to allow a user to understand the decision-making process behind the model's solution.

Random Forest is perhaps the most widely used bagging algorithm, mitigating overfitting through variance reduction by applying bootstrap sampling mechanisms. In essence, Random Forest builds multiple decision trees using different subsets of both training samples and features, aggregating their output through averaging, thus producing a more robust final prediction [33].

This method, described by Algorithm 1, according to Mienye *et al.*, enhances model performance by centring its approach around two main mechanisms: bootstrap sampling and feature randomization, the latter being the random selection of predictors within the feature space, seeking the construction of uncorrelated trees within the forest [33].

Furthermore, the Random Forest presents superior performance to the stand-alone bagging algorithm. It is capable of handling missing data, and its performance increases if the dimensionality of the feature space is bigger [36]. Evidently, their performance depends on the definition of its hyperparameters. Strategies to automatically find the best set of parameters within the hyperparameter space will be covered in section 3.1.4.

Algorithm 1 Random Forest Algorithm

Require: Training set $S = \{(x_1, y_1), \ldots, (x_n, y_n)\}$, number of trees T, number of features p
Ensure: A trained Random Forest model and a prediction function $f(x)$

for $t = 1$ to T **do**
    Generate a bootstrap sample S_j from S by sampling n instances **with replacement**
    Build a decision tree h_t using S_j:
    while splitting criteria not met **do**
        Randomly select m features from the total p features ($m \ll p$)
        Determine the best split using only the selected m features
        Apply the split and continue recursively
    end while
end for

▷ After training, use the ensemble of trees to make predictions on new data

function PREDICT(x)
    return $f(x) = \frac{1}{T}\sum_{t=1}^{T} h_t(x)$ ▷ Average of tree predictions
end function

3.1.3.2 Boosting Algorithms

As discussed earlier, the boosting algorithm falls into the sequential category, since it constructs a strong predictive model by combining multiple weak learns one after another. Therefore, the core idea behind boosting involves applying a base learning algorithm iteratively to modified versions of the training data, where each iteration aims to correct the errors made by the previous model [33, 36].

Initially, boosting starts by training a weak model on the original input dataset. The difference between the predicted and actual target value (residual) is then used to guide the training of the next weak learner. Specifically, subsequent learners are trained to predict these residuals, gradually improving the model's performance with each iteration. The process will continue until a predefined number of base learners has been trained. Each learner is assigned a weight based on its performance, meaning that the final outcome is computed as the weighted sum of all estimators' output [36].

Similar to bagging, this approach has the same limitations regarding interpretability. Furthermore, boosting gathers another limitation: its sensitivity to noise. Since the algorithm focuses on poorly predicted samples (large residuals), it may disproportionately focus on noisy data points. This could lead to overfitting, and therefore, reduced generalization capabilities, particularly in datasets with significant label noise [36].

This dissertation covers several boosting algorithms. For instance, the Gradient Boosted Decision Tree algorithm builds each tree sequentially, where each neew tree aims to correct the prediction errors of the preceding ensemble. As the name suggests, this strategy seeks to minimize target prediction errors, using a gradient descent algorithm. Gradient descent is an optimization algorithm used to minimize a differentiable loss function by iteratively moving in the direction of the steepest descent, as defined by the negative gradient of the function [33].

GBT operates by learning from the residual patterns of previous predictions using gradient-based optimization. Algorithm 2 describes the process. Essentially, it starts by initializing the model with a constant prediction and then, in each iteration, computes the pseudo-residuals, i.e, the negative gradients of the loss with respect to the current prediction, for each training instance. These pseudo-residuals serve as target values for fitting the next regression tree. The output of this tree is then scaled by a learning rate to control the contribution of each tree to the final ensemble and added to the existing ensemble model. This process will continue until the predefined number of learners has been reached. Finally, predictions for new instances are obtained by summing the output of all weak learners along with the initial prediction [33].

Algorithm 2 Gradient Boosted Decision Trees (GBT) [33] [36]

Require: Training data $\mathscr{S} = \{(x_1, y_1), \ldots, (x_n, y_n)\}$, number of boosting rounds M, learning rate γ, differentiable loss function $\mathscr{L}(y, \hat{y})$

Ensure: Trained ensemble model $f_M(x)$

1: Initialize the model with a constant prediction:

$$f_0(x) := \arg\min_c \sum_{i=1}^{n} \mathscr{L}(y_i, c)$$

2: **for** $m = 1$ to M **do**

3: Compute pseudo-residuals:

$$r_i^{(m)} = - \left. \frac{\partial \mathscr{L}(y_i, \hat{y})}{\partial \hat{y}} \right|_{\hat{y}=f_{m-1}(x_i)} \quad \text{for } i = 1, \ldots, n$$

4: Fit a regression tree $d_m(x)$ to the training data $\{(x_i, r_i^{(m)})\}$.

5: Update the model:

$$f_m(x) := f_{m-1}(x) + \gamma \cdot d_m(x)$$

6: **end for**

7: **function** PREDICT(x)

8: **return** $f_M(x) = f_0(x) + \sum_{m=1}^{M} \gamma \cdot g_m(x)$

9: **end function**

Mienye *et al.* states that the main advantage of this algorithm is its capability to really learn complex patterns from the input data. On the other hand, if a considerable amount of noise is present in the input data, the model is prone to overfitting. Hence, a better performance is expected for small datasets [33]. Moreover, the execution of this algorithm could entail considerable execution time, since this is a computationally expensive algorithm [39]. Naturally, its performance is dependent on the hyperparameters defined prior to execution. Strategies, as those mentioned further in Section 3.1.4, are helpful to find the optimal hyperparameter that ensures the minimum error within the search space.

This dissertation also resorts to XGBoost, a highly efficient and scalable tree-based ensemble method that implements the boosting framework. It frequently outperforms other ML approaches, being a reference in predictive performance among them. Different from GBT, this algorithm introduces improvements, particularly in terms of regularization, optimization, and computational performance [36, 40].

The distinguishing feature of XGB is the inclusion of a regularization term in its objective function, which helps control model complexity and seeks to reduce overfitting. Equation 3.1 represents the objective function, i.e., the function the algorithm tries to minimize during training, applied over each training sample.

$$\mathscr{L}_M(F) = \sum_{i=1}^{n} \mathscr{L}(y_i, F(x_i)) + \sum_{m=1}^{M} \Omega(f_m) \tag{3.1}$$

$$\Omega(h) = \gamma \times T + \frac{1}{2}\lambda \times ||w||^2 \tag{3.2}$$

The objective function $\mathscr{L}_M(F)$ combines two components: the loss function $\mathscr{L}(y_i, F(x_i))$, which measures the difference between predicted and actual values for each training instance, and the regularization term $\Omega(f_m)$, which penalizes complexity of the m-th tree in the ensemble. Here, $F(x_i)$ represents the current prediction of the ensemble model, and M corresponds to the current boosting round.

The regularization term, defined by Equation 3.2, quantifies model complexity, resorting to the number of leaves in the tree T and w, which represents the output values of the leaf nodes. Furthermore, it depends on γ, which acts as a complexity control, requiring a minimum loss reduction to allow further splits. Naturally, higher values lead to simpler trees. Meanwhile, λ is the penalty parameter applied to the leaf weights, encouraging smaller values to prevent overfitting.

XGB further distinguishes itself by utilizing a second-order Taylor expansion of the loss function to optimize the objective. This allows the model to consider both the gradient (direction of the error) and the Hessian (the curvature), adjusting updates based not only on how wrong the prediction is, but also on how rapidly the error is changing. This enhances optimization stability and precision during ensemble construction [33].

These theoretical design choices (second-order optimization and explicit regularization) allow XGB to efficiently balance accuracy and generalization. Beyond its theoretical appeal and its well-known performance accuracy, this algorithm offers several practical advantages. For instance, it requires minimal feature engineering, neither normalization nor scaling, while handling missing values effectively. Furthermore, its high speed and scalability across different machine learning tasks have made it a go-to algorithm for practitioners [33].

Furthermore, even though GBT, for instance, provides the ability to retrieve appreciated results regarding predictive performance, it might, depending on the size of the input dataset, prove challenging given its execution time. Therefore, reducing the number of data instances within the input is necessary. With that intent, this dissertation tested LightGBM. LGB achieves fast model training through two core innovations: Exclusive Feature Bundling and Gradient-based One-Sided Sampling. These techniques are designed to reduce computational overload while preserving the model's ability to capture complex patterns [36, 39].

EFB addresses the challenge of high-dimensional feature spaces by simultaneously combining mutually exclusive sparse features, features that rarely have non-zero values, into compact bundles. This reduces the overall number of features, improving efficiency without losing important information. Meanwhile, GOSS optimizes the boosting process by focusing on training instances that contribute the most to model updates. Instead of treating all samples equally, GOSS retains examples with large gradient values, which indicate poor prediction and carry more learning signal. A significant portion of instances with small gradients, which are already well-predicted, are removed from the computation of information gain. This selective sampling strategy reduces the number of training examples considered at each iteration, enabling faster computation while still

maintaining high-quality approximations of the loss gradient [36, 39].

Therefore, LGB is applicable to dense datasets and high-dimensional feature spaces, while still delivering accurate results in regression tasks. Nonetheless, the algorithm has limitations. Its leaf-wise growth strategy, where only the most promising leaf is split at each step (rather than growing all branches evenly), is highly effective for reducing loss but increases the risk of overfitting—particularly on small datasets. This contrasts with the more common level-wise growth used in traditional decision tree models, where the tree expands one full level at a time, growing all branches uniformly from the root downward. As a result, LGB tends to perform best in scenarios with large amounts of data, where its aggressive growth strategy can be better controlled through regularization [36, 39].

3.1.4 Hyperparameter Optimization

Although hyperparameters are of crucial importance to each model's performance, they are challenging to choose. For instance, the user may manually search for the best combination of values within the hyperparameter space or resort to more automated techniques, such as Grid Search and Randomized Search. The first systematically evaluates all possible combinations within the predefined set of hyperparameter values. This exhaustive search aims to identify the combination that yields the best model performance, meaning the lowest prediction error. No matter how promising this technique seems to be, caution is necessary, since the computational effort of the ML pipeline increases considerably with the size of the hyperparameter search space. [41, 42]

On the other hand, Randomized Search selects random combinations from the hyperparameter search space. Therefore, it focuses on computational efficiency rather than exhaustively searching, which leads to faster results. A degree of caution becomes necessary considering that this method may not guarantee comprehensive coverage of the entire search space [42].

However, Bergstra *et al.* derived a simple analytical expression for the expected success of random search in a hyper-parameter space of relative volume $\frac{v}{V}$. Relative volume within that space refers to the fraction of the total space V that corresponds to parameter configurations yielding high performance. Therefore, the probability of finding at least one good configuration in T trials is mathematically expressed by equation 3.3. Therefore, if one assumes that approximately 5% of the hyper-parameter combinations lie in a high-performance region ($\frac{v}{V} \approx 0.05$) and performs 250 iterations, then it would find a 99.9997% probability of sampling at least one-near near-optimal configuration [43].

$$P = 1 - \left(1 - \frac{v}{V}\right)^T \tag{3.3}$$

3.1.5 Integrating Evolutionary Forests for Feature Engineering in Decision Trees

Feature engineering is the deliberate transformation of an original feature space into a new one, seeking to allow a learner to capture relationships that the raw variables cannot express. The

Evolutionary Forest (EF) algorithm aims to evolve symbolic features, centring its approach around a multi-tree genetic programming representation [44, 45].

Genetic programming is an evolutionary algorithm in which every individual is a computer program represented as a tree that mutates and recombines under Darwinian pressure. Each individual is a vector $\Phi = [\phi_1, ..., \phi_n]$ of GP trees (a gene), where each ϕ_i represents a newly engineered feature. Collectively, those trees define an m-dimensional feature set that will be fed to a decision tree. The population is initialized by a classic half-and-half method, where half of the genes are fully grown trees while the other half is grown irregularly [44].

Evolution proceeds through the generation of offspring through two classic GP operators: crossover and mutation. Subtree crossover selects a random sub-tree in each of two parents and swaps them, thereby exchanging algebraic sub-expressions between individuals. On the other hand, sub-tree mutations replace a random parent's subtree with a freshly generated tree. This offspring introduces behavioural diversity. To preserve that diversity, duplicate individuals are rejected [44].

Next, a fitness test is elaborated. For each candidate feature set, an individual Φ, EF grows a random decision tree, whose split variables are exactly ϕ_i. A five fold cross validation test (4 for training, 1 for validation) occurs to assess the absolute error for every held-out observation, concatenating those n values into a fitness vector L rather than collapsing them into a scalar - the information level for each exemplar is stored in memory, since it is required for selection [44].

Parent selection is performed with automatic ε-lexical selection. At each draw, a random order of case indices is sampled, and the corresponding individual is retained if their loss lies within a median absolute deviation threshold, thereby favouring solutions that excel on different hard cases and thus promoting behavioural diversity. To be clear, firstly, the algorithm shuffles the exemplars to ensure diversity, forcing different individuals to excel on different hard cases. Once shuffled, for each exemplar j it finds the best error $best_j = \min_i l_{ij}$ among the current individuals i. Then, it computes the tolerance ε, as the median absolute deviation of the current individuals on exemplar j. Afterwards, it proceeds to filter survivors, keeping only those individuals whose error on case j is within the tolerance, $l_{i,j} \leq best_j + \varepsilon_j$. If the survivors are just one, then it stops. If not, it restarts with a different exemplar from the shuffled index set, repeating this process. Once the survivor is picked, it is selected as a parent [44].

Once the parent has been chosen, the algorithm preserves its most successful feature sets through an elitist external archive A. A has a limited capacity that stores the complete genotype - the multi-tree GP individual - together with its fitness vector. If there is available space within the archive, the newly created individual is appended directly. Otherwise, the algorithm compares the newcomer with the worst individual in the list, keeping whichever of the two is better. This mechanism is often referred to as elitism, aiming to ensure that better-performing individuals are not lost. At the end of the evolution, the archive's content directly becomes the trees that form the final Evolutionary Forest [44].

Finally, the algorithm goes through the archive, selecting and retrieving each feature expression from each gene from each individual. Then, it follows the training of a Decision Tree model

and assess whether a performance improvement will occur or not.

3.2 Explainability and Interpretability of Decision Trees

Even though ML algorithms are powerful given their predictive performance, these algorithms are typically non-interpretable. Therefore, these often function as black-box models, i.e., their internal mechanisms and reasoning through which a solution is formed are challenging for the general user to grasp [16, 46–48].

Previously, this document discussed how modelling EV charging tariffs through Decision Trees enables the reader to perceive and follow the decision-making process behind these prices. Naturally, the Decision Tree structure may be translated to a set of if-then rules. This property highlights the inherent clarity this ML algorithm presents to the end-user, since it allows decision-making readability. Consequently, decision trees, among ML algorithms, are naturally interpretable and thus more accessible to the average end user. [19, 32, 33, 35].

In spite of this, modelling complex data patterns through Decision Trees entails certain risks. In addition to overfitting, and despite being occasionally correlated, predictive performance and model interpretability often represent opposing objectives. Often, ML programmers face decisions about whether to prioritize predictive accuracy or model interpretability [19].

The most straightforward example of such a decision would be the definition of the hyperparameter that relates to a tree's maximum depth. Increasing depth, with particular attention to the possibility of overfitting, may end up improving the model's performance. On the other hand, such depth creates a less interpretable model, since following the decision-making process through dozens of internal nodes becomes a challenging task.

Therefore, it is essential to strike a balance between predictive performance and interpretability, ensuring the model captures the correct relationship between the target variable and features while remaining accessible to the general user, thus fostering its adoption. However, it is important to note, once again, that in the context of this research, greater emphasis is placed on interpretability rather than predictive performance.

Additionally, this research aims to go beyond merely leveraging the natural properties of Decision Trees. Specifically, by further exploring post-hoc explainability, this research introduces the usage of Large Language Models and their capabilities to create natural language textual explanations to enhance comprehension by the average end-user further. Given the simplicity of this step, this section places less emphasis on it.

3.2.1 Decision Tree

A single Decision Tree is inherently interpretable, as it allows the user to trace the decision-making process from the root node to the corresponding leaf node, where the prediction is made. However, another key consideration is how to properly represent the tree in such a manner that maximizes accessibility for the end user. For instance, *scikit-learn* package [49] provides built-in

tools for Decision Tree visualization, which this work showcases in Section 5.2.5. Furthermore, the *supertree* package [50] extends such functionality by offering enhanced visual representations along with additional features and customization options that may improve understanding from a programmer's perspective, but risk confusing general users.

Furthermore, single decision trees are weak learners. Therefore, the trade-off between performance and interpretability tends to bend towards the latter. Usually, a single Decision Tree is more interpretable than it is precise.

3.2.2 Ensemble

Ensemble algorithms consist of multiple individual decision trees that aim to improve overall predictive performance. However, such capabilities come accompanied by their black-box nature, i.e., low interpretability [19, 33, 51].

Recalling the objective at hand, which is to have an interpretable model, a natural question arises: how can the decision-making process of ensemble models be effectively represented? Indeed, given that these algorithms resort to multiple individuals to generate their final predictions and that they were not developed for interpretability, bringing the corresponding decision structure is not straightforward.

The first approach developed was to select the Decision Tree based on individual prediction accuracy for each tree within the ensemble. However, arguing that the tree with the highest prediction accuracy is the most representative out of the entire ensemble population is questionable. While such a tree may perform well on the test set, it does not necessarily reflect the typical behaviour or decision patterns of the ensemble as a whole. Given the tendency of individual Decision Trees to overfit, a high accuracy may reflect overfitting rather than a faithful representation of the ensemble's behavior. Ensembles, as mentioned earlier, use diversity to reduce overfitting. Therefore, the best prediction could reveal itself as a poor surrogate for explaining the collective logic of the ensemble.

For instance, Aria *et al.* pursues interpretability within ensemble-based learning for classification problems by introducing the E2Tree method. Rather than selecting an existing tree from the ensemble, E2Tree builds a surrogate model designed to approximate the global behaviour of a Random Forest. Basically, this algorithm builds a new Decision Tree by recursively partitioning the input space based on a dissimilarity matrix, where pairwise distances reflect how often observations co-occur in the same terminal nodes across the ensemble. The resulting tree offers interpretable if-then rules while maintaining the same predictive accuracy as the ensemble. Later on, the authors extend the E2Tree framework to regression settings. This version is adapted to account for continuous outcomes, using a node-level normalized mean squared error to weight co-occurrences between observation pairs [52, 53].

Furthermore, Banerjee *et al.* propose a method for selecting representative trees that approximate the collective behaviour of an ensemble model, without sacrificing the predictive diversity that ensembles typically offer. Rather than focusing on tree structure or accuracy, their methods define similarity between trees based on the agreement of their predictions. It applies a pairwise

similarity matrix along with clustering, therefore partitioning the ensemble into behaviourally coherent groups. From each cluster, a medoid tree is selected as representative, the tree whose predictions are the most central within the cluster. This enables the user to have a set of real trees that reflect different reasoning present in the original ensemble. The usage of real trees already present in the ensemble ensures faithfulness to the model's original decision space [54].

Moreover, Weinberg *et al.* suggests a framework named *SySM*, the Syntactic Similarity Method, aiming, as well, to select a single representative Decision Tree from a large ensemble. In turn, it relies on comparing the syntactic structure of each of the ensemble's individuals by encoding trees into a Bracket Tree Format (BTF). BTF is a string that captures the hierarchical structure of a Decision Tree, which allows the authors to compute pairwise distance between trees. This method quantifies the number of operations (insertions, deletions, substitutions) necessary to transform one tree's structure into another. By aggregating these distances, the algorithm selects the tree that is most centrally positioned in the ensemble's syntactic space, i.e, the one with the lowest average RTED to all others. Results showed robust performance and interpretability [51].

Table 3.2 summarizes the description of each algorithm.

Table 3.2: Comparison of Interpretability-Oriented Ensemble Methods.

Method	**SySM [51]**	**Representative Trees [54]**	**E2Tree [53]**
Core Idea	Select a tree structurally most similar to others using RTED over BTF	Select cluster medoids based on prediction similarity	Build a surrogate tree from ensemble-induced dissimilarity
Type of Output	One real tree from the ensemble	A few real trees from the ensemble	A new decision tree (not part of the original ensemble)
Similarity Basis	Tree structure only (syntactic)	Prediction agreement (semantic)	Co-occurrence in terminal nodes + prediction distance
Data Needed	No prediction or label data required	Predictions on a validation set	Predictions + internal ensemble structure
Supports Regression	Yes	Yes	Yes (in extended version)
Interpretability	High (single real tree)	Medium (small set of real trees)	High (interpretable surrogate tree)
Scalability	High (lightweight, parallelizable)	Moderate (requires prediction matrix)	Moderate (requires dissimilarity matrix computation)
Fidelity to Ensemble	Moderate	Good within clusters	High (global fidelity)
Python Availability	Easy to implement	Easy to implement	Only available in R

This work implemented the method outlined in article [51], from scratch, due to its straightforward integration into the ML pipeline and its reliance on structural and prediction-based tree comparison techniques, respectively. This implementation aims to offer an efficient path toward improving the ensemble transparency in post-hoc analysis.

Chapter 4

Methodology

4.1 Previous Work as Foundation

This research aims to extend the approach proposed by Silva *et al.*, which focused on designing dynamic carbon-aware electric vehicle charging tariffs (day-ahead) through stochastic optimization (i.e., explicitly incorporating uncertainty in key input variables) while briefly shifting attention to the explainability of such tariffs, resorting to ML algorithms to build a surrogate model and afterwards explaining it applying SHAP [4].

The optimization model proposed in the mentioned paper is a crucial component for the development of this dissertation, as its output serves as the basis for generating the input dataset fed into the ML pipeline. Without focusing too much on the model's details, which are carefully described in the referred paper, it is nonetheless essential to acknowledge its capability to consider RES, load, and grid carbon intensity forecast uncertainties while accounting for the price elasticity of demand. In addition, the authors introduced a chance constraint in the optimization problem, which aims to model the probability of complying with an indirect daily carbon emission budget in an EV charging station.

Its main conclusions include, for instance, that the stochastic approach leads to improved outcomes compared to a deterministic counterpart, achieving nearly 24% reduction in indirect carbon emissions per feasible day. Furthermore, the usage of the chance constraints has been shown to enable flexible compliance with daily carbon emission budgets. Additionally, including customer sensitivity to price variation also revealed itself to be critical, even though further research on how to correctly model customer behaviour would improve such results. Finally, resorting to SHAP enhanced the transparency of the model, revealing the features that impacted the outcome the most, as the hour of the day and forecast uncertainty.

Established the reliance on the stochastic optimization framework proposed within article [4], which underpins the data generation stage of this work, the following sections present the methodology adopted in this dissertation, highlighting the steps taken to extend and complement the original approach.

4.2 Methodology Overview

Figure 4.1 summarizes the approach followed during this dissertation. As previously stated, this study aims to address the black-box problem, therefore enhancing the transparency behind pricing mechanisms.

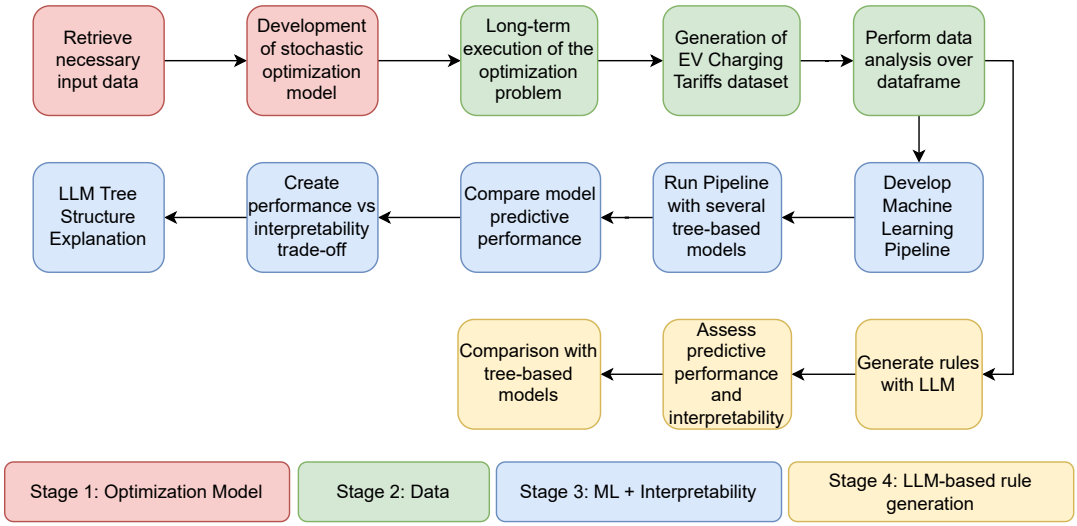

Figure 4.1: Overview of the methodology applied to this dissertation.

Firstly, stage 1 resorts to the framework defined in article [4]. Such a model is extremely important to the development of this dissertation since its extension and long-term execution will later enable the generation of the dataset, which is the basis of this work. Afterwards, a quick analysis is performed over the newly formed dataset, where the aim is to highlight the correlation between different features, as well as between the features and the target variable (stage 2). Further information regarding the composition of the dataset is available in Section 4.3.

Then, the workflow is divided into two different stages. Primarily, an ML Pipeline is developed (stage 3), resorting to different tree-based ML algorithms, aiming to predict the EV Charging Tariffs and ultimately assess the trade-off between interpretability and predictive accuracy, while briefly resorting to an LLM to explain in natural language the decision-making process. Secondly, this thesis intends to explore LLM's capability to generate rules (stage 4). Therefore, stage 4 seeks to foster the application of generative AI, assessing the readiness of current LLMs for this type of task and, ultimately, whether a general user could rely on them to perform these predictions without requiring expertise in either machine learning or energy systems.

Overall, this research stands out by integrating AI into both the creation and interpretation of predictive rules for EV charging tariffs, thereby promoting greater transparency and accessibility for everyday EV users and charging station operators. The forthcoming sections describe in detail the methodology applied to both the ML-based approach and the novel LLM-based Rule Generation framework.

4.3 ML-based Approach

This section provides an overview of the ML pipeline developed and applied to predict EV charging tariffs. Since interpretability is the primary focus of this work, tree-based models were chosen to execute this task, due to their inherently accessible nature for the general user. However, tree-based models could be divided into different categories: single trees or ensemble methods. Section 3.1 reviewed each model's theoretical background.

ML pipelines are structured and modular workflows (i.e., a sequence of processes) that transform raw data into results through model training, evaluation, and deployment. This kind of approach enables reproducibility, scalability, modularity, and efficiency [55]. Figure 4.2 illustrates each step of the ML pipeline, including xAI techniques, which are not a required component but were applied in this study.

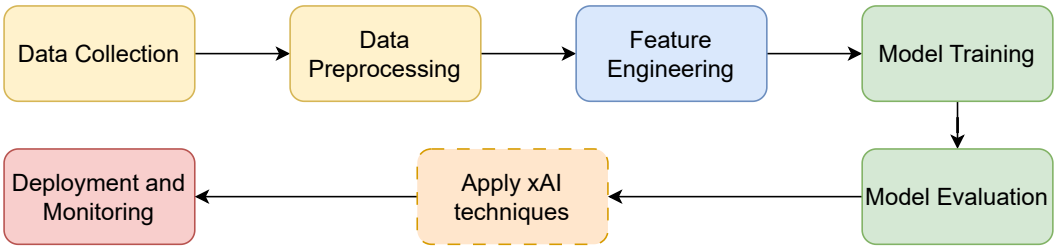

Figure 4.2: Typical ML Pipeline. xAI techniques are included due to the nature of this dissertation.

Hence, this process begins with data collection. As previously stated, this work resorts to a framework developed by Silva *et al.*, which was applied to a case study in Portugal. Table 4.1 describes the input data fed to the model and respective sources [4]. Moreover, Figure 4.3 illustrates the main building blocks of the EV charging station day-ahead carbon-aware dynamic pricing scheme, representing the data flow and functional architecture of the system used to determine charging prices, sensitive to both carbon intensity and grid conditions.

Table 4.1: Optimization Model Input Data and Sources Used.

Data Type	**Description and Source**
EV Charging Load	Metering and transaction data from a semi-public EVCS in northern Portugal; resampled to hourly resolution; data from Feb 1 to Oct 31, 2024.
Price Elasticity of Demand	Modeled using a Price Elasticity Matrix (PEM) with scaling factor $k = 5$ to match realistic driver responses.
PV Generation	Real hourly data from a 16 kWp PV system in Porto, scaled to simulate 250 kWp capacity.
Grid Carbon Intensity	Derived from historical electricity mix data from the Portuguese TSO and emission factors per source; forecasts from external provider.
Weather Data	Precipitation and temperature from WRF via MeteoGalicia THREDDS; irradiance and cloud cover from Open-Meteo.
Electricity Prices	Historical market prices retrieved via a public API from the Iberian Electricity Market (MIBEL).
Carbon Emission Costs	Set at €83.24/tonne CO_2, based on 2023 EU Emissions Trading System average price.
Simulation Configuration	500 statistical scenarios per day; tariffs range from €0.05 to €0.60/kWh with a base of €0.30; 20% chance constraint violation threshold.

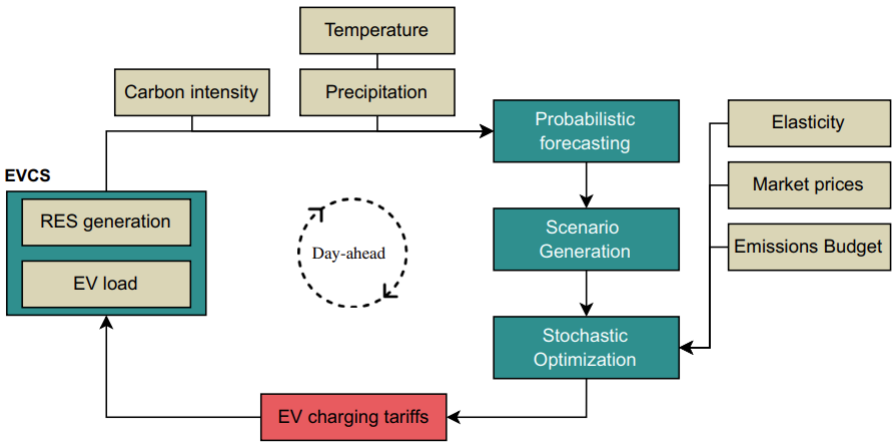

Figure 4.3: Building blocks of EVCS day-ahead dynamic pricing service, retrieved from [4].

Afterwards, once the data is collected, it needs to be processed. Essentially, this refers to an effort to create a dataset to act as an input to the ML pipeline. Therefore, during this dissertation, the optimization problem considered several values for the price elasticity of consumers as its input parameter. This allows to cover a more diverse range of behaviour from EV drivers, but also to extend the ML dataset in size. The optimization problem was executed sequentially for each

day considered, and each day considered multiple instances with different price elasticity factors. Table 4.2 defines each feature (as well as the ML algorithm's prediction target, Tariffs) that makes up the final dataset, which was generated after executing the optimization framework, thereafter containing 14,904 entries. Beware that for this work, in particular, the budget type $q75$, meaning that the emission budget for a given day is set to a value that is the 75 % quantile of the day-ahead predicted scenarios for carbon emissions.

Table 4.2: Description of features used in the ML pipeline.

Feature	Description
Elasticity Indicators	
Elasticity	Sensitivity of demand to tariff changes.
Self-elasticity	Product of elasticity and scaling factor; reflects demand sensitivity to tariff changes.
Temporal Features	
Hour, Day, Weekday, Month	Time indicators: hour of the day, day of the month, day of the week, and month of the year.
Scenario Forecast Statistics	
Mean (Load, RES, CI)	Average of load demand, renewable energy supply, and carbon intensity forecasts across scenarios.
IQR (Load, RES, CI)	Interquartile range of load demand, renewable supply, and carbon intensity forecasts.
Range (Load, RES, CI)	Difference between maximum and minimum forecasted values for load, renewable supply, and carbon intensity.
CV (Load, RES, CI)	Coefficient of variation: standard deviation divided by the mean for each forecasted variable.
External and Operational Variables	
Market Prices	Day-ahead electricity price (EUR/kWh) for each hour.
CE Budget	Daily carbon emissions budget (gCO_2) for the charging station.
Tariffs	Charging tariff (€/kWh) applied in each time period.

Usually, the data set needs to be processed to handle missing values, outliers, and inconsistent formats, all of which could impair the model's performance. However, as this work generated the utilized dataset and predictive performance is not its main focus, techniques such as normalization or encoding of categorical variables were not applied to preprocess data. However, this dissertation recognizes the importance of such a step, since even the most advanced algorithms can yield suboptimal results if trained on poorly curated inputs.

Then, the pipeline highlights the feature engineering step. This step involves selecting, transforming, or even constructing new variables that better capture the underlying patterns within the data. The core intention behind this stage is to improve the predictive power of the model, building on top of the raw features provided in the dataset [55]. However, this dissertation does not focus on feature selection, but rather on developing a brief evolutionary feature engineering step, whose theoretical background has been previously described in Section 3.1.5.

Subsequently, the model training phase uses processed data as input to a specific ML algorithm under study, aiming to minimize a loss function that measures the model's prediction error [55]. At this step, the dataset is partitioned into training and testing sets to ensure that the model can generalize well to unseen data, hence reducing the risk of overfitting, a critical concern when applying an ML model to a particular problem. Additionally, as this dissertation generated a time-series dataset, an essential step is the necessity to ensure that temporal dependencies are preserved. Such a constraint implies that the dataframe must not be shuffled during its split into the training and testing sets. Consequently, these subsets must be sequentially split to maintain the chronological order of observations and prevent future information from inadvertently influencing model training, a well-known issue commonly referred to as data leakage.

It is also during this stage that model optimization takes place through the use of hyperparameter tuning techniques such as Grid Search and Randomized Search. Section 3.1.4 focuses on describing each optimization technique. To ensure a robust evaluation during the optimization process, while respecting the temporal structure of the dataset, cross-validation is performed using the TimeSeriesSplit method [49]. This approach aims to preserve the sequential nature of the dataset across validation folds, ensuring that the model's generalization capacity is assessed under realistic, future-oriented conditions.

Following the training of the ML model, rigorous evaluation is conducted to assess the model's performance [55]. In this work, MAE (eq. 4.1) was the main metric employed for model evaluation. Wherever possible, models were optimized specifically to minimize MAE, as it provides a direct and interpretable measure of average prediction error in the same units as the target variable. Other regression metrics, such as mean squared error, were explored during model optimization. However, this work prioritizes MAE due to its interpretability. Additional metrics, such as mean squared error and R^2 score, exist but were not applied in this work.

$$\text{MAE} = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i| \tag{4.1}$$

Since this dissertation aims to explain/interpret ML models and, ultimately, foster user trust, post-hoc interpretability tools become an integral (yet not mandatory) element of the pipeline. Hence, techniques such as the reviewed SHAP and LIME could be used to such an effect. This work, however, does not feature these strategies, since it focuses on models that are inherently interpretable. For instance, models such as Decision Trees are at their core interpretable, by enabling the user to follow the model's decision-making process, therefore achieving much more clarity than either of those two post-hoc methods could achieve. Still, resorting to similar approaches to assess, for instance, which features impact the final solution the most could be useful, even for model development purposes. Furthermore, this work incorporated interpretability metrics into this ML pipeline, thereby enabling the model's evaluation from both a predictive and interpretability perspective.

In addition, the persistence of key components, such as trained models, evaluation metrics, and intermediate results, emerges as a critical component of the ML pipeline. Such caution enables re-

producibility, comparative analysis, and rollback to previous versions in the event of failure. This persistence is often implemented using serialization techniques such as the Python Pickle Module [56], which allows models to be stored in binary format and reloaded without retraining. In addition, performance metrics and experiment metadata are typically saved in structured formats like CSV. This step not only safeguards intellectual and computational investment, but also reinforces the integrity and accountability of the overall machine learning lifecycle [55].

Once the performance of the model is deemed satisfactory, it is then deployed through a service-oriented architecture or embedded within a larger decision-support system. Since real-world environments are dynamic, continuous model monitoring is required regarding data drift and performance degradation [55]. However, such a step is outside the scope of this dissertation and is not addressed.

In summary, it becomes clear that an ML pipeline is an iterative framework that spans from data acquisition to explainable deployment. Each pipeline stage builds upon the previous, forming a pathway that transforms raw information into interpretable predictions. Its development becomes indispensable for developing a transparent, robust, and sustainable AI system. In the framework developed in this work, the pipeline was designed to incorporate the already reviewed tree-based ML models, ensuring flexibility and adaptability, thereby enabling a systematic performance comparison between models.

4.4 Large Language Models based Approach

The following sections focus on the methodology applied to retrieve a natural language explanation of a DT, as well as a rule-based prediction through an LLM.

4.4.1 Tree Explanation

Figure 4.4 presents the methodology for tree-based model explanation through LLMs, where the latter is used to essentially translate the logic of Decision Trees into human-readable text. The process begins by training a single Decision Tree model, followed by the extraction of its structural representation. This structure is then incorporated into a carefully designed prompt, which is submitted to an LLM to generate a natural language explanation.

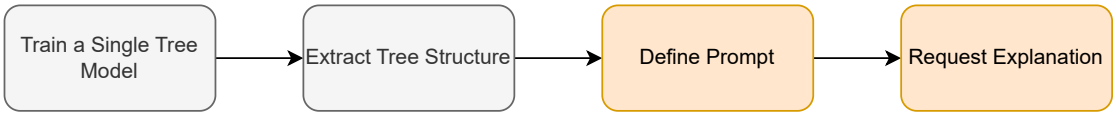

Figure 4.4: Workflow applied to retrieve a single tree explanation.

The prompt described below isolates expertise, context, goal, requirements, chain-of-thought, and inputs into numbered sections so the model reasons systematically while hiding technical detail from readers. By instructing threshold translation into everyday familiar terms and banning ML jargon, it should convert split logic into user-friendly language. The end goal is to create a

narrative that summarizes the decision-making process and that highlights important features for EV charging tariff predictions.

```
tree_explainer_prompt = """
## Expertise
You are a data storytelling specialist who converts technical  D e c i s i o n TreeRegressor  models into clear
     , b u s i n e s s friendly prose that anyone can follow.

## Context
1. You will receive:
   1. A raw text dump of a trained  s c i k i t learn  DecisionTreeRegressor.
   2. A dictionary mapping every potential feature name to a short, plain description.
2. Some features in that dictionary may not appear in the tree.
3. The explanation is for everyday  e n d users  of  e l e c t r i c vehicle (EV) charging services, who are
     unfamiliar with  m a c h i n e learning  details and should not be exposed to technical jargon.

## Goal
Provide a concise,  e a s y to grasp  narrative of what drives the model's predictions of EV charging
     tariffs, highlighting the overall decision logic and the most influential features.

## Requirements
1. Use everyday language; avoid ML terms such as "node variance," or "entropy."
2. Summarize how the model generally splits data (e.g., "The tree first looks at the time of day, then
     considers station occupancy").
3. Emphasize the features that matter most across the entire tree rather than detailing every split or path
    .
4. Translate thresholds into  r e a l world  terms.
5. Keep the narrative to roughly 3 paragraphs.
6. End with one clear "big takeaway" sentence for  d e c i s i o n makers and EV drivers.

## Chain of Thought  (hidden from the reader)
1. Parse the tree text into nodes, thresholds, and leaf predictions.
2. Map each feature code to its human description.
3. Draft a concise narrative focusing on overall structure and top features.
4. Review for clarity and user relevance, then output.

## Inputs
1. Feature Description
    "Self-elasticity": "A measure of how sensitive energy demand is to changes in the tariff.",
    "Hour": "Hour of the day.",
    "Day": "Day of the month.",
    "Weekday": "Day of the week.",
    "Month": "Month of the year.",
    "Mean (Load)": "Average energy load or demand forecast (kW) over a set of scenarios for EV charging.",
    "Mean (RES) ": "Average renewable energy supply forecast (kW) over a set of scenarios for solar PV
         generation.",
    "Mean (CI) ": "Average carbon intensity forecast (gCO2/kWh) over a set of scenarios for the  energy
         consumed from the electrical grid.",
    "IQR (Load)": "Interquartile range of energy load (kW) forecasted scenarios.",
    "IQR (RES) ": "Interquartile range of renewable energy (kW) forecasted scenarios.",
    "IQR (CI) ": "Interquartile range of carbon intensity (gCO2/kWh) forecasted scenarios.",
    "Range (Load)": "Range (max - min) of energy load (kW) forecasted scenarios.",
    "Range (RES) ": "Range (max - min) of renewable energy supply (kW) forecasted scenarios.",
    "Range (CI) ": "Range (max - min) of carbon intensity (gCO2/kWh) forecasted scenarios.",
    "Market Prices": "Day-ahead price of electricity (EUR/kWh) in the market for the given hour.",
    "CE Budget": "Total daily carbon emissions budget (gCO2) that should be not be surpassed by the
         charging station.",
    "Tariffs-Lag": "Lagged value of the Tariffs feature, shifted by 24 hours to account for daily patterns
         in energy consumption.",

    Target:
    "Tariffs": "The charging tariff (EUR/kWh) assigned to consumers in each time period.",
2. {tree_text}
"""
```

4.4.2 Rule Generation

Nowadays, various algorithms facilitate the prediction of target variables by leveraging patterns extracted from data. This dissertation uses ML models, specifically tree-based algorithms, to reach that goal. However, effectively implementing these models requires users to have programming experience, both in general-purpose programming languages and in understanding the

models themselves.

The algorithm outlined in Figure 4.5 enables the iterative use of an LLM for making predictions, which is expected to work beyond this optimization-generated EV Charging dataset, as well as for other regression or classification tasks, without requiring prior domain or programming knowledge.

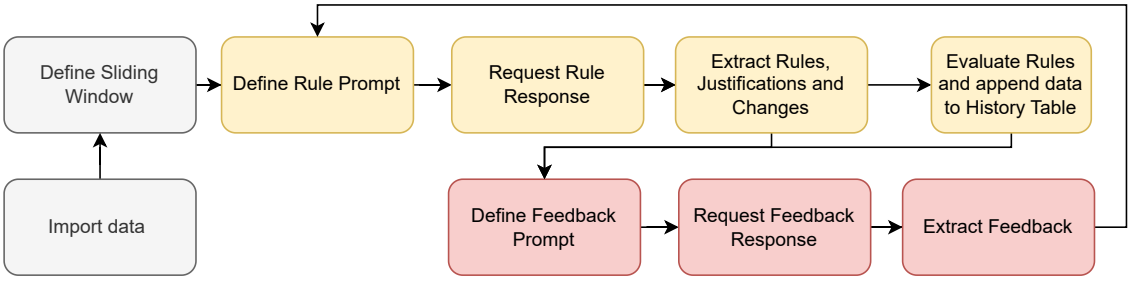

Figure 4.5: LLM-Rule generation workflow.

Since LLMs are constrained by a token limit, which restricts the maximum length of the input prompt, a sliding window approach is applied to a predefined unshuffled training set, allowing a portion of the data to be updated at each iteration. Then, the Rule Generator Prompt must be defined.

The process starts by assigning the LLM a specific role: an expert in rule-set generation. Next, the contextual information is provided, including the current iteration number, the target variable and descriptions of features. The training set, along with a summary of its contents, is also included within the prompt. Such is followed by prior knowledge from earlier iterations, specifically, the best-performing rules so far and the rules proposed in the most recent iteration, along with their respective MAE. The LLM is then explicitly tasked to produce a structured output containing a justification for the proposed rule-set, a Python function that generates those rules, and descriptions of the changes made from the previous to current iteration. To ensure extraction and execution capabilities, constraints are imposed on the output format. Finally, a Chain of Thought is recommended to the LLM.

Below, the Rule Generator prompt has been transcribed.

```
rule_prompt = """
You are an expert in generating rule-sets (pure nested if/else) to predict a **numerical** target.

## Context
    Iteration: {ITERATION_NUMBER}
    Target variable: **Tariffs**
    Feature descriptions:
{FEATURE_DESCRIPTIONS}

### Data
    Training set (sliding window varying each iteration):
{TRAINING_SET}

    Dataset summary (mins/max/average...):
{TRAINING_SET_DESCRIPTION}

### Prior knowledge
```

```
    Rules from the previous iteration (MAE ={MAE}):
{RULES}

    Best rules so far (MAE ={BEST_MAE}):
{BEST_RULES}

    Peer-LLM feedback you **must** address:
{FEEDBACK}

## Your task
Generate a new rule-set that lowers MAE further.
Return **exactly two things**:
    1. A short free-text justification, enclosed by [Start of Justification] and [End of Justification]
         tags.
        It should contain:
        "Nature": Wonder about the nature of the dataset, how is it structured. Is it a time series? What
             are the relationships between features?
        "Patterns": Analyze the dataset to identify patterns and relationships between features and the
             target variable, as well as among the features themselves.
        "Feedback to implement": Across iterations, prioritize implementing instructions from the peer-LLM
             feedback.
        - Incorporate your insights with the best rules so far, and peer feedback to refine and improve the
              new rule-set.
    2. A **single** Python function named `rule_based_prediction(row)` wrapped in
    "`python
    <code>
    ```
    After the closing back-ticks, output as below:
    [Start of Changes]
    <summary of every added / modified / removed condition, semicolon-separated,   180 chars>
    [End of Changes]
    Example:
        [Start of Justification]
        Justification of the rule-set.
        [End of Justification]

        "`python
        {required_prompt_structure}
        ```
        [Start of Changes]
        [text]
        [End of Changes]

## EXTREME IMPORTANCE     Hard Constraints
1. **NO `elif` statements**     only `if` / `else` blocks (nested tree, structurally similar to a Decision
     Tree Regressor).
2. **Always** enclose the entire code in **exactly one** ```python     ``` block and **close it**.
3. The function should always a return default value, if no conditions are met.
4. The "Changes made" comes **after** the closing ``` back-ticks.
5. Use **only** features listed in *Feature descriptions* with ***case-EXACT, space-EXACT spelling***
   (e.g. `Self-elasticity`, *not* `self_elasticity`, `Self-Elasticity` or any variant).
6. No imports, no external calls, no forbidden functions (`quantile`, `eval`, etc.).
7. Predictions must stay within the historical tariff range given in the given description.
8. No target leakage: never use `row['Tariffs']`.
9. Code must exec() without edits or extra context.

## Chain of Thought
    Think step-by-step, like a human would.
    Use the provided feature descriptions to understand the data, and comprehend the environment the
      dataframe is in.
    Use the provided training set to identify relationships between features themselves and the target
      variable.
    Consider the previous and best so far rules to improve the new rule-set, but most importantly, feedback
       from the peer-LLM.

Compliance is evaluated **solely** on the code you output in this response.
"""
```

After this definition, a response from the LLM is requested via API. If this answer follows the mandatory requirements for extraction, then the rules are executed, predictions are obtained, and the MAE is computed. Furthermore, both Justifications and Change Made are stored, the former for later review and the latter to be included in a history table, which is then incorporated into the feedback process.

Afterwards, the Feedback prompt must be defined. Similarly to the previous prompt, both

expertise and context are given, along with the current training chunk and description. Furthermore, current rules, previous rules, and the best rules so far are incorporated, enabling the LLM to analyse patterns across different solutions and compare them. Moreover, a History of Changes table is incorporated as well, describing changes performed from one iteration to another across the previous 10 iterations.

Additionally, the worst prediction instances from the current window are also displayed, so that the LLM recommends direct changes to tackle those poorly predicted cases. Finally, a Chain of Thought is proposed by imposing a step-by-step, structured response. A compliance assessment is employed, where the Feedback LLM checks if the Rule Generator LLM tightly followed the hard constraints.

The prompt, for the Feedback stage, is directly transcribed below.

```
    feedback_prompt = """
    You are an expert in providing feedback on rule based prediction functions. The LLM who generated
         them aimed to predict EV Charging Tariffs based on the features and on the provided training set.
You are part of a cyclical process, and currently you are in iteration {ITERATION_NUMBER}.

The features are described as follows:
{FEATURE_DESCRIPTIONS}

The current training set for the rule generation is as follows:
{TRAINING_SET}

The dataset is described as follows:
{TRAINING_SET_DESCRIPTION}

Your job is to analyse the provided function and give feedback that will improve its accuracy.
Such feedback will be sent to the rule generator LLM.

# Rules and metrics
    Current rules (MAE = {MAE}), from LLM iteration {ITERATION_NUMBER}:
    {RULES}


    Previous iteration (MAE = {PREVIOUS_MAE}, iteration {PREVIOUS_ITERATION}):
    {PREVIOUS_RULES}

    Best rules so far (MAE = {BEST_MAE}), iteration {BEST_ITERATION_INDEX}:
    {BEST_RULES}

# History of Changes
    {HISTORY_TABLE}

 ***If the MAE rises versus the previous iteration, set Compliance = FAIL and tell the rule-generator to
      undo those specific changes listing each one it must revert in the next round.***

# Worst Predicting Instances
    {WORST_PREDICTING_INSTANCES}

# Provide feedback
1. Compare current, previous and best rules.
2. Analyse history and MAE impact.
3. Recommend to be less conservative if MAE stagnates for more than 4 iterations.
4. Recommend concrete rule improvements (max 10 items).
5. Check compliance:
       No `elif`.
       Nested tree only.
       No forbidden functions/imports.
       Comments on every split.
       Predictions within tariff range.
       **Changes made** present and either lists modifications.
       No leakage: *It does not use row['Tariffs'] to predict row['Tariffs']*.

Finish with `Compliance: PASS` or `Compliance: FAIL` plus a 0  100  score."""
```

Finally, once the feedback request is completed, the text string is included within the Rule

Generator prompt of the next iteration, thus continuing the iterative cycle until the stop criteria is met: a minimum MAE, or a fixed number of maximum iterations.

Therefore, this framework aims to iteratively reduce MAE, primarily by leveraging a feedback loop and a rolling dataset. The experiment aims to evaluate whether LLMs are capable of generating good rule-based predictions and how their performance compares to that of traditional algorithms. Such an approach could prove highly valuable, as it reduces the need for domain expertise in typically complex predictive tasks and develops an innovative strategy to build another surrogate of an optimization model. Moreover, the proposed methodology is versatile and is applicable across a wide range of fields.

4.4.3 LLM and computational infrastructure

The first, more brief experiment, presented in Section 4.4.1, resorts to OpenAI's o3 model, described as their most powerful reasoning model. Along with high reasoning capabilities across multiple domains, the model offers a 200 thousand token context window, and supports up to 100 thousand output tokens, accepting both text and image inputs, while outputting in text only. Despite being their slowest model, the model provides great performance, especially in multi-step problem-solving, technical writing, and instruction-following [57]. These characteristics make the model particularly suitable for translating Decision Tree structures into detailed explanations that non-experts can easily understand.

The experiment described in Section 4.4.2 relies on DeepSeek-R1-Distill-Qwen-32B (DS-R1-Q32), a publicly released LLM with 32 billion parameters. Such a model adapts a decoder-only Transformer configuration, meaning that it is trained to predict the next token given all previously generated tokens. Therefore, this LLM is optimised for left-to-right text generation, having the ability to generate outputs of up to 32768 tokens [58, 59].

DS-R1-Q32 was produced through supervised distillation applied to the Qwen 2.5 backbone, an architecture that is widely used in the research community. During this process, the 671 billion-parameter DeepSeek-R1 teacher solved several problems, enumerating each intermediate step. A complete, ordered sequence of such steps is designated a reasoning trajectory, or chain-of-thought. Approximately 8×10^5 trajectories were used as paired training data, comprising the original prompt and the corresponding trajectory. Fine-tuning Qwen with this approach enables the student model to replicate the teacher's problem-solving procedure, at a fraction of the computational resources required by the original models [58, 59].

Benchmark results confirm the effectiveness of this LLM. The distilled model attains a 72.6% pass@1 on the AIME-2024 mathematics set, 94.3% on MATH-500, 62.1 % on GPQA-Diamond, 57.2% on LiveCodeBench, and a Codeforces rating of 1691. These scores equal or exceed those of OpenAI's o1-mini baseline while requiring roughly half as many parameters, indicating a favourable trade-off between accuracy and resource consumption [58, 59].

For this experiment, simulations were carried out on a single NVIDIA H200 Tensor Core GPU fitted with 141 GB of high-bandwidth memory (HBM3r VRAM). As the LLM generates answers, it relies on vLLM, which is a lightweight, open-source tool that feeds the user prompt to the GPU

in an orderly way, thereafter returning the responses quickly. More information related to the hardware used is described in Table 4.3.

Table 4.3: Virtual-machine hardware.

Component	**Description**
Platform	system OpenStack Novabridge 440FX
CPU	Intel Xeon Processor (IceLake)
Memory	64 GiB System Memory
GPU	Virtio GPU + NVIDIA Corporation
Network	Virtio network device (ens3)
Disk	Virtio block device /dev/vda – 322 GB

Chapter 5

Results and Discussion

5.1 Exploratory analysis of the dataset

In order to better understand the linear dependencies among the input variables and their relation to the prediction target, a *Pearson* correlation heatmap (Figure 5.1) was computed.

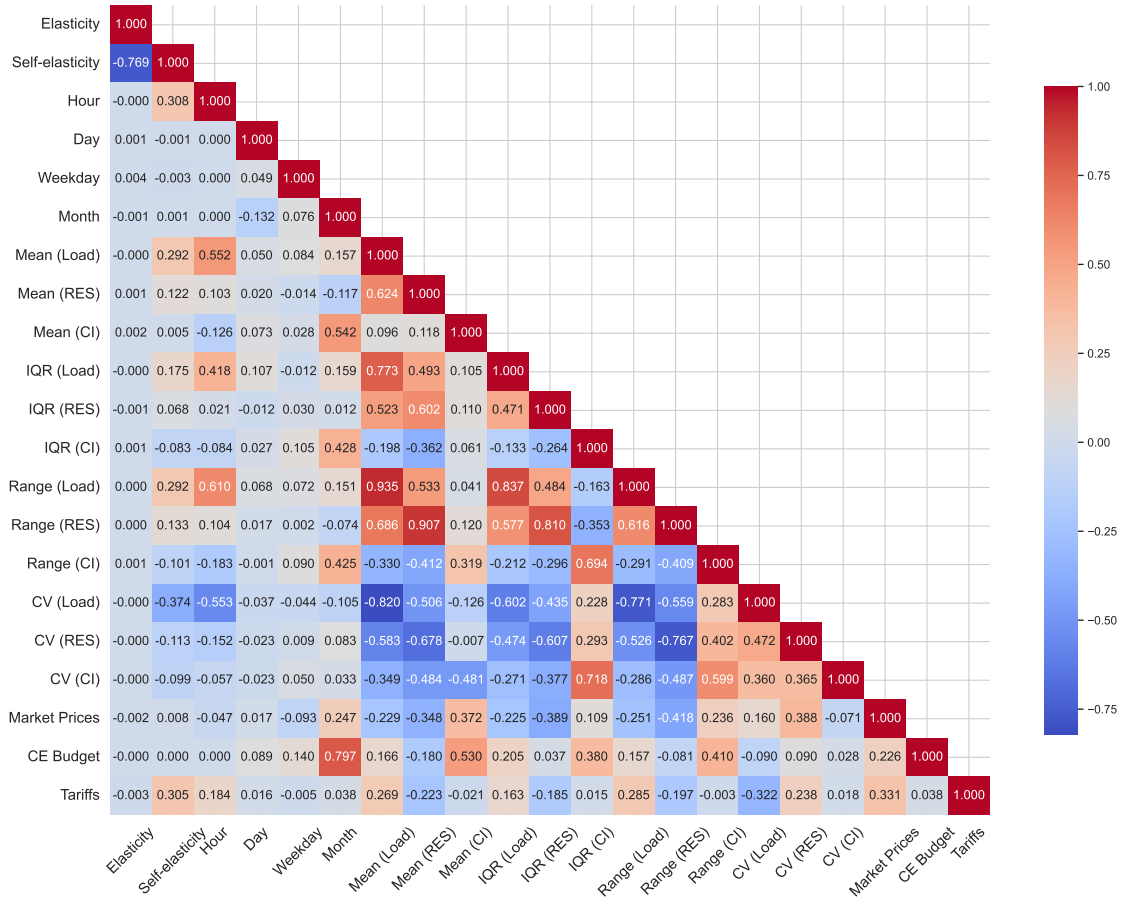

Figure 5.1: Correlation ρ between every dataframe column.

Figure 5.1 enables the detection of multicollinearity, feature redundancy, and potential informative patterns, all of which are crucial to planning feature selection in tree-based approaches. As expected, the most strongly correlated feature with the target, Tariffs, is Market Prices ($\rho \approx 0.33$). Such a result is natural, given the natural link between wholesale electricity costs and end-user charging tariffs.

Furthermore, Mean (RES), shows a moderate negative correlation with Tariffs ($\rho \approx -0.22$), suggesting that the higher availability of renewables may contribute to the reduction in energy prices, potentially due to their characteristic lower marginal costs.

In turn, calendar-related variables such as Hour, Weekday, and Month exhibit a noticeable correlation with the target. For instance, stronger associations with Hour suggest a clear intraday pattern in tariff behaviour, likely driven by typical load curves and market price fluctuations across the day.

Beyond correlations with the target, Figure 5.1 reveals meaningful associations among features. For instance, those related to statistical dispersion are highly correlated within each domain (load, RES, CI). Elasticity and self-elasticity are also highly correlated with each other, as expected.

Furthermore, an analysis of the degree of heterogeneity across the input space was performed. Therefore, Figure 5.2 displays a histogram for each feature, along with their maximum, minimum, and average values.

For instance, variables related to energy load and carbon intensity exhibit bimodal or skewed distributions. A distribution is considered bimodal if two different peaks can be identified from it, potentially reflecting the presence of separate behavioural regimes. For example, Mean (Load) shows those two different points, which could correspond to typical load patterns during working hours versus evening periods. Additionally, Mean (CI) also presents such distribution, possibly indicating shifts between periods, renewable-dominated versus fossil-dominated periods.

In contrast, skewness distributions reflect imbalanced distributions where one regime is typically dominant over others. For instance, Tariffs are mostly low, but sometimes they spike. Therefore, the distribution of tariffs is right-skewed.

The presence of bimodal and skewed distributions across several variables suggests the existence of structural regimes within the data. Such could be the reflection of conditions introduced by the optimization model, such as time-dependent consumption patterns, for instance.

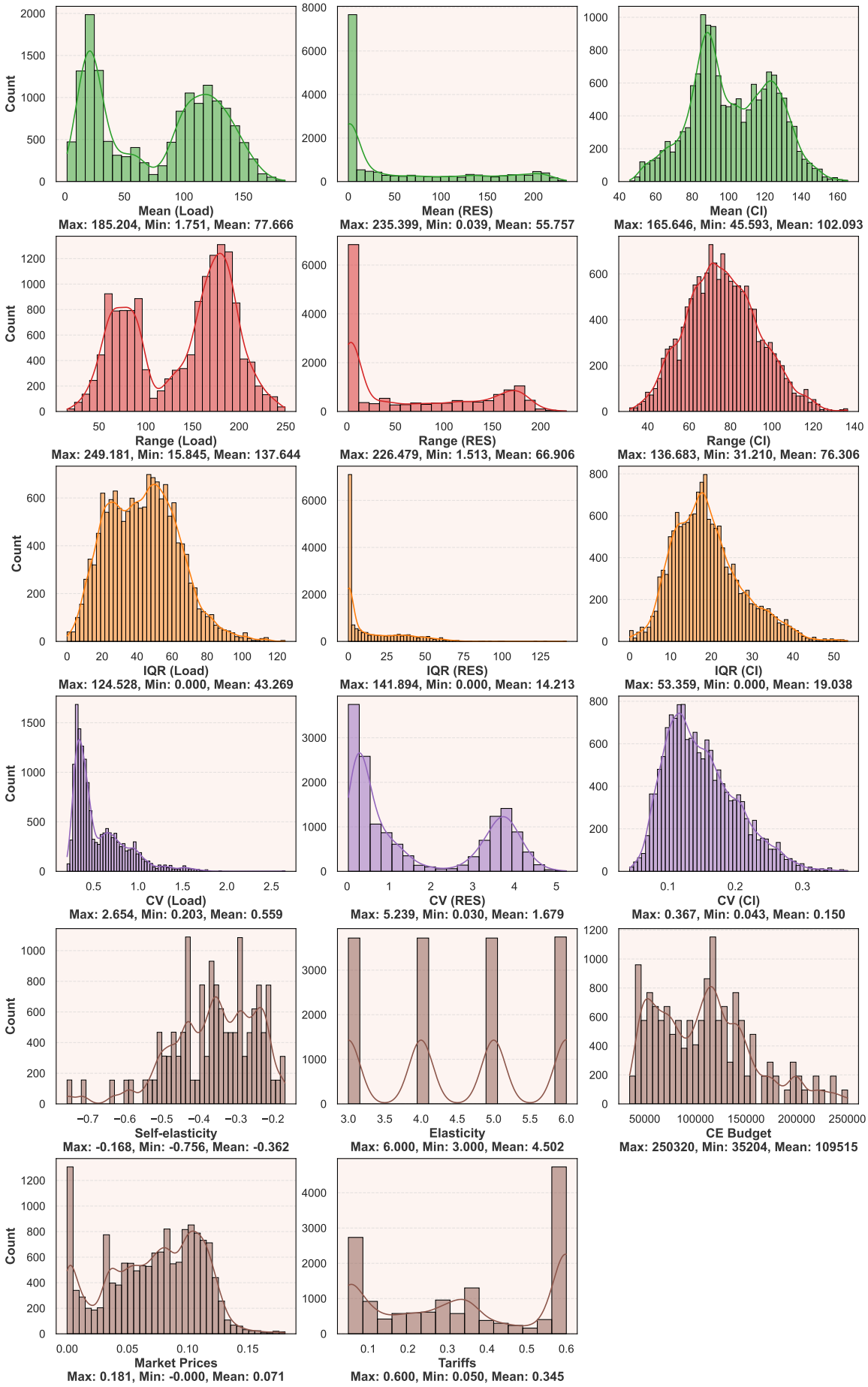

Figure 5.2: Data distribution of each variable present in the dataset.

To gain further insights into the temporal behaviour of key variables, a focused analysis is displayed over a selected time window. In this case, the 1st of July was chosen randomly, followed by the selection of the two subsequent days, the 2nd and the 3rd, considering an elasticity scale factor equal to 3. Such an approach will enable the observation of how tariffs and market prices evolve throughout the day under a fixed demand-response sensitivity setting within this data frame.

Figure 5.3 displays the hourly evolution of EV charging tariffs. Easily, one is able to identify a clear, repetitive pattern across all three days with defined high tariffs and low tariffs. In turn, Figure 5.4 shows the corresponding market electricity price over the same period. Driven by supply and demand, market prices show a consistent intra-day pattern, such as lower values during early mornings and peaks around the evening. Despite their distinct origins, both variables share a common temporal rhythm.

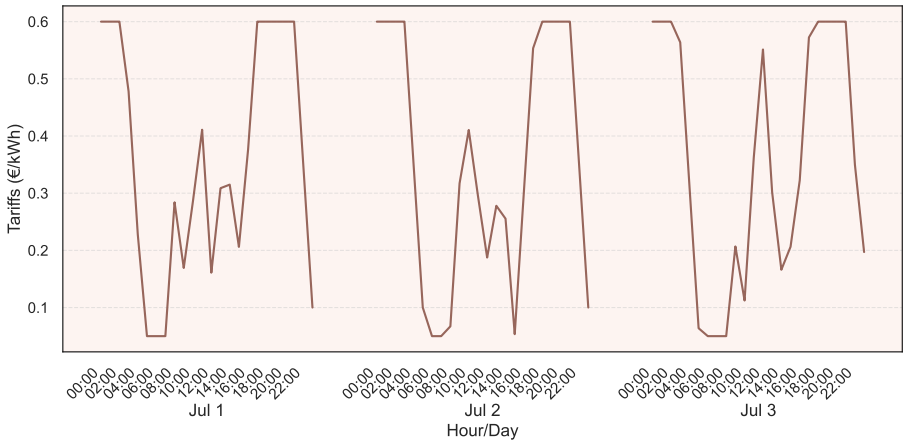

Figure 5.3: Hourly Evolution of EV Charging Tariffs.

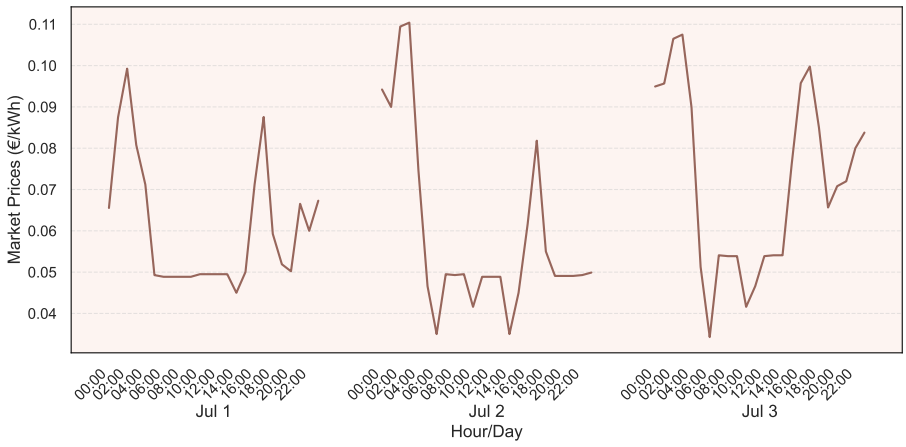

Figure 5.4: Hourly Evolution of Electricity Market Prices.

In turn, Figures 5.5 and 5.6 illustrate the hourly evolution of two key variables over the same period: Mean Carbon Intensity and Mean RES. Firstly, Figure 5.5 exhibits mid fluctuations throughout the day, typically ranging between 75 and 95 gCO2eq/kWh. These fluctuations

reflect the relative share of renewable versus fossil-fuel generation at different times, with lower values potentially indicating higher renewable penetration.

As shown in Figure 5.6, RES follows a strong diurnal pattern, peaking during midday and dropping to zero levels during the night. This regular structure is typical of solar generation, which is incorporated into the optimization framework. These figures show an inverse relationship, suggesting that renewable availability plays a significant role in reducing carbon intensity.

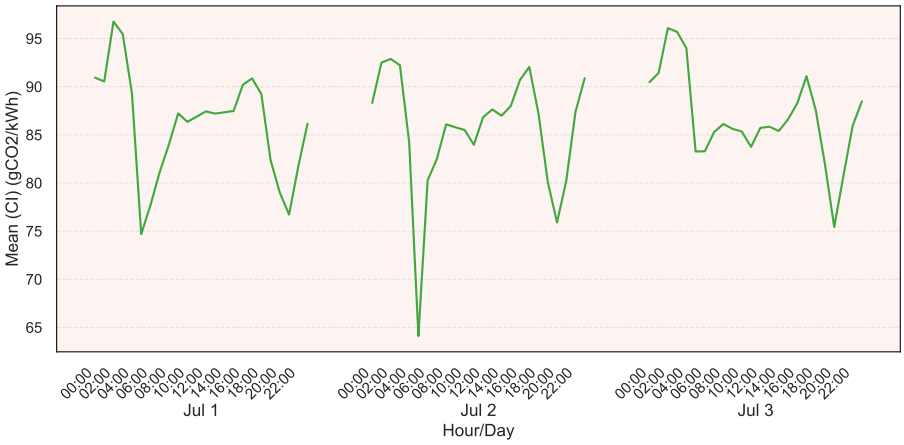

Figure 5.5: Hourly Evolution of Mean Carbon Intensity.

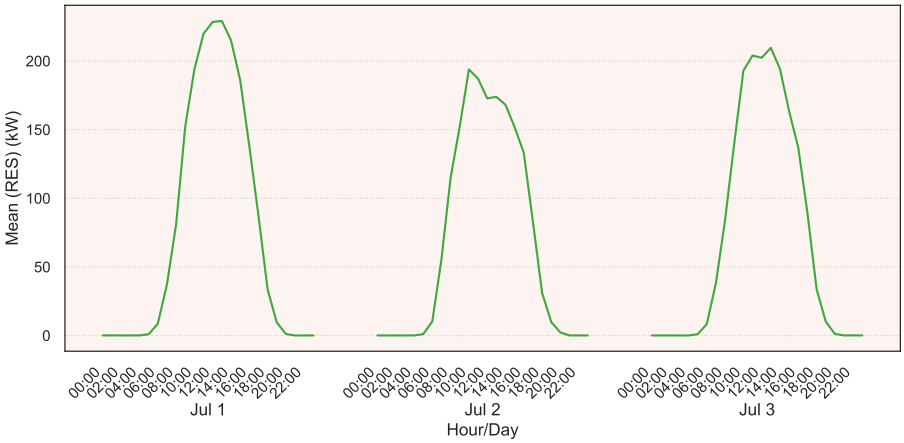

Figure 5.6: Hourly Evolution of Mean RES.

Moreover, Figure 5.7 presents, for each day, two well-defined peaks regarding Mean (Load). One during working hours and another in the evening, highlighting the strong influence of human activity and behavioural routines on energy demand.

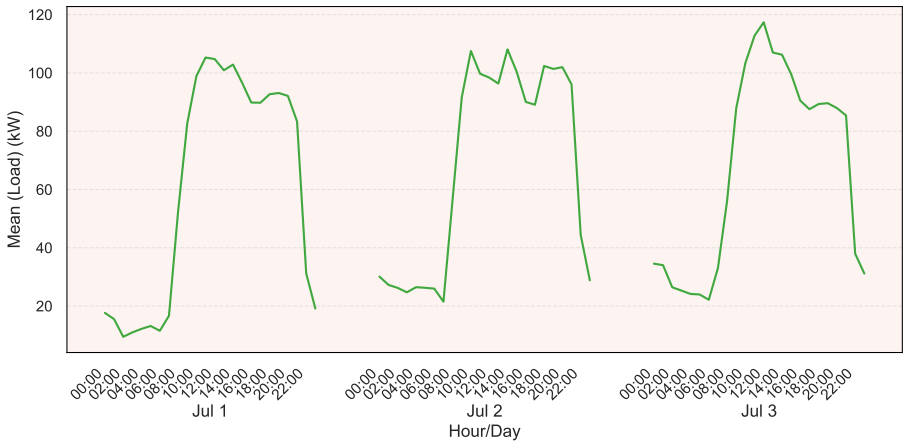

Figure 5.7: Hourly Evolution of Mean Load.

5.2 Tree-based interpretability

5.2.1 Simulation Description

Throughout this section, the focus falls upon the simulations described in Table 5.1. Essentially, the aim is to cover different algorithms and assess their performance regarding both predictive performance and, mainly, their interpretability. Additionally, this research also compares the following ML algorithms to the simplest form of prediction, linear regression.

Table 5.1: Summary of ML simulations.

Group	ID	Algorithm Type	Search Method	Particular Aspect
RS	RS-DT	Decision Tree Regressor	Random Search	All Features
	RS-M5	M5 Prime Model Tree		
	RS-RF	Random Forest Regressor		
	RS-GBT	Gradient Boosting Trees		
	RS-LGB	LightGBM Regressor		
	RS-XGB	Extreme Gradient Boosting (XGBoost)		
GS	GS-DT	Decision Tree Regressor	Grid Search	All Features
	GS-M5	M5 Prime Model Tree		
	GS-RF	Random Forest Regressor		
	GS-GBT	Gradient Boosting Trees		
	GS-LGB	LightGBM Regressor		
	GS-XGB	Extreme Gradient Boosting (XGBoost)		
LR	LR	Linear Regression	No Search	All Features
EF	EF-DT	Evolutionary Forest + Decision Tree	GridSearch + RandomizedSearch	Feature Engineering
SySM	S-RF	Random Forest Regressor	No Search	SySM + Default Parameters
	S-XGB	Extreme Gradient Boosting (XGBoost)		
	B-RF	Random Forest Regressor		Best Estimator + Default Parameters
	B-XGB	Extreme Gradient Boosting (XGBoost)		
	D-RF	Random Forest Regressor		Default Parameters
	D-XGB	Extreme Gradient Boosting (XGBoost)		

The results below, individualized by their ID, will differentiate each simulation, highlighting both the advantages and disadvantages of each algorithm and approach.

It should be remembered that this dissertation followed the typical machine learning pipeline workflow, as described in Section 4.3. The simulations presented considered the removal of the

feature Elasticity, due to its high correlation to Self-elasticity (Figure 5.1), and the integration of a lagged target variable, which is a common practice in machine learning for time series data. However, its inclusion raises questions about the overall explainability of the optimization results, contributing to the decoupling between the surrogate model and actual optimization process. Furthermore, 90% of the dataset is used for training, while the remaining 10% comprises the testing subset. Special care was taken to prevent data leakage, ensuring that the temporal order of observations was preserved throughout the data splitting and model evaluation processes.

5.2.2 Hyperparameter Optimization

Table 5.1 presents the performed simulations, among which a key distinction can be observed: the search method. In order to optimize results, a technique to select and set the model's hyperparameters becomes a requirement. This work resorted to both GridSearch and RandomizedSearch to fulfill this task. These well-known and established methods within the current literature were performed across each search space defined by Table 5.2, scored towards MAE minimization.

Table 5.2: Hyperparameter grids for all models.

Model	**Parameter**	**Values**
DecisionTreeRegressor	`criterion`	{`squared_error`*, `absolute_error`}
	`max_depth`	{4, 6, 8}
	`min_samples_split`	{2*, 10, 20}
	`min_samples_leaf`	{1*, 5, 10}
	`max_features`	{`None`*, `sqrt`, `log2`}
	`ccp_alpha`	{0.0*, 1e-3, 1e-2}
GradientBoostingRegressor	`n_estimators`	{100*, 300, 500}
	`learning_rate`	{0.05, 0.1*, 0.2}
	`max_depth`	{4, 6, 8}
	`min_samples_split`	{2*, 10}
	`min_samples_leaf`	{1*, 5}
	`subsample`	{0.8, 1.0*}
	`max_features`	{`None`, `sqrt`, 0.7}
XGBRegressor	`n_estimators`	{100*, 300, 500}
	`learning_rate`	{0.05, 0.1*, 0.2}
	`max_depth`	{4, 6, 8}
	`min_child_weight`	{1*, 5}
	`gamma`	{0*, 3}
	`subsample`	{0.8, 1.0*}
	`colsample_bytree`	{0.8, 1.0*}
	`reg_alpha`	{0.0, 0.1, 1.0}
	`reg_lambda`	{0.1, 1, 5}
LGBMRegressor	`n_estimators`	{100*, 300, 500}
	`learning_rate`	{0.05, 0.1*, 0.2}
	`max_depth`	{4, 6, 8}
	`num_leaves`	{31*, 63}
	`min_child_samples`	{20, 50}
	`subsample`	{0.8, 1.0*}
	`colsample_bytree`	{0.8, 1.0*}
	`reg_alpha`	{0.0, 0.1, 1.0}
	`reg_lambda`	{0.1, 1.0, 5.0}
RandomForestRegressor	`n_estimators`	{100*, 500, 800, 1000}
	`max_depth`	{4, 6, 8}
	`min_samples_split`	{2*, 10, 20}
	`min_samples_leaf`	{1*, 5, 10}
	`max_features`	{`sqrt`*, 0.7, `None`}
	`bootstrap`	{`True`*}
M5PRegressor	`max_depth`	{4, 6, 8}
	`min_samples_split`	{2*, 5, 10, 20}
	`min_samples_leaf`	{1*, 5, 10, 20}
	`use_smoothing`	{`True`*, `False`}
	`smoothing_constant`	{5, 15*, 30, 60, 100}

*Default library values.

The DecisionTreeRegressor grid includes six key hyperparameters. The `criterion` defines the loss function used for splitting: `squared_error` (default) or `absolute_error`. The `max_depth` parameter restricts the height of the tree, reducing overfitting by limiting complexity, with values 4, 6, and 8 offering a controlled balance between underfitting and overfitting. Note that `min_samples_split` and `min_samples_leaf` define the minimum number of samples required to split an internal node or create a leaf, serving as pre-pruning mechanisms. The values selected (2, 10, 20) and (1, 5, 10) include the defaults and offer increasing regularization. Additionally, `max_features` controls the number of features considered at each split; values include None (default), sqrt, and log2 to explore different levels of randomization. Finally, `ccp_alpha` applies cost-complexity pruning after training, removing weak subtrees; the grid includes 0.0 (default), 0.001, and 0.01 [60].

The GradientBoostingRegressor uses `learning_rate` to shrink each tree's contribution, controlling overfitting with values of 0.05, 0.1 (default), and 0.2, spanning conservative to more aggressive updates. In turn, `n_estimators` determines the number of boosting rounds, with values 100 (default), 300, and 500 balancing training time and capacity. Once more, `max_depth` governs model complexity, while `min_samples_split` and `min_samples_leaf` act as regularization terms for node and leaf formation. The `subsample` hyperparameter adds randomness by sampling rows per tree (0.8, 1.0), which reduces variance and enhances generalization. Similarly, `max_features` controls feature subsampling per split (None, sqrt, 0.7), lowering correlation between trees [61].

In XGBRegressor, `learning_rate` and `n_estimators` serve the same boosting role. Next, `max_depth` regulates tree complexity, and `min_child_weight` defines the minimum sum of instance weights per leaf, a regularization term defaulted to 1 and increased to 5 in the grid. Moreover, a new hyperparameter, `gamma`, introduces a minimum loss reduction required to split, controlling model structure (0, 3). `subsample` and `colsample_bytree` randomly sample rows and features, respectively, with values around the default (1.0) and 0.8 to reduce overfitting. Finally, `reg_alpha` and `reg_lambda` apply L1 and L2 penalties on leaf weights; L1 encourages sparsity by erasing weak leaves, and L2 smooths extreme outputs [62].

LGBMRegressor, a leaf-wise boosting variant, includes, as standard, both `n_estimators` and `learning_rate`. In turn, `num_leaves` controls the number of terminal nodes; values 31 (default) and 63 offer flexibility while avoiding overfitting when combined with `max_depth` (4, 6, 8). Note that `subsample` and `colsample_bytree` operate like in other boosting methods to reduce variance. Once again, `reg_alpha` and `reg_lambda` apply L1 and L2 penalties to control overfitting. All ranges include the default and moderate increases for regularization [63].

The RandomForestRegressor also resorts to `n_estimators` hyperparameter to define the number of trees in the forest. The grid explores 100 (default), 500, and 1000 trees, balancing computational cost and variance reduction. Tree complexity is managed through `max_depth`, with values 4, 6, and 8, limiting how deep individual trees can grow and thus preventing overfitting. The parameters `min_samples_split` and `min_samples_leaf` act once more as pre-pruning mechanisms. These are set to {2, 10, 20} and {1, 5, 10}, encompassing the defaults and explor-

ing increased regularization. The `max_features` parameter determines the number of features considered when looking for the best split, with options sqrt (default), 0.7, and None to adjust the level of randomness and reduce correlation among trees. Finally, `bootstrap` is fixed to True (default), enabling training on bootstrapped samples and allowing for out-of-bag error estimation [64].

Finally, M5PrimeRegressor uses standard splitting controls, such as `min_samples_split` and `min_samples_leaf` with values (2–20) to constrain branching and leaf creation. Here, `max_depth` controls tree height, balancing complexity and generalization. In turn, this model makes available `use_smoothing` to activate the model's smoothing mechanism, along with `smoothing_constant` (5–100) that controls the degree to which each leaf prediction is blended with its parent node's value. This allows for reduced variance and more stable predictions, particularly in small-sample leaves, following the original M5 design principles [37, 65].

However, it is worth noting that the focus of this work is not on developing the best predictor model but instead on assessing its interpretability and, in some cases, enhancing it.

5.2.3 ML performance

Figure 5.8 displays the mean absolute error, computed on the test set, obtained after performing the Randomized Search, optimized for MAE minimization, over the predefined hyperparameter space across 250 iterations. Section 3.1.1 demonstrates the rationale justifying that 250 is sufficient to approximate an optimal solution.

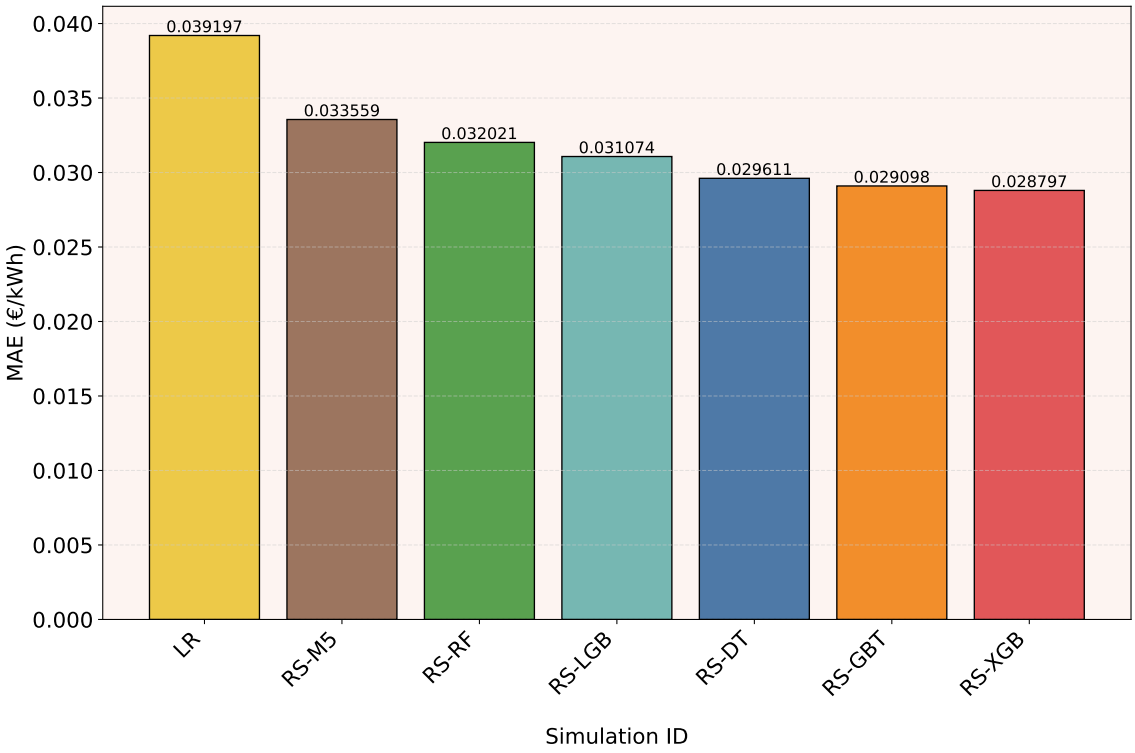

Figure 5.8: Mean Absolute Error (€/kWh) per simulation.

As expected, LR is the worst-performing model in terms of predictive performance. Therefore, this model is not capable of properly identifying non-linear patterns and relations between the features themselves, and between them and the output target. In line with expectations, RS-XGB was the best performing model prediction-wise, resulting in an MAE equal to 0.028797 €/kWh, which is an improvement over the results in [4], although the authors did not consider different elasticities, ultimately having a much smaller dataframe. In turn, RS-GBT and RS-DT follow closely behind, presenting a MAE of 0.209098 €/kWh and 0.029611 €/kWh. This outcome is rather interesting, since the simplest tree-based ML model ended up providing one of the best results, even as a single tree.

The remaining three models underperformed. Firstly, ensemble algorithms were expected to outperform single tree algorithms. This was not the case, since neither RS-RF nor RS-LGB overcame RS-DT's prediction performance. Moreover, RS-M5, which is supposed to be an improvement over the main single tree-based ML algorithm, RS-DT, did not provide better results.

Figure 5.9 displays the hourly predictions for randomly selected consecutive days from the testing subsets. All model predictions, except for LR, remain within the target's observed minimum and maximum values (as shown in Figure 5.2). Such is aligned with article [4], since the optimization model explicitly defines constraints that enforce both upper and lower bounds on the charging Tariffs.

Furthermore, all models appear to successfully capture the overall temporal structure. However, in general, models seem to struggle to accurately capture abrupt transitions at tariff levels. Many predictions show a tendency to miss the exact timing of sharp changes, often responding with a slight delay or shifting the transition to a nearby time point. Note that, during these rapid changes, models fail to reach the true extremes.

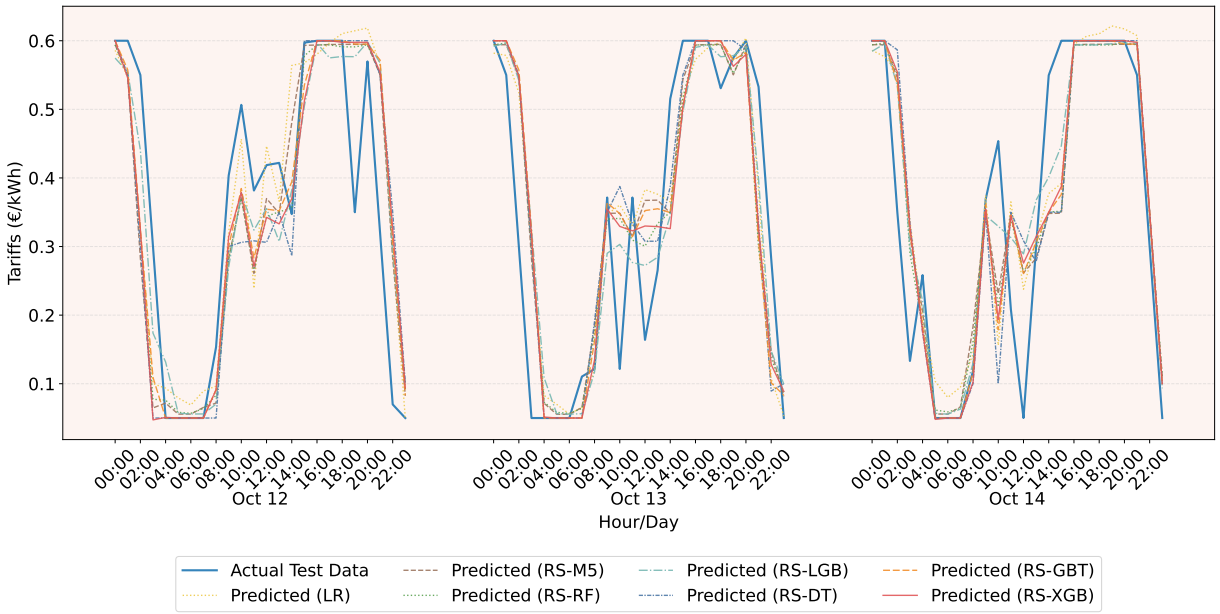

Figure 5.9: Hourly prediction per simulation across randomly chosen days.

Since RS-XGB is the best-performing model in terms of both predictive performance and generalization capabilities, a feature importance plot was generated (Figure 5.10) to highlight the most influential features, as a post-hoc explainability technique. Results show that predictors such as Tariffs-Lag, Hour, CV (Load), Self-elasticity, and Mean (RES) are the main contributors within the feature space.

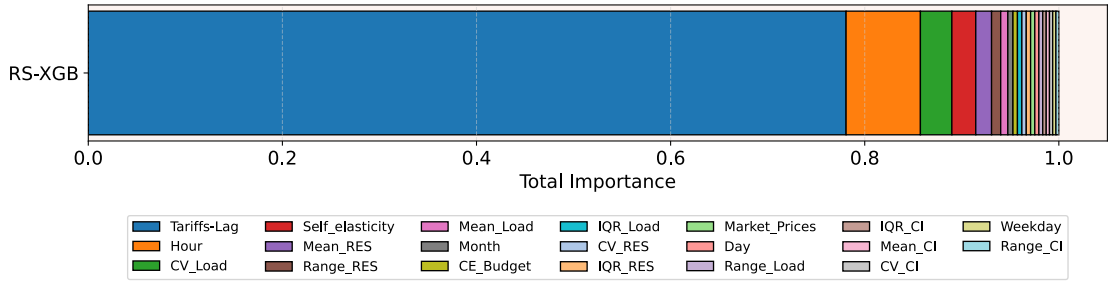

Figure 5.10: Feature Importance for the best predictor RS-XGB.

Such an outcome is to be expected, given that lag features are beneficial when working with time series data. Similarly, the importance of the Hour variable is also expected, due to its natural relationship with electricity prices, which typically follow daily consumption patterns. These temporal dynamics are tied to the variability in demand and availability of renewable generation, which are effectively captured by the CV (Load) and Mean (RES), respectively, thereby justifying themselves as a meaningful predictor.

Moreover, Self-elasticity is interestedly ranked among the most influential features, which aligns with the proposition in [4] and their incorporation of the Price Elasticity Matrix into the modelling of charging demand.

In contrast, features from Month onward contribute minimally to the overall predictive performance of RS-XGB. This could suggest that reducing or removing these features may not significantly impact predictive performance, while potentially improving training efficiency and enhancing model interpretability. These conclusions represent an example of post-hoc explainability. These results are consistent with most conclusions of the data analysis, performed in Section 5.1, highlighting the importance of renewable availability and the hour of the day.

Figure 5.11 demonstrates the generalizability of each model. Clearly, RS-LGB, LR and RS-RF exhibit the poorest generalization performance. An important observation is that the hyperparameter optimization for both tree-based models selected the maximum possible depth. It is well known that excessively deep trees can lead to overfitting. As described in Section 3.1, its leaf-wise tree growth strategy is likely correlated with this phenomenon.

On the other hand, in addition to being the most accurate predictor, the same figure shows that RS-XGB demonstrates the best generalization performance. Furthermore, RS-M5, RS-GBT, and RS-DT simulations closely follow this generalization performance by RS-XGB. Note that RS-DT warrants further attention, since its predictive performance and generalization performance were better than what was expected of a single-tree model.

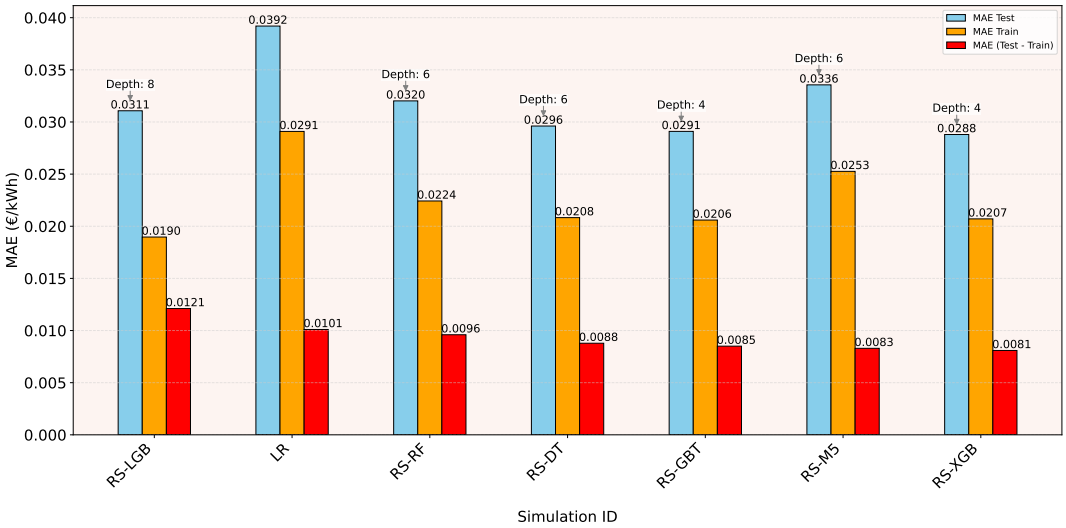

Figure 5.11: Generalization capabilities (MAE Test vs MAE Train) of each simulation.

An interesting analysis is presented in Figure 5.12, as it compares the two different hyperparameter search algorithms, namely Randomized Search and Grid Search, over the Normalized Elapsed Time and Mean Absolute Error. Here, Normalized Elapsed Time refers to the total search time divided by the number of hyperparameter combinations evaluated.

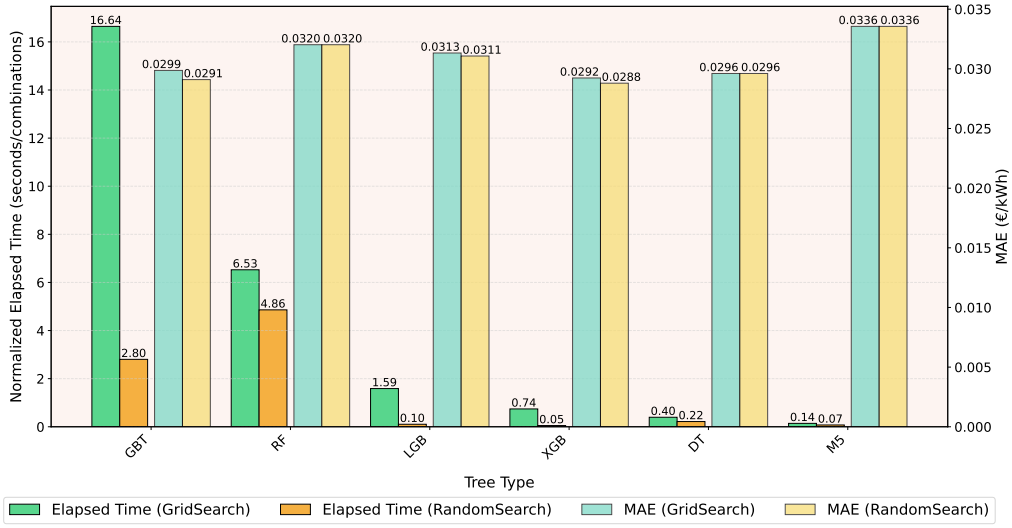

Figure 5.12: Search time and MAE comparison per search type and simulation.

Results show that GBT is by far the most computationally intensive, with a normalized training time exceeding 16 seconds per hyperparameter combination under Grid Search. In contrast, other boosting models achieve significantly faster training with times below 2 seconds per combination. Random Forest shows moderate computational demand, while Decision Tree and M5Prime are the least costly in terms of execution time. However, contrary to expectations, LGB is not the best boosting model in terms of computational efficiency. Such happens due to the accelerated

histogram-based split finding used in XGB, which enables highly efficient split computation by grouping continuous feature values into discrete bins. This approximation significantly reduces training time, allowing XGB to outperform LGB in this aspect under these hyperparameter sets.

Clearly, Randomized Search consistently offers significant reductions in training time across all models, as expected. The most notable improvement is observed in GBT, where training time per combination drops from 16.64 to 2.80 seconds. Although less effective, RF sees that reduction as well, while LGB and XGB computational time decreases as well.

Interestingly, despite its exhaustive nature, Grid Search does not consistently outperform Randomized Search in terms of predictive performance (MAE). In contrast, the mean absolute error achieved with the random search is often slightly lower. Such an effect is observed, for example, in GBT, LGB, and XGB models. A possible explanation is not related to the search space or its objective function, which are identical for both methods. Instead, it may be that, by chance, one generalizes better on the test set that was left out of cross-validation.

5.2.4 Evolutionary Forest Effect on DT Performance

Although commonly employed, the transformation of calendar features into cyclical representations was found to offer only marginal improvements during preliminary testing. Since predictive performance is not the main focus of this work, these transformations were omitted in favour of preserving the interpretability of tree-based models.

However, a brief experiment was conducted to assess whether the introduction of complex features would improve predictive performance. Section 3.1.5 briefly covers the theoretical aspects behind Evolutionary Forest. Essentially, a RandomizedSearch was performed to select the EvolutionaryForest Regressor hyperparameters and generate the newly engineered features. Afterwards, a GridSearch was applied to a DT so that a new hyperparameter configuration was developed to accommodate the new feature space. Note that only the newly created 20 most important features were added to the dataset.

Beware that feature engineering is not the focus of this dissertation. Therefore, a limited effort was conducted to define the hyperparameter search grids. Further refinements would probably enhance performance.

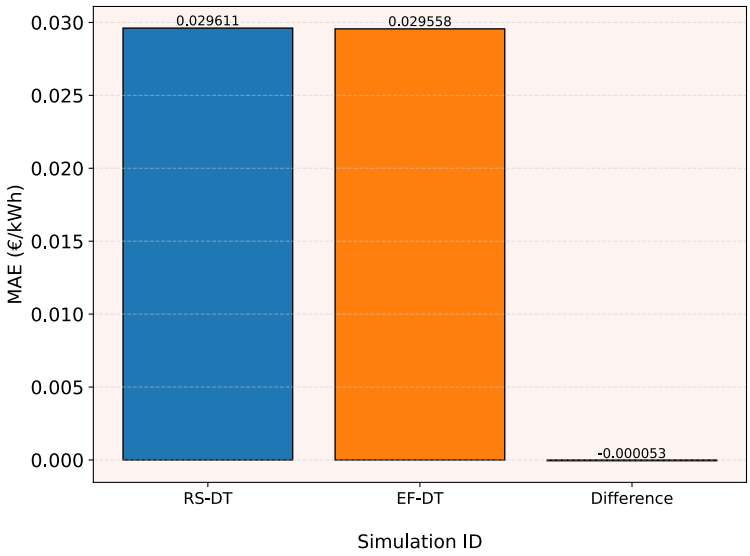

Figure 5.13: MAE Comparison after feature engineering process via EF.

Considering, for now, solely predictive performance, Figure 5.13 displays MAE from both RS-DT, EF-DT and the difference between them. Such results display a slight improvement in performance. This could be the effect of the applied search grids, or due to the restriction to a maximum of 20 of the most important newly created features. More importantly, Tariffs-Lag which is by far the most influential variable in baseline RS-DT already accounts for most of the predictive power in the model. Its dominance leaves very little headroom for extra engineered variables to add value, as per demonstrated by Table 5.3, where only engineered features with no null importance were displayed. Consequently, additional features created by EF can offer only a marginal benefit.

Table 5.3: Feature–importance comparison (RS-DT vs. EF-DT).

Feature	RS-DT	EF-DT
Self-elasticity	4.222×10^{-3}	4.317×10^{-3}
Hour	1.044×10^{-4}	0
Mean (RES)	1.344×10^{-3}	1.302×10^{-3}
Mean (CI)	3.979×10^{-5}	0
IQR (Load)	6.973×10^{-4}	6.972×10^{-4}
Range (RES)	8.128×10^{-4}	2.179×10^{-4}
Range (CI)	7.005×10^{-5}	7.004×10^{-5}
Market Prices	1.406×10^{-4}	1.152×10^{-4}
CE Budget	5.295×10^{-5}	0
Tariffs-Lag	9.925×10^{-1}	9.923×10^{-1}
(Range (CI) + Range (Load)) × (Mean (CI) − Weekday)	—	2.386×10^{-4}
(IQR (RES) × Mean (Load)) − 1	—	2.305×10^{-4}
Self-elasticity + Range (CI)	—	2.092×10^{-4}
Range (Load) × IQR (CI)	—	1.403×10^{-4}
Mean (Load) − Mean (CI)	—	1.042×10^{-4}
Weekday + Self-elasticity	—	6.120×10^{-5}

It is important to note the apparent lack of clear physical meaning among engineered features. Although this aspect is not captured by the interpretability metrics applied further in this dissertation (Section 5.2.7), it would nonetheless hinder the overall interpretability and explainability of the model.

5.2.5 Single Tree Visualization

As previously discussed, single tree-based models are inherently interpretable due to their transparent decision-making structure. Therefore, below follows the decision-making path of our best-performing single tree model, RS-DT, using a IF-THEN rule format, given its considerable depth which makes it impractical to display the tree in full within this document.

Note how this rule-set reinforces the importance of features such as Tariffs-Lag, Elasticity, Carbon Budget, and various statistical indicators that reflect the availability and variability of renewables, demand and carbon intensity.

```
IF Tariffs-Lag < 0.41 AND Tariffs-Lag < 0.154 AND Tariffs-Lag < 0.074 AND Self_elasticity < -0.243 AND Tariffs-Lag
    < 0.063 AND Tariffs-Lag < 0.05 THEN value=0.05
IF Tariffs-Lag < 0.41 AND Tariffs-Lag < 0.154 AND Tariffs-Lag < 0.074 AND Self_elasticity < -0.243 AND Tariffs-Lag
    < 0.063 AND Tariffs-Lag >= 0.05 THEN value=0.057
IF Tariffs-Lag < 0.41 AND Tariffs-Lag < 0.154 AND Tariffs-Lag < 0.074 AND Self_elasticity < -0.243 AND Tariffs-Lag
    >= 0.063 AND CE_Budget < 118008.5 THEN value=0.068
IF Tariffs-Lag < 0.41 AND Tariffs-Lag < 0.154 AND Tariffs-Lag < 0.074 AND Self_elasticity < -0.243 AND Tariffs-Lag
    >= 0.063 AND CE_Budget >= 118008.5 THEN value=0.094
IF Tariffs-Lag < 0.41 AND Tariffs-Lag < 0.154 AND Tariffs-Lag < 0.074 AND Self_elasticity >= -0.243 AND IQR_Load <
    50.224 AND IQR_Load < 29.998 THEN value=0.1
IF Tariffs-Lag < 0.41 AND Tariffs-Lag < 0.154 AND Tariffs-Lag < 0.074 AND Self_elasticity >= -0.243 AND IQR_Load <
    50.224 AND IQR_Load >= 29.998 THEN value=0.05
IF Tariffs-Lag < 0.41 AND Tariffs-Lag < 0.154 AND Tariffs-Lag < 0.074 AND Self_elasticity >= -0.243 AND IQR_Load >=
    50.224 AND Range_RES < 176.915 THEN value=0.344
IF Tariffs-Lag < 0.41 AND Tariffs-Lag < 0.154 AND Tariffs-Lag < 0.074 AND Self_elasticity >= -0.243 AND IQR_Load >=
    50.224 AND Range_RES >= 176.915 THEN value=0.176
IF Tariffs-Lag < 0.41 AND Tariffs-Lag < 0.154 AND Tariffs-Lag >= 0.074 AND Tariffs-Lag < 0.108 AND Self_elasticity
    < -0.205 AND Tariffs-Lag < 0.092 THEN value=0.089
IF Tariffs-Lag < 0.41 AND Tariffs-Lag < 0.154 AND Tariffs-Lag >= 0.074 AND Tariffs-Lag < 0.108 AND Self_elasticity
    < -0.205 AND Tariffs-Lag >= 0.092 THEN value=0.1
IF Tariffs-Lag < 0.41 AND Tariffs-Lag < 0.154 AND Tariffs-Lag >= 0.074 AND Tariffs-Lag < 0.108 AND Self_elasticity
    >= -0.205 THEN value=0.205
IF Tariffs-Lag < 0.41 AND Tariffs-Lag < 0.154 AND Tariffs-Lag >= 0.074 AND Tariffs-Lag >= 0.108 AND Tariffs-Lag <
    0.132 AND Self_elasticity < -0.261 THEN value=0.126
IF Tariffs-Lag < 0.41 AND Tariffs-Lag < 0.154 AND Tariffs-Lag >= 0.074 AND Tariffs-Lag >= 0.108 AND Tariffs-Lag <
    0.132 AND Self_elasticity >= -0.261 THEN value=0.191
IF Tariffs-Lag < 0.41 AND Tariffs-Lag < 0.154 AND Tariffs-Lag >= 0.074 AND Tariffs-Lag >= 0.108 AND Tariffs-Lag >=
    0.132 AND Self_elasticity < -0.279 THEN value=0.15
IF Tariffs-Lag < 0.41 AND Tariffs-Lag < 0.154 AND Tariffs-Lag >= 0.074 AND Tariffs-Lag >= 0.108 AND Tariffs-Lag >=
    0.132 AND Self_elasticity >= -0.279 THEN value=0.21
IF Tariffs-Lag < 0.41 AND Tariffs-Lag >= 0.154 AND Tariffs-Lag < 0.299 AND Tariffs-Lag < 0.224 AND Tariffs-Lag <
    0.187 AND Self_elasticity < -0.261 THEN value=0.175
IF Tariffs-Lag < 0.41 AND Tariffs-Lag >= 0.154 AND Tariffs-Lag < 0.299 AND Tariffs-Lag < 0.224 AND Tariffs-Lag <
    0.187 AND Self_elasticity >= -0.261 THEN value=0.25
IF Tariffs-Lag < 0.41 AND Tariffs-Lag >= 0.154 AND Tariffs-Lag < 0.299 AND Tariffs-Lag < 0.224 AND Tariffs-Lag >=
    0.187 AND Self_elasticity < -0.223 THEN value=0.206
IF Tariffs-Lag < 0.41 AND Tariffs-Lag >= 0.154 AND Tariffs-Lag < 0.299 AND Tariffs-Lag < 0.224 AND Tariffs-Lag >=
    0.187 AND Self_elasticity >= -0.223 THEN value=0.308
IF Tariffs-Lag < 0.41 AND Tariffs-Lag >= 0.154 AND Tariffs-Lag < 0.299 AND Tariffs-Lag >= 0.224 AND Tariffs-Lag <
    0.255 AND Tariffs-Lag < 0.234 THEN value=0.23
IF Tariffs-Lag < 0.41 AND Tariffs-Lag >= 0.154 AND Tariffs-Lag < 0.299 AND Tariffs-Lag >= 0.224 AND Tariffs-Lag <
    0.255 AND Tariffs-Lag >= 0.234 THEN value=0.246
IF Tariffs-Lag < 0.41 AND Tariffs-Lag >= 0.154 AND Tariffs-Lag < 0.299 AND Tariffs-Lag >= 0.224 AND Tariffs-Lag >=
    0.255 AND Tariffs-Lag < 0.268 THEN value=0.261
IF Tariffs-Lag < 0.41 AND Tariffs-Lag >= 0.154 AND Tariffs-Lag < 0.299 AND Tariffs-Lag >= 0.224 AND Tariffs-Lag >=
    0.255 AND Tariffs-Lag >= 0.268 THEN value=0.278
```

```
IF Tariffs-Lag < 0.41 AND Tariffs-Lag >= 0.154 AND Tariffs-Lag >= 0.299 AND Tariffs-Lag < 0.329 AND Tariffs-Lag <
    0.312 AND Self_elasticity < -0.223 THEN value=0.3
IF Tariffs-Lag < 0.41 AND Tariffs-Lag >= 0.154 AND Tariffs-Lag >= 0.299 AND Tariffs-Lag < 0.329 AND Tariffs-Lag <
    0.312 AND Self_elasticity >= -0.223 THEN value=0.35
IF Tariffs-Lag < 0.41 AND Tariffs-Lag >= 0.154 AND Tariffs-Lag >= 0.299 AND Tariffs-Lag < 0.329 AND Tariffs-Lag >=
    0.312 AND Self_elasticity < -0.22 THEN value=0.317
IF Tariffs-Lag < 0.41 AND Tariffs-Lag >= 0.154 AND Tariffs-Lag >= 0.299 AND Tariffs-Lag < 0.329 AND Tariffs-Lag >=
    0.312 AND Self_elasticity >= -0.22 THEN value=0.35
IF Tariffs-Lag < 0.41 AND Tariffs-Lag >= 0.154 AND Tariffs-Lag >= 0.299 AND Tariffs-Lag >= 0.329 AND Tariffs-Lag <
    0.38 AND Tariffs-Lag < 0.35 THEN value=0.333
IF Tariffs-Lag < 0.41 AND Tariffs-Lag >= 0.154 AND Tariffs-Lag >= 0.299 AND Tariffs-Lag >= 0.329 AND Tariffs-Lag <
    0.38 AND Tariffs-Lag >= 0.35 THEN value=0.35
IF Tariffs-Lag < 0.41 AND Tariffs-Lag >= 0.154 AND Tariffs-Lag >= 0.299 AND Tariffs-Lag >= 0.329 AND Tariffs-Lag >=
    0.38 AND Self_elasticity < -0.215 THEN value=0.388
IF Tariffs-Lag < 0.41 AND Tariffs-Lag >= 0.154 AND Tariffs-Lag >= 0.299 AND Tariffs-Lag >= 0.329 AND Tariffs-Lag >=
    0.38 AND Self_elasticity >= -0.215 THEN value=0.313
IF Tariffs-Lag >= 0.41 AND Tariffs-Lag < 0.528 AND Tariffs-Lag < 0.463 AND Tariffs-Lag < 0.439 AND Range_RES <
    160.432 AND Tariffs-Lag < 0.42 THEN value=0.41
IF Tariffs-Lag >= 0.41 AND Tariffs-Lag < 0.528 AND Tariffs-Lag < 0.463 AND Tariffs-Lag < 0.439 AND Range_RES <
    160.432 AND Tariffs-Lag >= 0.42 THEN value=0.424
IF Tariffs-Lag >= 0.41 AND Tariffs-Lag < 0.528 AND Tariffs-Lag < 0.463 AND Tariffs-Lag < 0.439 AND Range_RES >=
    160.432 AND Self_elasticity < -0.279 THEN value=0.406
IF Tariffs-Lag >= 0.41 AND Tariffs-Lag < 0.528 AND Tariffs-Lag < 0.463 AND Tariffs-Lag < 0.439 AND Range_RES >=
    160.432 AND Self_elasticity >= -0.279 THEN value=0.308
IF Tariffs-Lag >= 0.41 AND Tariffs-Lag < 0.528 AND Tariffs-Lag < 0.463 AND Tariffs-Lag >= 0.439 AND Hour < 14.5 AND
    Self_elasticity < -0.279 THEN value=0.437
IF Tariffs-Lag >= 0.41 AND Tariffs-Lag < 0.528 AND Tariffs-Lag < 0.463 AND Tariffs-Lag >= 0.439 AND Hour < 14.5 AND
    Self_elasticity >= -0.279 THEN value=0.306
IF Tariffs-Lag >= 0.41 AND Tariffs-Lag < 0.528 AND Tariffs-Lag < 0.463 AND Tariffs-Lag >= 0.439 AND Hour >= 14.5
    AND Tariffs-Lag < 0.449 THEN value=0.442
IF Tariffs-Lag >= 0.41 AND Tariffs-Lag < 0.528 AND Tariffs-Lag < 0.463 AND Tariffs-Lag >= 0.439 AND Hour >= 14.5
    AND Tariffs-Lag >= 0.449 THEN value=0.458
IF Tariffs-Lag >= 0.41 AND Tariffs-Lag < 0.528 AND Tariffs-Lag >= 0.463 AND Tariffs-Lag < 0.495 AND Range_RES <
    116.264 AND Mean_CI < 124.994 THEN value=0.478
IF Tariffs-Lag >= 0.41 AND Tariffs-Lag < 0.528 AND Tariffs-Lag >= 0.463 AND Tariffs-Lag < 0.495 AND Range_RES <
    116.264 AND Mean_CI >= 124.994 THEN value=0.55
IF Tariffs-Lag >= 0.41 AND Tariffs-Lag < 0.528 AND Tariffs-Lag >= 0.463 AND Tariffs-Lag < 0.495 AND Range_RES >=
    116.264 AND Range_RES < 179.969 THEN value=0.462
IF Tariffs-Lag >= 0.41 AND Tariffs-Lag < 0.528 AND Tariffs-Lag >= 0.463 AND Tariffs-Lag < 0.495 AND Range_RES >=
    116.264 AND Range_RES >= 179.969 THEN value=0.417
IF Tariffs-Lag >= 0.41 AND Tariffs-Lag < 0.528 AND Tariffs-Lag >= 0.463 AND Tariffs-Lag >= 0.495 AND Range_RES <
    117.38 AND Self_elasticity < -0.22 THEN value=0.511
IF Tariffs-Lag >= 0.41 AND Tariffs-Lag < 0.528 AND Tariffs-Lag >= 0.463 AND Tariffs-Lag >= 0.495 AND Range_RES <
    117.38 AND Self_elasticity >= -0.22 THEN value=0.6
IF Tariffs-Lag >= 0.41 AND Tariffs-Lag < 0.528 AND Tariffs-Lag >= 0.463 AND Tariffs-Lag >= 0.495 AND Range_RES >=
    117.38 AND Self_elasticity < -0.27 THEN value=0.486
IF Tariffs-Lag >= 0.41 AND Tariffs-Lag < 0.528 AND Tariffs-Lag >= 0.463 AND Tariffs-Lag >= 0.495 AND Range_RES >=
    117.38 AND Self_elasticity >= -0.27 THEN value=0.363
IF Tariffs-Lag >= 0.41 AND Tariffs-Lag >= 0.528 AND Tariffs-Lag < 0.578 AND Self_elasticity < -0.219 AND
    Tariffs-Lag < 0.55 AND Mean_RES < 77.614 THEN value=0.54
IF Tariffs-Lag >= 0.41 AND Tariffs-Lag >= 0.528 AND Tariffs-Lag < 0.578 AND Self_elasticity < -0.219 AND
    Tariffs-Lag < 0.55 AND Mean_RES >= 77.614 THEN value=0.523
IF Tariffs-Lag >= 0.41 AND Tariffs-Lag >= 0.528 AND Tariffs-Lag < 0.578 AND Self_elasticity < -0.219 AND
    Tariffs-Lag >= 0.55 AND Tariffs-Lag < 0.564 THEN value=0.55
IF Tariffs-Lag >= 0.41 AND Tariffs-Lag >= 0.528 AND Tariffs-Lag < 0.578 AND Self_elasticity < -0.219 AND
    Tariffs-Lag >= 0.55 AND Tariffs-Lag >= 0.564 THEN value=0.565
IF Tariffs-Lag >= 0.41 AND Tariffs-Lag >= 0.528 AND Tariffs-Lag < 0.578 AND Self_elasticity >= -0.219 AND
    Market_Prices < 0.103 THEN value=0.6
IF Tariffs-Lag >= 0.41 AND Tariffs-Lag >= 0.528 AND Tariffs-Lag < 0.578 AND Self_elasticity >= -0.219 AND
    Market_Prices >= 0.103 THEN value=0.575
IF Tariffs-Lag >= 0.41 AND Tariffs-Lag >= 0.528 AND Tariffs-Lag >= 0.578 AND Mean_RES < 132.04 AND Tariffs-Lag <
    0.592 AND Range_CI < 59.998 THEN value=0.554
IF Tariffs-Lag >= 0.41 AND Tariffs-Lag >= 0.528 AND Tariffs-Lag >= 0.578 AND Mean_RES < 132.04 AND Tariffs-Lag <
    0.592 AND Range_CI >= 59.998 THEN value=0.587
IF Tariffs-Lag >= 0.41 AND Tariffs-Lag >= 0.528 AND Tariffs-Lag >= 0.578 AND Mean_RES < 132.04 AND Tariffs-Lag >=
    0.592 AND Market_Prices < 0.008 THEN value=0.449
IF Tariffs-Lag >= 0.41 AND Tariffs-Lag >= 0.528 AND Tariffs-Lag >= 0.578 AND Mean_RES < 132.04 AND Tariffs-Lag >=
    0.592 AND Market_Prices >= 0.008 THEN value=0.6
IF Tariffs-Lag >= 0.41 AND Tariffs-Lag >= 0.528 AND Tariffs-Lag >= 0.578 AND Mean_RES >= 132.04 THEN value=0.287
```

Additionally, for a version with depth limited to 4, Figure 5.14 presents the tree in its standard graphical format, which proves to be both intuitive and pedagogically effective, allowing any

user to understand the model's logic easily. The visual predominance of Tariffs-Lag is expected, since Table 5.3 showcased the importance that this feature has in defining this specific model's predictions.

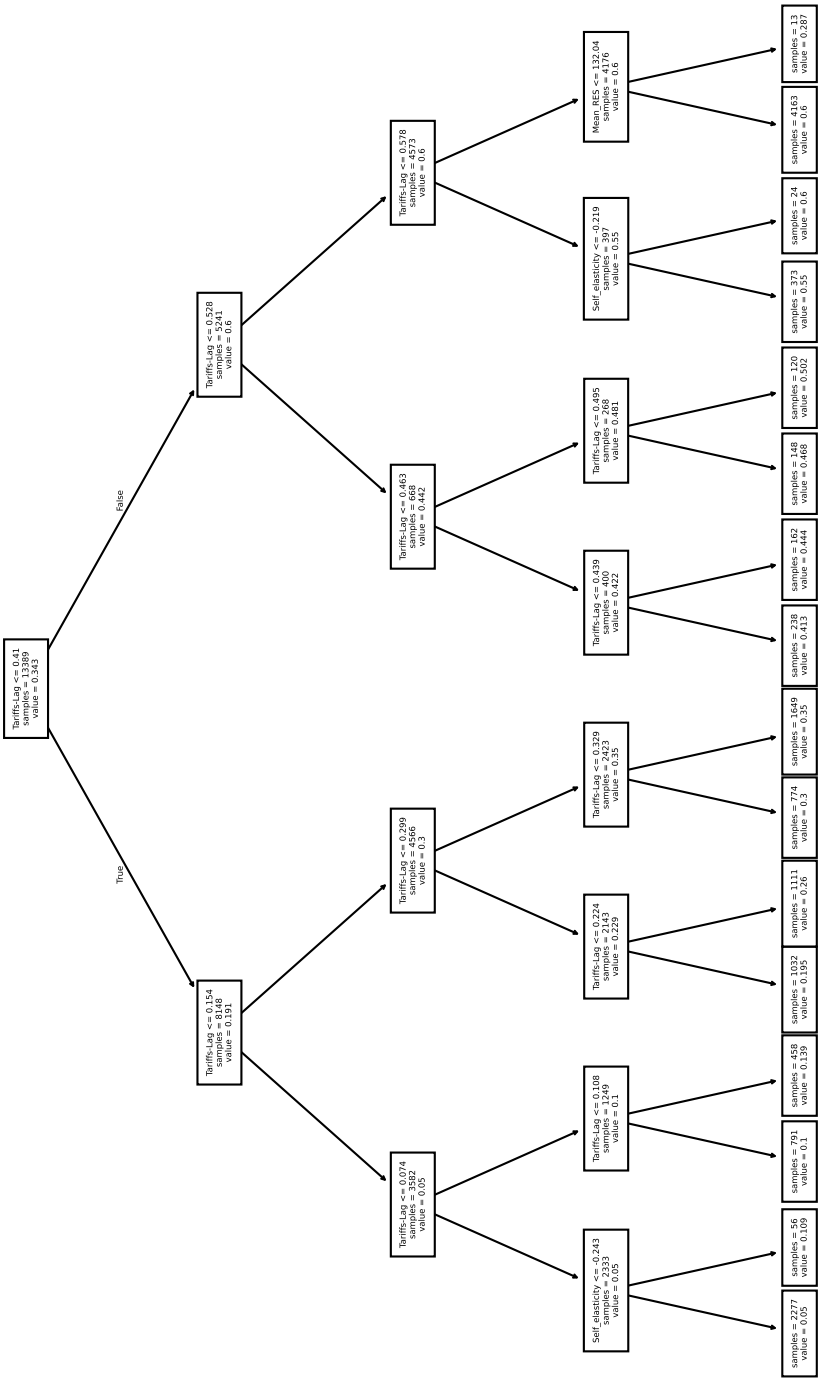

Figure 5.14: Graphical Representation of a Decision Tree with maximum depth set at 4.

Even though this representation is faithful to an individual decision tree, it fails to capture the aggregated decision-making process of ensemble models. Afterwards, within Section 5.2.6, this

dissertation aims to handle that challenge.

5.2.6 Ensemble interpretation

The goal of this work was to expand the framework established in article [4], whose aim was to develop the optimization model that resulted in the EV Charging Tariffs presented here. Such was accomplished, since improvements in prediction performance were achieved and multiple ML models were applied.

However, the main goal of this dissertation is to enhance the interpretability of these tariffs. As previously discussed, Decision Trees are inherently interpretable, which means that in the case of single tree-based machine learning models, interpretability can be achieved directly and efficiently.

Ensemble methods, by contrast, are generally considered to have low interpretability. Therefore, this work improved interpretability on RF and XGB models either by applying the methodology (section 3.2), developed by Weinberg *et al.* [51] to select the most representative tree within the ensemble, or simply selecting the best predictor. The impact of each approach on each model will be carefully evaluated in this section.

Since computing the syntactic similarity between one tree and all others in the ensemble can be more computationally expensive, the more estimators there are within the ensemble, this section does not use the best-performing models identified so far. Instead, the experiments were conducted with more computationally efficient configurations, by limiting the tree maximum depth to 4 and the number of estimators to 100, while keeping the remaining hyperparameters at their default values.

Therefore, Figure 5.15 enables the comparison of MAE between the whole ensemble, the best estimator in terms of individual predictive performance, and the estimator selected by the SySM algorithm as the most representative. Focusing first on the RF case, both B-RF and S-RF show similarities with the performance of the whole ensemble, D-RF. Therefore, it is arguable that these trees are representative of the whole model. Additionally, one can observe that the B-RF outperforms S-RF, which could question the usefulness of SySM. Its value, however, lies in the fact that it identifies the most representative tree within the ensemble without requiring access to predictions.

Furthermore, because SySM computes the syntactic similarity between every pair of trees in the ensemble and then selects the tree with the highest average similarity to all the others, the chosen estimator is, by construction, the one whose structure best mirrors the common decision patterns across the forest. This property means this algorithm can provide a more faithful structural representation for interpretability, potentially offering a better representation of the ensemble's behaviour than simply picking the top-performing estimator, whose predictive performance may not fully capture the ensemble's overall reasoning.

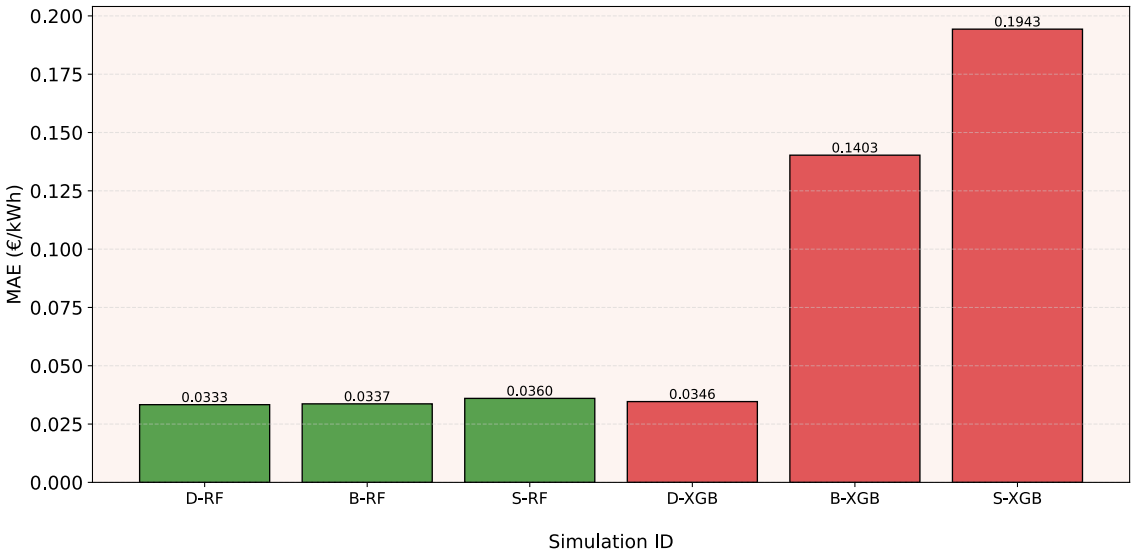

Figure 5.15: MAE comparison between the full ensemble (marked as D), the best individual predictor (marked as B), and the syntactically most representative estimator (marked as S).

Shifting attention to the XGB model, the error discrepancy between the whole ensemble, the best predictor, and SySM is quite significant. Such results show that this strategy does not extend to boosting algorithms. In boosting ensembles, such as XGB, GBT, or LGB, each successive tree is trained on the residual errors of the ensemble built so far. This means that every new tree is intentionally different from its predecessors so that, together, the trees correct previous weaknesses rather than share the same structure. Since the SySM algorithm relies on syntactic similarity, it selects the tree whose split pattern is most common across the ensemble. In a boosting context, that tree cannot capture the sequential error-correction mechanism that generates the model's predictive power. Therefore, while SySM offers a useful structural representation for bagging methods like Random Forests, it does not adequately describe boosted models, where no single tree can represent the ensemble's behaviour.

5.2.7 Interpretability Metrics

To facilitate the comparison between different models, clear metrics must be defined. This work employs a combination of features drawn by Weinberg *et al.* [51] and Fernandes *et al.* [66]. The latter defines interpretability of a Decision Tree as the comprehension of the decision-making process, subsequently dividing such concept into global interpretability, which regards the overall model, and local interpretability, which, in turn, shifts focus to an individual decision-path according to a specific input instance. Below is a brief description of these metrics.

Therefore, the Total Mental Operations refers to the metric that evaluates the entirety of the model. On the other hand, the Required Mental Operations is the metric responsible for assessing the interpretability of the model, considering solely a single data instance.

To remain faithful to the concept of interpretability, the main metric that genuinely reflects the general audience's typical behaviour is the Average Required Mental Operations. In fact, the general user does not need to navigate through the entire Decision Tree to reach the leaf that corresponds to its input data instance. It would be sufficient to traverse through the internal nodes whose splitting criteria are respected by the data within that input instance. Therefore, although TMO may still be used to describe the model complexity as a whole, RMO and ($\overline{\text{RMO}}$) are, ultimately, much more capable of reflecting the general end-user behaviour.

Since the ML pipeline proposed by this dissertation handles multiple Decision Tree models, their textual representation is not the same across different packages. Weinberg *et al.* [51] revealed an interesting string-like format to represent these models, the Bracket Tree Format (BTF). Although somewhat outside its scope, it was upon the article's referred tree structure that these interpretability metrics were computed.

For instance, observe the Decision Tree represented by figure 5.16. Note that the downward arrow corresponds to the true "outcome" of the condition, and the upward arrow corresponds to the "false" outcome.

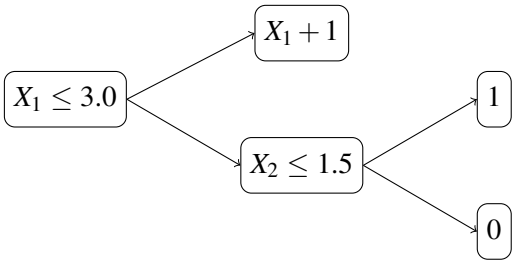

Figure 5.16: Decision Tree example prior to BTF conversion.

Such a tree could be easily represented using BTF, as depicted below. Essentially, the algorithm breaks down the string into its components, such as splits, assignments, and any arithmetic or functional operations present in the tree.

$$\{X_1 < 3.0\{X_2 < 1.5\{\text{value} = 0\}\{\text{value} = 1\}\}\{\text{value} = X_1 + 1\}\}$$

Firstly, the algorithm begins by dividing the string into individual tokens. These tokens are then processed to compute splits, assignments, and linear operations. Splits are internal nodes containing an inequality condition (< or >), each contributing three mental operations: comparing the feature to its threshold, identifying the left child, and identifying the right child. Assignment leaves are terminal nodes marked by *value* and contribute to one mental operation each. It is important to note that both split (in the context of feature engineering) and leaf nodes (as in the M5 model) may have a linear expression. Each of these operations are accounted for too. Therefore, equation 5.1 represents mathematically this formulation.

$$\text{TMO} = 3 \times \text{splits} + \text{assignments} + \text{linear operations} \tag{5.1}$$

Therefore, since the Decision Tree represented by Figure 5.16 displays two splits, three assignments, and one linear operation, the resulting TMO would be ten. Furthermore, if a user is faced with a data instance such as $X = \{2.5, 1\}$, then the expected prediction would be valued at 1. The RMO algorithm would traverse the BTF string based on that specific input, evaluating each condition encountered during traversal. Once again, each split returns three mental operations. If a condition is True, the algorithm proceeds to the corresponding subtree; otherwise, it skips it. When a leaf node is reached, it contributes one mental operation per assignment, with additional operations counted for arithmetic expressions. For this example, the result would be 7. Therefore, the RMO is the sum of all operations encountered during traversal, representing the cognitive effort required to evaluate the tree for the given input.

In summary, the definition of these metrics enables the comparison of different ML models regarding their interpretability and predictive performance, thereby making the trade-off between them more apparent. While not applied in this dissertation, these metrics could be included in the novel LLM Rule Generator framework, along with metrics designed to evaluate natural language explanations.

5.2.8 Trade-off analysis

The preceding sections reflect upon both ML predictive performance results and ensemble interpretability. This section will focus on analysing the trade-off between the predictive performance of the ML model and its interpretability score, using the metrics defined in Section 5.2.7. Therefore, the interpretability score is expressed using two metrics: TMO and RMO. Since this work uses a dataset with multiple observations, the $\overline{\text{RMO}}$ is used to account for every observation within the utilized dataframe.

Once deployed, a single Decision Tree enables the reader to follow the decision-making path to a given Tariff. However, most users will not need to examine the tree in its entirety. Given a single data instance, it would be sufficient to trace the branch that meets each successive split condition, thereby observing just a subtree of the entire model while still gaining a complete understanding of how the Tariff at a given time was determined. Figure 5.17 showcases that process, displaying the Average RMO by simulation. The closer each point is to the bottom-left corner, the better the model aligns with the following goal: a lower prediction error and a reduced average number of required operations.

Through direct observation, and assuming predictive performance as the main priority, the DT models appear to be the most suitable for deployment, given their low error rates and strong interpretability. On the other hand, if focus shifts entirely to interpretability, the best choice may be S-RF, as its low Average RMO demonstrates that it is highly accessible while still delivering reasonably competitive performance. It is important to note that these results refer exclusively to single-tree representations, whether selected as the best individual estimator or as the most syntactically representative of the full ensemble. As such, while the results for boosting models are interpretable by design in this context, their single tree performance is consistently inferior

to other candidates. Such erroneous performance indicates that these boosting algorithms are not made interpretable by the presented strategies.

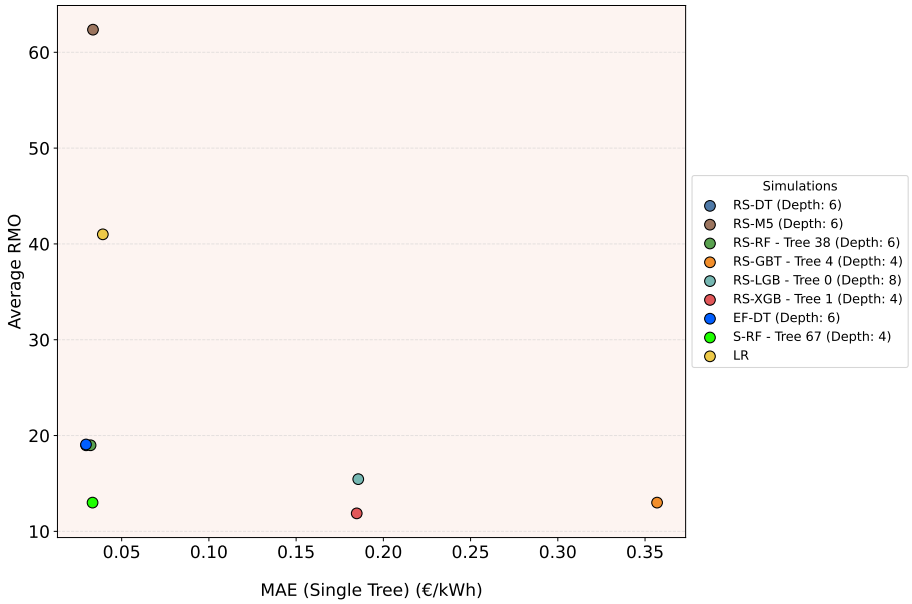

Figure 5.17: Trade-off between Average RMO and Predictive Performance. The Single Tree is the best-performing base estimator within the ensemble models, except for S-RF.

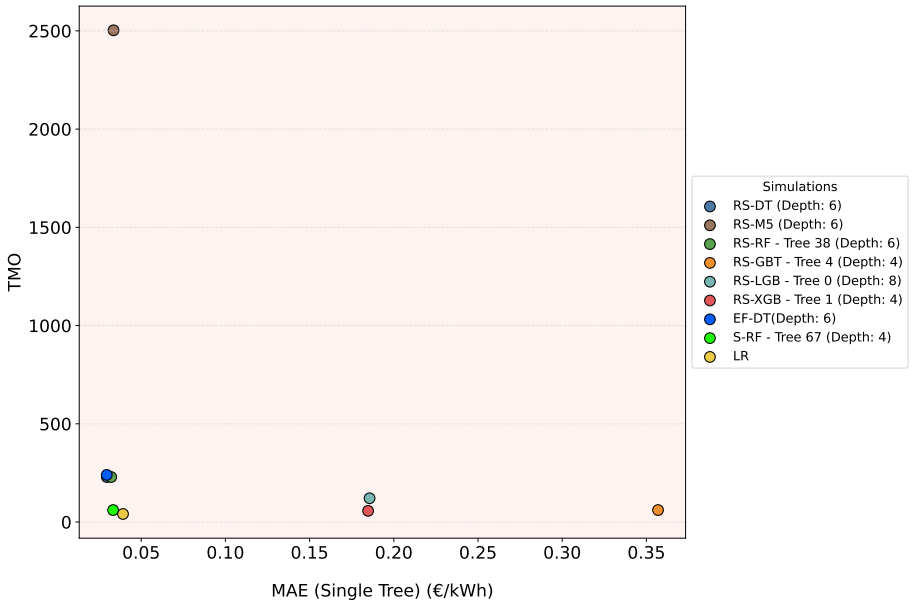

Figure 5.18: Trade-off between TMO and Predictive Performance. The Single Tree is the best-performing base estimator within the ensemble models, except for S-RF.

The M5 model results indicate that the inherent interpretability of a single Decision Tree is harmed by the Linear Regressions created by the model. Furthermore, LR, as expected, suffers

from a similar due its to the mathematical operations effect but its far more interpretable due to the absence of a nested representation, which increases the required number of mental operations.

Figure 5.18 mirrors the results observed for the Average RMO. However, it is important to note that TMO accounts for all mental operations required to process the entire tree, not jinvolved iniaolvedin aa single prediction path.

Finally, Figure 5.19 provides a brief analysis of the overall interpretability of the full forest for ensemble models. The reader should remember that these simulations from group RS performed a hyperparameter search over configurations expected to retrieve optimal predictive performance. These simulations highlight a strong correlation between the number of estimators returned by the RandomizedSearch and the Total Mental Operations of the ensemble. For instance, since XGB and GBT presented the highest number of estimators among these models, their total interpretability is lower despite their better predictive performance when compared against RF and LGB.

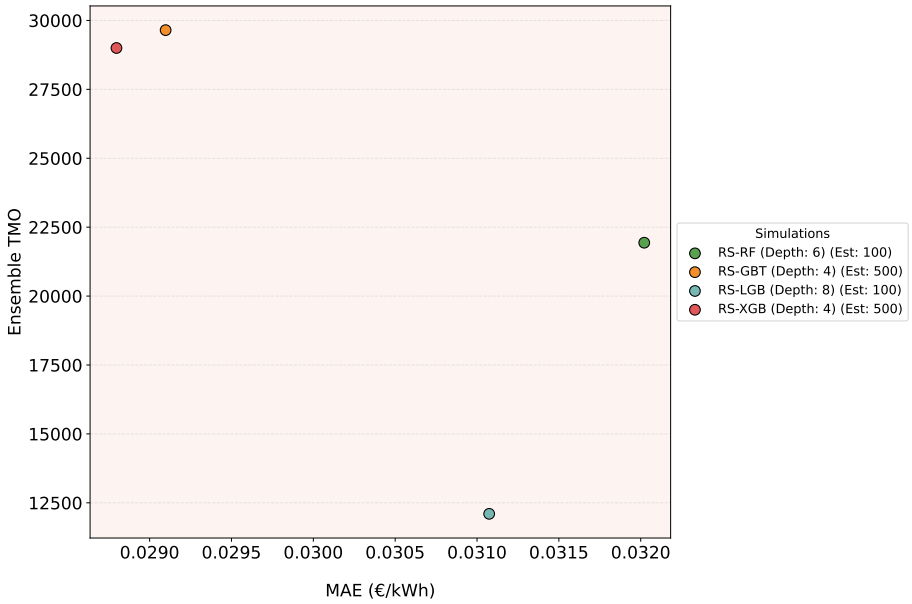

Figure 5.19: Trade-off between whole Ensemble and Predictive Performance. The Single Tree is the best-performing base estimator within the ensemble models.

Another interesting finding is that, although LGB exhibits a greater depth, the interpretability metrics suggest it is more interpretable than the remaining models. This can be partially explained by the leaf-wise growth strategy, which differs fundamentally from the level-wise strategy used in traditional decision trees. In LGB, tree grows by expanding the leaf that leads to the largest decrease in error, rather than expanding all nodes uniformly across levels. As a result, many instances follow shorter decision paths, reducing the effort required to interpret individual predictions.

5.3 Large Language Models

5.3.1 LLM Explanation of a Tree

This section provides results of applying the proposed methodology, detailed in Section 4.4.1, to explain a tree-based model using OpenAI's o3 large language model [67]. This brief experiment aims to assess the LLM's capabilities to generate an accessible, natural language explanation that clarifies the model's decision logic.

In addition to feature descriptions, the RS-DT tree structure itself is included in the prompts (available here in IF-THEN format, Section 5.2.5). As previously discussed, this model performed surprisingly well for a single-tree approach. Furthermore, according to Table 5.3, the Tariffs-Lag feature (which represents tariffs from 24 hours earlier) is the most influential factor in the prediction process.

```
Reasoning:


I need to write a simple explanation of how the EV charging tariff prediction model works without using
     technical terms. The model starts by looking at yesterday's tariff (Tariffs-Lag <= 0.41), which
     significantly influences the decision. For smaller tariffs, it also considers Self-elasticity and Load
      variations. For larger tariffs, it continues to use yesterday's tariff but includes factors like
     renewable supply and time of day. The big takeaway: past tariff data is the most important factor in
     predicting today's rates.


import re, textwrap, json, math
tree_text = {Here the LLM transcribed the tree strcuture}
import collections, itertools, math, json, os, sys, pprint
feature_counts = collections.Counter()
for line in tree_text.splitlines():
    # find feature names pattern: "--- <feature> <comparison>"
    m = re.search(r"\|[^\|]*--- ([^<=><]+?) [<>=]", line)
    if m:
        fname = m.group(1).strip()
        feature_counts[fname] += 1
feature_counts
Counter({'Tariffs-Lag': 56,
         'Self_elasticity': 28,
         'Range_RES': 10,
         'IQR_Load': 4,
         'Mean_RES': 4,
         'Market_Prices': 4,
         'CE_Budget': 2,
         'Hour': 2,
         'Mean_CI': 2,
         'Range_CI': 2})
Tariffs-Lag is a key factor, as it references the previous day's tariff. Self-elasticity measures how
     demand reacts to changes in tariff, indicating how sensitive consumers are to price fluctuations.
     Range_RES reflects variations in renewable energy supply, while IQR_Load tracks demand variability.
     Market_Prices show the predicted cost of electricity. Key thresholds help categorize tariffs, such as
     a 0.41 EUR/kWh cut-off for lower vs. higher tariffs, and social elasticity being more sensitive (
     negative values) when prices rise.
```

```
Explanation:

The model starts with a simple lookback . It first checks what you paid for a charge at the same
     hour yesterday. If that previous price was modest (below roughly €0.40 per kWh), the new tariff stays
     in the lower range; if yesterdays price was already higher, the model assumes conditions havent
     eased and keeps todays price in the higher bracket.

Inside each bracket, the next question is how drivers react to price changes. When the data show that
     demand falls sharply as price rises (high pricesensitivity), the model holds todays tariff close
      to yesterdays or even trims it, rewarding frugal behaviour. If drivers are less sensitive, the
     system feels freer to nudge the price upward. For the lowerprice bracket, it also peeks at how
```

```
      wildly todays demand forecast jumps around: a calm, predictable load keeps prices low, while a
      jumpy forecastespecially when renewable output is also unpredictablepushes the price up to
      prevent overcrowding at the plugs.

In the higherprice bracket, the model shifts its attention to how steady the greenenergy supply looks
      and what the wholesale market is charging right now. Big swings in expected solar production, a
      carbonheavy grid, or a midday   powerrush    hour  all convince the system to lift the tariff
      toward the upper 50  cent   range, especially when drivers show little resistance to paying more.

Big takeaway: yesterdays price sets the stage, but todays tariff ultimately balances drivers price
      sensitivity with the certainty of demand and renewable powersteady forecasts mean gentler prices,
      while volatility and low sensitivity quickly push them higher.
```

The LLM successfully produced a concise, coherent summary highlighting key variables influencing the model's predictions. It performs reasoning applying its coding capabilities, identifying Tariffs-Lag as the primary predictor, while also recognizing the role of self-elasticity as an important factor influencing pricing. Additionally, the LLM captured the impact of availability and variability in renewables, demand, and carbon intensity. Such results clearly demonstrate the model's capacity to extract and articulate the underlying structure of the decision process in clear, human-readable language.

5.3.2 Rule Generation

To assess the viability of applying LLMs for rule-based prediction tasks, a prototype framework, carefully described in Section 4.4.2, was developed and applied through different simulations, all of described by Table 5.4.

Table 5.4: Summary of simulations executed for the LLM-Rule Generator framework.

ID	Full Name	Lag Feature	Temperature	LLM/ML
Sim1–Sim5	Simulations 1 to 5	Yes	0.0	LLM
LR	Linear Regression	Yes	–	ML
XGB	XGBoost Regressor	Yes	–	ML
NL-Sim1–Sim5	Simulations 1 to 5	No	0.0	LLM
NL-LR	Linear Regression	No	–	ML
NL-XGB	XGBoost Regressor	No	–	ML

Two distinct scenarios were considered, as it is relevant to compare results with and without the inclusion of the Tariffs-Lag feature. Furthermore, the Temperature parameter, which controls the generative variability of the LLM was fixed at zero, while the random seed was also fixed, to ensure fully deterministic responses and reproducible responses. However, due to unforeseen and, to this day, unexplained reasons, it was not possible to stabilise the LLM's output. Therefore, five simulations were conducted to account for this variability. It should be noted that running additional simulations was not feasible due to time constraints and computational effort, as each 1000-iteration simulation required over one day to achieve completion.

Moreover, due to token limitations, several typically less relevant features were removed from the dataset to simplify the input. Specifically, the following features were excluded: Elasticity, Weekday, Range (CI), Range (RES) and Range (Load). Additionally, results were compared

against XGBoost and Linear Regression models, evaluated under the same conditions, to assess the performance of the LLM-based approach relative to standard ML models. The simulations run for 1000 iterations.

Table 5.5 provides the best MAE for each simulation of each scenario.

Table 5.5: Best MAE obtained for each scenario (with and without Lag Feature).

With Lag	**Best MAE**	**Without Lag**	**Best MAE**
Sim1	0.03341	NL-Sim1	0.09295
Sim2	0.06321	NL-Sim2	0.12734
Sim3	0.08456	NL-Sim3	0.12201
Sim4	0.04033	NL-Sim4	0.14828
Sim5	0.05919	NL-Sim5	0.08322
LR	0.03886	NL-LR	0.12395
XGB	0.02893	NL-XGB	0.07381

Figure 5.20 and 5.21 present the evolution of MAE across iterations, as well as the lowest MAE achieved up to each iteration, respectively, considering the Tariffs-Lag feature. The main and most immediate conclusion is the clear decrease in error throughout the iterations. Evidently, the LLM iteratively attempts to minimise the best MAE so far, and when an increase in error occurs, it often discards that iteration and reverts to a previous, better-performing solution. This suggests that the built-in feedback mechanism functions as intended, properly correcting the Rule Generator LLM throughout the iterative process.

Moreover, the best performing simulation is Sim1, even outperforming LinearRegression and achieving a close result to XGBoost (Table 5.5). Furthermore, Sim1 reaches convergence faster than other simulations. However, the remaining simulations show more erratic behaviours. For instance, Sim3, despite showing early improvements, failed to sustain convergence and terminated prematurely since its feedback mechanisms became trapped in a reasoning loop. This behaviour led to the exhaustion of the predefined response limit, causing the feedback output to grow excessively until the combined prompt exceeded the token limit of the Rule Generator LLM, which forced the early termination of the simulation. Sim2 and Sim5 showed the error reduction as well, however, neither was able to reach error levels comparable to Sim1. Finally, Sim4 achieved a closer result to Sim1, despite exhibiting greater oscillations throughout the process. Additionally, it is worth noting that, although some simulations show stagnation in improvement, for most remaining cases, it remains arguable that more iterations could slightly decrease the error further.

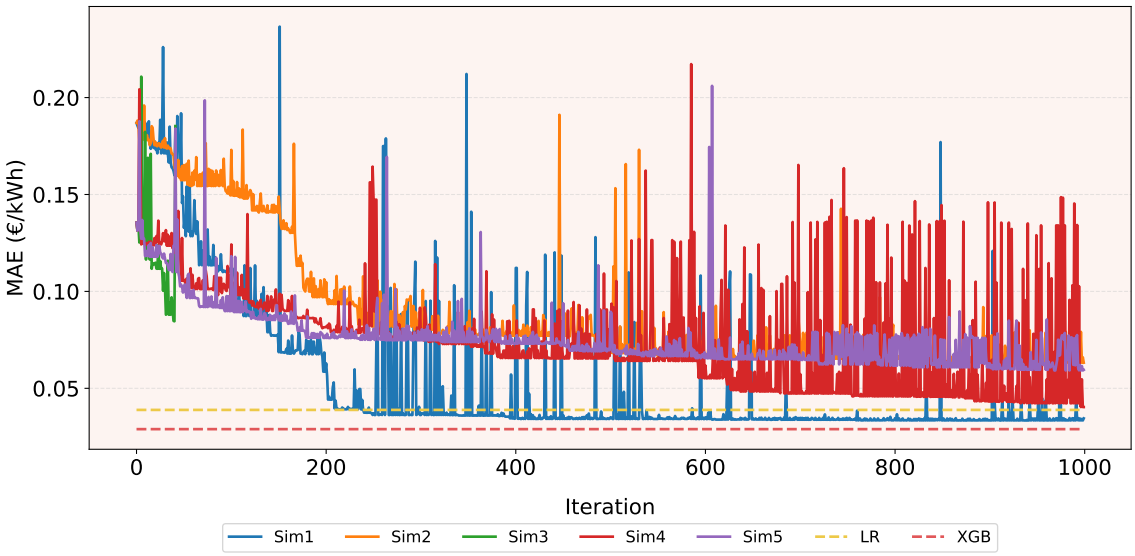

Figure 5.20: MAE Evolution for simulations including Tariffs-Lag.

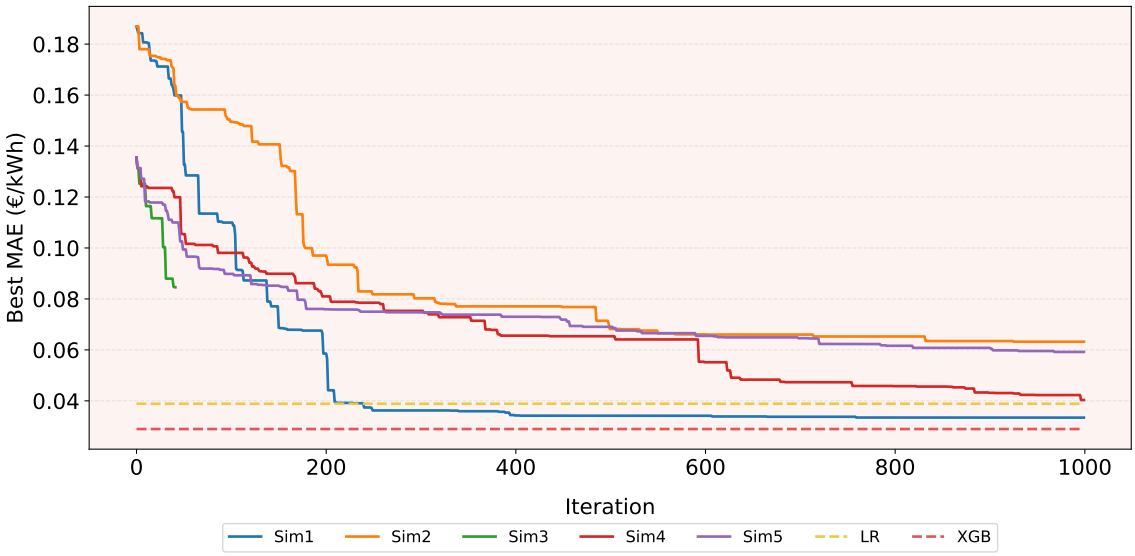

Figure 5.21: Best MAE until each iteration for simulations including Tariffs-Lag.

To better illustrate these results, two examples of the LLM's output, from Sim1, are presented below, more specifically from their first and best iteration, in order to show the evolution of the generated rules. The first code snippet transcribes results from iteration number 1, which is a simple rule-set whose depth is 3 and respective MAE reaches 0.1869 €/kWh.

As required, this rule-set exclusively resorts to nested structures, without employing "elif" statements, at the same time that it avoids forbidden functions or imports and ensures predictions remain within the valid tariff range. Moreover, data leakage is avoided by not utilizing Tariffs to compute predictions. However, the only unmet requirement is the absence of comments explaining each decision split, which were not included in the output.

```python
def rule_based_prediction(row):
    if row['Market Prices'] > 0.08:
        if row['Tariffs-Lag'] > 0.55:
            return 0.60
        else:
            return 0.55
    else:
        if row['Mean (Load)'] > 100:
            return 0.45
        else:
            if row['CE Budget'] > 100000:
                return 0.35
            else:
                return 0.30
```

Nonetheless, explainability is not forsaken, as the structure of the generated rules remains inherently interpretable, given its nested Decision Tree format that clearly exposes conditions leading to each prediction. Additionally, along with its built-in reasoning, the Rule Generator LLM is prompted to output a Justification and Changes Made, which are carefully transcribed below. This output demonstrates coherent reasoning by correctly identifying key variables and relevant patterns, such as temporal dependencies and market signals.

```
Reasoning:

Okay, I need to generate a rule-based prediction function for the Tariffs target variable. Let me start by
      understanding the problem and the data provided.

First, looking at the context, the target variable is Tariffs, which is the charging tariff in EUR/kWh. The
       features include various metrics like Self-elasticity, Hour, Mean (Load), Mean (RES), etc. The data
      is a time series, as it includes Hour, Day, Month, and Tariffs-Lag, which is the lagged value of
      Tariffs from 24 hours ago.

The goal is to create a nested if-else structure that predicts Tariffs based on these features. The
      function must not use any elif statements, only if and else blocks. Also, I must ensure that the
      predictions stay within the historical range of Tariffs, which from the dataset summary is between
      0.05 and 0.60.

Looking at the dataset, I notice that Tariffs-Lag is a feature, which suggests that the previous day's
      tariff might influence the current day's tariff. This could be a useful feature for prediction.

I should also consider the relationships between other features and the target. For example, Mean (Load)
      and Mean (RES) might indicate the demand and supply of energy, which could affect the tariff. Higher
      demand might lead to higher tariffs, while higher renewable supply could lower them.

Another important feature is Market Prices, which is the day-ahead electricity price. This is likely a
      strong predictor since higher market prices could lead to higher tariffs.

I should also look at the CV (Load), CV (RES), and CV (CI) which are coefficients of variation, indicating
      variability in load, renewable supply, and carbon intensity. Higher variability might lead to higher
      tariffs to manage uncertainty.

The CE Budget is the carbon emissions budget, which could influence tariffs if the goal is to encourage or
      discourage certain consumption patterns based on carbon impact.

Now, considering the prior knowledge, since this is the first iteration, there are no previous rules or
      feedback to incorporate. So, I'll start by identifying the most impactful features.

I think starting with Market Prices makes sense because it's a direct indicator of the cost of electricity.
       If Market Prices are high, tariffs are likely to be higher. So, I'll set a threshold for Market
      Prices. From the dataset, the average Market Prices is around 0.0712, so I'll use 0.08 as a threshold.

Next, considering Tariffs-Lag, if the previous day's tariff was high, it might influence the current day's
      tariff. So, if Market Prices are above 0.08 and Tariffs-Lag is above 0.55, set a high tariff like
      0.60.
```

```
If Market Prices are above 0.08 but Tariffs-Lag is 0.55 or below, maybe the tariff is slightly lower, say
     0.55.

For lower Market Prices, I'll check other factors. Mean (Load) could be a factor higher load might mean
     higher tariffs. If Mean (Load) is above 100, set tariff to 0.45.

If Mean (Load) is not high, perhaps look at CE Budget. A higher budget might allow for lower tariffs, so if
      CE Budget is above 100000, set tariff to 0.35.

If none of these conditions are met, a default tariff of 0.30 could be appropriate.

I need to ensure that all conditions are nested properly without using elif. Also, all return values must
     be within the historical range of 0.05 to 0.60.

I should test these conditions against the data to see if they make sense. For example, when Market Prices
     are high, tariffs are set higher, which aligns with the data where higher Market Prices correspond to
     higher Tariffs.

I think this structure covers the main factors and should help in lowering the MAE by making informed
     predictions based on key features.
```

```
Justification:
Nature: The dataset appears to be a time series with hourly granularity, spanning multiple days and months.
      Features like Hour, Day, Month, and Tariffs-Lag suggest temporal dependencies. The target variable (
     Tariffs) seems to be influenced by energy market dynamics (Market Prices), renewable supply (Mean (RES
     )), and demand patterns (Mean (Load)). .


Patterns:
1. Tariffs-Lag shows strong autocorrelation with current Tariffs.
2. Market Prices and Mean (Load) are strongly correlated with Tariffs.
3. Higher Mean (RES) tends to lower Tariffs, likely due to reduced reliance on carbon-intensive sources.
4. CE Budget and carbon intensity metrics (Mean (CI), IQR (CI)) may influence tariffs indirectly through
     regulatory mechanisms.

Feedback to implement: Since this is the first iteration, no prior feedback to incorporate. Initial rules
     focus on leveraging autocorrelation (Tariffs-Lag) and market price signals.
```

```
Changes:
Added condition on Market Prices > 0.08 to capture high-price scenarios; nested Tariffs-Lag check for
     autocorrelation; added Mean (Load) threshold for demand-driven pricing; included CE Budget for carbon
     considerations.
```

Once the rules are computed and the history table gets updated, feedback is requested. The box below transcribes the recommendations the second LLM provided the Rule Generator LLM to apply in the next iteration.

```
Feedback:

### Feedback for Rule-Generator LLM

#### 1. **Comparison of Current, Previous, and Best Rules**
- **Current Rules (Iteration 1):** The current rules are based on `Market Prices`, `Tariffs-Lag`, `Mean (
     Load)`, and `CE Budget`. The MAE is 0.18691.
- **Previous Iteration:** No previous iteration data is available.
- **Best Rules So Far:** The current rules are the best so far with an MAE of 0.18691.

#### 2. **Analysis of History and MAE Impact**
- The current rules are the first iteration, so there is no history to compare against. However, the worst
     predictions indicate that the model is overestimating tariffs in certain scenarios, particularly when
     `Market Prices` are high but `Tariffs-Lag` is low.

#### 3. **Recommendation to Be Less Conservative**
```

```
- The model is currently conservative in its predictions, particularly in the `else` clause where it
      returns 0.30 or 0.35. This may not capture the full range of possible tariff values. However, since
      this is the first iteration, it is reasonable to start with conservative estimates.

#### 4. **Concrete Rule Improvements (Max 10 Items)**
1. **Adjust the `Market Prices` Threshold:**
   - Change the threshold from `> 0.08` to `> 0.10` to better capture high-price scenarios.
2. **Modify the `Tariffs-Lag` Threshold:**
   - Change the threshold from `> 0.55` to `> 0.58` to better reflect the data distribution.
3. **Adjust the `Mean (Load)` Threshold:**
   - Change the threshold from `> 100` to `> 120` to better capture high-load scenarios.
4. **Modify the `CE Budget` Threshold:**
   - Change the threshold from `> 100000` to `> 150000` to better reflect the data distribution.
5. **Add a Condition for `Self-elasticity`:**
   - If `Self-elasticity` is very low (e.g., `< -0.5`), it may indicate highly price-sensitive demand, and
       the tariff could be set lower.
6. **Incorporate `IQR (Load)`:**
   - If `IQR (Load)` is very high, it may indicate high variability in load, and the tariff could be set
       higher to manage risk.
7. **Incorporate `CV (Load)`:**
   - If `CV (Load)` is very high, it may indicate high variability in load relative to the mean, and the
       tariff could be set higher to manage risk.
8. **Add a Condition for `Mean (RES)`:**
   - If `Mean (RES)` is very high, it may indicate abundant renewable energy supply, and the tariff could
       be set lower.
9. **Add a Condition for `Mean (CI)`:**
   - If `Mean (CI)` is very high, it may indicate high carbon intensity, and the tariff could be set higher
        to encourage off-peak charging.
10. **Add a Condition for `Hour`:**
    - Different hours may have different pricing needs. For example, peak hours (e.g., 17-20) could have
       higher tariffs.

#### 5. **Compliance Check**
- **No `elif`:** The current rules do not use `elif` statements. Compliance: PASS.
- **Nested Tree Only:** The current rules use a nested tree structure. Compliance: PASS.
- **No Forbidden Functions/Imports:** The current rules do not use any forbidden functions or imports.
     Compliance: PASS.
- **Comments on Every Split:** The current rules do not have comments. It is recommended to add comments to
      explain the reasoning behind each condition.
- **Predictions Within Tariff Range:** The current rules return values between 0.30 and 0.60, which are
     within the observed tariff range. Compliance: PASS.
- **Changes Made:** The proposed changes are listed above. Compliance: PASS.
- **No Leakage:** The current rules do not use `row['Tariffs']` to predict `row['Tariffs']`. Compliance:
     PASS.

#### Final Compliance Score
Compliance: PASS (90/100)

#### Updated Rule-Based Prediction Function with Improvements
```python
def rule_based_prediction(row):
    # Check if Market Prices are high
    if row['Market Prices'] > 0.10:
        # If Tariffs-Lag is very high, set a high tariff
        if row['Tariffs-Lag'] > 0.58:
            return 0.60
        else:
            # If Tariffs-Lag is moderate, set a moderate tariff
            return 0.55
    else:
        # If Mean (Load) is very high, set a moderate tariff
        if row['Mean (Load)'] > 120:
            return 0.45
        else:
            # If CE Budget is very high, set a low tariff
            if row['CE Budget'] > 150000:
                return 0.35
            else:
                # If Self-elasticity is very low, set a low tariff
                if row['Self-elasticity'] < -0.5:
                    return 0.25
                else:
                    # If IQR (Load) is very high, set a higher tariff
                    if row['IQR (Load)'] > 50:
                        return 0.50
                    else:
                        # If CV (Load) is very high, set a higher tariff
                        if row['CV (Load)'] > 0.8:
                            return 0.50
                        else:
                            # If Mean (RES) is very high, set a low tariff
```
```

```
                        if row['Mean (RES)'] > 100:
                            return 0.30
                        else:
                            # If Mean (CI) is very high, set a higher tariff
                            if row['Mean (CI)'] > 120:
                                return 0.40
                            else:
                                # Default to a moderate tariff
                                return 0.35
```

This feedback provided concrete suggestions, such as threshold changes or the introduction of new conditions or unused variables. Furthermore, the LLM carefully verified whether the hard constraints were respected, recommending a correction whenever a fault was detected. In this specific case (unlike what typically occurs), the LLM proposed an entirely new rule definition instead of solely providing isolated suggestions for adjustments. This suggestion was adopted by the Rule Generator LLM, resulting in a slight reduction in prediction error.

However, the first generated rules are too simple. Therefore, they do not properly capture the complex patterns between features themselves and between them and the target. Across several iterations, as the training data was updated and the feedback mechanisms refined and expanded the rule set, these initial limitations were progressively overcome, leading to the best rule set transcribed below, obtained in iteration 813.

```python
def rule_based_prediction(row):
    if row['Market Prices'] > 0.0058:  # Capture low-price scenarios
        if row['Tariffs-Lag'] > 0.06:  # Lower threshold for lagged value usage
            return row['Tariffs-Lag']
        else:
            if row['CE Budget'] > 72000:  # Prioritize carbon constraints
                if row['Mean (Load)'] > 120 and row['Mean (RES)'] < 30:
                    return 0.60
                else:
                    if row['Mean (Load)'] > 100 and row['Mean (RES)'] < 40:
                        return 0.55
                    else:
                        return 0.05
            else:
                if row['Self-elasticity'] < -0.15:  # Elasticity-based adjustment
                    return 0.20
                else:
                    if row['Mean (Load)'] > 120:
                        return 0.60
                    else:
                        if row['Mean (Load)'] < 10:  # Lower threshold for extreme
                            low load
                            return 0.05
                        else:
                            if row['CV (Load)'] > 0.70 or row['CV (RES)'] > 0.40:  #
                                Streamlined CV thresholds
                                if row['Mean (RES)'] > 100:
                                    return 0.40
                                else:
```

```
                                    return 0.55
                            else:
                                if row['Mean (CI)'] > 135:  # Refined CI threshold
                                    return 0.08
                                else:
                                    if row['IQR (RES)'] > 25:  # Adjusted IQR
                                        threshold
                                        return 0.50
                                    else:
                                        if row['IQR (CI)'] > 10:
                                            return 0.40
                                        else:
                                            return 0.35
    else:
        if row['Hour'] in [18, 19, 20]:  # Narrowed peak hours
            if row['Mean (Load)'] > 80 and row['Mean (RES)'] > 50:  # Combined load
                and RES conditions
                return 0.55
            else:
                if row['CV (RES)'] > 0.45:
                    return 0.45
                else:
                    if row['IQR (CI)'] > 10:
                        return 0.40
                    else:
                        return 0.35
        else:
            if row['Hour'] in [6, 7, 8, 9, 10]:  # Morning peak
                if row['Mean (Load)'] < 10:  # Lower threshold for extreme low load
                    return 0.05
                else:
                    if row['Mean (RES)'] > 40:
                        return 0.30
                    else:
                        return 0.15
            else:
                if row['Hour'] in [22, 23]:  # Late night
                    if row['Mean (Load)'] < 25:
                        return 0.05
                    else:
                        return 0.10
                else:
                    if row['Tariffs-Lag'] > 0.06:  # Lower threshold for lagged
                        value
                        return row['Tariffs-Lag']
                    else:
                        if row['Self-elasticity'] < -0.25:  # Lower elasticity
                            threshold
                            return 0.20
                        else:
                            if row['Mean (RES)'] > 150:  # Encourage charging with
                                high RES
                                return 0.30
                            else:
                                if row['Mean (CI)'] > 100:
```

```
                                    return 0.35
                                else:
                                    if row['CV (Load)'] > 0.70:
                                        return 0.40
                                    else:
                                        return 0.25
    prediction = max(0.05, min(0.60, prediction))  # Ensure predictions stay within
        valid range
    return prediction
```

Such rules resulted in an MAE equal to 0.0334 €/kWh. Compared to the initial rule set, the best set demonstrates significant complexity and refinement. The decision logic started to incorporate a broader range of features, thresholds were adjusted, and the overall logic became more granular. The rules evolved from capturing only general trends to reflecting more nuanced, context-dependent conditions, resulting in a more sophisticated prediction mechanism.

Once more, this work made available below the Reasoning, Justification and Changes Made between the best and previous iteration. The effect of the Changes made becomes more apparent at this stage. As the LLM carefully documented the modifications applied at each iteration, incorporating these in a table makes the overall optimisation trajectory clearer, providing direct insights into how the rule set evolved.

```
Reasoning:
 Okay, I need to generate a new rule-set for predicting tariffs based on the given data and feedback. Let
      me start by understanding the problem and the data.

First, the target variable is 'Tariffs', which is a numerical value. The features include various metrics
     like 'Self-elasticity', 'Hour', 'Mean (Load)', 'Mean (RES)', etc. The data seems to be a time series,
     given the 'Hour', 'Day', and 'Month' features, and it's likely capturing hourly data for EV charging
     stations.

Looking at the dataset summary, I notice that 'Tariffs' have a mean of around 0.345 and a max of 0.60. The
     features have varying distributions, with some like 'Mean (RES)' having a high standard deviation,
     indicating variability in renewable energy supply.

The previous rules had an MAE of 0.0336, and the best so far is 0.03345. The peer feedback suggested issues
      with overestimation in certain scenarios, especially when the actual tariff is low but the model
     predicts higher. This indicates that the model might not be capturing conditions for low tariffs
     effectively.

I should focus on refining the conditions that lead to lower tariffs, such as when the load is very low or
     when renewable energy is abundant. Also, the peak hour window might be too broad, causing higher
     tariffs to be applied unnecessarily.

From the feedback, adjusting the peak hour window to 18-20 could help. Lowering the threshold for 'Mean (
     Load)' to 10 might better capture scenarios where demand is extremely low. Using 'Tariffs-Lag' more
     effectively, perhaps with a lower threshold, could provide a better fallback.

I'll also consider simplifying some conditions to make the model more robust. For example, using a single
     CV threshold for both Load and RES could streamline the decision process.

Let me structure the new rules step by step:

1. **Market Prices Check**: If prices are high, use 'Tariffs-Lag' if it's above a certain threshold.
     Otherwise, prioritize carbon budget and load conditions.
2. **Carbon Budget and Load/RES Conditions**: If the carbon budget is high, apply higher tariffs when load
     is high and RES is low. Otherwise, adjust based on elasticity and other factors.
3. **Peak Hours**: Narrow the peak window to 18-20 and apply higher tariffs when both load and RES are
     within specific ranges.
4. **Morning and Late Night Handling**: Adjust tariffs based on load and RES, ensuring lower tariffs when
     demand is low.
5. **Fallback Conditions**: Use 'Tariffs-Lag' as a fallback and adjust based on elasticity and other
     metrics.
```

```
I'll ensure that all conditions are nested properly without using 'elif' statements and that the final
     prediction is clamped within the valid range.

Testing these changes should help reduce the MAE by better capturing the scenarios where tariffs should be
     low and avoiding overestimation.
</think>
```

```
Justification:
Nature: The dataset appears to be a time series with hourly granularity, capturing energy demand, renewable
      supply, and carbon intensity. Features like 'Self-elasticity' and 'Tariffs-Lag' suggest temporal
     dependencies. 'Market Prices' and 'CE Budget' introduce external constraints.

Patterns: High 'Market Prices' correlate with higher tariffs, while low 'Mean (Load)' and high 'Mean (RES)'
      suggest opportunities for lower tariffs. 'Self-elasticity' influences price sensitivity, and 'Tariffs
     -Lag' captures daily patterns.

Feedback to implement: Adjust peak hour window, refine load thresholds, simplify CV conditions, and better
     handle extreme load/RES scenarios. Incorporate Tariffs-Lag more effectively and ensure predictions
     stay within valid range.
```

```
    Changes Made: Adjusted peak hour window to 18-20; Lowered Mean (Load) threshold to 10; Simplified CV
         thresholds to 0.70; Introduced Mean (RES) > 150 condition; Lowered Tariffs-Lag threshold to 0.06;
         Lowered Self-elasticity threshold to -0.25; Added clamping step
```

Furthermore, Figures 5.22 and 5.23 illustrate the evolution of MAE and the progression of the best MAE across iterations, respectively, in the scenario where the LLM operates without access to the Tariffs-Lag feature, which, as expected, increased prediction error. Most of the previously discussed conclusions still hold in this scenario, as, for instance, every simulation experiences a decrease in error across iterations and sudden increases in error are promptly corrected, with the model quickly reverting to better-performing solutions.

Simulations NL-Sim1, NL-Sim3 and NL-Sim5 reached an error inferior to that of NL-LR, but still kept a considerable distance to the prediction performance of NL-XGB. Moreover, although NL-Sim2 results display a degree of stagnation in the final iterations, it is likely that small improvements would still occur had the number of iterations been bigger, potentially allowing it to reach error levels comparable to NL-LR.

Figures 5.21 and 5.23 reveal noticeable differences between the considered scenarios. In particular, the second Figure displays significantly more noise, with much more frequent abrupt changes in MAE across iterations. This behaviour is noteworthy, as the model exhibits much less stable progress compared to the scenario where the Lag Feature is included.

However, LLM hallucination was observed in NL-Sim4, whose execution prematurely terminated due to the model's attempt to use "Carbon Intensity" as a predictor, a variable that does not exist within the dataset.

The generated rule functions and prompt responses are not presented in this scenario, as their structures and conclusions would not be more informative of this process or its results. Given their lower overall performance observed, their inclusion was considered to offer little additional insight beyond the quantitative results already discussed.

Figure 5.22: MAE Evolution for simulations excluding Tariffs-Lag.

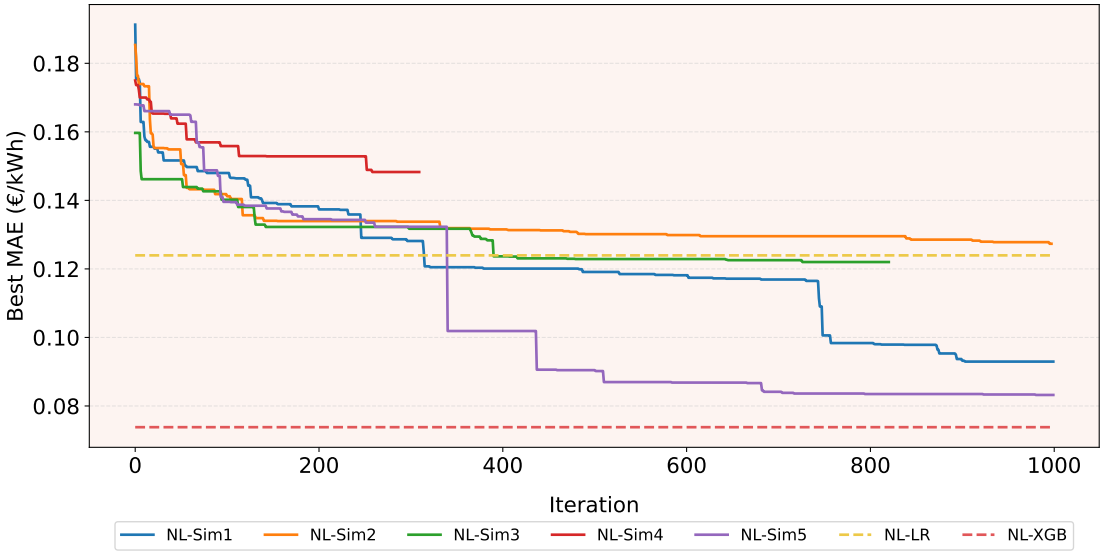

Figure 5.23: Best MAE until each iteration for simulations excluding Tariffs-Lag.

5.4 Discussion of Key Findings

As EV adoption continues to grow, driven by growing environmental concerns, new challenges emerge around infrastructure and pricing models. Among these, the lack of transparency in EV charging Tariffs stands out, as prices are typically defined by opaque optimisation frameworks. This challenge motivated the first part of this dissertation, which explored the use of interpretable ML models to clarify how EV charging tariffs are determined. By expanding the framework proposed by Silva *et al.* [4], which focused on the design of carbon-aware dynamic tariffs, this work aimed to explain the underlying mechanisms using transparent, human-readable models.

In terms of predictive performance, ensemble methods such as XGB and GBT achieved the highest performance (MAE equal to 0.028797 €/kWh and 0.029098 €/kWh, respectively). In particular, XGB not only delivered the best performance but also demonstrated the strongest generalisation from training to the test, confirming its robustness in handling unseen data. Contrarily, LGB exhibited the weakest generalisation, likely due to its greater tree depth and leaf-wise growth strategy, which may have contributed to overfitting.

However, their inherent structural complexity limits their interpretability. In contrast, simpler models such as DT delivered solid predictive performance while remaining inherently transparent. This model's rule-based nature allows for direct, step-by-step explanation of predictions, making them a practical solution when transparency is required.

The issue of interpretability in ensemble models was further analysed through the application of SySM, which proved useful in bagging methods such as Random Forest. When applied to RF, the tree selected by SySM exhibited a structure that by the method's definition closely mirrors the forest's aggregated decision logic, while maintaining a prediction error only marginally higher than that of the best estimator and the full ensemble. Even though the best estimator achieved a lower error (0.0337 €/kWh against 0.0360 €/kWh), it is arguable that the split pattern may represent the individual uniqueness of this tree rather than the ensemble's average "reasoning". Hence, SySM is preferable when the primary goal is to capture the common decision patterns of the forest.

In contrast, boosting ensembles inherently resist structural summarisation. Since XGB, for instance, is successively trained on the residuals to correct their predecessors, searching for structural similarity conflicts with the deliberate diversity, natural to the boosting algorithm. Consequently, the tree returned by SySM is significantly different from the ensemble in both structure and MAE (0.0346 €/kWh against 0.1943 €/kWh), confirming that no belonging single tree can faithfully represent a boosted model's sequential error correction.

Therefore, post-hoc explanation techniques remain the most reliable path to grasp somewhat of an understanding of the model predictions. Feature importance analysis for XGB highlighted lagged tariffs, time variables, consumption patterns, elasticity and renewable availability as the most influential predictors. This alignment with domain knowledge reinforces model credibility.

Beyond structural analysis, the interpretability of each model was quantitatively assessed using cognitive-effort metrics, namely RMO and TMO. These metrics formalised the trade-off between predictive performance and interpretability. Results show how DT combined a low prediction error with minimal cognitive effort, confirming it as the most balanced model when transparency is a requirement. The representative tree extracted from Random Forest using SySM proved valuable, maintaining reasonable predictive performance while significantly reducing interpretative complexity compared to full ensemble structures. Interestingly, despite having the same number of estimators as RF (100), LGB demonstrates higher Ensemble Interpretability, likely due to its leaf-wise growth strategy, which resulted in shorter prediction paths and reduced cognitive effort.

Even though single-tree representations of boosting models exhibit satisfactory interpretability, their higher prediction error indicates, once more, that boosting models cannot be simplified

through selection within their estimators without significant performance loss. However, it should be noted that these interpretability metrics may have been somewhat unfair in the case of LR, as counting individual arithmetic operations can overstate its complexity. Despite being a single, easily understandable expression, LR may appear less interpretable than it truly is.

Although predictive performance was not the primary goal, this dissertation applied techniques to optimize results. Firstly, it employed two different techniques to select each model's hyperparameters. Randomized Search surprisingly outperformed Grid Search regarding generalization capabilities, underlining the risk of overfitting in exhaustive search configurations, while achieving these results in substantially less time. Secondly, a brief, exploratory feature-engineering experiment combined an Evolutionary Forest with a Decision Tree Regressor. After the generation of a new set of features, the 20 most important were selected and incorporated into the dataset. However, this EF-DT variant achieved only a marginal reduction (-0.0000053 €/kWh) in error relative to the baseline tree. Such could be explained by the dominance of the Tariffs-Lag, which leaves barely any room for additional engineered features, although the limited exploration of the Evolutionary forest's hyperparameter space, as well as the restriction to only 20 generated features, may also have constrained potential gains.

Beyond tree-based models themselves, this work also explored the potential of LLMs to enhance explainability. By prompting OpenAI's model with the extracted tree structure, LLM successfully generated clear, human-readable explanations that clearly reflect the overall model's decision logic. It correctly identified the primary role of the lagged Tariff and acknowledged the influence of elasticity, renewable availability and demand patterns. This demonstrates that LLMs can act as a complementary tool, translating inherently transparent models into accessible narratives for non-technical stakeholders.

In addition to their role in explanation tasks, their ability to generate rule-based predictions is explored as the second core of this dissertation. In this framework, the LLM was iteratively prompted to minimise prediction error by refining its rule set at each iteration, based on internal feedback mechanisms and historical performance data. Throughout each iteration, the LLM was prompted to return an executable Python function containing the rule-set, to document a justification, and to write a small string highlighting the changes made from the previous iteration to the current one.

Building upon this iterative structure, the scenario incorporating the lag feature delivered the strongest results, aligning with ML standard time series practices that emphasise the use of historical values for improving predictive performance. Sim1 emerged as the best-performing solution, surpassing LR and producing results inferior, yet comparable, to those of XGB. After 813 iterations, Sim1 achieved an MAE equal to 0.334 €/kWh, progressively refining its rule-set as the LLM leveraged its feedback mechanism to minimise error. Given the consistent downward trend in error, it is reasonable to assume that additional iterations could have yielded further improvements.

Beyond structural interpretability, provided by the nested decision tree format, the model's transparency was enhanced through the inclusion of explicit justifications and reasoning at each

iteration. These justifications demonstrated the LLM's ability to detect relevant patterns within the training chunk it received, while also showing strong contextual understanding of the problem domain.

In summary, findings confirm that for this specific dataset and problem, a DT regressor model emerges as the most practical solution, combining interpretability with sufficient performance. Its clear, rule-based structure enables stakeholders to follow a logical decision-making process, making it the most suitable approach where transparency and explainability, such as the one at hand. Ensemble models, particularly boosting algorithms, offered higher predictive performance but should be accompanied by post-hoc explanation techniques when explainability is required. As transparency and traceability are priorities, simpler models are preferable. The exploratory use of LLMs revealed their potential as both explanation tools and rule generators. When applied to tree-based models, the LLM successfully translated models structured into clear, human-readable narratives, enhancing accessibility for non-technical users. Furthermore, by iteratively generating transparent rule-sets along with justifications, the LLM was able to build interpretable models without requiring, from the user, any expertise in programming or ML therefore highlighting this framework's potential for democratising access to transparent predictive modelling.

Chapter 6

Conclusions

6.1 Final Considerations

This dissertation addressed the challenge of making EV charging tariffs, often assigned through opaque optimization models, transparent, through a dataset generated from a previously developed carbon-aware dynamic pricing optimization framework. The work focused on two main strategies: applying interpretable surrogate models, based on ML, to approximate the optimisation model and developing a novel LLM-based rule generation framework aimed at enabling non-expert users to create transparent predictive models without requiring programming or machine learning knowledge.

In the first approach, various ML models were applied to approximate the optimisation model's charging prices. Among these ML models, most boosting algorithms and particularly XGB, achieve the highest predictive accuracy but remain opaque due to their structural complexity. Contrarily, simpler models as DT offered a practical trade-off, delivering a slightly inferior accuracy while ensuring full interpretability through their clear, rule-based structure. Therefore, for this dataset and problem, DT proved to be the most suitable solution where transparency and traceability are required.

LLMs were explored in two different roles. First, as explanation tools, where the LLM effectively translated a tree-based structure into human-readable narratives, making Tariff's decision-making process accessible to non-technical users. Second, as rule generators, developing a method through which the LLM autonomously produces and refines rule-sets for prediction tasks, accompanied by justifications for each modification. Beyond iteratively improving its own results, it surpassed simple models such as LR, effectively demonstrating the potential to democratise access to predictive modelling, enabling non-specialists, with no programming or ML knowledge, to build interpretable models.

In conclusion, this dissertation demonstrates that EV charging tariff models can be made transparent either through interpretable ML or by leveraging LLMs as explanation tools. Furthermore, the proposed LLM-based rule generation framework shows that non-expert users can generate transparent predictive models without technical expertise, contributing to the democratisation of

interpretable modelling. Overall, results confirm that transparency in dynamic pricing is achievable without notably compromising predictive performance.

6.2 Limitations

While this dissertation achieved positive outcomes achieved in this dissertation regarding the development of transparent models for EV Charging Tariff prediction, it is important to recognise that the methodologies adopted introduce limitations that must be considered when assessing broader applicability and scalability of the proposed methods and solutions.

For instance, within the ML component of this research work, the feature engineering process remained relatively limited, since explainability was the main focus. Besides adding a lag feature and a simple exploratory application of an evolutionary forest, this step of the ML pipeline remained relatively underdeveloped. Our application of SySM, revealed to be computationally expensive, thereby challenging to apply within forest with an increased number of estimators. Furthermore, from an evaluation perspective, model performance was exclusively assessed using MAE. Although appropriate, the use of complementary evaluation metrics, such as relative or percentage-based errors, could have provided a more comprehensive assessment of the model's predictive performance.

In terms of interpretability assessment, even though the resorted metrics (RMO and TMO) objectively quantify the cognitive effort by considering each step required to understand the model (conditional branches, assignment, arithmetical operations), evaluation involving real users would be essential to assess how interpretability a Decision Tree truly is when considering diverse user perspectives.

The explanation of a Decision Tree through an LLM also presents relevant limitations. Firstly, its brief implementation was restricted to Decision Tree Regressor models, having its applicability to ensembles or more complex single tree models not been tested during this dissertation. Moreover, LLM explanations are susceptible to hallucinations, potentially generating incorrect or unjustified statements. Therefore, expert validation remains necessary to ensure the reliability of the explanations produced. At this stage, only one LLM was tested, and a small effort was put into refining the prompt, which is known to be directly linked to LLM performance.

Regarding the LLM Rule Generator, although the methodology was designed to be adaptable to any dataset or problem, including our own, its generalization was not assessed beyond the dataset used in this study. Similarly to the Explainer, the rule generation process is sensitive to prompt formulation and prone to hallucinations, occasionally resulting in invalid rules or justifications. Furthermore, alternative LLM architectures were not explored. Additionally, the method is computationally demanding, with approximately one full day required to complete 1000 iterations. Finally, while the framework demonstrated the ability to generate transparent models accessible to non-expert users, objective measures were not developed to assess cognitive effort. Perhaps most importantly, despite several attempts, this work was unable to fixate the LLM's responses, thereby requiring multiple executions to demonstrate the framework's effectiveness.

6.3 Future Work

Building on both the results achieved and limitations identified during this dissertation, this section outlines several directions for future research, proposed to address challenges and extend the applicability of the developed approaches.

Focusing, firstly, on the ML stage, further work should focus on integrating automatic feature selection methods into the pipeline, so that only the most relevant variables are incorporated and model complexity is reduced. Furthermore, conducting studies with real stakeholders could help assess whether models deemed structurally simple are also perceived as interpretable in practice. An alternative approach, regarding the interpretability of the ensemble and the application of SySM, would be to compare the structurally most similar tree not to the best-performing estimator, but rather to the one whose prediction error is closest to that of the full ensemble. Evidently, the incorporation of other interpretable models could be explored, along with post-hoc explainability techniques. For instance, RuleFit, which blends decision rules with linear regression, is an interesting point to explore, as it is able to extract rules directly from tree-based ensembles and apply them within a sparse linear framework.

Regarding the LLM Explainer, several enhancements could be pursued. Firstly, the current methodology should compare its results to traditional explainability techniques, such as LIME or SHAP, in order to evaluate their relative clarity and effectiveness. Another research avenue is to include these results within the prompt to increase model knowledge and likely retrieve more complete and context-aware natural language explanations. Furthermore, implementing a feedback mechanism, similar to the iterative refinement used in the Rule Generator, should be explored. Such would involve a secondary LLM to evaluate the quality of each explanation and iteratively prompt for improved outputs based on structured feedback.

Similarly, within the LLM Rule Generator, the current framework could be extended to produce ensembles of rule sets rather than a simple solution, improving robustness and reliability at the cost of some explainability. Additionally, introducing explicit control over the trade-off between predictive accuracy and interpretability during rule generation would also enhance the framework's applicability, allowing for transparent models when required. Future versions could also explore incorporating mathematical formulations directly and measuring their impact on rule set generation.

Across both LLM-based frameworks, several common research directions should be considered. Firstly, future work should focus on the impact of varying temperature settings, not only globally across executions but also iteratively during generation, to evaluate the influence of creativity control on output quality and consistency. Testing different LLM architectures would help to clarify the extent to which model choice affects performance and reliability. Finally, empirical studies involving stakeholders will be essential to validate whether the LLM's outputs are genuinely interpretable and useful in practical contexts.

In summary, future work should aim to refine modelling methodologies and interpretability frameworks, incorporating human-centred evaluations as a critical step towards ensuring practical

relevance, transparency, and trustworthiness of both ML and LLM-driven models.

Bibliography

[1] Sudarshan Gnanavendan et al. “Challenges, Solutions and Future Trends in EV-Technology: A Review”. In: *IEEE Access* 12 (2024), pp. 17242–17260. ISSN: 2169-3536. DOI: 10.1109/ACCESS.2024.3353378. (Visited on Jan. 13, 2025).

[2] *Global EV Outlook 2024*. https://www.iea.org/reports/global-ev-outlook-2024. Apr. 2024. (Visited on Jan. 9, 2025).

[3] Ugur Fesli and Mustafa Bahadir Ozdemir. “Electric Vehicles: A Comprehensive Review of Technologies, Integration, Adoption, and Optimization”. In: *IEEE Access* 12 (2024), pp. 140908–140931. ISSN: 2169-3536. DOI: 10.1109/ACCESS.2024.3469054. (Visited on Jan. 9, 2025).

[4] Carlos A. M. Silva and Ricardo J. Bessa. “Carbon-Aware Dynamic Tariff Design for Electric Vehicle Charging Stations with Explainable Stochastic Optimization”. In: *Applied Energy* 389 (July 2025), p. 125674. ISSN: 0306-2619. DOI: 10.1016/j.apenergy.2025.125674. (Visited on June 3, 2025).

[5] Mitul Ranjan Chakraborty et al. “Evolution and Present Status of Electric Vehicles: A Comprehensive Review”. In: *2024 International Conference on Computational Intelligence for Green and Sustainable Technologies (ICCIGST)*. July 2024, pp. 1–6. DOI: 10.1109/ICCIGST60741.2024.10717604. (Visited on Jan. 13, 2025).

[6] Kavuri Poornesh, Kuzhivila Pannickottu Nivya, and K. Sireesha. “A Comparative Study on Electric Vehicle and Internal Combustion Engine Vehicles”. In: *2020 International Conference on Smart Electronics and Communication (ICOSEC)*. Sept. 2020, pp. 1179–1183. DOI: 10.1109/ICOSEC49089.2020.9215386. (Visited on Jan. 13, 2025).

[7] Fayez Alanazi. “Electric Vehicles: Benefits, Challenges, and Potential Solutions for Widespread Adaptation”. In: *Applied Sciences* 13.10 (Jan. 2023), p. 6016. ISSN: 2076-3417. DOI: 10.3390/app13106016. (Visited on Jan. 9, 2025).

[8] Steffen Limmer. “Dynamic Pricing for Electric Vehicle Charging—A Literature Review”. In: *Energies* 12.18 (Sept. 2019), p. 3574. ISSN: 1996-1073. DOI: 10.3390/en12183574. (Visited on Oct. 3, 2022).

[9] Arun Kumar Kalakanti and Shrisha Rao. “Computational Challenges and Approaches for Electric Vehicles”. In: *ACM Computing Surveys* 55.14s (Dec. 2023), pp. 1–35. ISSN: 0360-0300, 1557-7341. DOI: 10.1145/3582076. (Visited on Jan. 10, 2025).

[10] Qin Chen and Komla Agbenyo Folly. “Application of Artificial Intelligence for EV Charging and Discharging Scheduling and Dynamic Pricing: A Review”. In: *Energies* 16.1 (Dec. 2022), p. 146. ISSN: 1996-1073. DOI: 10.3390/en16010146. (Visited on Mar. 16, 2023).

[11] Yeming Dai et al. “A Dynamic Pricing Scheme for Electric Vehicle in Photovoltaic Charging Station Based on Stackelberg Game Considering User Satisfaction”. In: *Computers & Industrial Engineering* 154 (Apr. 2021), p. 107117. ISSN: 03608352. DOI: 10.1016/j.cie.2021.107117. (Visited on Mar. 23, 2023).

[12] Rongshan Yu, Wenxian Yang, and Susanto Rahardja. “A Statistical Demand-Price Model With Its Application in Optimal Real-Time Price”. In: *IEEE Transactions on Smart Grid* 3.4 (Dec. 2012), pp. 1734–1742. ISSN: 1949-3053, 1949-3061. DOI: 10.1109/TSG.2012.2217400. (Visited on Mar. 22, 2022).

[13] Moncef Garouani et al. “Investigating the Duality of Interpretability and Explainability in Machine Learning”. In: Oct. 2024, pp. 861–867. DOI: 10.1109/ICTAI62512.2024.00125. arXiv: 2503.21356 [cs]. (Visited on July 23, 2025).

[14] Giulia Vilone and Luca Longo. “Classification of Explainable Artificial Intelligence Methods through Their Output Formats”. In: *Machine Learning and Knowledge Extraction* 3.3 (Aug. 2021), pp. 615–661. ISSN: 2504-4990. DOI: 10.3390/make3030032. (Visited on Feb. 7, 2024).

[15] Xiangyu Li et al. “Deep Reinforcement Learning-Based Explainable Pricing Policy for Virtual Storage Rental Service”. In: *IEEE Transactions on Smart Grid* 14.6 (Nov. 2023), pp. 4373–4384. ISSN: 1949-3061. DOI: 10.1109/TSG.2023.3253140. (Visited on Jan. 11, 2025).

[16] Max Biggs, Wei Sun, and Markus Ettl. *Model Distillation for Revenue Optimization: Interpretable Personalized Pricing*. June 2021. DOI: 10.48550/arXiv.2007.01903. arXiv: 2007.01903 [stat]. (Visited on Jan. 11, 2025).

[17] Sebatian Pütz et al. “Regulatory Changes in Power Systems Explored with Explainable Artificial Intelligence”. In: *Companion Proceedings of the 14th ACM International Conference on Future Energy Systems*. June 2023, pp. 26–31. DOI: 10.1145/3599733.3600247. arXiv: 2303.17455 [eess]. (Visited on Jan. 12, 2025).

[18] Johannes Kruse, Benjamin Schäfer, and Dirk Witthaut. “Revealing Drivers and Risks for Power Grid Frequency Stability with Explainable AI”. In: *Patterns* 2.11 (Nov. 2021), p. 100365. ISSN: 26663899. DOI: 10.1016/j.patter.2021.100365. arXiv: 2106.04341 [eess]. (Visited on Jan. 12, 2025).

[19] Weiche Hsieh et al. *A Comprehensive Guide to Explainable AI: From Classical Models to LLMs*. Dec. 2024. DOI: 10.48550/arXiv.2412.00800. arXiv: 2412.00800 [cs]. (Visited on June 2, 2025).

[20] Chandan Singh et al. *Rethinking Interpretability in the Era of Large Language Models*. Jan. 2024. DOI: 10.48550/arXiv.2402.01761. arXiv: 2402.01761 [cs]. (Visited on June 2, 2025).

[21] Jia He et al. *Does Prompt Formatting Have Any Impact on LLM Performance?* Nov. 2024. DOI: 10.48550/arXiv.2411.10541. arXiv: 2411.10541 [cs]. (Visited on June 2, 2025).

[22] Sebastian Bordt et al. *Data Science with LLMs and Interpretable Models*. Feb. 2024. DOI: 10.48550/arXiv.2402.14474. arXiv: 2402.14474 [cs]. (Visited on June 2, 2025).

[23] Chiara Bordin and Asgeir Tomasgard. "Behavioural Change in Green Transportation: Micro-Economics Perspectives and Optimization Strategies". In: *Energies* 14.13 (June 2021), p. 3728. ISSN: 1996-1073. DOI: 10.3390/en14133728. (Visited on Sept. 20, 2023).

[24] Xuansheng Wu et al. *Usable XAI: 10 Strategies Towards Exploiting Explainability in the LLM Era*. May 2025. DOI: 10.48550/arXiv.2403.08946. arXiv: 2403.08946 [cs]. (Visited on June 2, 2025).

[25] Lin Dong et al. *Exploring the Capabilities and Limitations of Large Language Models in the Electric Energy Sector*. Mar. 2024. DOI: 10.48550/arXiv.2403.09125. arXiv: 2403.09125 [eess]. (Visited on July 1, 2025).

[26] Haochen Xue et al. *LLM-Enhanced Feature Engineering for Multi-Factor Electricity Price Predictions*. May 2025. DOI: 10.48550/arXiv.2505.11890. arXiv: 2505.11890 [cs]. (Visited on July 2, 2025).

[27] Bowen Zhang and Pengcheng Luo. *OR-LLM-Agent: Automating Modeling and Solving of Operations Research Optimization Problem with Reasoning Large Language Model*. Mar. 2025. DOI: 10.48550/arXiv.2503.10009. arXiv: 2503.10009 [cs]. (Visited on July 2, 2025).

[28] Chengrun Yang et al. *Large Language Models as Optimizers*. Apr. 2024. DOI: 10.48550/arXiv.2309.03409. arXiv: 2309.03409 [cs]. (Visited on July 8, 2025).

[29] Hao Chen et al. *OptiChat: Bridging Optimization Models and Practitioners with Large Language Models*. Jan. 2025. DOI: 10.48550/arXiv.2501.08406. arXiv: 2501.08406 [cs]. (Visited on July 8, 2025).

[30] Shangeetha Sivasothy et al. *Large Language Models for Generating Rules, Yay or Nay?* June 2024. DOI: 10.48550/arXiv.2406.06835. arXiv: 2406.06835 [cs]. (Visited on July 2, 2025).

[31] Ashley Suh et al. "Don't Just Translate, Agitate: Using Large Language Models as Devil's Advocates for AI Explanations". In: (Apr. 2025). DOI: 10.5281/zenodo.15170455. arXiv: 2504.12424 [cs]. (Visited on July 2, 2025).

[32] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The elements of statistical learning: data mining, inference and prediction*. 2nd ed. Springer, 2009. URL: http://www-stat.stanford.edu/~tibs/ElemStatLearn/ (visited on May 25, 2025).

[33] Ibomoiye Domor Mienye and Nobert Jere. "A Survey of Decision Trees: Concepts, Algorithms, and Applications". In: *IEEE Access* 12 (2024), pp. 86716–86727. ISSN: 2169-3536. DOI: 10.1109/ACCESS.2024.3416838. (Visited on May 25, 2025).

[34] Shai Shalev-Shwartz and Shai Ben-David. *Understanding Machine Learning: From Theory to Algorithms*. 1st ed. Cambridge University Press, May 2014. ISBN: 978-1-107-05713-5 978-1-107-29801-9. DOI: 10.1017/CBO9781107298019. (Visited on May 25, 2025).

[35] Tom M Mitchell. *Machine learning*. Vol. 1. 9. McGraw-hill New York, 1997. (Visited on May 25, 2025).

[36] Ibomoiye Domor Mienye and Yanxia Sun. "A Survey of Ensemble Learning: Concepts, Algorithms, Applications, and Prospects". In: *IEEE Access* 10 (2022), pp. 99129–99149. ISSN: 2169-3536. DOI: 10.1109/ACCESS.2022.3207287. (Visited on May 26, 2025).

[37] J. R. Quinlan. "Learning With Continuous Classes". In: *Proceedings 5th Australian Joint Conference on Artifical Intelligence*. Singapore: World Scientific, 1992, pp. 343–348. (Visited on May 26, 2025).

[38] Y. Wang and I.H. Witten. *Induction of Model Trees for Predicting Continuous Classes*. 1996. (Visited on May 26, 2025).

[39] Guolin Ke et al. "LightGBM: A Highly Efficient Gradient Boosting Decision Tree". In: *Advances in Neural Information Processing Systems*. Vol. 30. Curran Associates, Inc., 2017. (Visited on May 26, 2025).

[40] Tianqi Chen and Carlos Guestrin. "XGBoost: A Scalable Tree Boosting System". In: *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. San Francisco California USA: ACM, Aug. 2016, pp. 785–794. ISBN: 978-1-4503-4232-2. DOI: 10.1145/2939672.2939785. (Visited on May 26, 2025).

[41] Desi Anggreani et al. "Grid Search Hyperparameter Analysis in Optimizing The Decision Tree Method for Diabetes Prediction". In: *Indonesian Journal of Data and Science* 5.3 (Dec. 2024), pp. 190–197. ISSN: 2715-9930. DOI: 10.56705/ijodas.v5i3.190. (Visited on May 25, 2025).

[42] Abhay Kumar Pathak, Mrityunjay Chaubey, and Manjari Gupta. *Randomized-Grid Search for Hyperparameter Tuning in Decision Tree Model to Improve Performance of Cardiovascular Disease Classification*. Nov. 2024. DOI: 10.48550/arXiv.2411.18234. arXiv: 2411.18234 [cs]. (Visited on May 25, 2025).

[43] James Bergstra and Yoshua Bengio. "Random search for hyper-parameter optimization". In: *J. Mach. Learn. Res.* 13 (Feb. 2012), pp. 281–305. ISSN: 1532-4435. (Visited on June 25, 2025).

[44] Hengzhe Zhang, Aimin Zhou, and Hu Zhang. "An Evolutionary Forest for Regression". In: *IEEE Transactions on Evolutionary Computation* 26.4 (Aug. 2022), pp. 735–749. ISSN: 1941-0026. DOI: 10.1109/TEVC.2021.3136667. (Visited on May 27, 2025).

[45] Hengzhe Zhang et al. "Modular Multitree Genetic Programming for Evolutionary Feature Construction for Regression". In: *IEEE Transactions on Evolutionary Computation* 28.5 (Oct. 2024), pp. 1455–1469. ISSN: 1941-0026. DOI: 10.1109/TEVC.2023.3318638. (Visited on May 27, 2025).

[46] Marcin Czajkowski, Krzysztof Jurczuk, and Marek Kretowski. "Steering the Interpretability of Decision Trees Using Lasso Regression - an Evolutionary Perspective". In: *Information Sciences* 638 (Aug. 2023), p. 118944. ISSN: 00200255. DOI: 10.1016/j.ins.2023.118944. (Visited on Sept. 12, 2024).

[47] Andrew Pang, Hyeju Jang, and Shiaofen Fang. "Generating Descriptive Explanations of Machine Learning Models Using LLM". In: *2024 IEEE International Conference on Big Data (BigData)*. Dec. 2024, pp. 5369–5374. DOI: 10.1109/BigData62323.2024.10825667. (Visited on June 2, 2025).

[48] Bo Wang et al. *Can LLM Assist in the Evaluation of the Quality of Machine Learning Explanations?* Feb. 2025. DOI: 10.48550/arXiv.2502.20635. arXiv: 2502.20635 [cs]. (Visited on June 2, 2025).

[49] F. Pedregosa et al. "Scikit-learn: Machine Learning in Python". In: *Journal of Machine Learning Research* 12 (2011), pp. 2825–2830. (Visited on June 2, 2025).

[50] Tomasz Marchela and Piotr Płoński. *supertree: interactive visualization of decision trees. Version 0.3.0*. Łapy, Poland, 2024. URL: https://github.com/mljar/supertree (visited on June 2, 2025).

[51] Abraham Itzhak Weinberg and Mark Last. "Selecting a Representative Decision Tree from an Ensemble of Decision-Tree Models for Fast Big Data Classification". In: *Journal of Big Data* 6.1 (Feb. 2019), p. 23. ISSN: 2196-1115. DOI: 10.1186/s40537-019-0186-3. (Visited on June 2, 2025).

[52] Massimo Aria et al. "Explainable Ensemble Trees". In: *Computational Statistics* 39.1 (Feb. 2024), pp. 3–19. ISSN: 1613-9658. DOI: 10.1007/s00180-022-01312-6. (Visited on June 2, 2025).

[53] Massimo Aria et al. *Extending Explainable Ensemble Trees (E2Tree) to Regression Contexts*. Sept. 2024. DOI: 10.48550/arXiv.2409.06439. arXiv: 2409.06439 [cs]. (Visited on June 2, 2025).

[54] Mousumi Banerjee, Ying Ding, and Anne-Michelle Noone. "Identifying Representative Trees from Ensembles". In: *Statistics in Medicine* 31.15 (July 2012), pp. 1601–1616. ISSN: 0277-6715, 1097-0258. DOI: 10.1002/sim.4492. (Visited on June 2, 2025).

[55] *What Is a Machine Learning Pipeline? | IBM*. https://www.ibm.com/think/topics/machine-learning-pipeline. May 2025. (Visited on July 18, 2025).

[56] Guido Van Rossum. *The Python Library Reference, release 3.8.2*. Python Software Foundation, 2020. (Visited on July 24, 2025).

[57] OpenAI. *o3 Model—OpenAIAPI Documentation*. 2025. URL: https://platform.openai.com/docs/models/o3 (visited on July 15, 2025).

[58] DeepSeek-AI. *DeepSeek-R1: Incentivizing Reasoning Capability in LLMs via Reinforcement Learning*. 2025. arXiv: 2501.12948 [cs.CL]. URL: https://arxiv.org/abs/2501.12948 (visited on July 8, 2025).

[59] DeepSeek-AI. *DeepSeek-R1-Distill-Qwen-32B — Model Card on Hugging Face*. 2025. URL: https://huggingface.co/deepseek-ai/DeepSeek-R1-Distill-Qwen-32B (visited on July 8, 2025).

[60] scikit-learn developers. *DecisionTreeRegressor — scikit-learn Documentation*. scikit-learn v1.7.1. 2025. URL: https://scikit-learn.org/stable/modules/generated/sklearn.tree.DecisionTreeRegressor.html (visited on June 30, 2025).

[61] scikit-learn developers. *GradientBoostingRegressor — scikit-learn Documentation*. scikit-learn v1.7.1. 2025. URL: https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.GradientBoostingRegressor.html (visited on June 30, 2025).

[62] XGBoost developers. *XGBoost Parameters — XGBoost Documentation*. XGBoost v3.0.2. 2025. URL: https://xgboost.readthedocs.io/en/stable/parameter.html (visited on June 30, 2025).

[63] LightGBM developers. *LGBMRegressor — LightGBM Documentation*. LightGBM v4.6.0.99. 2025. URL: https://lightgbm.readthedocs.io/en/latest/pythonapi/lightgbm.LGBMRegressor.html (visited on June 30, 2025).

[64] scikit-learn developers. *RandomForestRegressor — scikit-learn Documentation*. scikit-learn v1.7.1. 2025. URL: https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestRegressor.html (visited on June 30, 2025).

[65] Sylvain Marié. *'python-M5p' - M5 Prime Regression Trees in Python, Compliant with Scikit-Learn*. May 2025. (Visited on June 30, 2025).

[66] Francisco S. Fernandes, Ricardo J. Bessa, and João Peças Lopes. "Evolving Symbolic Model for Dynamic Security Assessment in Power Systems". In: *Journal of Modern Power Systems and Clean Energy* (2025), pp. 1–13. ISSN: 2196-5420. DOI: 10.35833/MPCE.2024.000478. (Visited on June 3, 2025).

[67] OpenAI. *o3 (ChatGPT Model)*. https://chat.openai.com. 2025. (Visited on July 4, 2025).