INSTITUTO SUPERIOR DE ENGENHARIA DO PORTO

MESTRADO EM ENGENHARIA DE INTELIGÊNCIA ARTIFICIAL

ALMA: ALgorithm Modeling Application

NUNO ANDRÉ LAPA OLIVEIRA

julho de 2022

P.PORTO

ALMA: ALgorithm Modeling Application

Research Group on Intelligent Engineering and Computing for Advanced Innovation and Development, Porto School of Engineering

2021/2022

Nuno André Lapa Oliveira

1170793

Thesis Jury

Porto, June 2022

ALMA: ALgorithm Modeling Application

Research Group on Intelligent Engineering and Computing for Advanced Innovation and Development, Porto School of Engineering

2021/2022

Thesis submitted for the

Master's Degree on Artificial Intelligence Engineering

authorship of

Nuno André Lapa Oliveira

1170793

supervised by

Prof. Dr. Isabel Cecília Correia da Silva Praça Gomes Pereira

Coordinator Professor, Polytechnic of Porto - School of Engineering

Prof. Dr. Nuno Alexandre Pinto da Silva

Coordinator Professor, Polytechnic of Porto - School of Engineering

”I have no special talent.

I am only passionately curious.”

Albert Einstein

Acknowledgements

I would like to start to thank the Polytechnic of Porto - School of Engineering (ISEP), especially the people behind the construction of the Master's Degree on Artificial Intelligence Engineering (MEIA), namely Prof. Carlos Ramos, for making such an important step towards the development of Artificial Intelligence in our country, providing means to learn these sort of technologies through formal training in higher education.

I also wish to show my gratitude to the Research Group on Intelligent Engineering and Computing for Advanced Innovation and Development (GECAD) for being my home for the last two and a half years and for granting me the opportunity me to work on several challenging international research projects over the course of this period.

To my supervisors, Prof. Isabel Praça and Prof. Nuno Silva, I am also grateful for their help throughout the development of this thesis. I strongly appreciate all the meetings we had at both GECAD and B419. Trust me, they were not in vain.

Furthermore, I must also not forget to thank Prof. Orlando Sousa, for having believed in me and marked the start of my teaching career with the Post-Graduation in Information Security, Cybersecurity and Privacy (PG-SICP) at ISEP.

And, finally, I would like to thank my family, friends and girlfriend for being my support and foundation.

ABSTRACT

As of today, the most recent trend in information technology is the employment of large-scale data analytic methods powered by Artificial Intelligence (AI), influencing the priorities of businesses and research centers all over the world. However, due to both the lack of specialized talent and the need for greater compute, less established businesses struggle to adopt such endeavors, with major technological mega-corporations such as Microsoft, Facebook and Google taking the upper hand in this uneven playing field. Therefore, in an attempt to promote the democratization of AI and increase the efficiency of data scientists, this work proposes a novel no-code/low-code AI platform: the ALgorithm Modeling Application (ALMA). Moreover, as the state of the art of such platforms is still gradually maturing, current solutions often fail into encompassing security/safety aspects directly into their process. In that respect, the solution proposed in this thesis aims not only to achieve greater development and deployment efficiency while building machine learning applications but also to build upon others by addressing the inherent pitfalls of AI through a ”secure by design” philosophy.

Key-Words: Artificial Intelligence; Machine Learning; No-code/Low-code; AutoML; Secure Artificial Intelligence

Resumo

Atualmente, a tendência mais recente no domínio das tecnologias de informação é a utilização de métodos de análise de dados baseados em Inteligência Artificial (IA), influenciando as prioridades das empresas e centros de investigação de todo o mundo. No entanto, devido à falta de talento especializado no mercado e à necessidade de obter equipamentos com maior capacidade de computação, negócios menos estabelecidos têm maiores dificuldades em realizar esse tipo de investimentos quando comparados a grandes empresas tecnológicas como a Microsoft, o Facebook e a Google. Deste modo, na tentativa de promover a democratização da IA e aumentar a eficiência dos cientistas de dados, este trabalho propõe uma nova plataforma de no-code/low-code: "THe Algorithm Modeling Application" (ALMA). Por outro lado, e visto que a maioria das soluções atuais falham em abranger aspetos de segurança relativos à IA diretamente no seu processo, a solução proposta nesta tese visa não só alcançar maior eficiência na construção de soluções baseadas em IA, mas também abordar as questões de segurança implícitas ao seu uso.

Key-Words: Inteligência Artificial; Aprendizagem Máquina; No-code/Low-code; AutoML; Inteligência Artificial Segura

Contents

List of Figures

LIST OF TABLES

LIST OF CODE

ACRONYMS

A2PM	Adaptative Perturbation Pattern Method
AI	Artificial Intelligence
ALMA	ALgorithm Modeling Application
AMR	Autonomous Mobile Robot
ATARC	Advanced Technology Academic Research Center
AutoML	Automated Machine Learning
CLEVER	Cross-Lipschitz Extreme Value for nEtwork Robustness
CNN	Convolutional Neural Network
CV	Computer Vision
DL	Deep Learning
DNN	Deep Neural Network
DT	Decision Tree
ENISA	European Union Agency for Cybersecurity
FGSM	Fast Gradient Sign Method
GLC	Global Lipschitz Constant
GPU	Graphical Processing Unit
GUI	Graphical User Interface
IDE	Integrated Development Environment

IoT	Internet of Things
ISP	Interface Segregation Principle
IT	Information Technology
JSMA	Jacobian-based Saliency Map Attack
KNIME	Konstanz Information Miner
KNN	K-Nearest Neighbors
MAE	Mean Absolute Error
MAS	Multi-Agent System
ML	Machine Learning
MSE	Mean Squared Error
NAS	Neural Architecture Search
NIST	National Institute of Standards and Technology
NLP	Natural Language Processing
PCA	Principal Component Analysis
PoC	Proof of Concept
PSO	Particle Swarm Optimization
Q&A	Question and Answering
R2	R-squared
RC	Reading Comprehension
RNN	Recurrent Neural Network
RQ	Research Question
SaaS	Software as a Service
SFH	Server From Handler
SMBO	Sequential Model-Based Optimization
SRP	Single Responsibility Principle
TL	Transfer Learning
UC	Use Case
UML	Unified Modeling Language
ViT	Visual Transformer
XAI	Explainable Artificial Intelligence
XGBoost	eXterme Gradient Boosting

CHAPTER 1

INTRODUCTION

This chapter performs a brief overview of the problem addressed in this thesis, providing both contextual information and preliminary motivations. The main research questions and objectives of the work are also presented along with the document's outline.

1.1 Problem Statement

It is well known that technological endeavors substantially impact overall society. In what regards to digitalization, many domains of social life were restructured around digital communication and media infrastructures, benefiting from enhanced efficiency and reliability [1].
Nowadays, the most recent trend is the employment of different advanced technology-embedded business analytics tools in combination with Artificial Intelligence (AI), resulting into substantial gains in decision-making, innovation and performance [2]. As a matter of fact, AI was a catalyst of the data revolution that is now being experienced, with its impact being noticeable in many levels of our society [3, 4]. Moreover, the large-scale adoption of such technology is no surprise due to mature developments in the field that took place in recent times. In particularly, Deep Learning (DL), a sub-field of AI, has experienced several breakthroughs in domains such as Nat-

ural Language Processing (NLP) [5, 6] and Computer Vision (CV) [7, 8], gathering the attention of business all over the world. More precisely, according to Forbes [2], 76% of them prioritize AI over other Information Technology (IT) initiatives [9].

On the other hand, data science is often described as the process of leveraging AI methods to grasp insights from data, being data scientists the workers who engage in technical activities such as data cleaning and algorithm modelling [10]. As ”post-Moore” (modern) AI is heavily dependent in DL, a domain that still remains unexplored to a certain extent, new endeavors typically rely on people that go through formal training (*e.g.*, PhD). This kind of talent is rather scarce and, thus, usually expensive [11]. Additionally, both computing power and quality data have been considered key ingredients for AI applications, contributing to higher levels of performance [12]. Since richer firms have deeper pockets, they are in a better position to make the necessary investments, creating an oligopoly around AI, where technological mega-corporation such as as Amazon, Apple, Google, and Tesla are in control of the main resources that impact the development of AI [13].

In an attempt to mitigate damages on a societal level, as the potential of such technology can exacerbate already existing inequalities and lead to marginalization [11], researchers and policymakers have been defending the ”democratization” of AI *i.e.*, making AI widely and easily available at a lower cost. A way of doing so, can be the development of interactive tools and dashboards to support AI development, providing intuitive, visually rich experiences to guide novice developers via graphical user interfaces (GUIs) [14]. Furthermore, developing and validating AI algorithms takes a lot of time, effort, and skill and, although the adoption of Python programming language as the *lingua franca* for AI development contributed to improve development productivity, the process is still time-consuming and abundant in fine-grained tasks such as coding and debugging [15, 16].

A possible solution for the aforementioned problems can relate to no-code/low-code AI, that allows users to build codeless Machine Learning (ML) pipelines effortlessly through simple interfaces [14]. In fact, no-code/low-code AI can not only help developers to focus on higher-value activities [17, 18], such as tackling domain-specific challenges through the high-level analysis of business requirements and extensive exploratory data analysis, but also contribute to the democratization of AI by making the art of algorithm engineering more easily accessible.

However, although these technologies seem highly attractive, their inherent pitfalls concerning algorithm’s trustworthiness in terms of robustness, transparency, ethics, non-bias and governance must be taken into account [19]. This fragility opens new avenues for improvement in current no-code/low-code AI systems by considering the adoption of auditing and trustworthiness mechanisms when designing and implementing new development platforms, adopting a ”secure by design” approach [20].

1.2 Objectives

Although the main purpose of this work is to develop of a novel solution for no-code/low-code AI, several objectives were define to guide this thesis, ranging from the state of the art analysis to design and implementation. These can be defined as follows:

- **O1**: Investigate the main factors that currently endanger the democratization of AI.
- **O2**: Clarify the meaning of the terms: no-code, low-code and AutoML (Automated Machine Learning).
- **O3**: Study the state of the art in no-code/low-code AI and related technologies.
- **O4**: Identify the main flaws of current platforms and propose a new solution.
- **O5**: Implement a Proof of Concept (PoC) and demonstrate it in real case studies.

These subjects will be subsequently addressed over the remaining chapters of this document.

1.3 Research Questions

In order to better guide the research under the scope of this thesis, three main Research Questions (RQs) were formulated:

- **RQ1**: What is the current state of AI democratization?
- **RQ2**: What is the current landscape of no-code/low-code AI?
- **RQ3**: What are the main flaws of current no-code/low-code AI platforms?

The first question, RQ1, addresses the current oligopoly around AI development, requiring further investigation on how to contradict contemporary de-democratization tendencies. On the other hand, RQ2 requires the investigation of the existing no-code/low-code AI platforms, detailing how they differ from one another and which technologies they use. Finally, RQ3 aims at identifying the flaws of modern no-code/low-code AI platforms, leading to the proposal of a new solution.

1.4 Contributions

From this thesis, several contributions can be appointed:

- **C1**: A taxonomy to define the domain of no-code/low-code AI.
- **C2**: A survey about the state of the art of no-code/low-code AI.
- **C3**: A survey about the state of the art of AutoML.
- **C4**: The definition of high-level requirements to guide the development of novel no-code/low-code AI platforms in a way that mitigates the flaws of existing solutions.
- **C5**: A functional PoC that was demonstrated in two real case studies.

1.5 Outline

This thesis is organized into multiple chapters that can be described as follows:

- Chapter 1 provides an overview of the social and technological aspects that motivated this work as well as a brief description of this its objectives, the research questions to be addressed and the overall contributions of this work.
- Chapter 2 describes in greater detail the main motivation behind this thesis, namely the large-scale adoption of AI, the need for AI democratization and the security/safety issues intrinsic to AI use.
- Chapter 3 provides an overview of the overall ML process as well as a comparison between existing no-code/low-code AI applications. Moreover, AutoML frameworks are also presented and compared since it is strongly related to no-code/low-code AI.
- Chapter 4 describes the conceptualization behind the proposed solution along with the description of the high-level design decisions made while developing the first version of the ALMA platform.
- Chapter 5 showcases the usage of ALMA's PoC in two practical use cases, performing the benchmark of different ML algorithms for both classification and regression tasks.
- Chapter 6 provides a summary of the main findings of this work, appointing research lines to be explored in the future.

CHAPTER 2

MOTIVATIONS

The rapid, yet turbulent, advances in AI over the years contributed to a great shift in the technological strategies of many business and research centers word-wide [3]. However, the large-scale adoption of AI in recent years did not come without some challenges to be addressed, such as democratization, auditing, trustworthiness, ethics and non-bias [19].
This chapter does not make an attempt to point out the inherent problems of AI use, much less does it attempt to provide an utopic viewpoint on the future of AI. Instead, this chapter's goal is to provide the main motivations behind this thesis *i.e.*, the key factors that make the work developed in the context of this thesis relevant and up-to-date. Therefore, the following motivations will take the shape of subsections for further understanding:

1. **Large-Scale Adoption**: Attempts to answer questions such as *"How do we stand now in terms of the usage of AI?"* or *"What is its relevance to the business/academic world?"*.

2. **Democratization**: Addresses the inherent oligopoly created around the usage of AI and the need to make this technology widely available at a lower cost.

3. **Auditing and Trustworthiness**: Questions the secure use of AI and how concepts such as robustness, transparency, ethics, non-bias, and governance are commonly disregarded.

2.1 Introduction to Artificial Intelligence

Over the course of the years, many great philosophers, such as Plato and Aristotle in the classical years or Descartes, Hume and Kant, in the age of enlightenment, have studied the origins of intellect and knowledge, central pieces to the understanding of the human mind [21]. However, and despite being so thoroughly studied, theses terms are still used vaguely and without well-defined boundaries. Nevertheless, it is safe to say that ”Intelligence” is an anthropocentric concept in nature, being often employed to describe human intellectual capabilities [22].

Differently, and according to the Britannica Encyclopedia, ”Artificial Intelligence” is ”the ability of a digital computer to perform tasks usually associated with sentient beings, such as expressing human-like reasoning or learning from past experience”. Moreover, AI is considered as part of computer science, an epistemological domain that studies computers and computing, including their underlying theoretical and algorithmic foundations.

AI encompasses several sub-fields, that are often mentioned interchangeably and not mutually exclusive (*e.g.*, ML and NLP). Some of them can be described as follows:

- **Computer Vision**: CV focuses in the processing of visually rich data such as images or videos. Some of the main tasks under the domain of CV are object detection, facial recognition, action/activity recognition and human pose estimation [23].

- **Expert Systems**: Expert Systems use a knowledge-based approach, where domain information, provided by an expert in the field, is used by a knowledge engineer to populate a knowledge base. The content of such knowledge base is then used by an inference engine to derive meaningful conclusions [24].

- **Machine Learning**: ML emerged as a disruptive sub-field of AI, introducing a new paradigm for designing intelligent systems. With ML, algorithms can discover predictive rules from patterns in labeled data on their own, without being explicitly programmed for a given task. On the other hand, DL is arguably the most promising branch of ML, introducing new algorithms that perform a mimic of the way the human brain works and is structured [11, 16, 25].

- **Multi-Agent Systems (MAS)**: A MAS is a type of distributed AI composed of several autonomous entities designated as agents. Agents are able to perceive their surrounding environment and work collaboratively with each other to interact with it [26].

- **Natural Language Processing**: NLP uses computational techniques to learn, understand and produce contents in human language with respect to several levels of linguistic analysis (*e.g.*, lexical, syntactic and semantic) [27].

- **Planning/Optimization**: Planning/Optimization aims for the maximization or minimization of a given value obtained through a function of the problem's variables and constraints. There are several optimization methods, with some of them resorting to analytic methods and others to heuristics and meta-heuristics [28, 29].

- **Robotics**: The domain of Robotics is usually associated to physical machines with certain degrees of autonomy. These are able to adapt to their ever-changing environments through continuous loops of actions such as perceiving, planning and executing, many times resorting to intelligent mechanisms [30].

- **Speech Recognition**: The Speech Recognition domain encompasses methods for automatic speech processing. These methods can be used to provide better ways of interfacing with computers by modeling speech signals into continuous series of words [31].

The interconnectivity of some of these branches is evidenced by several works in the literature. More precisely, N. Sousa *et al.* in [26], applied a MAS for coordinating several Autonomous Mobile Roots (AMRs) in tasks such as transporting and dispatching raw materials, finished products and tools in the manufacturing ecosystem, merging the branches of Multi-Agent Systems and Robotics. Similarly, N. Oliveira *et al.* in [32], applied RoBERTa [33], a DL algorithm for natural language comprehension in the context of scientific search engines, combining the branch of NLP with DL, a sub-field of Machine Learning. The same happens with Planning/Optimization and Machine Learning in [34], where N. Oliveira *et al.* applied a genetic algorithm to optimize the hyperparameters of a Convolutional Neural Network (CNN) architecture.

2.2 Large-Scale Adoption

The work discussed in this thesis aims to ease the access to AI algorithms without presenting high technical barriers and while tackling auditing and trustworthiness issues that typically remain unaddressed. One of the main reasons that justify the need for these developments is the large-scale adoption of AI by both industry and academia [3, 35].

Nonetheless, firstly, after diving deep into the present and future of AI it is important to understand its historical context. In truth, history provides a better understanding of the present and can be key to decode what follows next, due to it's cyclic nature.

2.2.1 Historical Overview

In spite of modern technological trends, from its emergence until now, AI was not always a strategic priority in terms of funding and support. In fact, since 1950, when Alan Turing first questioned the concept of machine intelligence in its *magnum opus*, ”Computing Machinery and Intelligence” [36], AI endured two main periods of major funding drought, typically designated as AI Winters [37].

After the first conference on AI, the Dartmouth Summer Research Project on Artificial Intelligence (1956), where the ownership of the term ”Artificial Intelligence” was officially attributed to John McCarty and the first AI program [1] presented [39], AI development prospered, with several landmarks being developed such as the perceptron learning algorithm [40], foundation element of modern-day Deep Neural Network (DNN) architectures and ELIZA [41], the first questioning and answering (Q&A) system able to attempt the Turing Test.

However, in mid 70s, AI utility for real-world applications was questioned by several private and public institutions with some scientific publications, such as those of James Lighthill [42] and Richard Karp [43], reinforcing such doubts. The first, James Lighthill, published a report commissioned by the British Science Research Council stating that common-sense reasoning would always be beyond the understanding of machine intelligence and that an AI, in the context of strategic games, such as chess, would only be able to reach the level of an ”experienced amateur”. On the other hand, the second, Richard Karp, addressed the problem of combinatorial explosion, where the computing time necessary to solve a given problem exponentially increases as a function of the input size, thus proving that with the available hardware at the time, contemporary AI solutions could not be scaled up into useful real-life applications. This chain of events triggered the start of the first AI Winter which last until early 80s [44].

The revival of AI technology was catalysed by a new robust approach with roots in the Carnegie Mellon University: expert systems. With such approach, researchers decided to narrow down machine intelligence to specific areas of expertise in order to utilize domain-specific knowledge for stronger reasoning. A canonical example of such a system is DENDRAL [45], created at Stanford University to infer molecular structures from mass spectrometry data.

Nonetheless, expert systems also present serious drawbacks such as ineffectiveness in areas that do not lend themselves to specific formalization (*e.g.* object recognition) and laborious/expensive knowledge maintenance [44]. These limitations allied to the unwillingness of hardware manu-

[1]The Logic Theorist was the first program designed to mimic human problem-solving abilities in order to demonstrate propositional calculus theorems. It was able to demonstrate 38 out of the 52 of the theorems presented in the ”Principia Mathematica”, a three-volume work on the foundations of mathematics [38].

factures to keep up with the requirements of expert system's specialized needs, stroke another blow at the AI industry, originating the second, and arguably the last, of AI Winters [46].

In early 90s, new hopes for AI shaped again with the introduction of reinforcement learning and the re-adoption of sub-fields such as MAS, Robotics, NLP and CV, fuelled by large availability of data and significant advances in ML [44]. This resurgence of AI became clear in 1997 when, ”Deep Blue”, a chess-playing AI developed by IBM [47], defeated the world champion, Garry Kasparov, in a chess match with live broadcast on public television. Later on, in 2015, AI dominance in the context of strategic games became undeniable with ”AlphaGo” [48], by DeepMind, beating the world champion of Go, a substantially more complex game than chess [37].

Recently, DeepMind published two works regarding consecutive iterations of ”AlphaGo”, namely ”AlphaZero” [49] in 2017 and ”MuZero” [50] in 2020, with the objective of generalizing the strategic reasoning of their predecessor for other games. The latter, ”MuZero”, achieved new state of the art results when evaluated on 57 different Atari games as well as superhuman performance in the likes of chess, Shogi and Go [50].

These achievements in the context of game theory are but a representative fraction of the most recent advances in modern-day AI. Thanks to the parallel processing convergence, higher memory capacity and massive data collection resulting from the big data revolution [51], AI has experienced a steady upward climb since the early 2020s. This is supported by several facts such as the growth of global investment in AI-based startup companies, the increase in the number of peer-reviewed AI articles per year and the significant rise in the number of attendees to AI conferences world-wide [44].

2.2.2 Impact of Artificial Intelligence

As of today, the pervasiveness of AI is undeniable, with large-scale data collection and analytics fuelled by intelligent algorithms impacting all levels of our society [4, 52]. This becomes even clearer when performing a deeper analysis of the following aspects: (i) latest innovations at industrial level; (ii) state of the art of AI algorithms in many of its sub-fields such as game theory, computer vision and natural language processing; (iii) socioeconomic indicators.

2.2.2.1 Industry Advances

According to Müller *et al.* in [53], big data analytics has transformed the way business compete. It allows the extraction of hidden patterns from raw data, thus uncovering useful information that is key to improve decision making, enhance productivity and generate new knowledge [51]. In practice, industries with good data foundation such as finance, healthcare, automotive and media

took great benefit of the most mature developments of AI [52].
One of the biggest breakthroughs in the automotive industry is autonomous driving, a product resulting from the integration of new generational information technologies such as Internet of Things (IoT) and AI. By combining several information related to pedestrian activity and road conditions (collected through multiple sensors) with advanced algorithms that are continuously fed with such data, the vehicle's routes and control plans are optimized automatically with almost no need for human interaction [54]. On the other hand, financial marketing has also benefit from AI in tasks such as intelligent risk control, intelligent consulting and market forecasting. As an example, financial institutions employ ML methods to manage financial risks, integrating multiple data sources to provide real-time risk warning [52]. Similarly, the healthcare industry is not indifferent to the advent of AI. As healthcare systems all across the world face huge problems such as lack of access, high cost, waste and older population, AI is appointed as both a critical enabler of healthcare simplification and a bedrock for the development of intelligent care systems. In this context, recent developments show promising results into multiple sectors such as drug discovery, clinical trials and patient care [55]. Furthermore, in the media industry, intelligent social media platforms (*e.g.*, Facebook) combine contemporary events, public opinion and personal profiles to study media delivery and delivery rules. These are then used to suggest/generate content that users are likely to read as well as to optimize the dissemination of advertisements [52, 56]. For the social sciences point of view, these platforms also end up as serving as some sort of virtual laboratories for conducting social and psychological experiments (through data analytics) to acquire novel insights on human behaviour [3].
Fundamentally, AI systems have taken place in a wide variety of industries with many other works appointing its use in domains such as smart homes [57], to monitor and control home activities for convenience, energy, for energy consumption forecasting [58] and electricity market negotiation [59], and cybersecurity, for enhanced real-time threat detection in intrusion detection systems [60] and cyber-physical alert correlation [61].

2.2.2.2 Academic Advances

The aforementioned industrial advances were only possible due to recent mature developments in AI sub-fields such as ML. The "State of AI Report", published by Benaich *et al.*, points out the most significant achievements of 2021 in several tasks such as object detection, audio recognition, structural biology, game-play, image generation and code generation. Some of them can be described as follows [62]:

- **ViT**: The introduction of the "Transformer" architecture in [63], motivated several innovations in the context of NLP such as BERT [5], RoBERTa [33] and GPT-3 [6]. This approach disregards recurrence and convolutions from the usual encoder-decoder models and, instead, it uses several types of attention mechanisms, leading to a new state of the art in the field of Reading Comprehension (RC) [32]. In 2021, Google applied this same concept to the field of CV, achieving a top-1 accuracy (90.45%) on ImageNet [64], a widely accepted benchmark for image classification. This new algorithm, ViT (Visual Transformer) [7], despite being later dethroned by CoAtNet (90.88%) [8], which uses both convolution and attention layers, successfully marked the adoption of this architecture in the context of CV, overcoming other well-establish algorithms.

- **MuZero**: In the domain of game theory, Deep Mind has launched MuZero [50], matching the performance of its predecessor, AlphaZero [49], in the games of Go, chess and Shogi, and outperforming all existing models on the Atari benchmark while learning solely within a world model. This benchmark, Atari, comprises a suit of visually complex games which have been beyond the reach of model-based systems. However, MuZero is able to model only what is relevant for its decision making (from the whole scope of the game dynamics), allowing it to scale well for complex games [62].

- **DALL-E**: For the task of generating images from natural language prompts, OpenAI introduced a novel algorithm, DALL-E [65], which is in fact, a 12 billion parameter version of the GPT-3 model that was trained on text-image pairs. The authors illustrated the algorithm's potential by exhibiting a series of interactive visuals generated from a great variety of sentences that explore the compositional structure of language. CLIP [66] was used to select the best images in order to avoid manual cherry-picking that could potentially induce bias into the demonstration.

- **Codex**: OpenAI also applied GPT-3 to the domain of code generation through it's specialised offspring, Codex [67]. The algorithm was fine-tuned on publicly available code from GitHub and evaluated on HumanEval, a new evaluation set (also of OpenAI's authorship) to measure functional correctness for synthesizing programs from docstrings. It was proven that although showing promising results, Codex still lacks the ability to reason about docstrings that describe long chains of operations and binding operations to variables. Nevertheless, after breaking down a complex problem into more contained tasks, a developer can utilize Codex to map these tasks into existing code (libraries, APIs, or functions) automatically [62].

2.2.2.3 Socioeconomic Indicators

The AI Index Report 2021 [35], published by Zhang *et al.*, aims to provide an unbiased and rigorously vetted overview of the state of AI in the world. The report is organized into multiple chapters with attention to several viewpoints in many domains such as the economy, education, ethics and diversity. For the scope of this chapter, the economy should be the main focus, with emphasis on AI hiring, labour demand and corporate investments.

Regarding AI hiring, data collected from the LinkedIn platform suggest that the hiring rate has been increasing across all sample countries in 2020, being Brazil, India, Canada, Singapore and South Africa the countries with the most prominent growth in AI hiring from 2016 to 2020.

On the other hand, with respect to the AI labour demand, calculated through Burning Glass, an analytics firm that collects postings from over 45,000 online job sites of United States, United Kingdom, Canada, Australia, New Zealand, and Singapore, it is possible to conclude that the share of AI job postings among all job postings in 2020 is more than five times larger than in 2013. The study also pointed out that the United States is the only country who experienced a decrease in AI job postings from 2019 to 2020 (first drop in six years), with the COVID-19 pandemic and the country's considerably mature AI labor market appointed as possible reasons for such drop.

Figure 2.1 presents the percentage of all job postings by country.

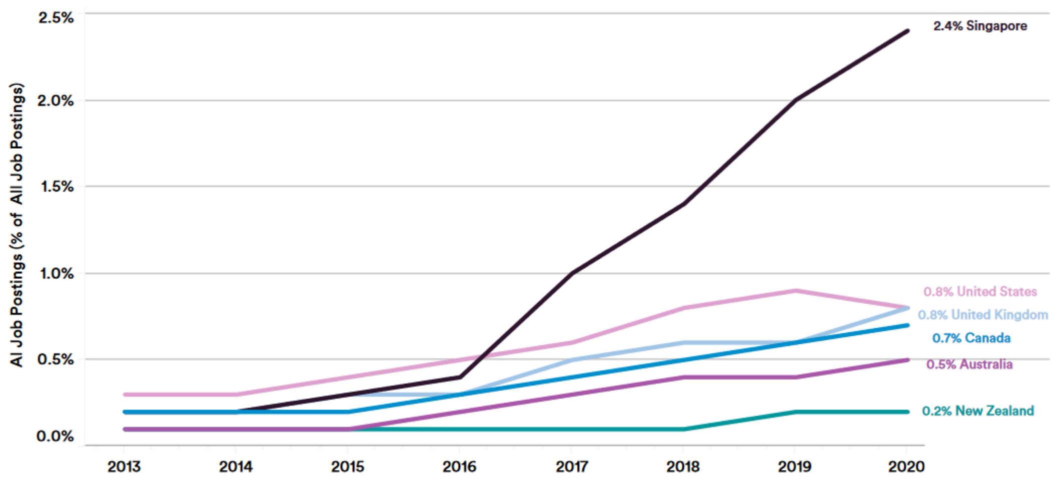


Figure 2.1: AI job postings by country [35].

A similar growing trend is visible in the context of global AI investments, where all private investment, public offerings, merger/acquisition and minority stakes, increased by 40% in 2020 when compared to 2019, representing a sum of 67.9 billion U.S. dollars. Moreover, several high-profile acquisitions took place in 2020, such as the NVIDIA's acquisition of Mellanox Technologies and

Capgemini's of Altran Technologies.

Figure 2.2 provides an overview of all global corporate investment in AI.

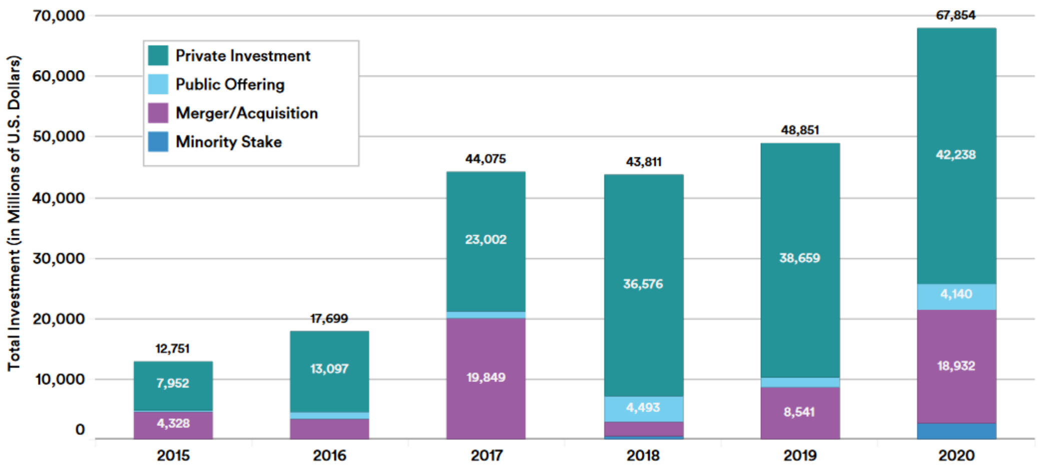


Figure 2.2: Global corporate investment in AI [35].

When further inspecting the topic of private investment in AI, it is well noticeable that the United States have clear dominance over China and the European Union. Nevertheless, these numbers can be misleading as China has strong public investments in AI by both the central and local governments. Figure 2.3 presents the total amount of private investment in AI by geographic area (expressed in U.S. dollars).

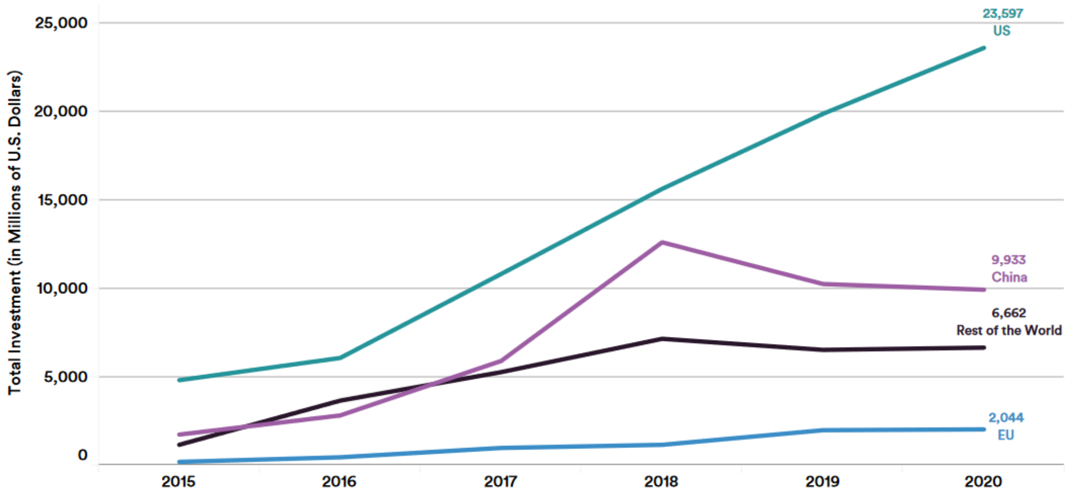


Figure 2.3: Private investment in AI by geographic area [35].

2.3 Democratization

In the early days, specialist engineers had to be hired to operate electric generators *in situ* at homes just to use simple light bulbs. As much as this idea may seem perplexing as of today, it was due to great minds, mature business opportunities and much patience that it was possible to convert electricity into a mere utility which is now often taken for granted [68]. On the other hand, when we turn to AI, some may argue that we are on the verge of taking full utilitarian value out of it as the field seems to be continuously maturing over the course of the years. Nowadays, its impact is already well noticeable on both economy and labor landscape, and as of such, AI and ML, are appointed as an integral portion of the Fourth Industrial Revolution [69].

However, while the world's population is currently over 7 billion people, only about 10 thousand people are working in the code for all of AI [70]. In fact, the vast majority of AI development comes from a few technology mega-corporations such as Microsoft, Google and Amazon. As the concentration of power can lead to marginalization and severe inequalities [68], this oligopoly of centralized mega-corporations is in position to shape the developments of AI to suit the interests of their own stakeholders. On the flip side of the coin, companies from all over the world that lack proper capital struggle to develop their own AI services, creating an uneven playing field which favours inequality and can lead to negative implications for humanity in general [13].

In order to ensure that the benefits of AI are not limited to a small group of people, thus avoiding the exacerbation of already existing social inequalities, regulatory governments and agencies across the globe are formulating national policies and laws around these technologies to simultaneously protect and empower national citizens [11, 71]. Moreover, concepts such as ”Inclusive AI” and ”AI for Social Good” have been recently advocated by researchers and policymakers, evidencing a growing consensus towards the ”democratization” of AI [14, 68].

According to the Britannica Encyclopedia, the epistemological origins of the word ”democracy” remotes to its Greek counterpart, *dēmokratia*. Coined after *dēmos* (people) and *kratos* (rule) it stands for ”rule by the people” or ”government by the people” which is dissimilar to oligarchy, the ”government by the few” (*oligos*). Therefore, and being democratization the act of making something democratic, the democratization of AI aims to lower the entry barriers for the ”world of AI” in terms of both resources and knowledge. On the academic level, democratizing AI translates into having more researchers working in the filed, which can lead to new research challenges (and opportunities), that can, consequentially, inspire more frequent breakthroughs. On the other hand, at the industry side, it means the production of more value, materialized in the form of new products and services, as well as greater market competition [72].

Per contra, this viewpoint may seem rather utopic when facing fashionable evidences of both unequal/unfair access to computing power (compute divide) and de-democratization. In [11], Ahmed *et al.* argue that private technological corporations such as Amazon, Apple, Google, and Tesla have too great of an advantage over the main resources that influence the development of post-Moore AI: human capital, computing power and data.

2.3.1 Human Capital

As modern AI depends heavily in DL, a domain not yet fully understood, innovation often relies on people that undergo formal training such as PhD and/or many years of work experience [11]. Due to both the mist around DL and the disparity in terms of demand/supply for AI-related talent, deep scarcity is noticeable in the field. This demand for qualified people makes human capital to be rather expensive and, as of such, richer companies take the upper hand when it comes to provide higher remunerations [73]. Furthermore, recent reports concerningly point out to the large-scale recruitment of faculty members of North American universities by large technological companies as well as the acquisition of dozens of startups with the primary purpose of talent acquisition [74, 75].

2.3.2 Computing Power and Data

Computing power has been considered an important ingredient for current AI since it was demonstrated that increased compute is complementary to DL algorithms and typically leads to increased performance [12, 76]. Nevertheless, to obtain such level of compute, investments have to be made so that appropriate infrastructure such as GPUs (Graphical Processing Units) can be acquired. Hence, similarly to what happens in the context of human capital, richer firms have greater margin to make this sort of investments [11]. Moreover, these same mega-corporations have been developing their own specialized hardware as well as producing software thoroughly optimized for it, taking them even further ahead than the competition [77]. Finally, to add up to the technocracy, it has been proven by recent studies that Facebook, Google, Amazon and other technology giants have an advantage in AI research due to their own proprietary data. This data, allows them to produce high-quality datasets, contributing to more accurate algorithms [78, 79].

2.4 Secure Artificial Intelligence

As society in general increasingly relies in digital media, cybersecurity plays a major role in safeguarding sensitive user and corporate information that is constantly shared over network infrastructures [25]. For this purpose, AI, in particular ML, has been applied successfully with many works, such as [24], [25], [34], [60] and [61], demonstrating the potential of such applications. However, it was only until recently that the secure use of AI has started to be questioned by the scientific community [80, 81]. In fact, it was proven that ML algorithms can be solely just as vulnerable as the same systems they were designed to protect in the first place, with several exploits being identified in both core phases of the learning process, training and inference [80]. In the first phase, carefully manipulated data can be inserted in the training dataset to intentionally sabotage the algorithm's training, as it is the case of poisoning attacks [82]. On the other hand, in the second phase, attackers can perform evasion attacks to manipulate test samples, deceiving the model into performing wrong predictions [83].

Thus, despite the adoption of AI presenting an unmatched opportunity for socioeconomic growth, its potential remains unfulfilled without proper methods for securing (and regulating) these sort of technologies [81]. For this reason, several works have recently being published by the community, addressing a plethora of subjects around the secure use of AI [84, 85, 86, 87, 88]. While some are focused in exploring both attack and defence mechanisms for ML algorithms directly [84], others aim to provide an holistic approach for AI trustworthiness assessment [85]. Moreover, as some are more technological [86], discussing mathematical and algorithmic explainability methods, others are more deontological [87], raising questions about the ethical grounds/fundamentals of AI.

In the context of no-code/low-code AI, where less technically educated audiences are empowered to build and train AI algorithms on their own [14], these kind of security issues take an even higher degree of importance, as the use of such platforms can either contribute to enhanced security by implementing strong measures or lead to insecure intelligent systems through negligent/unprotected use [19]. Hence, for the scope of this work, it is important to understand the underling security/safety aspects of AI. Furthermore, and as the domain is broad, the most relevant aspects were selected from the literature, shaping the remaining of this section.

2.4.1 Trustworthiness

Stanton *et al.*, from the National Institute of Standards and Technology (NIST), in a freshly published report [85], points out that common software customers, although having unclear understanding about the usage of AI, happen to deem trust in such systems at a certain extent. Furthermore, and although other works have tried to address the topic of trust in a system-centric way by formulating a series of requirements, trust is appointed to be, fundamentally, a psychological trait of the user who uses the system.

In the same line of thoughts, the authors indicate that, ever since our evolutionary beginnings, trust and distrust have been used as mechanisms to mange the risks of social interaction. Additionally, although the reliance on another individual can bring many advantages, it simultaneously opens new avenues for exploitation and deceit, with cognition taking a fundamental part for making such judgements. The role of cognition is, therefore, not to be disregarded when aiming for AI trustworthiness. As a matter of fact, while from the engineering point of view, a system is deem trusted if it meets certain technical characteristics such as Security, Explainability, Accountability and Privacy, it is, ultimately, the perception of the available technical information that shapes the user's trust in the system, contributing for better human-AI collaboration.

2.4.2 Robustness

Szegedy *et al.*, in [89], first found that DNNs can be exploited by means of thoroughly crafted samples, misleading algorithms into making wrong predictions in the inference phase. This venture has broken new ground in AI security, motivating the research of attack methods such as Fast Gradient Sign Method (FGSM) [90] and Jacobian-based Saliency Map Attack (JSMA) [91]. As a result, and given the existence of adversarial attacks, the concept of adversarial robustness has started to be adopted as a measurement of a DNN's resilience against such attacks [84]. Furthermore, novel developments have also started to take place in an attempt to mitigate these type of threats. Either by proposing new defense mechanisms against adversarial attacks, which is the case of adversarial training [92, 93], or by introducing more accurate ways of measuring robustness, as the Global Lipschitz Constant (GLC) or CLEVER-score (Cross-Lipschitz Extreme Value for nEtwork Robustness).

2.4.3 Transparency and Explainability

As stated by R. Schmelzer in [88], the landscape of the AI market is gradually shifting from model building to model consuming, by means of Software as a Service (SaaS). Hence, due to this increased dependency on external sources, questions are starting to be placed on whether consumers should blindly trust the algorithms they are provided with, as there is zero to none visibility on what is happening behind the scenes. Furthermore, in the case of ML, these kind of issues are particularly sensitive, as algorithms are not code *per se* and depend on a series of iterations (and approximations) to become more accurate. So, in that sense, users don't have much control over the process of model rebuilding and, even more concerningly, fail to comprehend on why the model is performing poorly due to the lack of transparency.

With the objective of assessing ML model transparency, the Advanced Technology Academic Research Center (ATARC) has produced a document [94] pointing out five factors of transparency:

- **Algorithm Explainability**: Addressing the need to understand how a given algorithm has reached a certain conclusion. Although some algorithms are naturally explainable, such as Decision Trees (DTs), others behave as complete black boxes, as it is the case with DNNs. Nevertheless, although a lot of research has been made in order to improve the explainability of such methods [86, 95], the field of XAI (Explainable Artificial Intelligence) is not mature enough for a widespread adoption to be witnessed.
- **Dataset Bias**: In this context, dataset bias does not imply poor quality data, but, instead, it refers to lack of representativeness in the training set, causing algorithms to make decisions based on preconceived notions [2]. This subject is explored further in the next section.
- **Data Sources**: This factor is mainly concerned with the origin of the data used to train the algorithm: where did it came from or how was it cleaned.
- **Data Selection**: While there can be a huge amount of data available, engineers often apply selection methods before training ML algorithms. Thus, this dimension attempts to comprehend what transformations have been made: if only a portion of data was used, what were the features selected for the training procedure or if any data augmentation method was employed.
- **Model Versioning**: As ML models should be iterated on for improved performance, full transparency should also be obtained over versioning operations. Not always freshly trained algorithms perform better than their predecessors, so, it is important to have control over spontaneous model versioning. For example, having the ability to select older model versions if a more recent one exhibits poorer performance.

The final result of the proposed framework is a radar chart where each component can take a score ranging from 1 to 5 (*e.g.*, scoring ”1” in explainability means that the algorithms solely behaves as a black box).

2.4.4 Fairness and Bias

The wide spread adoption of big data analytics has made several organizations argue about the potential of computational algorithms to introduce informational bias and discrimination in automated decisions, appointing the way that such digital systems are structured and used as one of the main reasons [97]. In fact, studies such as those presented by Bolukbasi *et al.* [98] and

[2]Cathy O'Neil, in the book ”Weapons of Math Destruction” [96], popularised the idea of mathematical tools fostering bias against certain groups of people. These algorithms are said to be opaque and unregulated, taking the potential to amplify already existing social inequalities through computational scalability.

Koenecke *et al.* [99], have shown empirical examples of such assumptions. The first, discovered that one well accepted algorithm in NLP, ”word2vec”, inadvertently encodes social biases such as gender stereotypes, while, the second, found that the automated speech recognition systems of major tech companies (Amazon, Apple, Google and Microsoft) have a higher error rate for Afro-American speakers when compared to white speakers. Moreover, the lack of inclusive training data is said to take a part into the performance gap between racial groups [11].

The White House has advocated, in an official report, for ”equal opportunity by design” as a guiding principle for domains such as credit scoring. However, and according to Hardt *et al.* in [97], ”a vetted methodology for avoiding discrimination against protected attributes in machine learning is lacking”, with the authors, in that same work, introducing a novel solution based on the principle of ”oblivious”. Furthermore, both the naive approach of ”fairness through unawareness” *i.e*, ignoring all protected attributes (race, color, gender), and the more thoughtful approach of demographic parity were systematically criticised.

2.4.5 Accountability

According to Busuioc in [100], accountable AI is a complex subject, since AI-related challenges strike at the very own heart of accountability procedures. As at its core, accountability is about ”answerability”, it becomes hard to judge and interrogate algorithmic outcomes when they usually present themselves as opaque or fail to exhibit substantial evidences of the decision-making process. In this respect, the author points out to transparency as a fundamental but insufficient condition for accountability. Not only should the industry adopt such good practices, but there is also a need for policy makers and regulators to enforce them. Moreover, as a practical scenario, the employment of black-box models in the public sector is questioned, especially in domains such as criminal justice where decisions with high individual stakes can be endangered by proprietary or uninterpretable algorithms. Ultimately, there is an urgency for regulatory efforts, as they are indispensable to ensure that AI comes-forward without striking a blow at our very own core institutions: justice, law enforcement and education.

2.4.6 Privacy

AI works on the basis of data and while it has already been proven that large amounts of data usually contributes to better algorithm performance [12], there is, simultaneously, an urgent need to prevent the right of individual privacy from eroding. According to [101], in current times, gathering personal data has become dangerously easier, with major companies like Facebook and Google running data-centric businesses. These same companies collect, on a daily basis, huge amounts of user information to exploit psychological traits for their own benefit (enhanced advertisement, marketing and sales). Additionally, as there are people who voluntarily share personal information on social media platforms, there are others who struggle to conceal delicate subjects, since there is no alternative way of searching for information online without introducing a series of terms into someone else's search box. In that same sense, the introduction of those so-called "digital assistants" such as Amazon's Alexa and Google Home adds up to "persistent surveillance", as the continuously collected data can either serve the purpose of the product or just as easily be used to observe individuals in ways that are unknown to them.

2.5 Summary

As the general attention to AI drastically fluctuated over the years, constantly dangling back and forth to serious funding droughts, this technology became, as of today, a priority for both modern-day companies and research centers [3, 44, 52]. This shift of paradigm was mainly caused by the most recent unprecedented advances in its several sub-fields, with high emphasis on DL [62].
However, the adoption of AI requires great costs for businesses such as the acquisition of specialized talent (which there is shortage) and powerful hardware for data storage and preprocessing. Therefore, an oligopoly is created around AI, where the richer and most established corporations continue to improve themselves with the most recent algorithms while the less established ones struggle to make profit out of such technology [11]. A way to contradict this oligopoly is the democratization of AI *i.e.,* making AI widely available at lower cost [13, 68].
Nevertheless, the democratization of AI, in terms of providing easier access to complex algorithms with lower technical barriers, also comes with its own pitfalls. With aspects such as robustness, transparency, ethics, non-bias and governance [19] to be addressed, there is demand for a solution that can not only deliver this kind of simplified access to the technology but that can also take into account the inherent security/safety issues. This is the gap that ALMA platform expects to fill.

CHAPTER 3

MACHINE LEARNING

This chapter provides a quick overview of ML and all phases of its lifecycle. Additionally, a common taxonomy regarding low-code AI, no-code AI and AutoML, that will be used consistently over the course of this thesis, is presented. Finally, an overview of the state of the art in the domains of both no-code/low-code AI and AutoML is performed along with a comparison of the most relevant applications/methods.

3.1 Overview

ML is a disruptive subdomain of AI that comprises three main learning paradigms: supervised learning, unsupervised learning and reinforcement learning [102]. In supervised learning, algorithms obverse many examples of input-output pairs and learn a function that maps inputs to outputs as intended. Supervised Learning is mainly used for two main tasks, classification and regression. In the first, the output variable, or target, is one of a finite set of values (*e.g.*, sunny, cloudy or rainy) while in the second, the output is expected to be a continuous numerical variable, such as temperature or atmospheric pressure. Differently, in unsupervised learning, algorithms learn patterns from data without explicit feedback. This paradigm is usefull for other types of tasks such as clustering, identifying groups of similar data points, or dimensional reduction, used to reduce the size of a given dataset's feature space. On the other hand, in reinforcement learning, algorithms learn through a series of rewards and/or punishments depending on its decisions (and their consequences) within a given environment. For each of the introduced paradigms, there are different possible algorithms such as Decision Trees and eXtreme Gradient Boosting (XGBoost) for the supervised, K-means clustering and Principal Component Analysis (PCA) for the unsupervised, and finally, Deep-Q learning for the reinforcement learning paradigm. Furthermore, advances in other paradigms such as semi-supervised learning, have contributed to solve problems where a few expensive labeled samples are available and abundant unlabeled samples are effortlessly obtained [103].

3.2 Lifecyle

As the domain of AI is broad, and, therefore, requires a structured and methodical approach to understand its different facets, the European Union Agency for Cybersecurity (ENISA) proposed a generic reference model for a functional overview of typical AI systems [104]. Nevertheless, due to the vast range of intricacies (technologies, techniques and algorithms) involved in these systems, mapping their entirety in a single AI lifecycle model is arguably too ambitious. For this reason, the reference model proposed by ENISA, Figure 3.1, is geared towards ML, as the particularities of the many sub-fields of AI demand for the generation of specific target models and ML has been spearheading the explosion of AI in the last ten years.

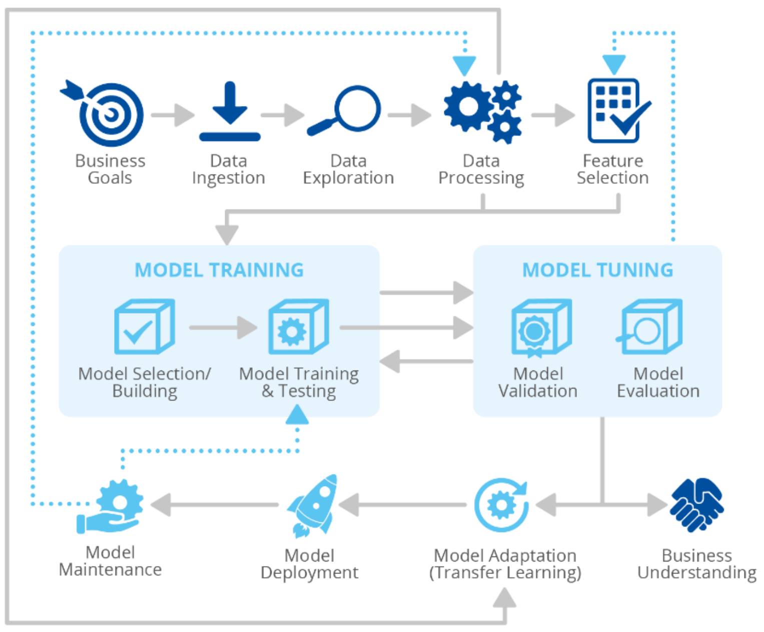


Figure 3.1: AI lifecycle generic reference model [104].

The lifecycle of an AI system foresees several interconnected phases that address its design, development, installation, deployment, operation, maintenance and disposal. These should be followed by any organization that intends to benefit from the employment of AI techniques (with emphasis in ML algorithms). All phases presented in Figure 3.1, can be briefly described as follows [104]:

- **Business Goal Definition**: Before deciding to carry on with the development of an AI system, one should first fully understand the business context of its application as well as the required data. Furthermore, appropriate metrics must be defined to determine to which the degree the business goals have been achieved.

- **Data Ingestion**: In the data ingestion phase, data is obtained from multiple heterogeneous sources for immediate use or storage. Furthermore, and based on the AI system specification, data can be ingested in real-time (streaming) or periodically, in batches, with the possibility of requiring annotations before being further processed by algorithms.

- **Data Exploration**: In the data exploration phase, insights are gathered from collected data, identifying variable types and multimedia data such as images, audio or video. Additionally, several plots are usually produced along with descriptive statistic measures to verify if data fits simple parametric distributions (*e.g.*, Gaussian).

- **Data Preprocessing**: Raw ingested data, typically, can not be used directly to train AI algorithms, it first requires a series of preprocessing steps to cleanse, integrate and transform the data. This phase aims to improve data quality, contributing to better performance and efficiency of the AI system. Some preprocessing steps can be the conversion of categorical data to equivalent numerical representation and the handling of missing/null values through interpolation and augmentation.

- **Feature Selection**: In the feature selection phase, the less significant features of a dataset are discarded or condensed to achieve denser representations. These denser representations, use the features which are believed to be most meaningful to be processed by the AI algorithms, contributing to the reduction of the overall computational cost and more accurate models.

- **Model Selection / Building**: At this stage, candidate AI algorithms are selected to be trained, evaluated and compared against each others. This is a difficult task, often subjected to trial and error, since there is a whole plethora of possible algorithms, within several paradigms, to be considered, requiring deep knowledge of the different techniques to make the right decisions.

- **Model Training**: After selecting the best candidate algorithms, these must be trained and benchmarked to determine how well they can fulfill the business requirements. In the context of supervised ML, in the training phase algorithms are fed with batches of input vectors, using a given learning function to adjust their internal parameters (*e.g.*, weights and bias) based on a measure of the difference between the output of the model and the ground truth. On the other hand, in the inference phase, a portion of the dataset saved for testing is used to make predictions and compute multiple evaluation metrics (*e.g.*, accuracy and precision).

- **Model Tuning**: Although model tuning strongly overlaps with model training, ENISA opted to separate the two stages to highlight the specificities in terms of functional operations. Algorithms usually have high-level settings that can not be learned by the input data. These hyperparameters have to be manually set up and are typically tuned to obtain enhanced performance. For this process, many optimization methods can be used (*e.g.*, Random Search and Grid Search) to compare the performance of multiple hyperparameter combinations, making use of a validation set or k-fold cross validation for an unbiased evaluation.

- **Transfer Learning**: In this phase, a pre-trained AI model is used as a starting point for further training. Transfer learning can be a good option when there is few data available for training or when the task to be done shares strong similarities with other well-known models. This process is, many times, considered to be part of the model training.

- **Model Deployment**: Deployment is the phase where a trained model, that fulfills the business objectives, is made available to users.

- **Model Maintenance**: AI algorithms require continuous monitoring and maintenance to deal with potential concept drifts/changes that can arise during their operation. To deal with model maintenance several techniques such as window-based relearning or back testing can be employed. In the first, new models are systematically created from the most recent data points while, in the second, the performance of the deployed model is continuously monitored in order to understand when retraining is required, detecting potential performance drops.

- **Business Understanding**: In the business understanding phase, companies gather insights on the impact of AI on their business with the objective of maximizing their possibility of success while accounting for the cost of the development process.

3.3 Actors

According to ENISA, in [104], multiple actors can be actively engaged on the entirety of the AI lifecycle. These include AI application designers and developers that work closely with data scientists to design and create completely integrated AI systems. On the other hand, data scientists mainly work in the design and development of AI models, interpreting data, extracting high-level insights, training algorithms and analysing results. In turn, these rely on data engineers to manage and optimize the flow of data, collecting it from difference sources, cleaning, standardizing and storing it. Differently, other important actors, such as data owners, are the ones who actually own the datasets that are used to build the AI systems. These can also take the role of data providers (or brokers), monetizing different types data for multiple purposes. Similarly, there are also model providers, who provide already tuned models (*e.g.,* through cloud services), and third-party providers, that deliver third-party software frameworks and libraries to be used by AI developers. Eventually, there are also the end users, that benefit from a given AI system's functionalities, such as specific companies or the general public.

3.4 No-code/Low-code Platforms

The concept behind no-code/low-code platforms has been around for decades, far long before the term was officially used [105]. However, there is currently an increasing trend in what concerns to the development of this sort of technology [17]. Particularly, in the context of AI, the no-code/low-code landscape is gradually maturing as new platforms emerge at a vigorous pace, contributing to enhance development productivity as well as promoting the democratization of AI [106].
Fundamentally, this section aims to clarify how ALMA positions itself in comparison with other already existing applications. For that, a predefined set of metrics was selected and employed to better understand how different no-code/low-code AI platforms relate to one another.

3.4.1 Taxonomy

Nowadays, there still seems to exist some misconceptions in what concerns the terms ”no-code” and ”low-code”, leading to different interpretations (and wrong employment) of such concepts [107]. Hence, for the scope of this thesis, there is a need to establish a proper taxonomy, providing better and cleared organization of the no-code/low-code AI landscape.

According to Outsystems, one of the biggest players of the low-code market for application development [108], the terms can be defined as follows [107]:

- **No-code**: Distinctively to low-code, the targets of no-code platforms are people lacking formal development training *i.e.*, non-technical people that may not know any actual programming language but want to develop a given application for some specific purpose. The major downside to no-code is ”shadow IT”, when people develop some sort of application without proper understanding of it’s implications (*e.g.*, security issues, performance issues, increased technical debt).

- **Low-code**: Low-code can be described as a way for developers of all skills to build applications with minimum effort by taking advantage of intuitive visual interactions such as the dragging and dropping of blocks into more complex workflows that are representative of an application’s intended behaviour. This allows developers to focus in higher value activities, leaving fine-grained work to be performed by the low-code development platform.

In the context of AI development platforms, another term often mistaken with no-code/low-code is AutoML, or Automated Machine Learning. As stated by Rosaria Silipo in [109], principal data scientist at KNIME [110], AutoML is an entirely different business from low-code AI.

While in a low-code development platform users are fully engaged on the data transformation steps, algorithm hyperparameters and many other aspects that define an AI pipeline, in an Automated Machine Learning application, users are confined to predefined default settings that automatically generate the intended pipeline, lacking customization and often falling short when presented with non-trivial use cases [109]. Furthermore, AutoML solutions frequently emerge as programming language libraries, which is the case of TPOT [111] and PyCaret[1] [112] for Python.

However, AutoML and low-code are not mutually exclusive since the latter can embed AutoML functionalities, benefiting both from simpler and optimized solutions for well-conditioned problems and sufficient customization to cover use cases that require a higher level of personalization. Since ALMA can also benefit from such endeavors, investigating about the state of the art of AutoML frameworks is also important for this thesis.

[1]Although PyCaret positions itself as a ”low-code” machine learning library, such categorization will not be considered under the scope of this thesis. According to the adopted definition of the term, and as PyCaret requires actual coding, without presenting intuitive visual interactions to assist the developer, it cannot be considered as low-code.

3.4.2 No-code/Low-code AI

The no-code/low-code AI landscape is still an up-growing market, with many applications sharing similar traits, ultimately, making it rather difficult to draw the line of when an application ends and another starts. Some of these position themselves in broad domains such as NLP and CV while others are more focused in specific use case management [106]. Moreover, some software such as KNIME (Konstanz Information Miner) [110] and RapidMiner [113] have been around for a long time, 2006 and 2001 respectively, providing integrated development platforms for data science development. Others, are more contemporary and focus on specific tasks such as Obviously.ai [114], created in 2018 for predictive analysis in tabular data, and Levity [115], as recent as 2020, for image and text processing applications. Furthermore, there is a thin line, dependent on interpretation, when it comes to decide whether a given platform is more related to no-code than low-code (or vice versa)[2].

3.4.2.1 Comparison

In the context of this work, the following parameters were selected to compare existing platforms:

- **Designation**: The name of the company/product that serves as a mean of identification.
- **Year**: The year in which the platform was first introduced.
- **Category**: A classification of the platform, taking either the value of no-code or low-code. When such platform shares characteristics of both domains, primary value proposition is used to decide between the two possible categories.
- **Targets**: List of target domains that are under the scope of the platform. These correspond to, but are not limited to, data types such as structure/tabular data, text (NLP) or images (CV). In the case of general-purpose Integrated Development Environments (IDEs) (e.g., KNIME), the value ”General-purpose” is used to identify its targets.
- **AutoML**: A ”Yes” or ”No” parameter that identifies if there are any Automated Machine Learning functionalities embedded into the application.

Table 3.1 provides a summary of the considered no-code/low-code applications.

[2]There are two main reasons that particularly contribute to make this categorization somewhat difficult: (i) many no-code applications provide additional customization features that require technical expertise, entering the domain of low-code; (ii) some of the presented applications represent commercial products that cannot be experimented for free or require booked demonstrations.

Table 3.1: No-code/Low-code AI platforms comparison.

No-Code/Low-code AI Landscape Summary				
Designation	**Year**	**Category**	**Target**	**AutoML**
RapidMiner [113]	2001	Low-code	General-purpose	Yes
Peltarion AI [116]	2004	Low-code	Tabular, Text, Images, Video, Audio, Multi-Modal	Yes
KNIME [110]	2006	Low-code	General-purpose	Yes
Teachable Machine [117]	2017	No-code	Images	No[3]
Obviously.ai [114]	2018	No-code	Tabular	Yes
Google AutoML [118]	2017	No-code	Tabular, Text, Images	Yes
Levity [115]	2020	No-code	Text, Images	Yes
Trinity [119]	2021	No-code	Images	Yes

A brief description of the introduced platforms can be provided as follows:

- **RapidMiner** [113]: RapidMiner is an IDE for data science introduced in 2001, with roots in the Technical University of Dortmund, that supports end-to-end data science, including ML and DL. It was developed on an open core model and it is used for both industry and research. Some of RapidMiner's biggest customers are Land Rover/Jaguar, Michellin and CEPSA.

- **Peltarion AI** [116]: Peltarion AI is a Swedish company that provides a low-code platform to build, train and evaluate DL models. Some of Peltarion's customers are companies like Tesla, BMW or HP.

- **KNIME** [110]: KNIME, with foundations in the University of Konstanz, is a low-code platform for data science. Through it's "Building Blocks of Analytics" concept, users can take advantage of intuitive visuals to build data science pipelines. KNIME has two complementary tools: KNIME Analytics Platform, which is open source and can be used to create data science workflows; KNIME Server for taking data science workflows into production.

- **Teachable Machine** [117]: Teachable Machine is a Google's web-based no-code AI platform designed to effortlessly apply AI algorithms for image-related data. It works on the

[3]Google's Teachable Machine works by using Transfer Learning (TL), *i.e*, adapting powerful general-purpose pre-trained models for specific use cases through re-training. It is unclear if any AutoML is used to further optimize the resulting neural network architecture.

basis of Transfer Learning and provides a way of downloading created models into one's machine.

- **Obviously.ai** [114]: Obviously.ai is a tool that enables non-technical business people to run predictions on their historical data, enhancing decision making. It is mostly used for tabular data and supports use cases such as fraud detection, sales prediction and credit risk scoring.
- **Google AutoML** [118]: Google's AutoML platform is a cloud service that allows users to build upon algorithms distributed by Google, tailoring them to the needs of their own business. It supports integration with custom-made software.
- **Levity** [115]: Levity is a no-code web-based platform built upon the belief that AI should not be solely the privilege of tech companies that hire and build data science teams. It provides access to algorithms for image tagging, text classification and others. It works mainly with unstructured data.
- **Trinity** [119]: Trinity is a recent no-code platform developed by Apple. It allows both machine learning researchers and non-technical geospatial domain experts to experiment with domain-specific signals and datasets. Trinity is composed of an intuitive user interface, a feature store, hosting derivatives of complex feature engineering, a deep learning kernel and a scalable data processing mechanism.

3.4.3 AutoML

AutoML is a sub-field of ML dedicated to the study of algorithmic methods that provide certain degrees of automation in all the stages of ML systems design. It's main purpose lies on increasing the efficiency of data scientists by reducing the need for human-in-the-loop in multiple steps that compose a ML pipeline [120]. This topic is therein of the upmost importance, as the world is experiencing both large-scale data collection and shortage of AI expertise required to work with such data [11],
Although AutoML could be in theory applied to any ML use case, most of the research is performed under supervised learning [121], so, in that respect, and for the scope of this thesis, despite the valiant efforts in unsupervised [122] and semi-supervised learning [123], the subsequent comparison of AutoML approaches will address only the most dominant paradigm. Additionally, there are different notions on the scope of AutoML [124], leading to distinct viewpoints. For example, the concept of "full models" [125, 126], implies that an AutoML system should mandatorily address all the necessary processes for building a supervised learning model, while,

in fact, any task that automates machine learning design can, ultimately, be considered AutoML - which is the case of hyperparameter optimization [127] and Neural Architecture Search (NAS) [128].

In order to systematize the view of this work on AutoML, and with so many aspects to be considered, there is the need to establish a proper taxonomy on the levels of automation provided by such systems. Despite several publications presenting different unifying views [120, 129], the three-tiered framework of Liu *et al.* [121, 129] will be used in the scope of this thesis. The considered levels can be described as follows:

1. α **(alpha-level)**: Addresses the task of defining a specific mapping (by means of a function) between inputs and outputs. For instance, choosing hard-coded models based on if-then rules or manually setting algorithms hyperparameters for a particular task.

2. β **(beta-level)**: This level of automation is two-fold: it includes, firstly, methods that are able to find the most appropriate hyperparameters for a given classifier by means of automatic exploration of the search space and, secondly, others with the ability to explore the space of all estimators from a limited set of learning algorithms. Examples of such level of automation are PSMS [126] and Auto-Weka [130].

3. γ **(gamma-level)**: The γ-level involves meta-learning algorithms. These methods use a knowledge base of tasks-solutions to learn how to recommend β-level algorithms for new tasks. Some examples of γ-level methods are surrogate models [131], Auto-sklearn [124] and older approaches based on algorithm recommendations from a predefined set of options [132].

3.4.3.1 Comparison

In [121], Escalante performed a review of the most relevant methodologies and works on AutoML from 2006 to 2020. These were categorized by the author into three major waves of developments. The first, between 2006-2010, focused on ”full model” building and laid the foundations for AutoML, with some approaches based on Particle Swarm Optimization (PSO) being presented. The second, between 2010-2016, was mainly marked by the adoption of Bayesian Optimization / Sequential Model-Based Optimization (SMBO) as the *de facto* optimizer for AutoML and the identification of meta-learning as a cornerstone of future endeavors. The third, from 2017 up until today, mainly features NAS, the employment of AutoML to the disruptive trend of DL [11]. Table 3.2 was adapted from [121] and provides a chronological overview of the main AutoML methods[4].

Table 3.2: AutoML methods comparison.

AutoML Landscape Summary				
Designation	**Year**	**Wave**	**Level**	**Approach**
PSMS [125, 126]	2006	1st	β	Particle Swarm Optimization
Auto-WEKA [130]	2013	2nd	γ	Sequential Model-Based Optimization
Auto-sklearn [124]	2015	2nd	γ	Sequential Model-Based Optimization
TPOT [111]	2016	2nd	β	Genetic Programming
NAS_1 [133]	2017	3rd	γ	Evolutionary Algorithms
NAS_2 [134]	2017	3rd	γ	Reinforcement Learning

A brief description of each method can be provided as follows:

- **PSMS** [125, 126]: Particle Swarm Model Selection addresses the problem of full pipeline generation by finding the best combination of data preprocessing, feature selection and classification methods. The algorithm works by encoding candidate solutions into vector-based representations that undergo trough PSO in order to find those who are more likely to obtain optimal performance.

- **Auto-WEKA** [130]: Auto-WEKA considers both the problem of finding an optimal learning algorithm as well as the setting of its hyperparameters by applying Bayesian Optimization. Its performance was tested on several datasets such as MNIST and CIFAR-10, obtaining significant gains when compared to standard hyperparameter selection methods.

- **Auto-sklearn** [124]: Auto-sklearn builds upon the Python's ML library of scikit-learn by providing an AutoML wrapper. It works by taking into account past performance indicators on similar datasets and by building ensembles of models that were evaluated during the optimization process. Auto-sklearn won six out of ten phases of the first ChaLearn AutoML challenge [135].

- **TPOT** [111]: TPOT introduces the novel concept of tree-based pipeline optimization. This method, when tested on a series of benchmark datasets, showed that, without requiring any input or prior knowledge from its user, is able to achieve similar or even better degrees of performance when compared to standard ML analysis.

[4]Although the already mentioned PyCaret [112] being a well accepted AutoML framework, it was not included in the comparison as it was not possible to obtain a clear understanding of its underlying mechanics

- **NAS$_1$** [133]: Real *et al.*, applied evolutionary algorithms for NAS in the context of image classification problems. From the presented results, it was possible to conclude that, using simple evolutionary techniques and non-insignificant amounts of computational resources, it is possible to achieve competitive results within the state of the art for benchmarks such as the CIFAR-10 (95.6%) and CIFAR-100 (77.0%) datasets.
- **NAS$_2$** [134]: Zoph *et al.* from Google Brain, trained a controller Recurrent Neural Network (RNN) with reinforcement learning to generate descriptions of neural network architectures. The model, while starting from scratch, is able to design architectures that rivals with the best human-invented ones for both the CIFAR-10 (image classification) and the Penn Treebank (character language modeling) datasets.

3.5 Summary

Despite no-code/low-code AI platforms being quite complex and hard to categorize [106], it was possible to perceive that several products such as KNIME [110] and RapidMiner [113] have been around for a long time, even before the term ”low-code” was officially coined and used [17]. Additionally, and unsurprisingly, since low-code became a disruptive trend [17], many platforms have been developed in recent years (*e.g.*, Levity [115] and Trinity [119]). Another tendency highlighted through the provided comparison is the adoption of AutoML by the majority of the applications, solely demonstrating the importance of this technology for no-code/low-code AI.
In the case of ALMA platform, it can be described as a low-code application that does not require high-technical background for it’s use. However, although ALMA aims to fully engage typical users in all steps of the data science process, advocating for transparency and raising awareness to security/safety issues, it also intends to provide alternative paths for a less educated audience through the encompass of AutoML features.

CHAPTER 4

PROPOSED SOLUTION

This chapter provides an overview of the conceptualization behind ALMA. At first, a new no-code/low-code AI application is envisioned and proposed with the main purpose of tackling pitfalls of current platforms. Secondly, the overall requirements of this solution are presented and organized by different levels of priority. And, finally, the design of the implemented PoC is described along with additional implementation details.

4.1 Conceptualization

ML is a broad domain comprising several learning paradigms and algorithms [102]. Furthermore, the lifecycle of a ML-based solution is rather complex, comprising multiple interconnected phases that are hard to completely categorize and describe [104]. This partially explains why the current landscape of no-code/low-code AI is so convoluted, with many applications sharing similar traits. Some, such as KNIME [110] or RapidMiner [113] attempt to model the whole data science process, without a specific emphasis on ML, while, oppositely, others, as it is the case of Google's Teachable Machine [117], focus solely on the processing of a specific type of data trough ML algorithms. This variety of platforms offers different levels of granularity and use cases, serving both experienced data scientists and less technically educated individuals. However, recent concerns [19, 136] claim for greater transparency and security in this kind of solutions. Therefore, and motivated by the increasing use of ML, the need for greater democratization and current security concerns, this thesis envisions the development of a novel no-code/low-code platform, ALMA. To move from a hypothetical solution to a full-scale, production-ready application, functional and non-functional requirements should be well-defined, discussed and prioritized. In that sense, and with respect to both the overall ML process and existing applications, four main topics were selected for discussion: type of data; target audience; operational phases; and security concerns. The aim of ALMA on all of these topics clarifies its position in the state of the art and helps to define high-level requirements.

4.1.1 Type of Data

Some existing no-code/low-code AI platforms position themselves according to the type of data on which they operate [117], while others target all types of data in an undifferentiated way [110]. The later, although being powerful and highly configurable platforms, are harder to use due to the vast amount of scenarios that cover, losing on practicality and usability when compared to more context-specific solutions. In that sense, ALMA was designed mainly for tabular data, as it is still the most used form of data [137]. For this reason, and although the possibility of including other types of data in the future should not be completely discarded, these were not be considered to build the list of requirements.

4.1.2 Target Audience

Similarly to what happens for the type of data, there are no-code/low-code AI platforms for all kinds of users. Therefore, ALMA aims to provide support for both technical and non-technical operators, adjusting the level of configuration required to setup the ML process depending on the user's expertise. This way, more prolific users, such as data scientists, are able to fine tune the ML pipeline, benefiting from greater flexibility and transparency. Simultaneously, less technical individuals can still benefit from ALMA by using more straightforward interfaces, where the ML process is mostly controlled by AutoML.

4.1.3 Operational Phases

The overall ML development process is complicated and hard to categorize [104], being even harder when considering the implementation details of the distinct learning paradigms. Additionally, since these allow the modelling of completely different tasks, requiring differentiated

and fine-grained configurations, supporting all paradigms would most likely contribute to a convoluted system. Therefore, and as supervised learning is still the most dominant ML paradigm [138], ALMA was designed to support it, allowing both classification and regression use cases.
On the other hand, and with respect to the ML lifecycle reference model proposed by ENISA in [104], Figure 3.1, ALMA addresses the phases of Data Ingestion, Data Exploration, Data Preprocessing, Feature Selection, Model Training, Model Tuning, Model Deployment and Model Maintenance. These, although comprising the technical intensive core of the ML development process, are expected to suffer conceptual changes throughout the system design for greater practicality and increased usability[1]. The remaining phases were left apart for multiple reasons: Business Goal Definition and Business Understanding correspond to phases that occur outside the scope of the application (*e.g.*, high-level discussions in team meetings); and Transfer Learning, as it is mostly used for images and text (that is not addressed by ALMA), does not have yet much applicability for tabular data, being still an underdeveloped subject [139].

4.1.4 Security Concerns

Nowadays, AI security is a topic of paramount importance, since, without addressing current concerns [80], the potential of such technology will remain unfulfilled [81]. Furthermore, when it comes to no-code/low-code AI, security issues can be either reduced, by providing users proper security measures within the development process, or increased, through negligent/unprotected use by less-technical individuals [19]. Similarly, another contemporary concern lies on the lack of explainability and transparency, as many no-code/low-code platforms provide zero to none details on what happens under the scenes. This makes the adoption of no-code/low-code AI to be lesser than it could, as businesses are reluctant to provide a ML-based solution to a customer without fully understanding its implications [136].
To mitigate such problems, ALMA aims to provide counter-measures against adversarial attacks, such as different means of adversarial training to increase the overall robustness of algorithms. Additionally, regarding model-based explainability, ALMA encompasses functionalities that help to better understand the internal functioning of algorithms. Some of these can rely on feature importance measures, prediction-level explanations or global model explainability. Finally, to address the problem of limited transparency, this thesis proposes a code generation approach, where the configurations made by users of the platform are directly converted into executable code that can be later downloaded for further inspection. This allows users to have full

[1]ENISA's proposed ML lifecycle reference model attempts to provide an holistic view of the whole ML domain, while ALMA addresses only one of its subfields, supervised learning. Therefore, the initial reference model was refactored to reflect this narrower viewpoint, aggregating some of its phases with little taxonomy changes

perception of ALMA's inner workings, potentially increasing their trust in the system.

4.1.5 Requirements

Having described the positioning of ALMA in several topics, it is now possible to understand not only how this solution compares to existing platforms, but also how it intends to tackle the general weaknesses of the state of the art in no-code/low-code AI. Therefore, to provide a more systematic overview of the proposed solution, a list of functional requirements was elaborated. These were prioritized into three levels, low, medium and high, so that it is easier to select which requirements should first be addressed in subsequent development iterations. Although all requirements appear to be similarly important, there is no point in addressing some of them without having others implemented first. For example, there is no point in providing access to different adversarial training methods if the platform does not yet support the modeling of a traditional ML pipeline. The list of high-level requirements is presented in Table 4.1.

Table 4.1: High-level requirements of the proposed solution.

Identifier	**Requirement**	**Priority**
R1	Users must be able to upload tabular datasets into the system.	High
R2	Users must be able to create, execute and evaluate ML pipelines, for classification or regression, using an uploaded dataset.	High
R3	Users must be able to advance with a given ML pipeline for the production phase.	High
R4	Users must be able to consult records from an uploaded dataset.	Medium
R5	Users must be able to download the code associated to a given ML pipeline.	Medium
R6	Users must be able to visualize insightful explanations about a given ML pipeline operation.	Medium
R7	Users must be able to use current adversarial training methods to improve the robustness of their ML pipelines.	Medium
R8	New users must be allowed to register themselves in the system.	Medium
R9	Users must be able to edit their personal information as well as delete their own account from the system.	Low

These requirements can still be further dissected. For example, the main, and only actor mentioned in Table 4.1 is the User, but, as previously stated, each user can be associated to a different technical level. For this reason, the User is further divided into two profiles, the Technical User and the Non-technical User. Depending on the profile, R1, R2, R3 and R7 are expected to be addressed differently. In particularly, R2 requires further detailing as it is arguably one of the most important requirement:

- **R2.1**: Technical Users must be able to specify feature-level preprocessing operations such as feature imputation (*e.g.*, replacing missing values by the mean) and encoding (*e.g.*, ordinal encoding or one-hot encoding).
- **R2.2**: Technical Users must be able to specify dataset-level preprocessing operations such as feature selection (*e.g.*, recursive feature elimination or PCA) and normalization (*e.g.*, min-max normalization).
- **R2.3**: Technical Users must be able to select specific algorithms to be used as a final step of the classification/regression ML pipeline.
- **R2.4**: Technical Users must be able to perform hyperparameter tuning with different optimization strategies (*e.g.*, grid search or random search).
- **R2.5**: Technical Users must be able to select evaluation metrics to be computed when executing a given ML pipeline (*e.g.*, accuracy or f1-score).
- **R2.6**: Non-technical Users must be able to have the required configurations to build a given ML pipeline abstracted through AutoML.

The same logic applies for requirements R1, R3 and R7. While a Technical User is expected to be provided with greater flexibility and customization, a Non-technical User requires a higher level of abstraction to keep the overall process rather simple, possibly requiring the application of distinct methods to perform the same task. Furthermore, there is also to note that all sub-requirements inherit the priority of the parent, high, with the exception of those related to Non-technical Users, which are classified with medium priority-level.

4.2 Proof of Concept

Taking into account the time constraints implicit to the development of this thesis, attempting to fully implement the proposed solution would end up being an overly ambitious attempt. Therefore, instead, only a fraction of the introduced requirements was addressed, producing a working PoC. These were selected according to the priority level, R1 to R6, with some simplifications being made for this first iteration. For each requirement, the introduced simplifications can be described as follows:

- **R1**: The data ingestion functionality of the system was constrained only to tabular data files, disregarding other means such as SQL or NoSQL databases.
- **R2**: Only Technical Users were considered and the system was only expected to support classical ML algorithms and basic preprocessing steps.
- **R3**: For the deployment phase, integration with cloud or API generation was considered, however, for now, only the trained model was expected to be provided as a downloadable serialized file. This file can later be used as part of another system that, somehow (*e.g.*, REST API), exposes the model's capabilities.
- **R4**: For the PoC, the user should only be able to consult a small sample of a dataset's records, not allowing for more complex query and visualization mechanisms.
- **R5**: This requirement was expected to be fully addressed, as the ALMA approach to no-code/low-code is based on code generation.
- **R6**: For now, only the importance measure of the dataset's features should be provided as an explainability functionality.

4.2.1 Use Cases

After selecting the subset of requirements to be addressed for the PoC and identifying the list of simplifications required to comply with the time constraints, it was necessary to detail these high-level requirements into fine-grained Use Cases (UCs) to steer the remaining design and implementation. The Technical User's use cases are presented in Table 4.2.

Table 4.2: Technical User's use cases.

Identifier	**Use Case**
UC1	As a Technical User, I should be able to upload tabular datasets.
UC2	As a Technical User, I should be able to consult the list of datasets.
UC3	As a Technical User, I should be able to see a sample of records from an uploaded dataset.
UC4	As a Technical User, I should be able to create a ML pipeline.
UC5	As a Technical User, I should be able to consult the results of a ML pipeline.
UC6	As a Technical User, I should be able to consult the importance of a dataset's features.
UC7	As a Technical User, I should be able to download the code related to a given ML pipeline.
UC8	As a Technical User, I should be able to download a trained model in serialized format.

When performing this process, new problems emerged, such as how would the system manage the available algorithms and metrics that Users employ when configuring a given ML pipeline. One option would be to statically define a predefined set of resources supported by the platform. However, that solution is rather limiting as it would make it difficult to expand the number of algorithms and metrics in the future. Therefore, the need to create and manage these kind of resources was identified, along with a new actor, the Administrator. The Administrator's use cases are presented in Table 4.3.

Table 4.3: Administrator's use cases.

Identifier	**Use Case**
UC9	As an Administrator, I should be able to define new algorithm types.
UC10	As an Administrator, I should be able to configure new algorithms.
UC11	As an Administrator, I should be able to configure new metrics.

4.2.2 Domain Model

With the use cases and their respective actors identified, a domain model was developed to provide a better understanding of the business domain addressed by the PoC, revealing the main concepts and their relationships through a series of interconnected constructs. The proposed domain model is presented in Figure 4.1.

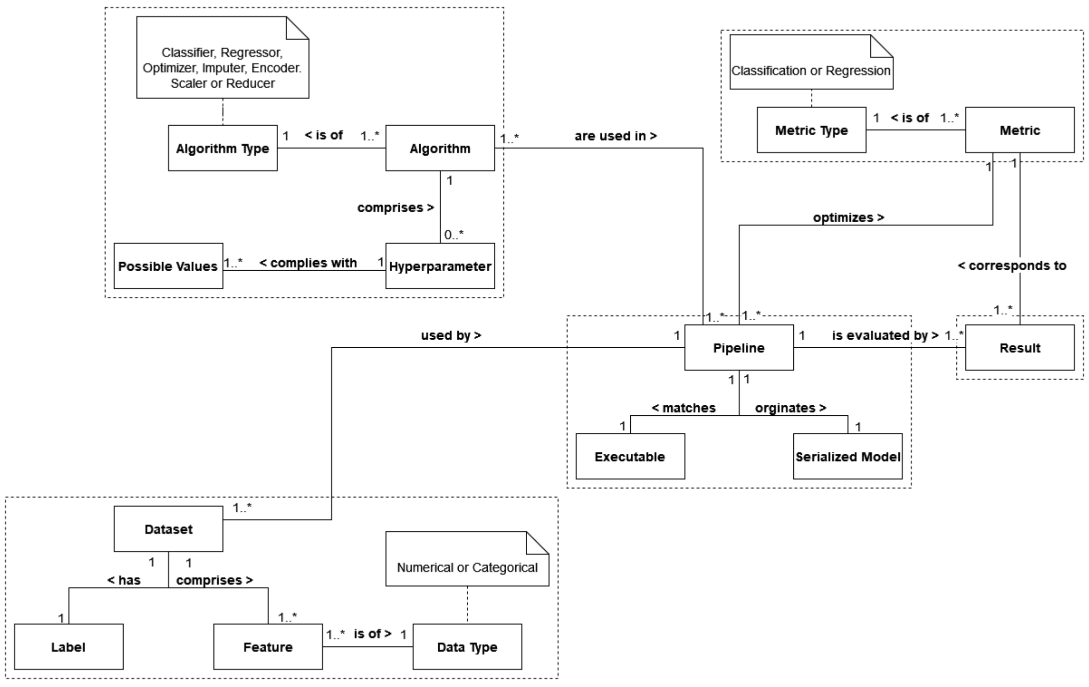


Figure 4.1: Domain Model.

In the presented domain model, 5 different contexts were identified, that is, the definition of tangible boundaries of the sub-domains presented in the global model. These couple strongly related constructs together, providing a better overview of the business. For each context, the relationship between concepts can be briefly described as follows:

- **Dataset**: The dataset represents a set of tabular data that was uploaded into the system. This data comprises one label (*e.g.*, the classes we are trying to make a given ML model recognise) and a subset of features. In turn, each feature is of a given data type, either categorical or numerical. The proper identification of the data types is required to decide upon the most suitable preprocessing operations.

- **Algorithm**: An Algorithm does not represent exclusively ML algorithms for classification or regression. It is an abstraction of, ultimately, any algorithmic operation, being it either a ML model or an optimization algorithm for hyperparameter tuning. This distinction is performed through the association between the Algorithm and Algorithm Type concepts. Furthermore, an Algorithm can comprise several hyperparameter *i.e*, degrees of freedom that influence its behaviour. The different values a given hyperparameter can take are constrained by Possible Values. These can be of different types, as for example, if an hyperparameter is exclusively categorical (*e.g.*, distance formula to be used by KNN), then its possible values are constrained to a specific list of designations. On the other hand, if a hyperparameter requires a numerical value, it can be constrained between a minimum and a maximum.

- **Metric**: An evaluation Metric represents a given mathematical formula to measure the deviation between a set of predictions and the ground truth. Depending on the Metric Type, classification or regression, different metrics are applied (*e.g.*, accuracy for classification and mean absolute error for regression).

- **Pipeline**: The Pipeline is a concept that represents the configuration of multiple operations, Algorithms, such as preprocessing operations, ML models and meta-heuristic algorithms for hyperparameter tuning, to be performed over a given Dataset. The Pipeline, when it considers hyperparameter tuning, makes use of a Metric to serve as an objective function that is maximized/minimized during the optimization process. Additionally, a Pipeline originates an Executable Model, that can be used for deployment, and matches an Executable *i.e*, the result of transforming a given Pipeline into executable code to provide increased transparency.

- **Result**: When a Metric is applied to evaluate the final (tuned) version of a Pipeline, it originates a Result. This concept represents, fundamentally, a score that evaluates a given Pipeline according to a specific Metric.

This domain model was the foundation for the subsequent design, which included the representation of the business concepts (and logic) as a given set of objects and the separation of concerns through the adoption of a service-oriented approach. It is also worth nothing that the domain model does not provide, on its own, the full overview of all the implementation details required to materialize the use cases presented in Tables 4.2 and 4.3.

4.2.3 Design

To provide an overview of the employed design, a combination of the 4+1 architectural view model [140] and C4 model [141] was used to categorize all diagrams throughout the following section. These foresee 4 + 1 different views of the software system with 4 levels of abstraction, respectively. However, in this thesis, only two views will be addressed, the logic view and the process view, at the second level of abstraction (container level). Furthermore, the Unified Modeling Language (UML) [142] formalism was also used for all graphical views of the system.

4.2.3.1 Logic View

The logic view, at the container level, represents the different individual blocks of the system and their interaction. In the context of ALMA's PoC, the system is comprised by 5 containers with limited responsibilities to assure greater maintainability and fault isolation. The logic view at container level is presented in Figure 4.2.

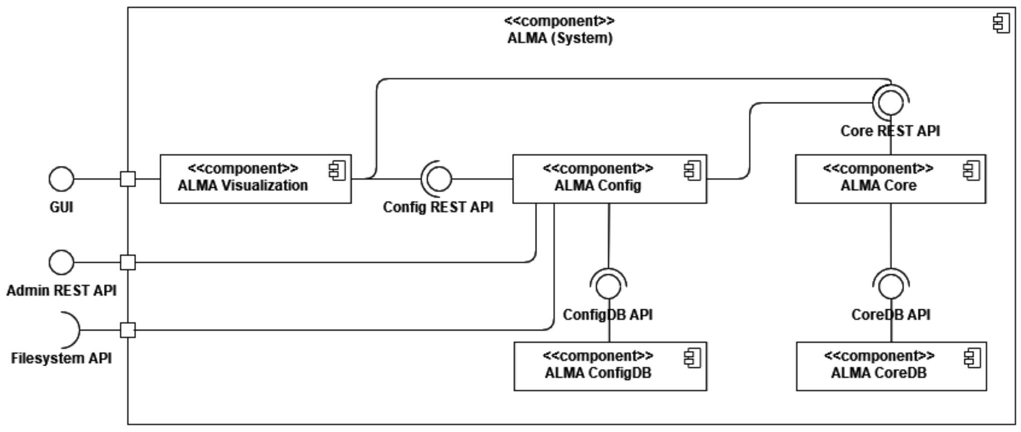


Figure 4.2: Logic view at container level.

The responsibilities of each container presented in Figure 4.2 can be described as follows:

- **ALMA Core**: This container is responsible for managing the building blocks of ML pipelines, such as algorithms and metrics. ALMA Core provides two conceptually distinct interfaces, Core REST API and Admin REST API. The Core REST API is mainly used for data consulting and validation by ALMA Visualization, while the Admin REST API provides access to the Administrator's use cases. As it would not be viable to implement a distinct GUI for the Administrator in the PoC, its use cases are triggered through this API at the first start of the system, populating ALMA CoreDB with the supported algorithms and metrics.
- **ALMA Config**: This container is responsible for managing both datasets and pipelines. It provides a single interface, Config REST API, which is used by ALMA Visualization, and consumes the Filesystem API for file input/output operations.
- **ALMA Visualization**: Finally, ALMA Visualization is the container responsible for handling the interaction with the Technical User through a GUI.

The remaining containers, ALMA CoreDB and ALMA ConfigDB, do not require further details as their name is quiet self-explanatory, representing the data stores used by ALMA Core and ALMA Config for persistence purposes. The choice of splitting the overall system into smaller containers can be justified by the Single Responsibility Principle (SRP), the ”S” of the SOLID design principles mnemonic [143]. This principle states that every system component should have only one responsibility. Similarly, with respect to ALMA Core, two distinct client-specific interfaces are provided by the container instead of a single general-purpose one. This design follows the Interface Segregation Principle (ISP), the ”I” in SOLID.
Regarding technology-specific choices, both ALMA Core and ALMA Config were implemented using the Python programming language [144], while ALMA Visualization was implemented using Vaadin [145], a web application platform for Java [146]. For the container's databases, ALMA ConfigDB and ALMA CoreDB, MongoDB [147] was selected as the database management program.

4.2.3.2 Process View

The process view, at the container level, describes the interactive process of the system when processing a specific use case. In this thesis, one use case of each actor was selected to be showcased, UC4 (as a Technical User, I should be able to create a ML) and UC10 (as an Administrator, I should be able to configure new algorithms). For UC4, the process view, at container level, is presented in Figure 4.3.

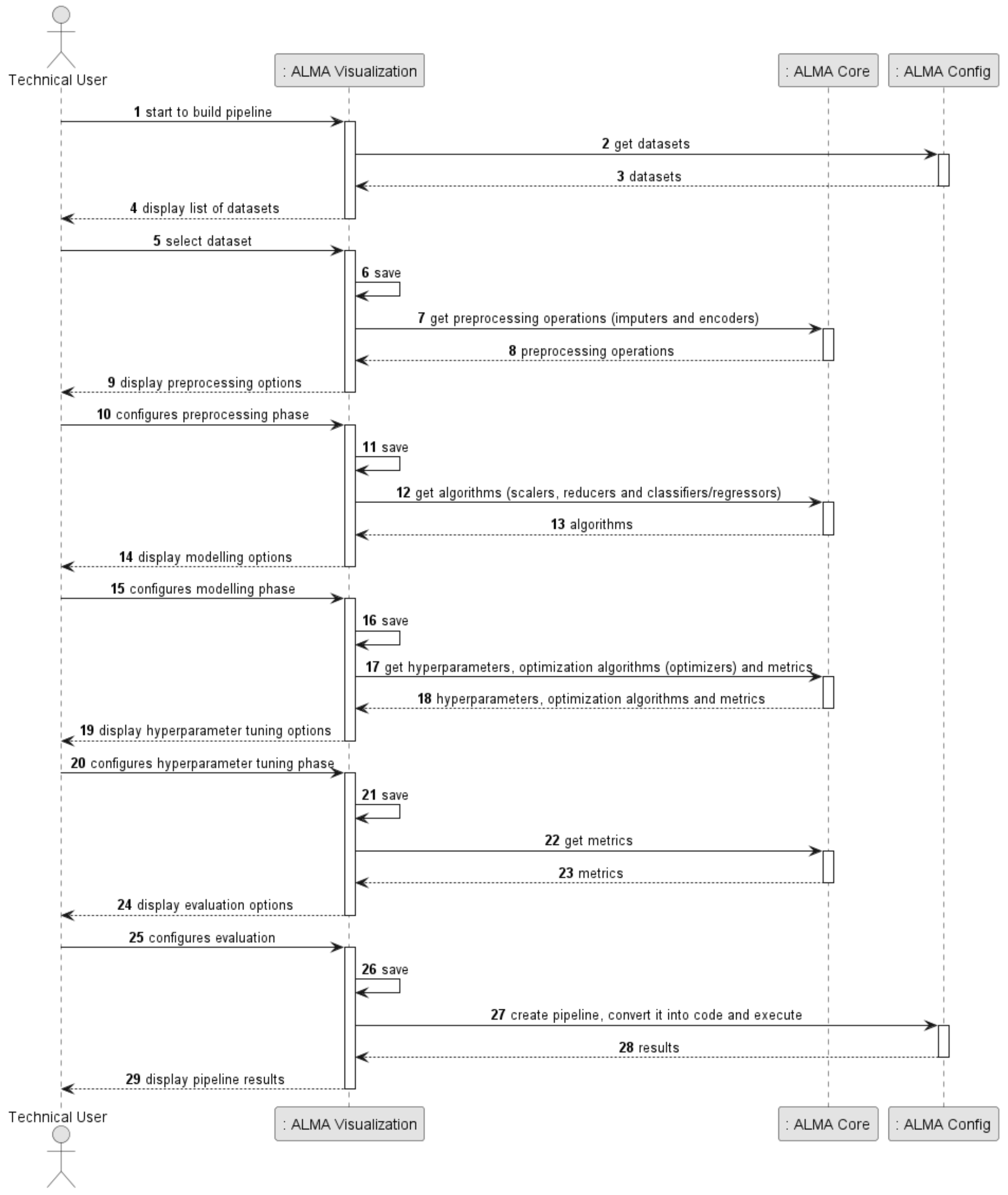

Technical User
: ALMA Visualization
: ALMA Core
: ALMA Config
1 start to build pipeline
2 get datasets
3 datasets
4 display list of datasets
5 select dataset
6 save
7 get preprocessing operations (imputers and encoders)
8 preprocessing operations
9 display preprocessing options
10 configures preprocessing phase
11 save
12 get algorithms (scalers, reducers and classifiers/regressors)
13 algorithms
14 display modelling options
15 configures modelling phase
16 save
17 get hyperparameters, optimization algorithms (optimizers) and metrics
18 hyperparameters, optimization algorithms and metrics
19 display hyperparameter tuning options
20 configures hyperparameter tuning phase
21 save
22 get metrics
23 metrics
24 display evaluation options
25 configures evaluation
26 save
27 create pipeline, convert it into code and execute
28 results
29 display pipeline results
Technical User
: ALMA Visualization
: ALMA Core
: ALMA Config

Figure 4.3: UC4 - Process view at container level.

The interaction presented in Figure 4.3 describes the process of creating a new ML pipeline. For this, the Technical User interacts with ALMA Visualization, sequentially performing a series of configurations relative to each ML phase: preprocessing, modelling, hyperparameter tuning and evaluation. ALMA Visualization, in turn, fetches data from specific sources, ALMA Config and ALMA Core, for each step of the pipeline. Finally, when the Technical User finishes the configuration of the evaluation phase, the last step of the pipeline, ALMA Visualization commands the creation of the ML pipeline to ALMA Config, which responds with the correspondent evaluation results. There is also to note that calls 27 and 28 were condensed for visualization purposes. Although conceptually there is only one call, in practice, several calls are made to ALMA Config, ordering the creation of the pipeline, its conversion into executable code and the execution itself. On the other hand, the Administrator's use case is substantially more simple to follow, requiring only a single call to the ALMA Core container, using the Admin REST API. The UC10 process view, at container level, is presented in Figure 4.4.

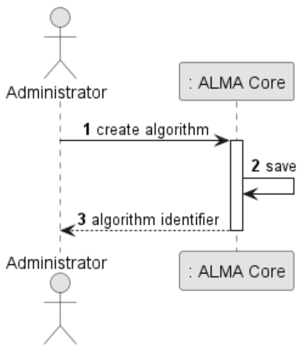


Figure 4.4: UC10 - Process view at container level.

In the diagram presented in Figure 4.4, as the development of an interface for the Administrator was not considered for the PoC, the actor is, in truth, another system that triggers its use cases at the first start of the application, populating ALMA with the supported algorithms and metrics. Furthermore, for simplification purposes, the two database containers, ALMA CoreDB and ALMA ConfigDB were not represented in the process view, although being consistently used in the majority of the use cases for data query and persistence purposes.

4.3 Summary

The field of no-code/low-code AI is gradually maturing, with many platforms being produced over the last few years [106]. Nevertheless, there is yet room for improvement in current solutions, especially when it comes to security/safety aspects such as robustness, explainability, trustworthiness and transparency [19, 136].

However, to move from a hypothetical solution to a *de facto* no-code/low-code AI platform, there is much work to be done. Hence, this chapter made a step towards that goal, providing a detailed description on how ALMA positions itself according to the type of data, target audience, targeted ML operational phases and AI security concerns - distinguishing factors in the state of the art. Furthermore, a list of high-level requirements was produced to systematize what should be addressed in a future implementation, guiding the design of subsequent platforms.

Evidently, for this thesis, aiming to implement the full system would be a too ambitious attempt.

Therefore, only a subset of requirements was chosen to be partially addressed in a PoC, with the main objective of testing the applicability of the proposed solution in real case studies. The high-level design decisions required to build this software system were also described throughout the remaining of the chapter, providing an overview of the overall architecture at container level.

CHAPTER 5

DEMONSTRATION

To demonstrate the applicability of the proposed platform, two publicly available datasets were selected as benchmark for comparing several classification and regression algorithms. In this chapter, these datasets are described along with the configuration steps required to perform the comparative study, being the experimental results presented and discussed. All the necessary implementation was carried out exclusively using ALMA's features.

5.1 Case Studies

As ALMA supports both classification and regression, one dataset was chosen for each of those tasks. These datasets were extracted from Kaggle, being the first one related to website phishing detection [148, 149] and the second one to laptop price forecasting [150].

5.1.1 Website Phishing

Website phishing can be described as the practice of mimicking a given trusted website to obtain sensitive user information. Thus, due to the massive amount of online transactions performed on a daily basis, this topic is of paramount importance for securing the online community [148].

The selected website phishing dataset comprises 10 high-level features related to 1353 websites of different sources, holding three possible categorical values, legitimate, suspicious and phishy, that have been replaced for 1, 0 and -1, respectively. The value of each feature was determined based on predefined if-then-rules. The dataset's features can be briefly described as follows [148, 149]:

- **Server From Handler (SFH)**: When a user submits information on a given website, this will transfer it to a server for further processing. If the SFH is "about: blank" or empty, the website is considered to be phishy, while if the server transfers the information to a different domain it is suspicious and, otherwise, legitimate.
- **Pop-up Window**: Evaluates the existence of pop windows with forms, as in legitimate websites users are, typically, not asked to submit their credentials via popup windows.
- **SSL Final State**: Verifies if the HTTPS protocol is legitimate and is offered by a trusted issuer such as GeoTrust or VeriSign.
- **Request URL**: As webpages consist of text and multimedia objects such as images or videos, these are usually loaded from the same server of the webpage. Hence, this feature evaluates if these objects are loaded from a different domain other than the one typed in the URL address bar.
- **URL of Anchor**: If the links within the page lead to domains different from the one displayed in the URL address bar.
- **Website Traffic**: Phishing websites usually have much lower website traffic than legitimate ones, as they are expected to have a relatively short life.
- **URL Length**: Since phishers can hide the suspicious part of a URL to redirect user information to untrusted domains, long URLs are considered to be suspicious or phishy.
- **Age of Domain**: Websites that are online only for a short period of time can be deemed as phishy.
- **Having IP Address**: If the domain name of the URL uses an IP address, sometimes presented in hexadecimal notation, then the website is most likely to be phishy.
- **Result**: It's the dataset's target variable, labeling each website as either legitimate, suspicious or phishy.

5.1.2 Laptop Prices

For the regression task, a laptop price forecasting dataset was selected from Kaggle [150]. This dataset comprises 11 laptop-related features, such as the product description, laptop type (*e.g.*, notebook or gaming), inches, screen resolution, CPU, RAM, memory, GPU, operating system and weight that can be used to predict the dataset's target variable, the price in euros. Differently from the website phishing dataset, as these features are quite straightforward to understand, there is no need for a more thorough description. However, since the dataset is not originally as clean as the phishing one, it requires some operations to be done before being uploaded into ALMA for further processing. To simplify the cleaning operation, the same procedure employed by Ruslan in [151] was implemented resorting to the Python programming language. The most significant steps that were performed can be described as follows:

1. Firstly, duplicated records were discarded from the dataset. These include situations where all features are the same but the laptop price varies.

2. Then, records where the laptop price belonged to the 95 percentile of the probability distribution were also removed, avoiding probable outliers.

3. The RAM column was preprocessed (*e.g.*, ”GB” was removed) and directly converted as a numerical feature, removing rows that only occurred once (24 and 32GB).

4. Regarding the operating system, ”mac os x” was replaced by ”macos” and ”windows 10 s” by ”windows 10”. The single record related to the ”android” operating system was removed.

5. From the screen resolution feature, more significant high-level features were extracted: width, height and the resolution category, such as ”full hd” and ”4k ultra hd”.

6. Similarly, the CPU, GPU and memory features were split according to the unique occurrence of their values (*e.g.*, intel or amd for cpu, intel, nvidia or amd for gpu and ssd, hdd or flash for the memory feature).

7. Finally, all column names were normalized, being converted to lower case and have each word split by ”_”, before exporting the transformed dataset.

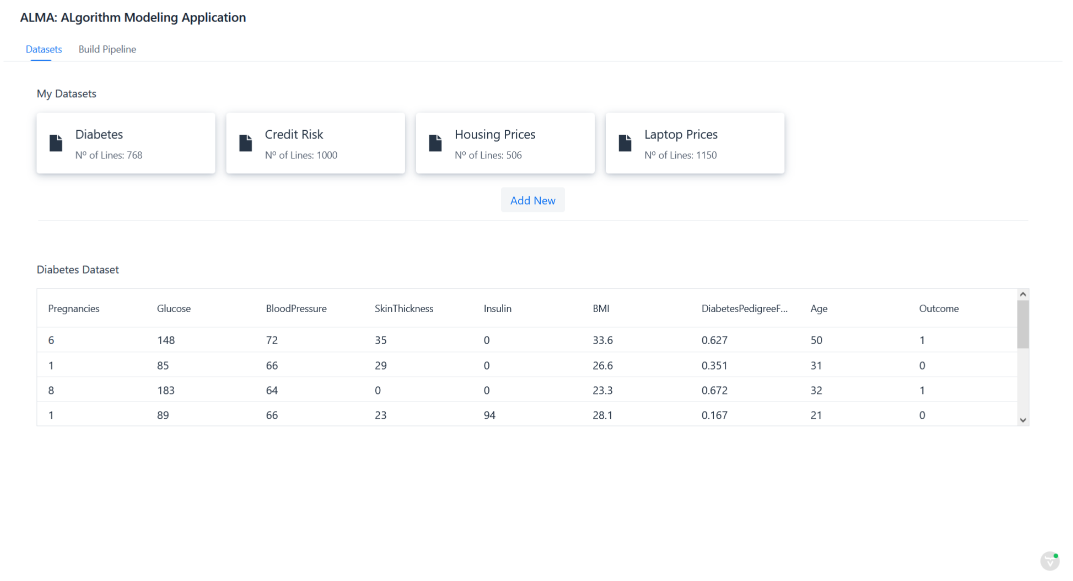
The resulting laptop price dataset comprises 26 features (including price), being uploaded to ALMA just like the website phishing one.

5.2 Execution

After describing the chosen datasets, one can now upload them into ALMA to create/configure distinct ML pipelines for both classification and regression, assessing their predictive capabilities in any of these benchmarks. In particularly, for this work, four ML algorithms, Decision Tree, XGBoost, Random Forest and K-Nearest Neighbors (KNN), were selected, implemented and compared for both tasks solely by using ALMA's features[1]. Furthermore, and to assure an unbiased comparison, several rules, accounting the best practices of ML development, were defined and strictly followed, being materialized as simple configurations in ALMA's GUI. The code generated by the application for the presented example is provided in Appendix A.

5.2.1 Data Ingestion

Having downloaded the datasets from Kaggle, they can be uploaded into ALMA platform by selecting the "Add New" button in the main dashboard, Figure 5.1.


ALMA: ALgorithm Modeling Application

Datasets Build Pipeline

My Datasets

Diabetes	Nº of Lines: 768
Credit Risk	Nº of Lines: 1000
Housing Prices	Nº of Lines: 506
Laptop Prices	Nº of Lines: 1150

Add New

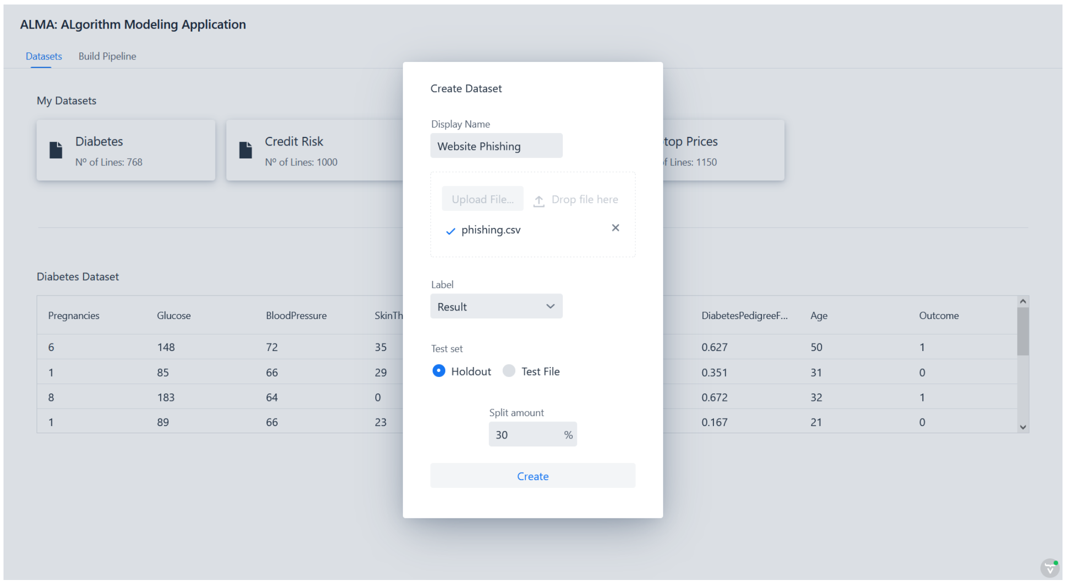
Diabetes Dataset

Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeF...	Age	Outcome
6	148	72	35	0	33.6	0.627	50	1
1	85	66	29	0	26.6	0.351	31	0
8	183	64	0	0	23.3	0.672	32	1
1	89	66	23	94	28.1	0.167	21	0

Figure 5.1: ALMA platform - Main Dashboard.

Afterwards, a dialog will pop-up, prompting the user to select the dataset file, automatically filling in the display name on upload, and allowing further editing. The user should also select the label column and specify the percentage of data that must be reserved for the test set (or upload a new test file entirely if the dataset is split beforehand), Figure 5.2.

[1]Although these four algorithms were specifically selected for the demonstration, ALMA supports the vast majority of algorithms that are implemented in Python and compatible with scikit-learn [15]. New algorithms can be configured in the system without having to rewrite any code, requiring only the execution of UC10 and small configurations in a JSON file.


ALMA: ALgorithm Modeling Application

Datasets Build Pipeline

My Datasets

Diabetes N° of Lines: 768

Credit Risk N° of Lines: 1000

top Prices f Lines: 1150

Diabetes Dataset

Pregnancies	Glucose	BloodPressure	SkinTh	DiabetesPedigreeF...	Age	Outcome
6	148	72	35	0.627	50	1
1	85	66	29	0.351	31	0
8	183	64	0	0.672	32	1
1	89	66	23	0.167	21	0

Create Dataset

Display Name: Website Phishing

Upload File... Drop file here

✓ phishing.csv ×

Label: Result

Test set: [● Holdout] Test File

Split amount: 30 %

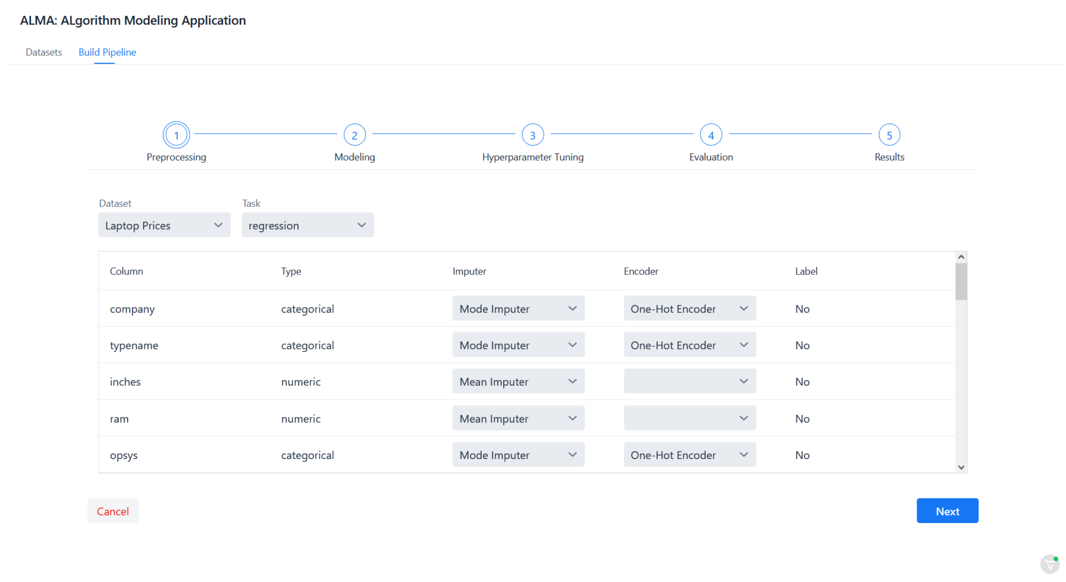
Create

Figure 5.2: ALMA platform - Add New Dataset.

The procedure illustrated by Figures 5.1 and 5.2, was equally applied for both web phishing and laptop prices datasets.

5.2.2 Preprocessing

In the preprocessing step, one of the uploaded datasets should be selected along with the task to be done (classification or regression). Then, in the table bellow, for each column of the selected dataset, some information is displayed and basic preprocessing steps regarding features imputation and feature encoding can be configured. Figure 5.3 presents the employed preprocessing for the laptop prices dataset.


ALMA: ALgorithm Modeling Application

Datasets Build Pipeline

1 Preprocessing — 2 Modeling — 3 Hyperparameter Tuning — 4 Evaluation — 5 Results

Dataset: Laptop Prices

Task: regression

Column	Type	Imputer	Encoder	Label
company	categorical	Mode Imputer	One-Hot Encoder	No
typename	categorical	Mode Imputer	One-Hot Encoder	No
inches	numeric	Mean Imputer		No
ram	numeric	Mean Imputer		No
opsys	categorical	Mode Imputer	One-Hot Encoder	No

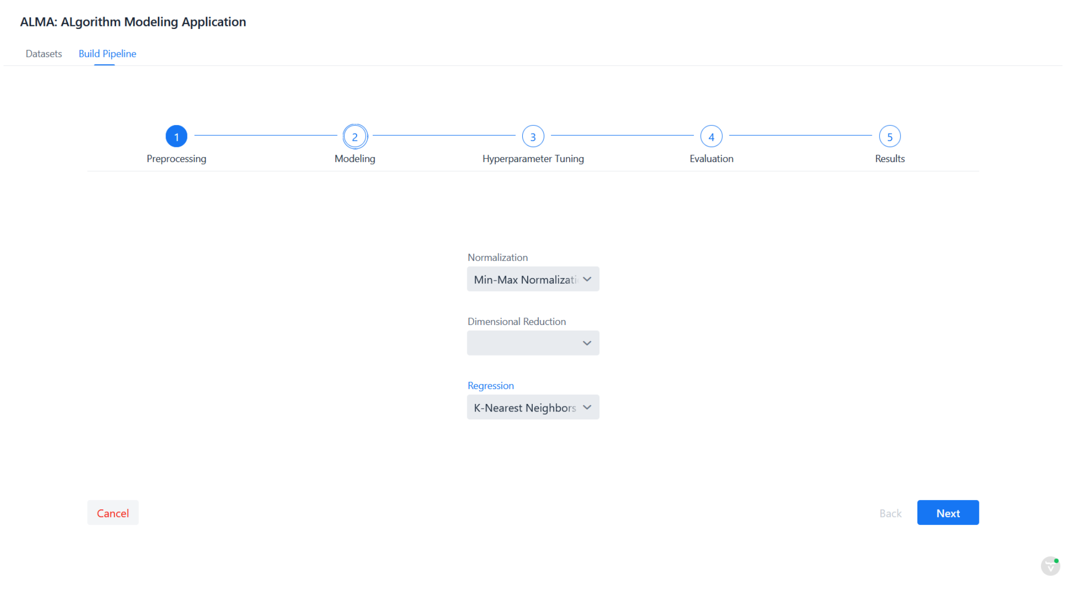
Cancel Next

Figure 5.3: ALMA platform - Build Pipeline - Preprocessing Step.

For this experience, regarding imputation, ”Mode Imputer” was employed for categorical features, replacing potentially missing values of a given feature by the most frequent one, and ”Mean Imputer” was employed for numerical features, replacing missing values by the mean of all other values of that given feature. On the other hand, for the encoding, that applies only to categorical variables, one hot encoding was used, as there wasn’t any ordinal features and the number of distinct values of each categorical feature is considerably small (otherwise, as it would result in sparse and large feature spaces, other strategies such as binary encoding could be preferred).

5.2.3 Modeling

In the modeling step, three select boxes are displayed, allowing the configuration of feature scaling, dimensional reduction and the final predictive algorithm. Regarding the first, min-max normalization was employed only for the KNN, as the algorithm usually underperforms when the input data presents different scales (*e.g.*, tens vs thousands). The remaining algorithms, being tree-based, do not suffer from the same problem, and as of such, don’t require normalization to be performed. Differently, dimensional reduction was not employed for any algorithm, since the input feature space is already small enough to be processed directly. For the laptop prices dataset, the configuration of min-max normalization and KNN regressor is presented in Figure 5.4.



Normalization: Min-Max Normalizati

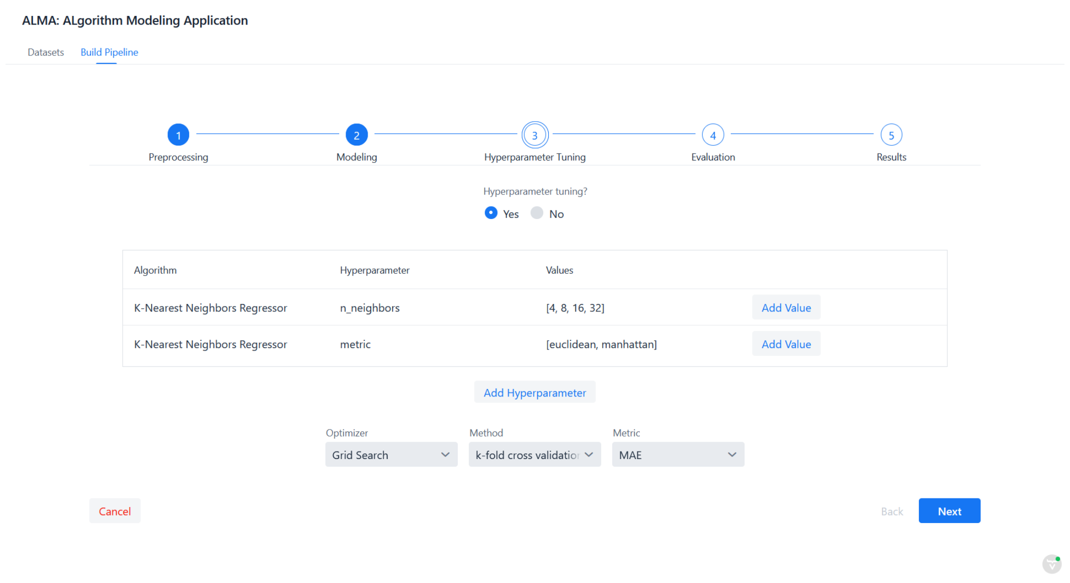
Dimensional Reduction:

Regression: K-Nearest Neighbors

Figure 5.4: ALMA platform - Build Pipeline - Modeling Step.

5.2.4 Hyperparameter Tuning

After the modeling step, hyperparameter tuning can be performed to determine the best configuration of each algorithm before making the final evaluation in the test set. For this, k-fold cross validation (5 folds) was employed along with grid search, optimizing the Mean Absolute Error (MAE) for regression and accuracy for classification. As both datasets are relatively small (1.1 to 1.3 thousand records) and hyperparameter grids comprised only 12 different combinations, the bruteforce solution was feasible to be computed in a short amount of time (around 1 or 2 minutes). For large datasets, different choices are advised to keep the computational time of the experience manageable, such as using a validation set instead of performing k-fold cross validation and/or applying a random search (or other heuristic/meta-heuristic) instead of grid search. Figure 5.5, presents the configuration of the hyperparameter tuning process for the KNN regressor in the context of the laptop prices dataset.


ALMA: ALgorithm Modeling Application

Datasets | Build Pipeline

1 Preprocessing — 2 Modeling — 3 Hyperparameter Tuning — 4 Evaluation — 5 Results

Hyperparameter tuning?

[✓ Yes] No

Algorithm	Hyperparameter	Values	
K-Nearest Neighbors Regressor	n_neighbors	[4, 8, 16, 32]	Add Value
K-Nearest Neighbors Regressor	metric	[euclidean, manhattan]	Add Value

Add Hyperparameter

Optimizer: Grid Search

Method: k-fold cross validatio

Metric: MAE

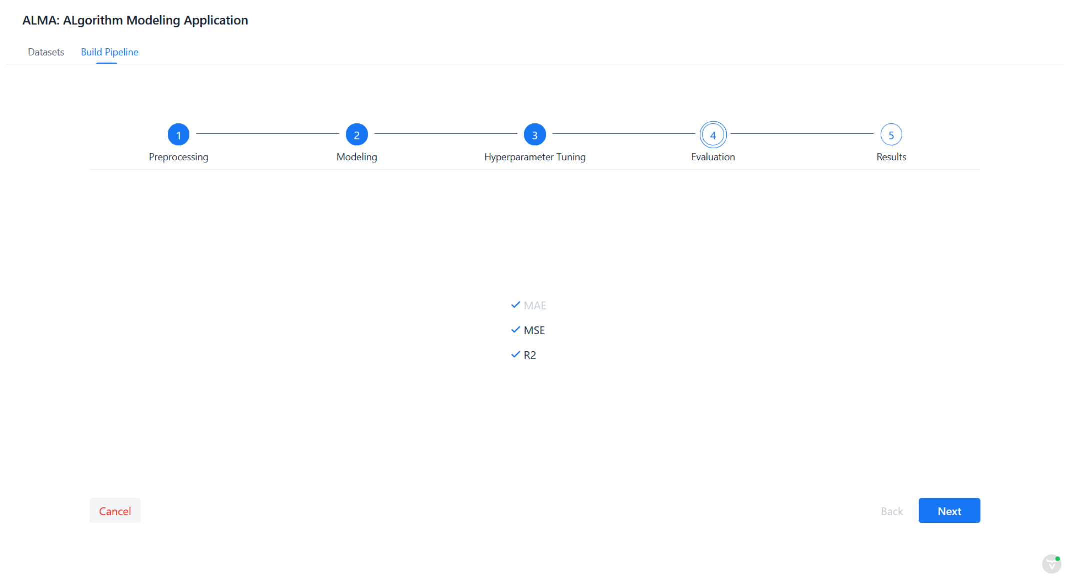
Cancel | Back | Next

Figure 5.5: ALMA platform - Build Pipeline - Hyperparameter Tuning Step.

To prevent one algorithm to be benefited over other, all employed hyperparameter grids had 12 possible combinations of parameters. Furthermore, the same grid of hyperparameters was employed for the same algorithm for both tasks, regression of laptop prices and website phishing classification. The exception was the split criterion of tree-based algorithms *i.e.*, the function that measures the quality of a split, which has different implementations depending on the task (*e.g.*, gini/entropy for classification and squared error/absolute error for regression).

5.2.5 Evaluation

In the evaluation phase, the metrics to be computed through the comparison between the model's predictions in the test set with the ground truth can be selected. For the website phishing, accuracy, precision, recall and f1-score were selected to determine the classification performance, while for the laptop prices, MAE, Mean Squared Error (MSE) and R-squared (R2) were utilized to measure the regression capabilities. By default, accuracy and MAE are selected in ALMA's GUI for classification and regression, respectively. In Figure 5.6, the selection of the regression metrics for the KNN algorithm is presented.

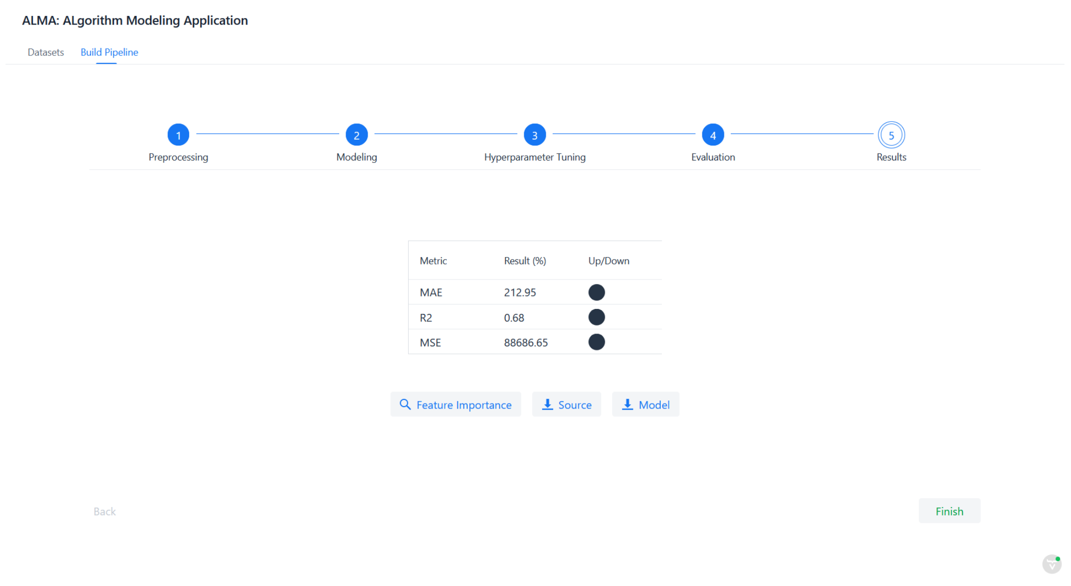


✓ MAE
✓ MSE
✓ R2

Figure 5.6: ALMA platform - Build Pipeline - Evaluation Step.

5.2.6 Results

After performing all configurations, ALMA platform will generate the source code, execute it and display a result grid with the computed evaluation metrics. Figure 5.7, presents the results for the KNN in the laptop prices dataset.



Metric	Result (%)	Up/Down
MAE	212.95	●
R2	0.68	●
MSE	88686.65	●

Figure 5.7: ALMA platform - Build Pipeline - Results Step.

In that same layout, there are three additional buttons, one to download the generated source code (”Source”), another to download the already trained model in a serialized format (”Model”) and, finally, one to visualize the dataset’s feature importance *i.e.*, the relevance of each feature to predict the target variable (”Feature Importance”).

In the context of the laptop prices dataset, the five most important features are the ”typename”, ”hdd_value”, ”memory_ssd”, ”full_hd” and ”inches”, as seen in Figure 5.8.

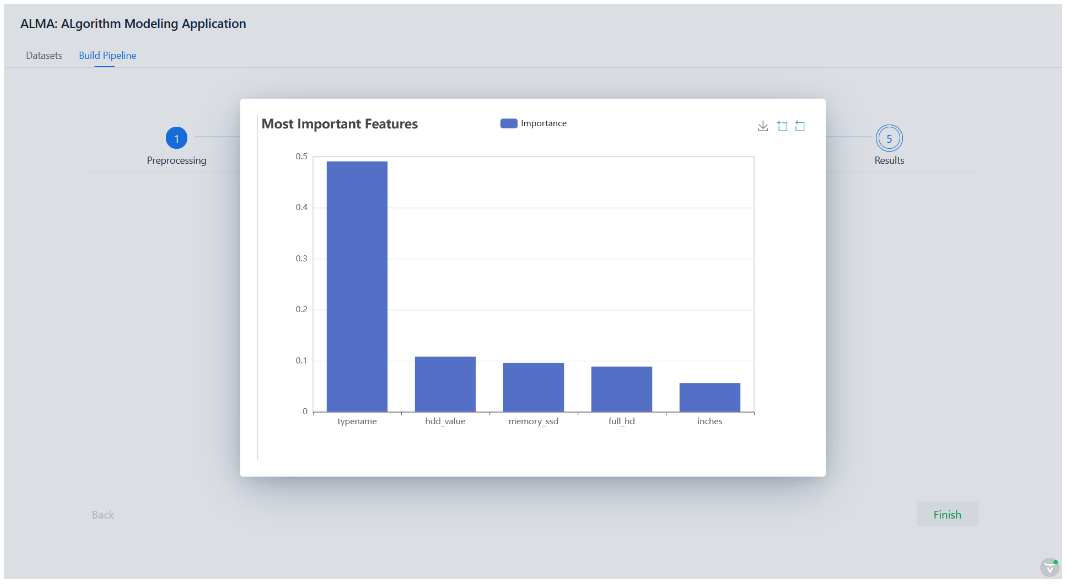


Figure 5.8: ALMA platform - Build Pipeline - Feature Importance.

To perform the intended study, this whole process was performed 8 times, one for each algorithm for each dataset, with respect to the aforementioned specifications. The obtained results are presented and discussed in the following section.

5.3 Discussion

Having used ALMA to configure and execute each ML pipeline, the experimental results were stored in order to compare the performance of distinct algorithms for both tasks. Table 5.1 presents the obtained evaluation measures for the website phishing classification task.

Table 5.1: Results for website phishing classification.

Algorithm	**Accuracy**	**Precision**	**Recall**	**F1-score**
Decision Tree	0.85	0.78	0.74	0.76
Gradient Boosting	0.91	0.85	0.92	0.87
Random Forest	0.91	0.89	0.87	0.88
K-Nearest Neighbors	0.86	0.78	0.81	0.80

From the presented results, it is possible to conclude that the Decision Tree is the worst performing algorithm for all metrics, slightly surpassed by the KNN in term of accuracy, recall and f1-score. Although both algorithms present an accuracy score above the 80% mark, 85% and 86%, respectively, the ensemble-based algorithms have proven to be even more reliable to identify phishy websites, achieving an equal accuracy score of 91%. Nevertheless, the Random Forest, presents a higher f1-score, 88%, when compared to XGBoost, 87%, partially justified by the greater balance between its precision and recall scores, 89% and 87%[2]. Ultimately, the choice between Gradient Boosting and Random Forest, for this particular scenario, relies on the business objectives and the metric to be optimized. If precision is preferred over recall, Random Forest should be chosen, while, on the other hand, if recall is more important than precision, XGBoost is the best option.

Considering the second task, laptop prices forecasting, the situation is somewhat similar. The experimental results are presented in Table 5.2.

Table 5.2: Results for laptop prices forecasting.

Algorithm	**MAE**	**MSE**	**R2**
Decision Tree	235.46	111415.79	0.58
Gradient Boosting	159.83	47654.29	0.83
Random Forest	159.50	49565.63	0.83
K-Nearest Neighbors	208.53	80689.83	0.73

As evidenced by the table above, the Decision Tree has obtained poor results when forecasting laptop prices, achieving a MAE of 235.46, a MSE of 111415.79 and a R2 of 0.58. Differently, the KNN has achieved reasonable results, improving upon the Decision Tree for all measures. Nonetheless, the tree-based ensembles has showed to be, once again, significantly superior to both algorithms. While tied on R2, 0.83, the Random Forest achieves the lowest MAE, 159.50, against 159.83 presented by the XGBoost. In turn, the later has shown greater resilience against larger errors than the Random Forest, achieving an MSE of 47654.29. Once more, it is hard to decide between both ensembles, with XGBoost taking the upper hand if having a slightly lower MAE is preferred over the risk of making large prediction errors sporadically[3].

[2]F1-score can be described as the harmonic mean between precision and recall [25].

[3]As MSE first squares the errors before being averaged, it penalizes greater deviations from the ground truth.

5.4 Summary

From the presented practical demonstration, it is possible to showcase that ALMA platform provides a practical abstraction over the process of creating classical ML pipelines, which often requires time consuming tasks such as coding or debugging, while assuring great transparency over what is happening under the hood. Through this example, it was possible to testify the applicability of ALMA, performing a complete study on two publicly available datasets, where all necessary steps of typical ML development were set through an intuitive GUI and the desired experimental results obtained without having to perform fine-grained tasks. Furthermore, both the generated code and the already trained models were made available to download, contributing to greater transparency and potentially increasing the trustworthiness of the user on the overall system. With the same objective, XAI was also addressed, providing greater insights on the dataset's features through a feature importance functionality.

CHAPTER 6

CONCLUSION

This chapter presents the main conclusions drawn from this work, appointing future research and development directions to further enhance the described solution.

6.1 Summary of Results

As AI became such an important technology for both modern-day companies and research centers, fueled by the recent advances in several subfields, such as ML, there is an increasing need to promote its democratization, making it widely available at a lower cost. With the high costs of capturing specialized talent, no-code/low-code platforms, can be a solution to provide easier access to complex algorithms, lowering the technical barriers of its use and evening the playing field between organizations of different sizes. Nevertheless, allowing less technically educated users to build intelligent systems can be a disaster in terms of security if such aspects are not properly addressed by the no-code/low-code platform being used. In that sense, ALMA was designed to fill the gap of existing platforms, providing a simply way to build complex ML pipelines while addressing the security concerns behind the use of these technologies, such as explainability, robustness and trustworthiness.

In this thesis, the high-level requirements for a working PoC were defined, leading to the design and implementation of a first version of ALMA that can be further expanded in the future to encompass additional functionalities. This PoC was showcased in a practical setting comprising two publicly available datasets, abstracting the implementation of ML pipelines through an intuitive GUI and producing the desired experimental results without having to perform fine-grained task such as coding and debugging. Furthermore, all generated code and trained models were made available trough download buttons in ALMA's interface along with greater insights on the feature importance of each dataset.

6.2 Objectives Overview

The objectives introduced in Chapter 1, section 1.2, were all fulfilled over the course of this thesis. O1 (investigate the main factors that currently endanger the democratization of AI) was addressed in Chapter 2, section 2.3, with a discussion on the current state of AI democratization being performed. O2 (clarify the meaning of the terms: no-code, low-code and AutoML) and O3 (study the state of the art in no-code/low-code AI and related technologies) were tackled in Chapter 3, section 3.4, with the proposal of a novel taxonomy and a description of the state of the art in both no-code/low-code AI and AutoML. Finally, O4 (identify the main flaws of current platforms and propose a new solution) and O5 (implement a PoC and demonstrate it in real case studies) were answered in Chapters 4 and 5. For the first, section 4.1 provides an overview over the main flaws of current solutions and proposes a new approach for no-code/low-code AI, while, for the second, section 4.2, presents an overview of the PoC's design and implementation, being its demonstration described throughout Chapter 5.

6.3 Research Questions Overview

In Chapter 1, section 1.3, the main research questions to be addressed in this thesis were defined, with possible answers being provided throughout the remaining of the document. An overview of the main conclusions drawn for each research question can be provided as follows:

- **RQ1**: What is the current state of AI democratization?
  - Although experts have been starting to advocate the democratization of AI [14], this technology is not yet, at the hands of the majority of people, with mega-firms such as Facebook and Google having great influence over its future [11]. This is largely caused due to their control over the main resources that influence the developemnt of AI: human capital, computing power and data.

  - RQ1 is mainly addressed in Chapter 2, section 2.3.

- **RQ2**: What is the current landscape of no-code/low-code AI?

  - The no-code/low-code AI market is still growing, with the recent introduction of many similar applications, making it hard to categorize and describe. There are platforms such as KNIME [110], that have been around for a long time, and others, such as Levity [115], which were more recently introduced. Furthermore, while some attempt to model the whole data science process, [113], others focus on specific use cases such as predictive analysis [114] or image processing [117].
  - RQ2 is mainly addressed in Chapter 3, section 3.4.

- **RQ3**: What are the main flaws of current no-code/low-code AI platforms?

  - While current solutions provide answers to many use cases, there is still work to do in the domain of no-code/low-code AI. According to recent concerns [19, 136], the main aspects to improve rely into providing greater transparency and explainability, as the ”black-box” approach prevents many business from taking full utilitarian value out of existing platforms. Furthermore, and with respect to the danger of adversarial attacks [82, 83], no-code/low-code can also serve as a mean for introducing security controls directly into the development process (*e.g.*, adversarial training), contributing to more secure AI-based solutions.
  - RQ3 is mainly addressed in Chapter 4, section 4.1.

6.4 Limitations and Further Work

Although this work's objectives were achieved, with the implementation and demonstration of a working PoC in a practical setting, ALMA must undergo further developments before being suitable to be used in a real production environment. In that regard, four major topics can be appointed for future research and development: (i) large datasets; (ii) deep learning; (iii) explainability; and (iv) robustness. These can be described as follows:

- **Large datasets:** Although ALMA was designed to support and operate over datasets of various sizes, it currently works in a single-threaded environment. This implies that the user must wait for the ML pipeline to process before making additional interactions in the application. Since, nowadays, most applications of AI are deeply related to big data technologies, the support for larger datasets should be enhanced for ALMA to be used in a real production environment. This limitation can be tackled by attaching a state to each pipeline (*e.g.*, in progress, finished) and executing them in separate threads, having their state updated upon finishing. In turn, the user would make use of a pipeline listing functionality to keep track of all pipelines that were built, identifying the ones still under processing. Finally, strategies regarding distributed processing and resource management should be considered to assure scalability, as multiple pipelines can be executed at the same time by different users.

- **Deep learning:** For now, only classical ML algorithms are supported in ALMA, allowing for many different pipelines to be built. However, being DL such as disrupting trend, there is a need to also support such algorithms. This implies providing an abstraction over modern programming libraries, such as Tensorflow [152] or Pytorch [153], and allowing the use of GPUs, as these are typically required to train neural networks efficiently.

- **Explainability:** Some times, in domains such as healthcare or cybersecurity, explainability is an essential requirement when developing an AI-based solution. Furthermore, the ability to understand why an algorithm reached a certain conclusion increases the level of transparency on what is happening under the scenes, potentially making the user to augment his trust on the overall system. For the first iteration of ALMA, XAI was tackled by implementing a feature importance functionality that provides insights on the impact of each feature of a given dataset. Nevertheless, further developments can be made to provide both global (overall model) and local (prediction-based) explanations, potentially resorting to well established programming libraries, which is the case of LIME [154] or SHAP [155].

- **Robustness:** As of today, adversarial attacks represent a serious threat to overall AI security. Therefore, ALMA would benefit from providing not only a simplified access to contemporary defense mechanisms, such as adversarial training, but also to appropriate robustness measures, so that the resilience of trained algorithms against such attacks can be properly judged. The implementation of such functionalities can make less technically educated users more aware about the blind use of AI, mitigating security risks from an early stage of development. For further iterations of ALMA, the Adaptative Perturbation Pattern Method (A2PM) [156] should be considered, as it provides a way of creating coherent data perturbations for tabular data. This way, it is possible to encompasses domain constraints into the data perturbation algorithm so that it doesn't generate unrealistic adversarial examples *i.e.*, samples that can't exist within a given domain.

6.5 Final Remarks

This thesis demonstrated that, although AI being an essential technology for many organizations all over the world, its use is not yet at the access of everyone. In fact, there is a need to democratize AI, preventing a privileged group of mega-corporations to shape its development in way that mostly suits the interests of their own stakeholders. On the other hand, as the dependency of the overall society on digital media only tends to increase, disregarding cybersecurity aspects when creating new products can be a costly act. This also applies to data-based solutions, with much research being done to understand the true implications of adopting AI at a large scale while lacking appropriate security controls. Therefore, in its very own essence, this work aims to raise awareness for these subjects, making a step forward to promote change by proposing a new approach to no-code/low-code AI.

REFERENCES

[1] M. Abou-foul, J. L. Ruiz-Alba, and A. Soares, “The impact of digitalization and servitization on the financial performance of a firm: an empirical analysis,” *Production Planning & Control*, vol. 32, no. 12, pp. 975–989, 2021.

[2] S. Chatterjee, R. Chaudhuri, and D. Vrontis, “”Does data-driven culture impact innovation and performance of a firm? An empirical examination,” *Annals of Operations Research*, 2021.

[3] F. Emmert-Streib, “From the Digital Data Revolution toward a Digital Society: Pervasiveness of Artificial Intelligence,” *Machine Learning and Knowledge Extraction*, vol. 3, no. 1, pp. 284–298, 2021.

[4] R. Kitchin, *The Data Revolution: Big Data, Open Data, Data Infrastructures and Their Consequences*. SAGE Publications, 2014.

[5] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, “BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding,” *ArXiv*, vol. abs/1810.04805, 2019.

[6] T. B. Brown *et al.*, “Language Models are Few-Shot Learners,” *ArXiv*, vol. abs/2005.14165, 2020.

[7] A. Dosovitskiy *et al.*, “An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale,” *ArXiv*, vol. abs/2010.11929, 2021.

[8] Z. Dai, H. Liu, Q. V. Le, and M. Tan, “CoAtNet: Marrying Convolution and Attention for All Data Sizes,” *ArXiv*, vol. abs/2106.04803, 2021.

[9] L. Columbus, “76% Of Enterprises Prioritize AI & Machine Learning In 2021 IT Budgets,” https://www.forbes.com/sites/louiscolumbus/2021/01/17/76-of-enterprises-prioritize-ai–machine-learning-in-2021-it-budgets/?sh=3ec07eed618a, Forbes, Jan 2021, accessed: 08 Jan 2022.

[10] A. X. Zhang, M. Muller, and D. Wang, “How do data science workers collaborate? roles, workflows, and tools,” *Proc. ACM Hum.-Comput. Interact.*, vol. 4, no. CSCW1, May 2020. [Online]. Available: https://doi.org/10.1145/3392826

[11] N. M. Ahmed and M. Wahed, “The De-democratization of AI: Deep Learning and the Compute Divide in Artificial Intelligence Research,” *ArXiv*, vol. abs/2010.15581, 2020.

[12] N. M. Shazeer *et al.*, “Outrageously Large Neural Networks: The Sparsely-Gated Mixture-of-Experts Layer,” *ArXiv*, vol. abs/1701.06538, 2017.

[13] G. A. Montes and B. Goertzel, “Distributed, decentralized, and democratized artificial intelligence,” *Technological Forecasting and Social Change*, vol. 141, pp. 354–358, 2019.

[14] C. T. Wolf, “Democratizing AI? experience and accessibility in the age of artificial intelligence,” *XRDS: Crossroads, The ACM Magazine for Students*, vol. 26, no. 4, pp. 12–15, 2020.

[15] F. Pedregosa *et al.*, “Scikit-learn: Machine learning in Python,” *Journal of machine learning research*, vol. 12, no. Oct, pp. 2825–2830, 2011.

[16] S. Raschka, J. Patterson, and C. Nolet, “Machine Learning in Python: Main Developments and Technology Trends in Data Science, Machine Learning, and Artificial Intelligence,” *Information*, vol. 11, no. 4, 2020. [Online]. Available: https://www.mdpi.com/2078-2489/11/4/193

[17] B. Atkins, “The Most Disruptive Trend Of 2021: No Code / Low Code,” https://www.forbes.com/sites/betsyatkins/2020/11/24/the-most-disruptive-trend-of-2021-no-code–low-code/?sh=2dd31a666570, November 2020, accessed: 08 Jan 2022.

[18] “No-code AI in 2021,” https://levity.ai/blog/no-code-ai, accessed: 28 September 2021.

[19] "Trustworthiness: An even greater challenge for the "no-code" AI models," https://trustilio.com/blog/trustworthiness-an-even-greater-challenge-for-the-no-code-ai-models/, Aug 2021, accessed: 31 December 2021.

[20] J. C. S. Santos, K. Tarrit, and M. Mirakhorli, "A catalog of security architecture weaknesses," in *2017 IEEE International Conference on Software Architecture Workshops (ICSAW)*, 2017, pp. 220–223.

[21] A. Demetriou, G. Spanoudis, and M. Shayer, "Developing intelligence: Is a comprehensive theory possible?" *Intelligence*, vol. 41, pp. 730–731, 09 2013.

[22] P. Wang, "On Defining Artificial Intelligence," *Journal of Artificial General Intelligence*, vol. 10, pp. 1–37, 01 2019.

[23] A. Voulodimos, N. Doulamis, A. Doulamis, and E. Protopapadakis, "Deep learning for computer vision: A brief review," *Computational intelligence and neuroscience*, vol. 2018, 2018.

[24] T. Dias, N. Oliveira, N. Sousa, I. Praça, and O. Sousa, "A Hybrid Approach for an Interpretable and Explainable Intrusion Detection System," *ArXiv*, vol. abs/2111.10280, 2021.

[25] N. Oliveira, I. Praça, E. Maia, and O. Sousa, "Intelligent cyber attack detection and classification for network-based intrusion detection systems," *Applied Sciences*, vol. 11, no. 4, p. 1674, 2021.

[26] N. Sousa, N. Oliveira, and I. Praça, "A Multi-Agent System for Autonomous Mobile Robot Coordination," *ArXiv*, vol. abs/2109.12386, 2021.

[27] L. Zhao *et al.*, "Natural Language Processing (NLP) for Requirements Engineering: A Systematic Mapping Study," *ArXiv*, vol. abs/2004.01099, 2020.

[28] M. N. Zafar and J. Mohanta, "Methodology for path planning and optimization of mobile robots: A review," *Procedia computer science*, vol. 133, pp. 141–152, 2018.

[29] M. Nazari-Heris, B. Mohammadi-Ivatloo, and G. Gharehpetian, "A comprehensive review of heuristic optimization algorithms for optimal combined heat and power dispatch from economic and environmental perspectives," *Renewable and Sustainable Energy Reviews*, vol. 81, pp. 2128–2143, 2018.

[30] J. Andreu-Perez, F. Deligianni, D. Ravi, and G.-Z. Yang, "Artificial Intelligence and Robotics," *ArXiv*, vol. abs/1803.10813, 2018.

[31] A. V. Haridas, R. Marimuthu, and V. G. Sivakumar, “A critical review and analysis on techniques of speech recognition: The road ahead,” *International Journal of Knowledge-Based and Intelligent Engineering Systems*, vol. 22, no. 1, pp. 39–57, 2018.

[32] N. Oliveira, N. Sousa, and I. Praça, “A Search Engine for Scientific Publications: A Cybersecurity Case Study,” in *Distributed Computing and Artificial Intelligence, Volume 1: 18th International Conference*, K. Matsui, S. Omatu, T. Yigitcanlar, and S. R. González, Eds. Cham: Springer International Publishing, 2022, pp. 108–118.

[33] Y. Liu *et al.*, “RoBERTa: A Robustly Optimized BERT Pretraining Approach,” *ArXiv*, vol. abs/1907.11692, 2019.

[34] N. Oliveira, N. Sousa, J. Oliveira, and I. Praça, “Anomaly Detection in Cyber-Physical Systems: Reconstruction of a Prediction Error Feature Space,” *arXiv*, vol. abs/2112.14821, 2021.

[35] D. Zhang *et al.*, “The AI Index 2021 Annual Report,” *CoRR*, vol. abs/2103.06312, 2021.

[36] A. M. Turing, “I.— Computing Machinery and Intelligence,” *Mind*, vol. LIX, no. 236, pp. 433–460, 10 1950.

[37] M. Haenlein and A. Kaplan, “A Brief History of Artificial Intelligence: On the Past, Present, and Future of Artificial Intelligence,” *California Management Review*, vol. 61, no. 4, pp. 5–14, 2019.

[38] A. N. Whitehead and B. Russell, *Principia Mathematica.* Cambridge University Press, 1925–1927.

[39] A. Newell and H. Simon, “The logic theory machine–A complex information processing system,” *IRE Transactions on Information Theory*, vol. 2, no. 3, pp. 61–79, 1956.

[40] F. Rosenblatt, “The Perceptron: A Probabilistic Model for Information Storage and Organization in The Brain,” *Psychological Review*, pp. 65–386, 1958.

[41] Weizenbaum, Joseph, *Computer Power and Human Reason: From Judgment to Calculation.* USA: W. H. Freeman & Co., 1976.

[42] J. Lighthill, “Artificial Intelligence: A General Survey,” *Artificial Intelligence: a paper symposium*, 1973.

[43] R. M. Karp, *Reducibility among Combinatorial Problems.* Boston, MA: Springer US, 1972, pp. 85–103.

[44] A. Toosi, A. Bottino, B. Saboury, E. L. Siegel, and A. Rahmim, "A brief history of AI: how to prevent another winter (a critical review)," *PET clinics*, vol. 16 4, pp. 449–469, 2021.

[45] E. Feigenbaum, B. Buchanan, and J. Lederberg, "On generality and problem solving: A case study using the DENDRAL program," *Machine Intelligence*, vol. 6, 09 1970.

[46] L. Floridi, "AI and its New Winter: From Myths to Realities," *Philosophy and Technology*, vol. 33, no. 1, pp. 1–3, 2020.

[47] M. Campbell, A. J. Hoane Jr, and F.-h. Hsu, "Deep blue," *Artificial intelligence*, vol. 134, no. 1-2, pp. 57–83, 2002.

[48] D. Silver *et al.*, "Mastering the game of Go with deep neural networks and tree search," *Nature*, vol. 529, pp. 484–489, 01 2016.

[49] ——, "Mastering Chess and Shogi by Self-Play with a General Reinforcement Learning Algorithm," *ArXiv*, vol. abs/1712.01815, 2017.

[50] J. Schrittwieser *et al.*, "Mastering Atari, Go, chess and shogi by planning with a learned model," *Nature*, vol. 588, no. 7839, p. 604–609, Dec 2020.

[51] P. Maroufkhani, M.-L. Tseng, M. Iranmanesh, W. K. W. Ismail, and H. Khalid, "Big data analytics adoption: Determinants and performances among small to medium-sized enterprises," *International Journal of Information Management*, vol. 54, p. 102190, 2020.

[52] C. Zhang and Y. Lu, "Study on artificial intelligence: The state of the art and future prospects," *Journal of Industrial Information Integration*, vol. 23, p. 100224, 2021.

[53] O. Müller, M. Fay, and J. vom Brocke, "The Effect of Big Data and Analytics on Firm Performance: An Econometric Analysis Considering Industry Characteristics," *Journal of Management Information Systems*, vol. 35, no. 2, pp. 488–509, 2018.

[54] R.-X. Ding *et al.*, "Large-Scale decision-making: Characterization, taxonomy, challenges and future directions from an Artificial Intelligence and applications perspective," *Information Fusion*, vol. 59, pp. 84–102, 2020.

[55] M. Y. Shaheen, "Applications of Artificial Intelligence (AI) in healthcare: A review," *ScienceOpen Preprints*, 2021.

[56] M. R. Benabdelouahed and C. Dakouan, "The Use of Artificial Intelligence in Social Media: Opportunities and Perspectives," *Expert Journal of Marketing*, vol. 8, pp. 82–87, 2020.

[57] X. Guo, Z. Shen, Y. Zhang, and T. Wu, "Review on the Application of Artificial Intelligence in Smart Homes," *Smart Cities*, vol. 2, no. 3, pp. 402–420, 2019. [Online]. Available: https://www.mdpi.com/2624-6511/2/3/25

[58] T. Pinto, I. Praça, Z. Vale, and J. Silva, "Ensemble learning for electricity consumption forecasting in office buildings," *Neurocomputing*, vol. 423, pp. 747–755, 2021.

[59] I. Praça, C. Ramos, Z. Vale, and M. Cordeiro, "MASCEM: a multiagent system that simulates competitive electricity markets," *IEEE Intelligent Systems*, vol. 18, no. 6, pp. 54–60, 2003.

[60] J. Carneiro, N. Oliveira, N. Sousa, E. Maia, and I. Praça, "Machine Learning for Network-based Intrusion Detection Systems: an Analysis of the CIDDS-001 Dataset," in *International Symposium on Distributed Computing and Artificial Intelligence*. Springer, 2021, pp. 148–158.

[61] I. Macedo, S. Wanous, N. Oliveira, O. Sousa, and I. Praça, "A tool to support the investigation and visualization of cyber and/or physical incidents," in *World Conference on Information Systems and Technologies*. Springer, 2021, pp. 130–140.

[62] N. Benaich and I. Hogarth, "State of AI Report 2021," https://www.stateof.ai/, 2021, accessed: 06 January 2022.

[63] A. Vaswani *et al.*, "Attention Is All You Need," *ArXiv*, vol. abs/1706.03762, 2017.

[64] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "ImageNet: A large-scale hierarchical image database," in *2009 IEEE Conference on Computer Vision and Pattern Recognition*, 2009, pp. 248–255.

[65] A. Ramesh *et al.*, "Zero-Shot Text-to-Image Generation," *ArXiv*, vol. abs/2102.12092, 2021.

[66] A. Radford *et al.*, "Learning Transferable Visual Models From Natural Language Supervision," *ArXiv*, vol. abs/2103.00020, 2021.

[67] M. Chen *et al.*, "Evaluating Large Language Models Trained on Code," *ArXiv*, vol. abs/2107.03374, 2021.

[68] S. Ahmed, R. Mula, and S. S. Dhavala, "A Framework for Democratizing AI," *ArXiv*, vol. abs/2001.00818, 2020.

[69] K. Schwab, *The Fourth Industrial Revolution*. USA: Crown Publishing Group, 2017.

[70] L. Shen, "Former U.S. CTO: The 'Robot Apocalypse' Could Happen. Here's How You Stop It," https://fortune.com/2017/11/14/megan-smith-cto-robot-apocalypse-elon-musk/, Nov 2017, accessed: 04 January 2022.

[71] "Portugal AI Strategy Report," https://knowledge4policy.ec.europa.eu/ai-watch/portugal-ai-strategy-report_en, Jun 2019, accessed: 05 January 2022.

[72] M. Riedl, "AI Democratization in the Era of GPT-3," *The Gradient*, 2020.

[73] C. Matez, "Tech Giants Are Paying Huge Salaries for Scarce A.I. Talent," https://www.nytimes.com/2017/10/22/technology/artificial-intelligence-experts-salaries.html, Oct 2017, accessed: 05 January 2022.

[74] M. Gofman and Z. Jin, "Artificial Intelligence, Human Capital, and Innovation," *SSRN Electronic Journal*, 2019.

[75] J. B. Dina Bass, "Big Tech Swallows Most of the Hot AI Startups," https://www.bloomberg.com/news/articles/2020-03-16/big-tech-swallows-most-of-the-hot-ai-startups, Mar 2020, accessed: 05 January 2022.

[76] J. Hestness *et al.*, "Deep Learning Scaling is Predictable, Empirically," *ArXiv*, vol. abs/1712.00409, 2017.

[77] N. C. Thompson and S. Spanuth, "The Decline of Computers as a General Purpose Technology," *Commun. ACM*, vol. 64, no. 3, p. 64–72, feb 2021.

[78] J. Traub, J.-A. Quiané-Ruiz, Z. Kaoudi, and V. Markl, "Agora: Towards An Open Ecosystem for Democratizing Data Science & Artificial Intelligence," *ArXiv*, vol. abs/1909.03026, 2019.

[79] R. Shokri and V. Shmatikov, "Privacy-preserving deep learning," in *2015 53rd Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, 2015, pp. 909–910.

[80] L. Mauri and E. Damiani, "STRIDE-AI: An Approach to Identifying Vulnerabilities of Machine Learning Assets," in *2021 IEEE International Conference on Cyber Security and Resilience (CSR)*, 2021, pp. 147–154.

[81] A. Oseni, N. Moustafa, H. Janicke, P. Liu, Z. Tari, and A. Vasilakos, "Security and Privacy for Artificial Intelligence: Opportunities and Challenges," *ArXiv*, vol. abs/2102.04661, 2021.

[82] C. Wang, J. Chen, Y. Yang, X. Ma, and J. Liu, "Poisoning attacks and countermeasures in intelligent networks: Status quo and prospects," *Digital Communications and Networks*, 2021. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S235286482100050X

[83] B. Biggio *et al.*, "Evasion Attacks against Machine Learning at Test Time," *Lecture Notes in Computer Science*, p. 387–402, 2013. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-40994-3_25

[84] Y. Dong *et al.*, "There is Limited Correlation between Coverage and Robustness for Deep Neural Networks," *ArXiv*, vol. abs/1911.05904, 2019.

[85] B. Stanton and T. Jensen, "Trust and Artificial Intelligence," 2021-03-02 05:03:00 2021. [Online]. Available: https://tsapps.nist.gov/publication/get_pdf.cfm?pub_id=931087

[86] A. Adadi and M. Berrada, "Peeking Inside the Black-Box: A Survey on Explainable Artificial Intelligence (XAI)," *IEEE Access*, vol. 6, pp. 52 138–52 160, 2018.

[87] R. Blackman, "A Practical Guide to Building Ethical AI ," Oct 2020, accessed: 27 January 2022. [Online]. Available: https://hbr.org/2020/10/a-practical-guide-to-building-ethical-ai

[88] R. Schmelzer, "Towards A More Transparent AI," May 2020, accessed: 27 January 2022. [Online]. Available: https://www.forbes.com/sites/cognitiveworld/2020/05/23/towards-a-more-transparent-ai/?sh=65352afd3d93

[89] C. Szegedy *et al.*, "Intriguing properties of neural networks," *ArXiv*, vol. abs/1312.6199, 2014.

[90] I. J. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and Harnessing Adversarial Examples," *ArXiv*, vol. abs/1412.6572, 2015.

[91] N. Papernot, P. McDaniel, S. Jha, M. Fredrikson, Z. B. Celik, and A. Swami, "The Limitations of Deep Learning in Adversarial Settings," *arXiv*, vol. abs/1511.07528, 2015.

[92] A. Shafahi *et al.*, "Adversarial Training for Free!" *ArXiv*, vol. abs/1904.12843, 2019.

[93] E. Wong, L. Rice, and J. Z. Kolter, "Fast is better than free: Revisiting adversarial training," *ArXiv*, vol. abs/2001.03994, 2020.

[94] ATARC, "Machine Learning (ML) model transparency," 2020, accessed: 28 January 2022. [Online]. Available: https://atarc.org/project/information-technology-artificial-intelligence-machine-learning-ml-model-transparency/

[95] P. Linardatos, V. Papastefanopoulos, and S. Kotsiantis, "Explainable AI: A Review of Machine Learning Interpretability Methods," *Entropy*, vol. 23, no. 1, 2021.

[96] C. O'Neil, *Weapons of Math Destruction: How Big Data Increases Inequality and Threatens Democracy*. USA: Crown Publishing Group, 2016.

[97] M. Hardt, E. Price, and N. Srebro, "Equality of Opportunity in Supervised Learning," *ArXiv*, vol. abs/1610.02413, 2016.

[98] T. Bolukbasi, K.-W. Chang, J. Zou, V. Saligrama, and A. Kalai, "Man is to Computer Programmer as Woman is to Homemaker? Debiasing Word Embeddings," *ArXiv*, vol. abs/1607.06520, 2016.

[99] A. Koenecke *et al.*, "Racial disparities in automated speech recognition," *Proceedings of the National Academy of Sciences*, vol. 117, no. 14, pp. 7684–7689, 2020.

[100] M. Busuioc, "Accountable Artificial Intelligence: Holding Algorithms to Account," *Public Administration Review*, vol. 81, 08 2020.

[101] C. Bartneck, C. Lütge, A. Wagner, and S. Welsh, *Privacy Issues of AI*. Cham: Springer International Publishing, 2021, pp. 61–70.

[102] S. Russell and P. Norvig, *Artificial Intelligence: A Modern Approach*, 3rd ed. Prentice Hall, 2010.

[103] Z. Song, X. Yang, Z. Xu, and I. King, "Graph-based semi-supervised learning: A comprehensive review," *IEEE Transactions on Neural Networks and Learning Systems*, pp. 1–21, 2022.

[104] and European Union Agency for Cybersecurity, A. Malatras, and G. Dede, *AI cybersecurity challenges : threat landscape for artificial intelligence*. European Network and Information Security Agency, 2020.

[105] T. C. Lethbridge, "Low-Code Is Often High-Code, So We Must Design Low-Code Platforms to Enable Proper Software Engineering," in *Leveraging Applications of Formal*

Methods, Verification and Validation, T. Margaria and B. Steffen, Eds. Cham: Springer International Publishing, 2021, pp. 202–212.

[106] G. Keil, "Mapping the no-code AI landscape," May 2021, accessed: 08 January 2022. [Online]. Available: https://levity.ai/blog/no-code-ai-map

[107] F. Alexander, "Low-Code and No-Code: What's the Difference and When to Use What?" Jan 2021, accessed: 08 January 2022. [Online]. Available: https://www.outsystems.com/blog/posts/low-code-vs-no-code/

[108] K. Talesra and G. Nagaraja, "Low-code platform for application development," *International Journal of Applied Engineering Research*, vol. 16, no. 5, pp. 346–351, 2021.

[109] R. Silipo, "Low Code Data Science Is Not the Same as Automated Machine Learning," Dec 2021, accessed: 08 January 2022. [Online]. Available: https://www.knime.com/blog/low-code-analytics-platform

[110] M. R. Berthold *et al.*, ""'KNIME: The Konstanz Information Miner"," in *Data Analysis, Machine Learning and Applications*, C. Preisach, H. Burkhardt, L. Schmidt-Thieme, and R. Decker, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, pp. 319–326.

[111] R. S. Olson, N. Bartley, R. J. Urbanowicz, and J. H. Moore, "Evaluation of a Tree-based Pipeline Optimization Tool for Automating Data Science," in *Proceedings of the Genetic and Evolutionary Computation Conference 2016*, ser. GECCO '16. New York, NY, USA: ACM, 2016, pp. 485–492. [Online]. Available: http://doi.acm.org/10.1145/2908812.2908918

[112] M. Ali, *PyCaret: An open source, low-code machine learning library in Python*, July 2020, pyCaret version 2.3. [Online]. Available: https://www.pycaret.org

[113] "RapidMiner," 2021, accessed: 09 January 2022. [Online]. Available: https://rapidminer.com/

[114] "Obviously.ai," 2021, accessed: 09 January 2022. [Online]. Available: https://www.obviously.ai/

[115] "Levity.ai," 2021, accessed: 09 January 2022. [Online]. Available: https://levity.ai/

[116] "Peltarion AI," 2021, accessed: 09 January 2022. [Online]. Available: https://peltarion.com/

[117] "Teachable Machine," 2021, accessed: 09 January 2022. [Online]. Available: https://teachablemachine.withgoogle.com/

[118] "Google Cloud AutoML," 2021, accessed: 09 January 2022. [Online]. Available: https://cloud.google.com/automl

[119] C. V. K. Iyer *et al.*, "Trinity: A No-Code AI platform for complex spatial datasets," *ArXiv*, vol. abs/2106.11756, 2021.

[120] S. K. Karmaker, M. M. Hassan, M. J. Smith, L. Xu, C. Zhai, and K. Veeramachaneni, "AutoML to Date and Beyond: Challenges and Opportunities," *ACM Computing Surveys (CSUR)*, vol. 54, no. 8, pp. 1–36, 2021.

[121] H. J. Escalante, "Automated Machine Learning – a brief review at the end of the early years," *ArXiv*, vol. abs/2008.08516, 2020.

[122] G. Poulakis, "Unsupervised AutoML: a study on automated machine learning in the context of clustering," Master's thesis, University of Piraeus, 2020.

[123] Y.-F. Li, H. Wang, T. Wei, and W.-W. Tu, "Towards Automated Semi-Supervised Learning," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, no. 01, pp. 4237–4244, Jul. 2019.

[124] M. Feurer, A. Klein, K. Eggensperger, J. T. Springenberg, M. Blum, and F. Hutter, *Auto-sklearn: Efficient and Robust Automated Machine Learning*. Cham: Springer International Publishing, 2019, pp. 113–134.

[125] H. J. Escalante, M. Montes, and L. Sucar, "Particle Swarm Model Selection," *Journal of Machine Learning Research*, vol. 10, pp. 405–440, 01 2009.

[126] H. J. Escalante, M. Montes, and L. E. Sucar, "Particle Swarm Model Selection," *J. Mach. Learn. Res.*, vol. 10, p. 405–440, jun 2009.

[127] Y. Wang, H. Zhang, and G. Zhang, "cPSO-CNN: An efficient PSO-based algorithm for fine-tuning hyper-parameters of convolutional neural networks," *Swarm and Evolutionary Computation*, vol. 49, pp. 114–123, 2019.

[128] T. Elsken, J. H. Metzen, and F. Hutter, "Neural Architecture Search: A Survey," *ArXiv*, vol. abs/1808.05377, 2019.

[129] Z. Liu *et al.*, “Overview and unifying conceptualization of automated machine learning,” in *Proceedings of the Automating Data Science Workshop, Wurzburg, Germany*, vol. 20, 2019.

[130] C. Thornton, F. Hutter, H. H. Hoos, and K. Leyton-Brown, “Auto-WEKA: Combined Selection and Hyperparameter Optimization of Classification Algorithms,” in *Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ser. KDD ’13. New York, NY, USA: Association for Computing Machinery, 2013.

[131] D. Gorissen, T. Dhaene, and F. D. Turck, “Evolutionary Model Type Selection for Global Surrogate Modeling,” *J. Mach. Learn. Res.*, vol. 10, p. 2039–2078, dec 2009.

[132] R. Vilalta and Y. Drissi, “A Perspective View And Survey Of Meta-Learning,” *Artificial Intelligence Review*, vol. 18, 09 2001.

[133] E. Real *et al.*, “Large-Scale Evolution of Image Classifiers,” *ArXiv*, vol. abs/1703.01041, 2017.

[134] B. Zoph and Q. V. Le, “Neural Architecture Search with Reinforcement Learning,” *ArXiv*, vol. abs/1611.01578, 2017.

[135] I. Guyon *et al.*, *Analysis of the AutoML Challenge Series 2015–2018*. Cham: Springer International Publishing, 2019, pp. 177–219. [Online]. Available: https://doi.org/10.1007/978-3-030-05318-5_10

[136] F. Candelon, M. Courtaux, and G. Nahas, “You can now put A.I. tools in the hands of all your employees. But should you?” https://fortune.com/2022/06/03/artificial-intelligence-ai-democratization-no-low-code, June 2022, accessed: 18 June 2022.

[137] V. Borisov, T. Leemann, K. Sessler, J. Haug, M. Pawelczyk, and G. Kasneci, “Deep Neural Networks and Tabular Data: A Survey,” *ArXiv*, vol. abs/2110.01889, 2021.

[138] M. Smolaks, “The current state of AIś three main learning paradigms, and why they need to change,” https://aibusiness.com/document.asp?doc_id=761214, February 2020, accessed: 18 June 2022.

[139] V. Borisov, T. Leemann, K. Sessler, J. Haug, M. Pawelczyk, and G. Kasneci, “Deep neural networks and tabular data: A survey,” *ArXiv*, vol. abs/2110.01889, 2021.

[140] P. Kruchten, “The 4+1 View Model of architecture,” *IEEE Software*, vol. 12, no. 6, pp. 42–50, 1995.

[141] S. Brown, “Software Architecture for Developers: Volume 2 ,” 2015.

[142] M. Fowler, *UML Distilled: A Brief Guide to the Standard Object Modeling Language*, 3rd ed., ser. Object Technology Series. Boston, MA: Addison-Wesley, 2003. [Online]. Available: https://www.safaribooksonline.com/library/view/uml-distilled-a/0321193687/

[143] R. Martin, J. Rabaey, A. Chandrakasan, J. Newkirk, B. Nikolić, and R. Koss, *Agile Software Development: Principles, Patterns, and Practices*, ser. Alan Apt series. Pearson Education, 2003. [Online]. Available: https://books.google.pt/books?id=0HYhAQAAIAAJ

[144] G. Van Rossum and F. L. Drake Jr, *Python reference manual*. Centrum voor Wiskunde en Informatica Amsterdam, 1995.

[145] “Vaadin: The modern web application platform for Java,” https://vaadin.com/, 2022, accessed: 22 June 2022.

[146] K. Arnold, J. Gosling, and D. Holmes, *The Java programming language*. Addison Wesley Professional, 2005.

[147] K. Chodorow and M. Dirolf, *MongoDB: The Definitive Guide*, 1st ed. O’Reilly Media, Inc., 2010.

[148] A. Noor, Mohamedmohiy, A. Elsherbiny, and N. Mostafa, “Website Phishing Dataset,” https://www.kaggle.com/datasets/ahmednour/website-phishing-data-set, May 2019, accessed: 05 May 2022.

[149] N. Abdelhamid, A. Ayesh, and F. Thabtah, “Phishing detection based Associative Classification data mining,” *Expert Systems with Applications*, vol. 41, no. 13, pp. 5948–5959, 2014. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0957417414001481

[150] M. Varli, “Laptop Price,” https://www.kaggle.com/datasets/muhammetvarl/laptop-price, May 2020, accessed: 05 May 2022.

[151] L. Ruslan, “EDA, XGBoost and RF Laptop Price,” https://www.kaggle.com/code/leoruslan/eda-xgboost-rf-laptop-price, May 2021, accessed: 05 May 2022.

[152] M. Abadi *et al.*, “TensorFlow: Large-scale machine learning on heterogeneous systems,” 2015, software available from tensorflow.org. [Online]. Available: https://www.tensorflow.org/

[153] A. Paszke *et al.*, “PyTorch: An Imperative Style, High-Performance Deep Learning Library,” in *Advances in Neural Information Processing Systems 32*. Curran Associates, Inc., 2019, pp. 8024–8035. [Online]. Available: http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf

[154] M. T. Ribeiro, S. Singh, and C. Guestrin, “”Why Should I Trust You?”: Explaining the Predictions of Any Classifier,” in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, August 13-17, 2016*, 2016, pp. 1135–1144.

[155] S. M. Lundberg and S.-I. Lee, “A unified approach to interpreting model predictions,” in *Advances in Neural Information Processing Systems 30*, I. Guyon *et al.*, Eds. Curran Associates, Inc., 2017, pp. 4765–4774. [Online]. Available: http://papers.nips.cc/paper/7062-a-unified-approach-to-interpreting-model-predictions.pdf

[156] J. Vitorino, N. Oliveira, and I. Praça, “Adaptative perturbation patterns: Realistic adversarial learning for robust intrusion detection,” *Future Internet*, vol. 14, no. 4, 2022. [Online]. Available: https://www.mdpi.com/1999-5903/14/4/108

Appendix A

Generated Code

In the normal operation of ALMA, when a ML pipeline is configured, it must then be converted into executable code to obtain the corresponding evaluation results. In that sense, the proposed solution works by generating code dynamically, supporting a large number of alternatives. Furthermore, by implementing a different code generation mechanism it is possible to perform a model-to-text translation to a different programming language. For the current iteration, a Python translation module was implemented as this language is largely used by the AI community [16]. In this appendix, in order to provide provide greater details about the inner workings of ALMA, the code generated from the ML pipeline described in Chapter 5 is presented and discussed.

A.1 Prologue

The first lines of the generated Python file contain imports of objects and methods that are used for the remaining implementation. Some of them are common to multiple pipelines, 1-5, while the others, are dynamically generated according to user selections, 7-15. These lines are presented in Listing A.1.

```
1 import joblib
```

```
import pandas as pd

from sklearn.pipeline import Pipeline
from sklearn.compose import ColumnTransformer

from sklearn.model_selection import train_test_split
from sklearn.impute import SimpleImputer
from sklearn.preprocessing import OneHotEncoder
from sklearn.preprocessing import MinMaxScaler
from sklearn.neighbors import KNeighborsRegressor
from sklearn.model_selection import GridSearchCV
from sklearn.metrics import mean_squared_error
from sklearn.metrics import mean_absolute_error
from sklearn.metrics import r2_score
```

Listing A.1: Import statements

A.2 Data Ingestion

For the provided example, the laptop price forecasting dataset was chosen as benchmark to test the performance of regression algorithms. This dataset is first loaded in line 19, then, in lines 20 and 21 the label is separated from the remaining features and, finally, in line 23 the train and test set are split according to the percentage specified by the user when registering the dataset into the system. Listing A.2 provides an overview of the data ingestion code.

```
18 # read data and split train and test data
19 data = pd.read_csv("res/datasets/laptop_price.csv")
20 X = data.drop("price_euros", axis=1)
21 y = data["price_euros"]
22
23 X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3)
```

Listing A.2: Data ingestion code

A.3 Preprocessing

For the preprocessing step, two distinct transformers were defined. The first, ”transformer_0” (lines 29-38), applies mode imputation and one hot encoding to the dataset’s categorical features, while the second, ”transformer1” (lines 40-48), applies mean imputation to the numerical features. These are then appended to form the preprocessing pipeline, ”preprocessor” (lines 51-53). Listing A.3 provides an overview of the data preprocessing code.

```
26 # define preprocessing pipeline
27 tr_list = []
28 
29 transformer_0 = Pipeline(
30     steps=[
31         ("mode_imputer", SimpleImputer(strategy='most_frequent')),
32         ("one_hot_encoder", OneHotEncoder(handle_unknown='ignore')),
33         ]
34 )
35 
36 tr_list.append(
37     ("transformer_0", transformer_0, ['company', 'typename', 'opsys'])
38 )
39 
40 transformer_1 = Pipeline(
41     steps=[
42         ("mean_imputer", SimpleImputer(strategy='mean')),
43         ]
44 )
45 
46 tr_list.append(
47     ("transformer_1", transformer_1, ['inches', 'ram', 'weight', 'width', 'height', 'touchscreen', 'ips', 'full_hd', '4k_ultra_hd', 'quad_hd', 'ghz', 'cpu_intel', 'cpu_amd', 'gpu_intel', 'gpu_nvidia', 'gpu_amd', 'memory_ssd', 'memory_hdd', 'memory_flash', 'ssd_value', 'hdd_value', 'flash_value'])
48 )
49 
50 
51 preprocessor = ColumnTransformer(
52     transformers=tr_list, remainder="drop"
53 )
```

Listing A.3: Data preprocessing code

A.4 Modeling

In the modelling step, the final pipeline is created, appending the previously created preprocessing pipeline, min-max scaler and the KNN regression algorithm (lines 55-61). The modeling code is presented in Listing A.4.

```
55 pipeline = Pipeline(
    steps=[
        ("preprocessor", preprocessor),
        ("min_max_normalization", MinMaxScaler()),
        ("k_nearest_neighbors_regressor", KNeighborsRegressor()),
        ]
)
```

Listing A.4: Modeling code

A.5 Hyperparameter Tuning

For hyperparameter tuning, the parameter grid is defined in lines 64-68, being used to optimize the final regression pipeline with grid search k-fold cross validation in lines 69-73. Listing A.5 provides an overview of the hyperparameter tuning code.

```
63 # perform hyperparameter tuning
64 parameters = {
65     "k_nearest_neighbors_regressor__n_neighbors": [4, 8, 16],
66     "k_nearest_neighbors_regressor__weights": ['uniform', 'distance'],
67     "k_nearest_neighbors_regressor__metric": ['euclidean', 'manhattan'],
68     }
69 cv = 5
70
71 clf = GridSearchCV(pipeline, parameters, cv=cv, scoring="
       neg_mean_absolute_error")
72
73 clf.fit(X_train, y_train)
```

Listing A.5: Hyperparameter tuning code

A.6 Evaluation

To obtain the evaluation results, predictions are made for the testing set in line 76. These same predictions are used in the subsequent lines, 78-83, to compute all evaluation metrics selected by the user. The evaluation code is presented in Listing A.6.

```
75 # determine results
76 y_pred = clf.predict(X_test)
77
78 mae = mean_absolute_error(y_test, y_pred)
79 print("mae: " + str(mae))
80 r2 = r2_score(y_test, y_pred)
```

```python
print("r2: " + str(r2))
mse = mean_squared_error(y_test, y_pred)
print("mse: " + str(mse))
```

Listing A.6: Evaluation code

A.7 Epilogue

In the final step of the generated Python file, the trained ML pipeline is exported as joblib file so that it can be downloaded by the user through ALMA's GUI and used for production purposes. The pipeline export code is presented in Listing A.7.

```python
# export ML pipeline
joblib.dump(clf, "res/serialized/laptop_price_7.joblib")
```

Listing A.7: Pipeline export code