PUCRS

ESCOLA POLITÉCNICA
PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO
MESTRADO EM CIÊNCIA DA COMPUTAÇÃO

GUILHERME DALL'AGNOL DECONTO

PATTERN DETECTION STRATEGY APPLIED TO CRIME INVESTIGATION IN IOT ENVIRONMENTS

Porto Alegre
2024

PÓS-GRADUAÇÃO - *STRICTO SENSU*

Pontifícia Universidade Católica
do Rio Grande do Sul

PONTIFICAL CATHOLIC UNIVERSITY OF RIO GRANDE DO SUL
SCHOOL OF TECHNOLOGY
COMPUTER SCIENCE GRADUATE PROGRAM

PATTERN DETECTION STRATEGY APPLIED TO CRIME INVESTIGATION IN IOT ENVIRONMENTS

GUILHERME DALL'AGNOL DECONTO

Master Thesis submitted to the Pontifical Catholic University of Rio Grande do Sul in partial fulfillment of the requirements for the degree of Master in Computer Science.

Advisor: Prof. Avelino Francisco Zorzo

Porto Alegre
2024

I would like to express my heartfelt gratitude to my parents and sister for their consistent support and encouragement, which have been fundamental to my accomplishments. I am truly grateful for their unwavering belief in my aspirations and their constant presence throughout my journey. Thank you sincerely.

Furthermore, I am extremely grateful to my advisor Avelino Francisco Zorzo for his invaluable guidance, unwavering support, and expert mentorship throughout the course of this thesis journey. His insightful feedback, encouragement, and dedication to excellence have been instrumental in shaping my work and helping me navigate the challenges. I am truly grateful for his commitment to my academic and personal growth. I sincerely thank you for being an exceptional mentor.

“You can’t connect the dots looking forward;
you can only connect them looking backwards.
So you have to trust that the dots will some-
how connect in your future.”
(Steve Jobs)

ACKNOWLEDGMENTS

This research was achieved in cooperation with Hewlett Packard Brasil LTDA. using incentives of Brazilian Informatics Law (Law nº 8.2.48 of 1991).

ESTRATÉGIA DE DETECÇÃO DE PADRÕES APLICADA À INVESTIGAÇÃO DE CRIMES EM AMBIENTES IOT

RESUMO

A adoção da Internet das Coisas (IoT) trouxe muitas vantagens, mas também apresenta desafios para o campo da Perícia Digital. A heterogeneidade dos dados afeta diretamente o processo investigativo em cenários que envolvem aplicações de IoT. Por meio da análise de um conjunto de dados heterogêneo e abrangente coletado de dispositivos IoT, este estudo analisa o uso de algoritmos de aprendizado de máquina para detectar padrões específicos e estimar o número de pessoas em ambientes físicos que envolvem dispositivos IoT, com o objetivo de auxiliar em investigações criminais. Os resultados destacam a capacidade dos modelos de Aprendizado de Máquina em identificar padrões relevantes e fornecer informações valiosas para investigações em ambientes de IoT, como casas inteligentes, escritórios inteligentes e edifícios inteligentes. Essas descobertas contribuem para o avanço da Perícia Digital e demonstram o potencial de abordagens baseadas em aprendizado de máquina na análise de dados de dispositivos IoT em contextos forenses.

Palavras-Chave: Internet das Coisas, Perícia Digital, Aprendizado de Máquina.

PATTERN DETECTION STRATEGY APPLIED TO CRIME INVESTIGATION IN IOT ENVIRONMENTS

ABSTRACT

The adoption of the Internet of Things (IoT) has brought many advantages, but it also presents challenges for the field of Digital Forensics. The heterogeneity of the data directly affects the investigative process in scenarios involving IoT applications. Through the analysis of a comprehensive and heterogeneous dataset collected from IoT devices, this study analyzes the use of machine learning algorithms to detect specific patterns to estimate the number of people in physical environments involving IoT devices, with the aim of helping in crime investigations. In this work, we discuss the use of Machine Learning approaches to enhance criminal investigations based on data collected from IoT environments. The experimental evaluation not only showcases the potential enhancement of Digital Forensics through the utilization of IoT data but also serves to emphasize the effectiveness of machine learning-based approaches in these environments.

Keywords: Internet of Things, Digital Forensics, Machine Learning.

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

IoT. – Internet of Things

DF. – Digital Forensic

ML. – Machine Learning

NBT. – Next Best Thing

NIJ. – National Institute of Justice

DSP. – Data Stream Processing

RNN. – Recurrent Neural Network

LSTM. – Long-Short-Term Memory

GRU. – Gated Recurrent Unit

SVM. – Support Vector Machine

HVAC. – Heating, ventilation, and air conditioning

IQR. – Interquartile Range

Q3. – Third Quartile

Q1. – First Quartile

SMOTE. – Synthetic Minority Oversampling Technique

CONTENTS

1. INTRODUCTION

1.1 Overview

The term Internet of Things (IoT) has been increasingly used to refer to a variety of devices and equipment with low computational power that are connected to the Internet. Due to the embedded sensors in these devices, it is possible to collect data from the environment in which they are placed and transmit them to the network, allowing them to be monitored and controlled remotely. Data collected from sensors and actuators in IoT environments can be used to remotely monitor and control them. Furthermore, IoT devices offer significant advantages in various sectors, including industry, healthcare, transportation, and agriculture, allowing process optimization and improving product quality [25].

The intersection between IoT and Digital Forensic (DF) presents a complex and challenging dynamic. As the IoT continues to expand and diversify, an increasing number of connected devices generate a massive amount of data, becoming a valuable source of evidence for forensic investigations. However, the heterogeneous nature and large volume of data from these devices require specific and customized approaches. The convergence of DF with Machine Learning (ML) algorithms offers unique opportunities to improve investigation and forensic analysis in IoT environments, allowing analysis of patterns, detection of suspicious events, and a deeper understanding of the investigated scenarios [4, 58].

Thus, DF is a science field dedicated to the collection, analysis, and preservation of digital data to be used as evidence in a court of law [26]. It is an area of growing importance as the world becomes increasingly digitized. DF involves the retrieval, analysis, and interpretation of data stored on electronic devices such as computers, mobile phones, and IoT devices, among others.

Although there are numerous publications focused on the context of DF devices of IoT, there is still a lack of tools designed and developed to handle the large volume and heterogeneity of the data generated by these devices [26]. Furthermore, these tools can allow for an accurate assessment of the consequences of the loss or destruction of a specific IoT device. Therefore, it is necessary to develop specific tools to aid in conducting investigations in this scenario, as current tools are not able to meet this demand [29, 28]. This is one of the main challenges in the field, as the quality of the investigations directly impacts the success of legal proceedings.

Oriwoh et al. [42] proposed Next Best Thing (NBT) screening model considering the challenges of forensics during data analysis. This model aims to help researchers

identify potential sources of evidence. According to the authors, IoT devices along with any evidence stored within them can become inaccessible or compromised due to various incidents such as destruction, theft, or tampering. Investigators responsible for researching the IoT ecosystem must be able to identify additional elements beyond those that make up the subject of the analysis to determine whether these elements may provide meaningful insights. This recognition capability is crucial for researchers to gain a better understanding of the functioning of the ecosystem studied and to conduct a more detailed analysis.

The field of DF still lacks tools that can analyze the entire context of a smart environment more accurately. Analyzing the collected data is crucial to the efficiency of DF work, but the tools available are currently inadequate for this purpose [29, 28, 44]. This analysis is essential for DF efficiency and therefore solutions must be sought for this issue.

Detecting patterns in IoT data can be instrumental in uncovering criminal activities within smart environments. Such data can provide valuable information on the *modus operandi* and potential culprits. This research is part of our effort to identify patterns in physical environments, ultimately contributing to the improvement of crime detection in physical smart environments. Specifically, this study aims to **evaluate and discuss the adoption of ML models to accurately detect specific patterns of a given environment using data collected from IoT devices in the domain of DF to aid crime investigations in physical environments.**

To achieve the objectives outlined in this research, a publicly available IoT multisensor room dataset was used for comprehensive data analysis. The use of supervised machine learning techniques, including Random Forest, Decision Tree, and XGBoost, is a crucial aspect of the approach, facilitating accurate detection of room occupancy. The methodology involves the deployment of these machine learning models, and comparisons are conducted to assess their efficacy in this context. The collected results were meticulously evaluated, and the discussion shed light on the insights gained, providing a deeper understanding of the performance and implications of the techniques used.

1.2 Project Objective

Identifying patterns in IoT data can help uncover crimes committed in smart environments. This information can provide clues about how the crime was perpetrated and who may be involved. Here are some examples. In a smart office environment equipped with IoT sensors, access to various areas is logged by analyzing patterns in the access logs, hence unusual behavior can be identified. For example, if an employee typically accesses his office during regular working hours but suddenly starts entering the building late at night, it could be flagged as an anomaly. This pattern might suggest unauthorized access or even potential suspicious activity.

Another scenario could be in a smart neighborhood, where environmental sensors monitor various factors such as noise levels, air quality, and temperature. Unusual patterns, such as a significant increase in noise or a sudden change in air quality, could be indicative of a disturbance or an event that requires attention. By analyzing the data and identifying patterns, authorities can gain insight into potential criminal activities, such as a loud party or environmental hazards that need attention.

Imagine a state-of-the-art university campus where cutting-edge IoT devices seamlessly integrate into every part of its infrastructure. Among these innovations, smart classrooms stand out, meticulously equipped with a variety of sensors, high-resolution cameras, and interconnected devices designed to serve educational and security functions. One day, late in the evening, a tragedy unfolds as a highly regarded professor, renowned for groundbreaking research, falls victim to an assassination during a private lecture within one of these classrooms equipped with IoT devices.

As the scenario unravels, the IoT ecosystem within the smart classroom becomes a crucial player in the investigation. The access control system, designed to monitor entry and exit, exposes an unauthorized breach during an unusual time, highlighting the critical time frame of the crime. Meanwhile, security cameras, with facial recognition ML techniques, capture the image of the assailant, transforming the classroom into a virtual witness stand.

At the same time, audio sensors record ambient sounds, capturing the echo of the crime itself. This auditory evidence becomes a key element in reconstructing the sequence of events and understanding the dynamics at play during the incident. In addition, occupancy sensors and motion detectors can reveal anomalies in the room’s activity. These data points, when analyzed, create a vivid timeline of crime, helping investigators piece together the puzzle of the assassination.

In this scenario, the smart classroom transforms from a space of education to a critical node in data-driven investigation. The fusion of technology and security not only el-

evates the capabilities of law enforcement but also underscores the ethical considerations surrounding the use of advanced surveillance in pursuit of evidence, facts, and justice.

Therefore, the objective of this project is to **evaluate and discuss the adoption of ML models to accurately detect specific patterns in a given environment using data collected from IoT devices in the domain of DF to aid crime investigations in physical environments.** An IoT multisensor room has been implemented for data collection and ML techniques have been used to allow for the detection of room occupancy. The results have been evaluated and discussed.

1.3 Thesis Outline

This section summarizes the layout and content of the thesis chapters.

- **Chapter 2:** Explores fundamental concepts and reviews the relevant literature in the field.
- **Chapter 3:** Provides a comprehensive overview of DF and ML.
- **Chapter 4:** Introduces the proposed model, outlines the study's objectives, and delineates the research questions.
- **Chapter 5:** Details the experiment, presents findings, addresses research questions, and discusses study limitations.
- **Chapter 6:** Serves as the conclusion, discussing the achievement of initial objectives and offering reflections on future work.

2. RELATED WORK

The literature has extensively examined diverse methodologies aimed at tackling the intricate challenge of pattern detection, applicable to environments both indoors and outdoors. The complexity of this issue arises from the myriad variables that require careful consideration, including but not limited to user privacy, desired accuracy levels, and system costs. This chapter provides a comprehensive survey of these methods, presenting a panoramic view of the current state-of-the-art within the broader context of this work. This analysis serves as a foundational reference point for the project, influencing decisions that range from the selection of methods for the evaluation of the results to the initiation of a thorough feasibility analysis.

2.1 Digital Forensics

Several authors emphasize the urgency of developing innovative approaches to address the complex DF scenario in IoT. The exponential growth in the number of connected devices has generated a substantial volume of data, necessitating specialized strategies for the collection, analysis, and interpretation of this information. The challenge lies not only in the quantity of data, but also in the diversity and interdependence of these devices, making the adaptation of forensic techniques essential to ensure effectiveness in the investigation and preservation of the integrity of digital evidence. In this dynamic context, the continuous evolution of DF practices in IoT is crucial to keep pace and overcome the emerging challenges in this field [38, 21, 20, 47, 5, 57].

Adedayo [1] highlights the complexity associated with one of the main challenges facing DF: the considerable volume of data that requires analysis. The authors underscore the imperative need to reassess our approaches in the field of DF, recognizing the constant evolution of the technological landscape in recent years. The authors' paper introduced and discussed several solutions and techniques to enhance better collection, analysis, preservation, and presentation in the face of the challenges of DF.

In the same line of thinking, Shams Zawoad and Ragib Hasan [59] highlight that the era of Big Data presents numerous opportunities across various fields, but simultaneously introduces several challenges for the field of DF. According to the authors, existing tools and infrastructures do not meet the expected response time when dealing with large datasets. Forensics investigators encounter difficulties in identifying crucial pieces of evidence within large datasets, and collecting and analyzing said evidence becomes a challenging task. Furthermore, they emphasize that this challenge is significantly exacerbated by the growth in both the variety and quantity of IoT devices, driving a substantial increase in the volume of digital data.

To address these challenges and improve the processing of digital evidence, NIJ initiated research collaborations with Purdue and Rhode Island Universities. In particular, the NIJ journal [41] refers to the work of Marco Vega at Rhode Island University [55]. Vega's research resulted in the creation of a groundbreaking software called *DeepPatrol*, leveraging advanced ML and Deep Learning algorithms. The software is designed to assist investigators in the examination of child sexual abuse materials, highlighting the potential of cutting-edge technology to play a pivotal role in forensic endeavors.

To face the challenges associated with the volume of data, heterogeneous data, and the cognitive load on investigators to understand the relationships between artifacts, Mohammed *et al.* [38] introduced a framework designed to address these issues. Their framework emphasizes that data analysis should be entrusted to a set of Artificial Intelligence algorithms. However, they leave open the specific algorithms to be employed, as further studies are warranted to evaluate the efficacy of these algorithms and their ability to identify pertinent information crucial to investigations.

Yaacoub *et al.* shed light on the current landscape of cyber attacks on IoT and the associated challenges. The authors not only draw conclusions and advocate for more research in the realm of smart automated evidence detection tools, incorporating ML techniques, but also underscore the need for increased investment in the field of forensics. They emphasize that a greater commitment to resources is essential for researchers to effectively address current demands and challenges in the field.

2.2 Machine Learning

Machine learning techniques have attracted considerable attention from researchers, showcasing a diverse range of applications. Although some studies may not be explicitly tailored to address the challenges of DF, there is a growing recognition that certain ML applications hold significant promise within the forensic domain. These applications have the potential to improve data analysis, pattern recognition, and anomaly detection, contributing to the efficiency and efficacy of DF investigations. As technology continues to evolve, the integration of ML methodologies into the forensic toolkit becomes increasingly essential, paving the way for innovative approaches to address the evolving challenges of the digital landscape.

Consistent with the potential of ML, Dey *et al.* [17] devised a way to infer the number of people in a specific room of a smart environment using ML. In their work, they constructed a model that utilizes temperature, CO_2 levels, air volume, and air conditioning temperature data for their analysis. The authors successfully predicted real-time room occupancy in a building at the University of Washington with up to 95% precision. To enhance data analysis, they also explored the use of parallel processing techniques for

data analysis and normalization of the collected data. The use of data stream processing (DSP) techniques proved to be highly beneficial due to the abundance of data ingested by the application, enabling real-time data processing.

Also, Jiho *et al.* [43] proposed an activity recognition system for smart homes based on a variant of the recurrent neural network (RNN). This system outperforms other variants of classic RNN, such as Long-Short-Term Memory (LSTM) and Gated Recurrent Unit (GRU). Although activity monitoring in smart homes is not a new topic, many applications have been developed and are available to the public; however, these tools focus only on individual systems, not the entire environment [14]. As observed by the authors, the proposed system manages to eliminate less relevant data for analysis, thus improving performance and significantly outperforming related models, achieving results of up to 90% accuracy in the explored dataset.

These studies highlight significant advances in the application of ML in smart environments. Dey *et al.* demonstrated the effectiveness of the model in predicting room occupancy, while Jiho *et al.* developed a robust activity recognition system for smart homes. Both works underscore the importance of advanced data processing techniques and ML approaches to handle the complexity and volume of data generated by smart environments.

However, it is important to note that these studies focus more on applications within specific environments, such as university buildings and smart homes. On the contrary, this research aims to explore the detection of specific patterns to estimate the number of people in physical environments in the highly heterogeneous infrastructure of the IoT, within a broader context of criminal investigations in physical environments involving IoT devices. Therefore, while these studies are relevant to the fields of IoT and ML, they do not directly address the scope and objectives of this research.

3. DIGITAL FORENSICS AND MACHINE LEARNING STUDY

In the previous chapter, we explored the applications of ML across various domains, showcasing its effectiveness in detecting various phenomena and occurrences. Expanding on this groundwork, the current chapter provides an in-depth analysis of the exploration of ML and DF in the research context, outlining the essential frameworks and algorithms that serve as the bedrock for our investigation.

3.1 Digital Forensics Overview

To ensure the admissibility of credible digital evidence in a court of law, it becomes imperative to employ scientifically validated DF investigation techniques that support a suspected security incident. Although conventional DF methods have traditionally focused on computer desktops and servers, the evolving landscape of digital technologies requires the extension of these practices to encompass a broader spectrum.

Recent advances in digital media and platforms have underscored the growing necessity for applying DF investigation techniques to diverse subdomains. This expanded scope encompasses not only computer desktops and servers, but also the examination of mobile devices, databases, networks, cloud-based platforms, and the vast realm of the IoT. This strategic broadening of focus acknowledges the pervasive nature of digital evidence in a multitude of interconnected and technologically sophisticated domains.

By recognizing the need to adapt and apply DF methodologies beyond traditional computing environments, practitioners can more fully address the complexities presented by modern technology landscapes. This adaptability ensures a more robust and nuanced approach to the investigation process, enabling the extraction and validation of digital evidence from a variety of sources critical to legal proceedings.

To support forensic investigators in their navigation of investigations within these specific subdomains, academic researchers have endeavored to formulate various investigative methodologies and frameworks. Therefore, the framework used to structure the research in this study was conceived by Guilherme Schneider in [50].

Schneider, G., proposes an investigative model that organizes the investigative process into three distinct phases: Planning, Execution, and Conclusion. The planning phase encompasses preinvestigation steps and the identification of potential sources of evidence, with the aim of developing an action plan to be implemented in the execution phase. The execution phase, in turn, involves the acquisition, preservation, and examination of information that will be subsequently analyzed. In the conclusion phase, the analysis stage aims to validate and correlate the information obtained to characterize the

evidence and compile the final report. After completion of the report, an evaluation of all the documentation generated during the process is performed.

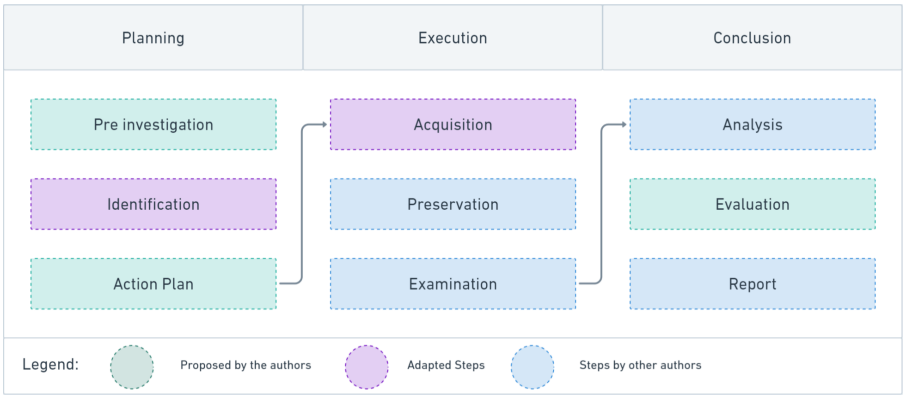

Figure 3.1 – Foca-IoT - Investigative Model Phases [50]

The investigative model introduced by Schneider, G., is presented in Figure 3.1, illustrating the three stages, their interconnections and the steps involved in each phase. To facilitate a clear understanding of where this study stands within an investigative process guided by a specific model, Figure 3.2 emphasizes the aligned stages of this work, which are analysis and evaluation.

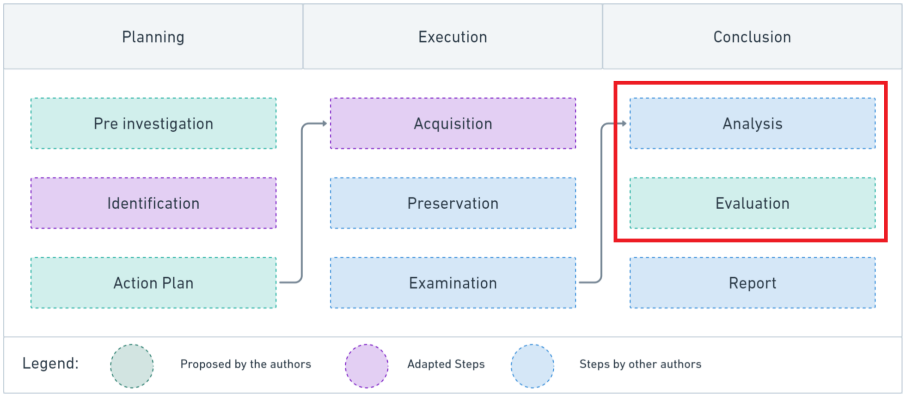

Figure 3.2 – Foca-IoT - Investigative Model Phases [50]

According to Schneider [50], the analysis phase aims to relate and reconstruct the crime scene or events relevant to the investigation. In addition, it determines the significance of the collected evidence and reconstructs fragmented data to formulate concrete conclusions. The appropriate presentation of the results of this phase is essential. The evaluation process assesses the sufficiency of the evidence derived from the analysis to draw conclusions about the case.

This work falls within the intersection of these two forensic phases, as analysis, conducted using ML models, can yield artifacts and vital evidence during an investigation, aiding in its progression. The results of this phase, such as the identified patterns, should be presented in a manner suitable for discussion and analysis. Therefore, the current research is guided by the work of Schneider.

3.2 Machine Learning Overview

Machine learning, a branch of artificial intelligence, revolutionizes the way computer systems perform tasks by embracing a variety of algorithms and statistical methods. Unlike traditional programming paradigms that follow explicit sequential instructions, ML empowers systems to perform tasks by leveraging vast datasets. These datasets serve as repositories, allowing machines to learn patterns, correlations, and trends autonomously, enabling them to make informed decisions and predictions without being explicitly programmed for each step.

The essence of ML lies in its ability to generalize from data, facilitating the development of models that can adapt and evolve based on new and unseen information. This approach enables computers to exhibit a form of intelligence, making decisions or predictions in real-time scenarios without requiring predefined step-by-step commands [51, 3].

Tom M. Mitchell provided a formal and widely quoted definition of an ML algorithm in [37]: *'A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P if its performance on tasks in T, measured by P, improves with experience E'*. These algorithms use statistical approaches to uncover relationships and patterns in training data, enabling their application across different domains. When discussing ML, a few important approaches need to be considered: supervised, semi-supervised, unsupervised learning, and reinforcement learning. [51, 32].

- **Supervised Machine Learning** is an algorithm that uses a dataset with labels to teach itself. Each training example consists of an input, also referred to as features, and a desired output, also known as a label. The goal of supervised learning is to teach the model how to correctly map the inputs to the desired outputs. During training, the algorithm analyzes patterns in training data and learns to make accurate predictions on new data based on acquired knowledge, enabling the model to generalize and make predictions on unexplored data [24, 7, 9].

- **Unsupervised Machine Learning** is an algorithm that is trained using a dataset that does not have any class information or predefined labels associated with it. Unlike supervised learning, there are no desired outputs present during training. The goal of this type of learning is to explore the data for intrinsic patterns, structures, or clusters, to gain insight and knowledge about the data. Unsupervised learning is useful for tasks such as data segmentation, anomaly detection, or dimensionality reduction, where the emphasis is on discovering hidden information in the data without prior knowledge of the desired results [24, 7, 56].

- **Semi-supervised learning** is a hybrid approach that combines elements of supervised and unsupervised learning. In this paradigm, the algorithm is trained on a dataset that contains both labeled and unlabeled examples. Although some instances in the dataset have associated labels, a substantial portion of the dataset remains unlabeled. This unique characteristic allows the algorithm to learn from both the explicit guidance provided by the labeled data and the inherent patterns within the unlabeled data. Semi-supervised learning is particularly advantageous in scenarios where obtaining labeled data for training is resource intensive or impractical. Using unlabeled data, the algorithm can improve its understanding of the underlying structures and relationships within the dataset, ultimately improving its performance on tasks that involve predicting labels for new unseen data [60, 11].
- **Reinforcement learning** on the other hand, is a distinct paradigm in which an agent learns to make decisions by interacting with an environment. The agent receives feedback in the form of rewards or penalties based on the actions it takes, enabling it to learn optimal strategies over time. This approach is well-suited for tasks where an agent must navigate a dynamic environment and make sequential decisions. Reinforcement learning has found applications in various domains, including game playing, robotics, and autonomous systems, where learning from experience and adapting to changing conditions are crucial aspects [53].

Due to the form of our dataset and the scope of this research, the study will leverage a range of supervised ML algorithms to identify patterns within the source dataset, contributing to the enhancement of forensic investigations. Additionally, unsupervised ML algorithms will be applied to identify outliers. The selection of algorithms for this research is based on the most prevalent and widely recognized models used in various application domains [49]. The subsequent subsections will provide detailed insights into the functionalities of the chosen algorithms.

3.2.1 Decision Tree

The Decision Tree algorithm solves classification and regression problems by continuously splitting data based on a specific parameter. Decisions are made on the leaves of the tree, and the data are divided into nodes. The goal is to create a model that can predict the values of a target variable by learning simple decision rules from the characteristics of the data [48]. Decision trees are widely used in various application domains due to their intuitive nature and ease of interpretation. This model can handle variables of different types, including categorical and numerical variables, providing great flexibility in data manipulation [31].

The main components of a decision tree model are nodes and branches, and the most important steps in building a model are splitting, stopping, and pruning.

- **Nodes** can be categorized into three types. First, there is the root node, often referred to as a decision node. This node signifies a pivotal choice that leads to the subdivision of all records into two or more mutually exclusive subsets. Second, the internal nodes, also known as chance nodes, represent one of the options available within the tree structure. These nodes connect to their parent node through the upper edge and link to their child nodes or leaf nodes through the lower edge. Finally, the leaf nodes, alternatively termed the end nodes, symbolize the final outcome resulting from a sequence of decisions or events. This classification provides a clear understanding of the various roles these nodes play in the context of decision trees.

- **Branches** symbolize the possible outcomes or events that originate from both the root nodes and the internal nodes. The construction of a decision tree model entails establishing a hierarchical structure composed of these branches. Each trajectory, which extends from the root node through internal nodes and ends at a leaf node, delimits a unique classification decision rule. These decision tree paths are not only a graphical representation but can also be expressed as ’if-then’ rules. For instance, a rule might be framed as follows: "If condition 1 and condition 2 and... condition k occurs, then the outcome y is anticipated." This concise articulation illustrates the logical connections within the decision tree, translating complex pathways into understandable decision rules.

- **Splitting** is the pivotal stage in decision tree construction, where a node is partitioned into multiple subnodes with the objective of creating relatively pure subsets and creating decision rules that effectively capture patterns and relationships within the data. The essence of this process lies in identifying the optimal split for a node, a task that can be approached through various methods. The goal is to enhance the homogeneity within each resulting subset, thereby facilitating the generation of a decision tree that effectively captures distinct patterns in the data. The determination of the best split involves evaluating different criteria, such as Gini impurity or information gain, to ensure that the resulting nodes encapsulate coherent and discernible patterns, contributing to the overall predictive accuracy of the model.

- **Stopping** process serves as a key determinant in the construction of the decision tree, defining when the expansion of the tree should come to a halt. This strategic intervention is essential to guard against overfitting, a scenario in which the model becomes overly intricate and excessively tailored to the idiosyncrasies of the training data, potentially compromising its ability to generalize effectively to new data.

  During the stopping process, specific conditions or criteria are established to signal the end of tree growth. These criteria may include parameters such as a predefined

maximum depth of the tree, ensuring that the tree does not become overly complex. Alternatively, a minimum number of data points in a node might be specified to prevent the creation of nodes that are too specific to the training set.

By implementing a well-considered stopping process, the decision tree strikes a balance between capturing meaningful patterns in the data and avoiding an overly intricate model. This thoughtful approach enhances the tree's ability to generalize, making it more robust when applied to unseen data and contributing to the overall effectiveness of the predictive model.

- **Pruning** process is a pivotal step in the optimization of decision trees, designed to counteract the potential pitfall of overfitting and bolster the model's ability to generalize beyond the training data. In contrast to the stopping process, which determines when the growth of the tree should cease, pruning involves the discerning removal of branches or subtrees that might not significantly contribute to the model's predictive accuracy.

  As the decision tree evolves during training, certain branches may become overly intricate, capturing noise or peculiarities specific to the training dataset. Pruning aims to identify and trim these superfluous branches, fostering a more streamlined and generalizable tree structure. This selective removal is typically guided by metrics such as cross-validation error or impurity reduction, ensuring that the pruned tree maintains its predictive power while avoiding unnecessary complexity.

  By executing a judicious pruning process, decision trees strike an optimal balance between capturing meaningful patterns and avoiding undue complexity. The resultant pruned tree is more resilient against overfitting, making it better suited to provide accurate predictions on new, unseen data and contributing to the overall robustness of the model.

Figure 3.3 illustrates the concept and components of a decision tree model.

3.2.2 Naive Bayes

The **Naive Bayes** algorithm is based on conditional probability. In this approach, there is a probability table that serves as the model and is updated through the training data. This table is based on the values of its features, where it is necessary to consult class probabilities to predict a new observation. The basic assumption is conditional independence, which is why it is called naive. However, in most analyses, it is challenging to maintain the assumption that all input characteristics are independent of each other [46].

The core equation of Naive Bayes is expressed by 3.1.

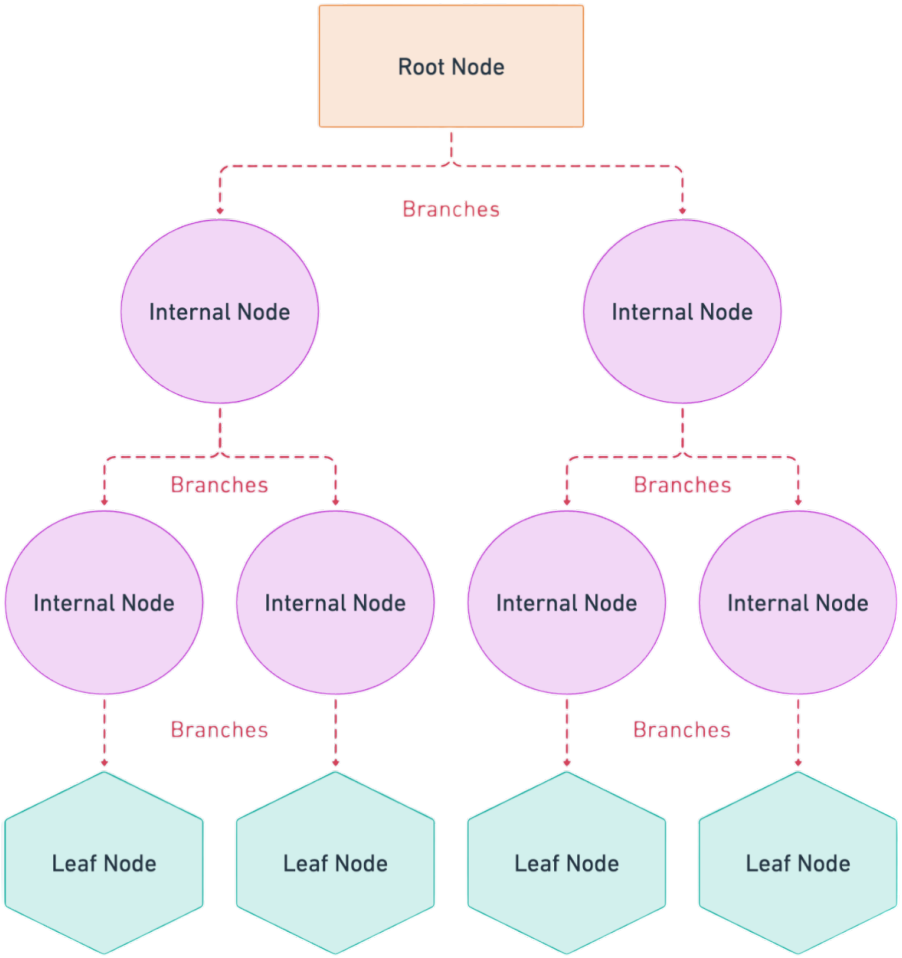

Figure 3.3 – Representation of decision tree.

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)} \tag{3.1}$$

Equation 3.1 encapsulates the conditional probability, representing the probability that event A will occur given the occurrence of event B. Naive Bayes leverages this formula to make predictions by estimating the probabilities involved, making it a powerful and widely used algorithm in various fields, despite its simplistic assumption of feature independence.

There are three most common types of Naive Bayes models, they are:

- **Gaussian:** Suited for continuous data, the naive Gaussian Bayes model assumes that the features follow a normal distribution.
- **Multinomial:** This model is appropriate for discrete data, commonly used in text classification tasks. It is particularly effective when dealing with features that represent counts or frequencies.

- **Bernuolli:** Designed for binary data, this model is well suited for situations where features are binary variables, typically representing the presence or absence of specific attributes.

These three variants accommodate diverse data characteristics, allowing the Naive Bayes algorithm to be applied effectively across a range of applications and datasets.

3.2.3 Support Vector Machine

The **SVM** algorithm is a powerful tool used for both classification and regression tasks. It operates by using support vectors to establish boundaries to classify input data. Here is a more detailed explanation:

SVM treats each data point as a distinct entity within a n-dimensional space, where n corresponds to the number of features present in the dataset. In this space, the value of each feature corresponds to the coordinate of the respective point. This geometric interpretation allows SVM to create a line or hyperplane that effectively segregates the data into distinct classes [10].

Key Attributes of SVM includes:

- **Linear Algorithm:** SVM is inherently a linear algorithm, and its primary function is to identify the optimal linear separation or hyperplane in the feature space. This is achieved by strategically placing the hyperplane to maximize the margin between different classes, enhancing the model's ability to generalize well to unseen data.
- **Support Vectors:** These are the critical data points that lie closest to the decision boundary, influencing the positioning of the hyperplane. SVM aims to find the optimal hyperplane that not only separates classes but also maximizes the distance from the support vectors.
- **Flexibility in Kernel Functions:** While SVM starts as a linear classifier, it can be adapted to non-linear problems through the use of kernel functions. This flexibility allows SVM to handle complex relationships between features, providing a solution for tasks where linear separation is insufficient.
- **Effective in High-Dimensional Spaces:** SVM performs well even in high-dimensional feature spaces, making it suitable for datasets with a large number of features.

In essence, SVM excels at creating robust decision boundaries, especially in scenarios where classes are not easily separable. Its ability to handle both linear and non-linear relationships, along with its effectiveness in high-dimensional spaces, makes SVM a versatile and widely used ML algorithm.

Figure 3.4 illustrates the concept and hyperplane for a SVM model.

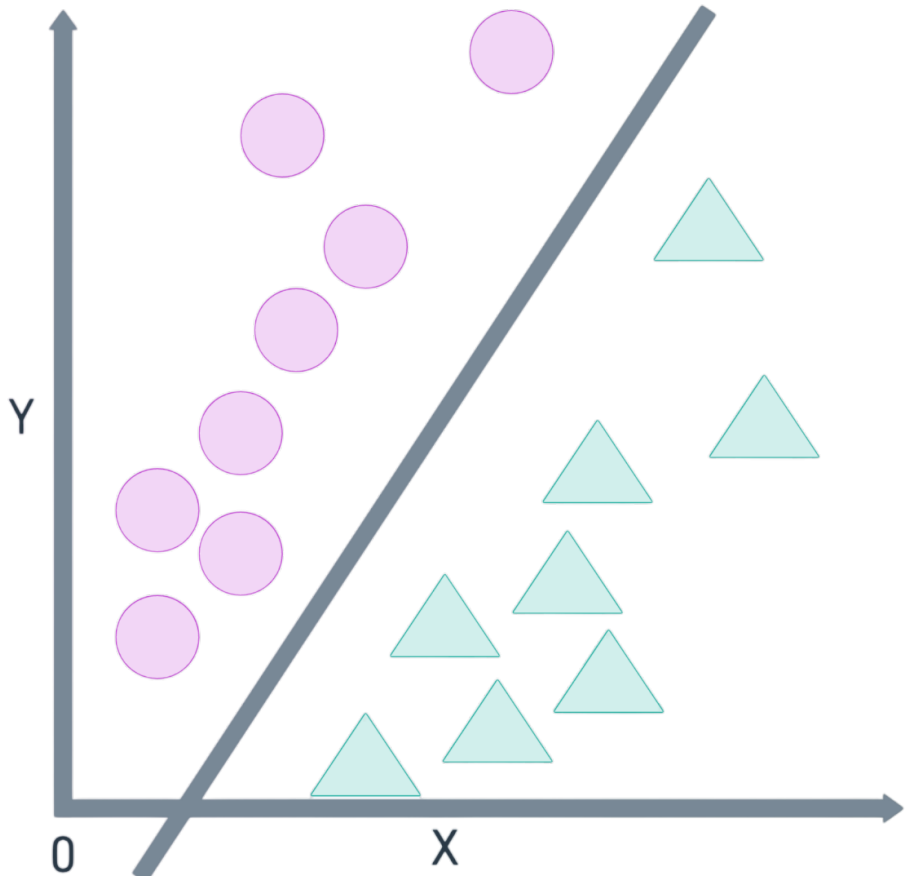

Figure 3.4 – Representation of SVM and its best hyperplane.

3.2.4 Random Forest

Random Forest employs an ensemble approach to solve classification and regression problems. This algorithm combines multiple individual decision trees, each trained on a random sample of the training data, to make predictions. These individual trees are known as random decision trees and are combined to form the forest. Each tree contributes to the final decision through a voting or averaging process of the results [8, 15].

Moving to the intricacies of Random Forest, a closer examination reveals key attributes that contribute to its effectiveness and wide-ranging applicability. In particular, the strength of Random Forest lies in its robustness and versatility. Its ensemble nature allows it to adapt to diverse and noisy datasets, making it particularly resilient in scenarios where individual trees may struggle. This characteristic, coupled with the randomization of the features in each split, mitigates the risk of overfitting, improving the generalization capacity of the algorithm [8].

One notable feature of Random Forest is its ability to provide insights into the importance of variables. By assessing the contribution of each feature to the overall predictive accuracy, users can gain valuable insight into the relevance of different variables, aiding in feature selection and model interpretation.

Widely recognized for its effectiveness, Random Forest has become a popular choice in a spectrum of ML applications. Its ability to handle intricate relationships within the data, coupled with its adaptability to various problem domains, makes it a robust and reliable algorithm in the ML toolkit [15].

3.2.5 AdaBoost

The operation of **AdaBoost** involves iteratively training a series of weak classifiers on modified subsets of the training data. At each iteration, the algorithm adjusts the weight of the training examples, placing greater importance on the examples misclassified in previous iterations. Thus, the algorithm emphasizes the most challenging examples to classify, allowing subsequent classifiers to focus on these difficult cases [19].

At the end of AdaBoost's process, a final classifier is produced, which is a combination of individual weak classifiers with weights determined by their error rates. This technique amplifies the effect of classifiers with higher accuracy, thus creating a powerful and adaptive classifier that can make accurate predictions on test data [19].

AdaBoost's strength lies in its ability to iteratively enhance its performance by prioritizing difficult-to-classify instances, ultimately achieving a robust and accurate predictive model. This adaptability makes AdaBoost a valuable asset in addressing complex classification challenges in various domains.

3.2.6 Extreme Gradient Boosting

The **XGBoost** algorithm is based on tree committees, much like the Random Forest. It belongs to the family of *boosting* algorithms, which combine multiple weak models to form a strong model. XGBoost is known for its exceptional efficiency and performance, widely used in data science competitions and real-world applications [13].

Rather than constructing each tree independently, XGBoost utilizes a sequential training approach. This method involves successive trees rectifying the mistakes of the ensemble of prior trees, thereby gradually improving the model's predictive accuracy.

The optimization mechanism within XGBoost is notable, as it aims to minimize a defined objective function. This function combines the training loss with a penalty for model complexity, emphasizing a balanced trade-off between accuracy and simplicity. By

strategically combining these decision trees, XGBoost seeks the optimal configuration that minimizes the overall objective function [13].

This unique methodology contributes to the effectiveness of XGBoost, making it a preferred choice in various ML applications. Its ability to iteratively refine its predictions and manage model complexity positions XGBoost as a powerful algorithm, particularly in situations where predictive accuracy is paramount.

AdaBoost and XGBoost are two ensemble learning techniques that belong to the boosting family. However, they have considerable differences in their algorithms, techniques, and outcomes. XGBoost can be thought of as a more sophisticated and improved version of the traditional AdaBoost. Here are some of the main differences:

- **Model Complexity:** AdaBoost focuses on combining multiple weak learners, usually shallow decision trees, to form a strong learner. Each weak learner is designed to rectify the errors of its predecessor. XGBoost, however, utilizes a more intricate base learner, usually decision trees. It incorporates a regularization term into the objective function, which allows it to manage more intricate relationships and potentially outperform AdaBoost.

- **Objective Function:** The aim of AdaBoost is to reduce the exponential loss function by giving greater importance to incorrectly classified examples in each iteration. XGBOOSt optimizes a differentiable objective function that combines a loss element and a regularization element. This allows for a more precise adjustment of the model complexity and helps to avoid overfitting.

- **Weighted Voting versus Gradient Boosting:** AdaBoost utilizes weighted voting, where each learner has a weight based on its accuracy that contributes to the final prediction. In contrast, XGBoost applies gradient boosting, where each weak learner is added to the model in sequence, and each successive tree tries to rectify the residual errors of the collective.

- **Handling Missing Values:** XGBoost has a feature that allows it to manage missing values when training and predicting, making it more reliable when working with incomplete datasets. On the contrary, AdaBoost does not manage missing values well, as it is based on weighted voting, which can be affected by the lack of values.

- **Parallelization:** AdaBoost's boosting process is sequential, thus limiting its ability to be parallelized. In contrast, XGBoost is designed to be highly parallelizable, allowing for faster training on multicore machines.

In conclusion, both AdaBoost and XGBoost use the boosting technique of combining weak learners. However, XGBoost has several improvements, including a more flexible base learner, a complex objective function, and the ability to manage missing

values effectively. These enhancements are likely the reason why XGBoost is so popular and successful in many ML tasks, often making it the preferred choice over other boosting algorithms.

3.2.7 Isolation Forest

The **Isolation Forest** algorithm, a powerful anomaly detection technique, operates by employing binary trees to isolate and identify anomalies within a dataset. This approach contributes to a remarkable linear time complexity, making it particularly efficient for processing large datasets with minimal computational burden. The unique strategy of isolating anomalies through the creation of individual trees allows the algorithm to quickly discern irregularities, making it an advantageous choice for real-time applications.

In more detail, the algorithm constructs random trees and isolates anomalies based on their shorter average path lengths within these trees. Anomalies typically require fewer partitioning steps to be separated, resulting in shorter paths. This distinctive characteristic improves the efficiency and effectiveness of the Isolation Forest, making it a robust tool for anomaly detection across diverse datasets.

Using binary trees also translates into low memory usage, a crucial factor when dealing with large datasets. This characteristic not only contributes to computational efficiency but also facilitates the scalability of the algorithm, ensuring its applicability to datasets of varying sizes.

The Isolation Forest algorithm’s proficiency in swiftly identifying anomalies while maintaining efficiency and scalability renders it well-suited for applications in numerous fields, including cybersecurity, fraud detection, and IoT forensics. Its inherent advantages make it a valuable asset in situations where rapid anomaly detection is paramount, contributing to the growing arsenal of tools available for data-driven analyses and investigative insights.

4. PROPOSED MODEL

In the previous chapter, various concepts of DF and ML were discussed. Furthermore, as seen in Chapter 2, ML models have demonstrated their ability to identify a wide range of phenomena and events within different scenarios. For example, they can assist investigators in examinations of child sexual abuse materials, detect instances of cyber attacks, infer the number of people in a specific room, activity monitoring in smart houses, and more. These examples illustrate the versatility and potential impact of ML algorithms in various domains, including DF and IoT.

Given the potential of ML models to recognize patterns and anomalies in IoT environments, we have created a model tailored to this specific purpose. In this chapter, we will introduce our proposed model, explain our objectives, and present the research questions we will use to validate these claims.

4.1 Model Design

Our proposed model aims to aid criminal investigations by incorporating various layers of technology. From data gathering sensors to cloud-based computation and analysis systems, this design will yield actionable insights for various stakeholders. Analyzing the layers and elements of this model will reveal how it converts raw data into informative results that shape decision-making processes in various ways.

The Environment layer consists of various intelligent sensors such as temperature sensors, luminosity sensors, sound sensors, CO_2 sensors, presence sensors, gas sensors, and smart devices including personal assistants like Alexa and Google Home, luminaires, lights, switches, sockets, coffee makers, TVs, among others. Additionally, user devices are connected and interact with these smart devices, such as computers connected to smart sockets.

Next, we have the Cloud Computing layer, which comprises several sub-layers. The first sub-layer is the Applications layer, which aims to communicate with all sensors and smart devices within the intelligent environment, collecting data from these sensors and devices and saving it to the centralized database, the next sub-layer of cloud computing. The next layer of cloud computing is the ML layer, where these applications will query the database and analyze the data to identify potential patterns that can assist in the analysis of criminal investigations. Finally, they will generate reports with the data and information found.

The Users layer is the final part of our model. It consists of individuals and entities who are interested in analyzing environmental data for different purposes. These stake-

holders can be criminal investigators who want to gain insights from the data collected at crime scenes, researchers who conduct specialized studies, or facility managers who want to improve resource utilization and operational efficiency. This layer acts as a bridge between the insights obtained from the cloud computing layer and real-world applications. Its main role is to enable informed decision making and achieve meaningful results.

Figure 4.1 outlines the model proposed above, showcasing the layered architecture designed to analyze environmental data in smart environments. This visual representation provides a comprehensive overview of the components, including the Environment layer with its array of sensors and devices, the Cloud Computing layer with its sub-layers for data processing and ML, and the Users layer comprising various stakeholders interested in leveraging the insights derived from the model. Figure 4.1 serves as a visual aid to improve understanding and visualization of the architecture and functionality of the proposed model.

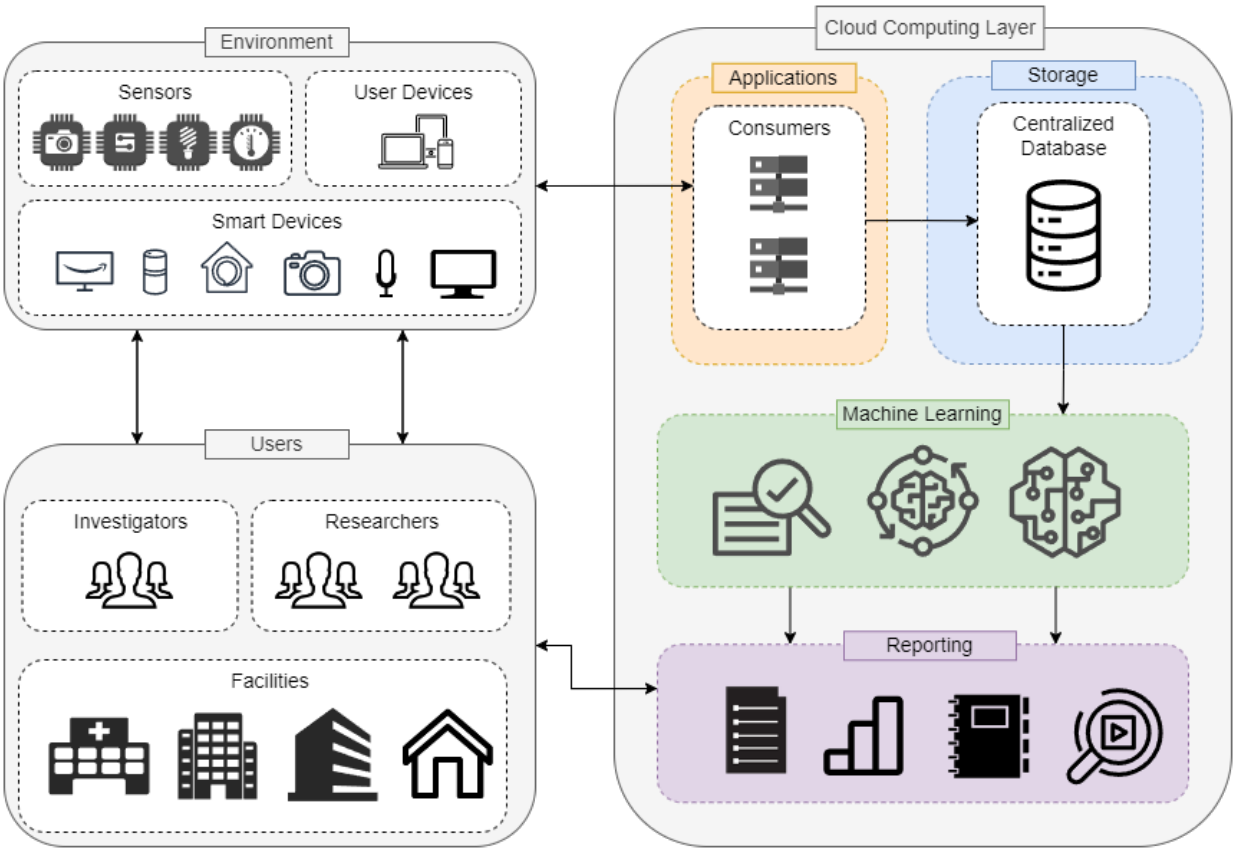

Figure 4.1 – Proposed model.

4.2 Goal and Research Questions

The objective of this research is to evaluate and discuss the adoption of ML models to accurately detect specific patterns in a given environment using data collected from IoT devices, such as temperature, luminosity, CO_2, as well as sound and passive infrared, **in the domain of** Digital Forensic **to** aid crime investigations in physical environments.

With information from several IoT devices, we believe that we could identify several pertinent pieces of information about the environment being analyzed, such as the number of people in the room, temperature fluctuations, ambient light levels, sound levels, air quality metrics including CO_2 levels, presence of specific gases, occupancy patterns over time, energy consumption patterns, and even potential safety hazards such as fire or gas leaks. Additionally, insights into user behavior, usage patterns of various devices, and environmental trends could also be gleaned from the data collected by these IoT devices. This wealth of information has the potential to significantly enhance our understanding of the environment and facilitate more informed decision-making processes.

While there are numerous aspects to explore within the realm of IoT data analysis, this research seeks to compare various ML algorithms to estimate the number of people in a smart environment, using a range of different sensors. The following research questions have been formulated to guide the study:

- **RQ1:** What methods are suitable to detect specific patterns within a large database generated from IoT data?
- **RQ2:** What ML algorithms can be utilized to identify these patterns effectively?
- **RQ3:** How can the identified patterns contribute to the forensic analysis of IoT environments?
- **RQ4:** Can ML algorithms be utilized to detect outliers within a large database generated from IoT data?
- **RQ5:** How accurately can ML algorithms estimate the number of people in a smart environment based on data collected from different types of sensors?
- **RQ6:** Which ML algorithm demonstrates the highest accuracy and reliability in estimating the number of people in a smart environment?

5. EXPERIMENT

This chapter presents the details of our experiment with the aim of offering a detailed explanation of the tools, frameworks, and dataset utilized. Our main objective is to clarify the methodologies employed, ensuring a clear understanding of the experimental setup and its various elements. Moreover, we will offer a comprehensive analysis of the results derived from the experiment, elucidating the findings and the significance of our research.

5.1 Planning

In this section, we offer a comprehensive overview of the strategic planning and methodology used in our study. Our research is based on the utilization of an existing dataset, meticulously chosen from another research project, to align with our study objectives. We started by establishing the necessary environment and computational resources, followed by the careful curation of the selected dataset to ensure its relevance and suitability to address our research questions. The selection of variables and the formulation of hypotheses were driven by a thorough understanding of the field and informed by the pertinent literature. We aim to enhance transparency and clarity in our planning approach by providing a comprehensive explanation. This will establish a strong basis for the subsequent analysis and interpretation of the results.

5.1.1 Environment Setup

We began our environment setup by installing Python, a widely used and versatile programming language. Python was chosen because of its extensive libraries and frameworks that are suitable for data analysis, ML, and data processing. To establish a clean and isolated environment for this research, we utilized the *venv* tool, which is part of the Python standard library. This tool allowed us to create a self-contained environment, separate from the system-wide Python installation, avoiding any potential problems with packages and dependencies. The specific Python version used throughout this research was 3.11.

Within this virtual environment, we used the Python package manager *pip* for the installation and management of essential libraries and modules. These libraries included popular tools for data manipulation (e.g., Numpy, pandas), ML (e.g., scikit-learn, XGboost), data visualization (e.g., Matplotlib, Seaborn), and more. Jupyter Notebooks were used to facilitate interactive and exploratory data analysis, enabling us to document our research

process by integrating code, visualizations, and explanations in a single document. This iterative approach allowed for the development and testing of code while maintaining a comprehensive record of the research.

We chose PyCharm Professional as our Integrated Development Environment (IDE) to facilitate streamlined development, coding efficiency, and seamless integration. PyCharm's advanced features, such as code autocompletion, code analysis, version control, and support for Jupyter Notebooks, were instrumental in improving our coding experience. To ensure robust version control, code tracking, and collaborative development, we use Git as our version control system and GitHub as our online platform. This enabled the management of code modifications and the maintenance of a research record.

Table 5.1 shows all tools and libraries used and their respective versions during this research.

Table 5.1 – Summary of Utilized tools and libraries.

Name	**Type**	**Version**
PyCharm Professional	Tool	2023.3.2
Jupyter Notebooks	Tool	7.0.2
Python	Tool	3.11
Venv	Tool	20.16.7
Pip	Tool	22.3.1
Git	Tool	2.39.3
Scikit-learn	Library	1.3.0
Matplotlib	Library	3.7.2
Pandas	Library	2.0.3
Numpy	Library	1.25.2
XGBoost	Library	1.7.6
Seaborn	Library	0.12.2

5.1.2 Computational Resources

We employed two MacBook models, namely the MacBook Pro M1 and MacBook Pro M2 Pro, to perform our computational tasks. The MacBook Pro M1 features Apple's innovative M1 chip, encompassing an 8-core CPU and an 8-core GPU, delivering exceptional performance across diverse computational workloads. Boasting 16GB of unified memory, this MacBook ensures seamless multitasking and responsiveness. In contrast, the MacBook Pro M2 Pro elevates computational capabilities by integrating the M2 Pro chip, showcasing a 12-core CPU, a 19-core GPU, and 16GB of RAM.

It is essential to note that, despite disparities in hardware specifications, our research yields consistent results in both MacBook models. Our analysis focuses on the accuracy and final results of ML models, rather than the time required for classification

tasks. Thus, any variations in computational performance between the MacBook Pro M1 and MacBook Pro M2 Pro do not impact the reliability or precision of our research findings. This approach safeguards the credibility of our results, ensuring their independence from specific computational resources.

To validate the accuracy of the results, we conducted a test in a cloud environment using an Oracle Cloud ARM instance equipped with a 4-core ARM CPU and 24GB of RAM. This cloud-based infrastructure adds an additional layer of assurance regarding the reproducibility and applicability of our research.

5.1.3 Dataset

The primary dataset for this study, which serves as the basis for the analysis, was originally collected and compiled by Adarsh *et al.* [52]. This dataset was obtained from a controlled laboratory environment. The laboratory consists of a room measuring 6 x 4.6 m, containing four office tables. It is important to note that the experiments were conducted without the use of HVAC systems to maintain consistent conditions during data collection.

The laboratory's network infrastructure is based on a Zigbee star network configuration, which consists of seven slave nodes that send data to a central master node. The choice to use multiple multivariate nodes was motivated by the notion that such a system would provide improved dependability in larger areas compared to a single-node deployment.

Data were collected at an interval of 30 seconds to ensure a thorough capture of the environment's changes. This high-frequency sampling was intended to provide a precise and comprehensive description of the connections between IoT devices and their environment.

These data provide a valuable source of information for the analysis performed. For the execution of this experiment, five different types of noninvasive sensors were used: temperature, luminosity, CO_2, as well as sound and passive infrared (PIR) sensors. This diversity of data offers a comprehensive and detailed understanding of the environment analyzed, allowing a more accurate and comprehensive analysis of occupancy patterns. Table 5.2 presents the representation of the dataset used.

The room used by Adarsh *et al.* for data collection purposes is shown in Figure 5.1. This figure visually presents the layout and dimensions of the laboratory environment, which helps in comprehending the spatial context of the data collection process.

However, it should be noted that acquiring a suitable and extensive dataset for this type of analysis can often be a formidable task. One of the significant challenges we faced was the availability of data, as datasets that met the specific criteria required

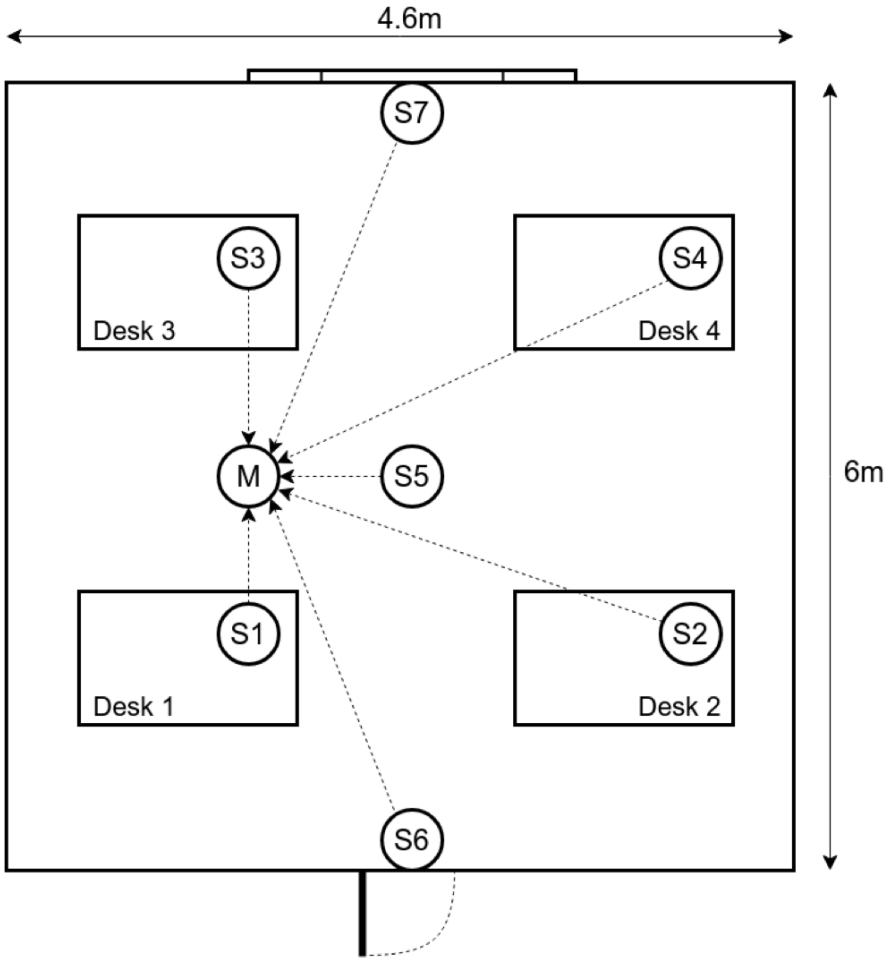

Figure 5.1 – Room used for data acquisition by Adarsh *et al. [52].*

for this research were limited. Furthermore, data quality was a meticulous process as we needed to address inconsistencies, errors, and missing values within the dataset. Additionally, the diversity of the dataset in terms of sensor types, environmental conditions, and occupancy patterns had to be evaluated, as a more diverse dataset is essential for a comprehensive analysis. Although this dataset proved invaluable for our study, these challenges underscore the importance of having access to more extensive and diverse datasets in the field of occupancy analysis, while also highlighting areas that may benefit from improvements in future research endeavors.

Table 5.2 – Database representation

Parameters	**Unit**	**Column Name**
Temperature	ºC	[S1,S2,S3,S4]_Temp
Luminosity	Lux	[S1,S2,S3,S4]_Light
Sound	Volts	[S1,S2,S3,S4]_Sound
CO_2	PPM	S5_CO2
Infrared	Binary	[S6,S7]_PIR
CO_2 Slope	Decimal	S5_CO2_Slope
Occupant Count	Integer	Room_Occupancy_Count

5.1.4 Selection of Variables

To evaluate the efficacy of trained models, performance metrics will be used. This research will incorporate precision, F1 score, accuracy, and recall metrics, which will provide a comprehensive assessment of the model's capacity to fit the data and execute the classification task effectively.

- **Precision:** Precision is a measure that evaluates the accuracy of a model to avoid false positives. It is calculated by dividing the number of correctly classified instances by the total number of classified instances, as shown in Formula 5.1. In this formula, TP stands for true positives (instances correctly classified as positive) and FP stands for false positives (instances incorrectly classified as positive).

$$Precision = \frac{TP}{TP + FP} \tag{5.1}$$

- **Recall:** Recall is a performance measure used to evaluate a model's ability to recognize all positive examples while reducing false negatives. It gauges the proportion of accurately identified positive cases compared to the total number of positive cases in the dataset. The formula for calculating the recall is shown in Formula 5.2, where TP stands for true positives (instances correctly classified as positive) and FN stands for false negatives (instances incorrectly classified as negative). Therefore, recall quantifies the model's sensitivity in accurately detecting positive cases, preventing cases that should have been identified from slipping through. The higher the recall value, the better the model's capacity to precisely retrieve positive cases.

$$Recall = \frac{TP}{TP + FN} \tag{5.2}$$

- **F1-Score:** The F1 score is a metric that combines precision and recall into one measure, providing an overall evaluation of the model's performance. This metric is determined by the harmonic mean of precision and recall, giving both metrics equal importance. The formula 5.3 illustrates how this metric is calculated. Generally, a score of F1 close to 1 implies a good balance between precision and recall, indicating a good classification performance of the model. On the contrary, a value close to 0 suggests poor model performance.

$$F1 = \frac{2 \cdot Precision \cdot Recall}{Precision + Recall} = \frac{2 \cdot TP}{2 \cdot TP + FP + FN} \tag{5.3}$$

- **Accuracy:** Accuracy is a measure of how many instances are correctly classified compared to the total number of instances. To calculate it, we divide the number of

correct predictions by the total number of predictions. In other words, it is the sum of true positives and true negatives divided by the total number of predictions, as expressed in equation 5.4.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{5.4}$$

- **k-Fold Cross-Validation:** Cross-validation is a method that is used to evaluate the performance of ML models. It involves splitting the dataset into subsets and iterating the training and testing of the model. In each iteration, one subset is used for testing, while the rest are combined and used for training.

  This research used the k-fold technique. This approach divides the dataset into k sections, each of which is approximately the same size. The K models are then trained, each iteration using one subset as the testing data and the rest as the training data. Finally, the results are combined to give an overall assessment of the effectiveness of the model.

  The benefit of k-fold cross-validation is that it enables all data to be utilized for both training and testing, guaranteeing a more precise assessment of the model's performance. In addition, it helps reduce the variance in the evaluation, as the performance is determined based on multiple combinations of training and tests. Figure 5.2 shows how k-fold works.

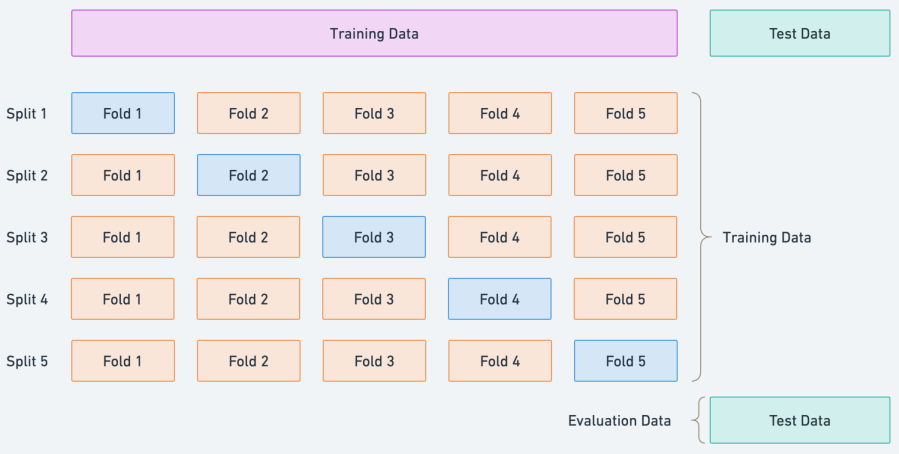

Figure 5.2 – Representation of cross validation.

- **Final Evaluation:** An assessment of the program with real-world data is often done by splitting the data into training and validation sets. In this case, 90% of the data is assigned for training and the other 10% is used for validation. This data division technique enables the model to be trained on most of the available data to recognize the patterns and structures present in the data. Subsequently, the validation set is used to assess the model's performance on data that have not been seen before.

This technique allows us to check if the model can apply the patterns it learned during training to make precise predictions on fresh data. The ultimate validation gives a more realistic assessment of how the model will perform in a production setting where new data will be provided.

An approach was used to evaluate the models trained with real-world data, which involved dividing the data into training sets (90%) and validation (10%). As illustrated in Figure 5.3, 10% of the dataset was set aside for the final evaluation of the models. These data points were not altered in any way, with no normalization or SMOTE technique applied. The results presented in Section 5.2.5 give an explanation of the comparison between the models when applied to these particular data points.

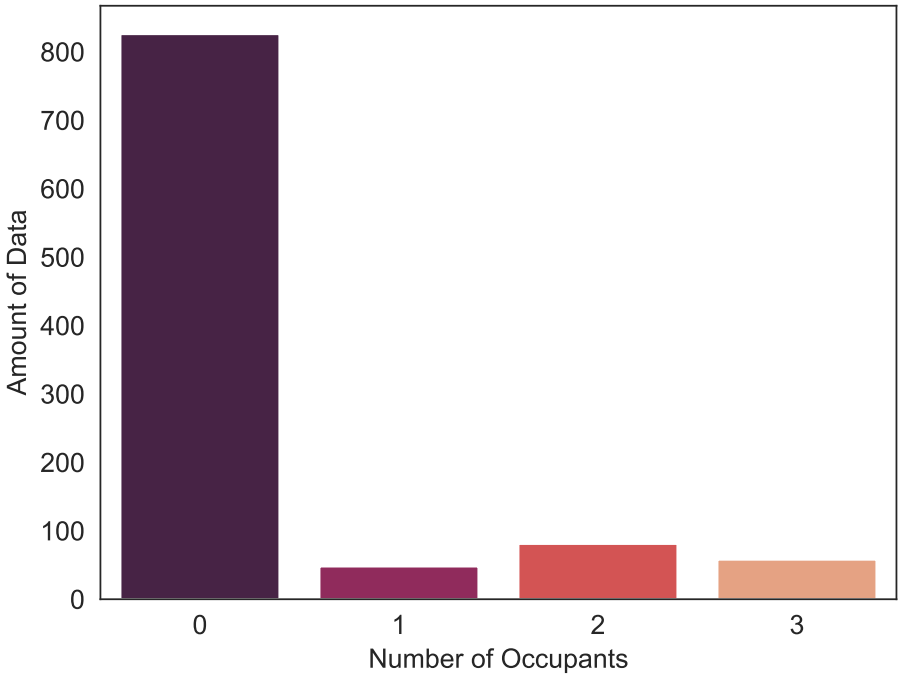

Figure 5.3 – Unmodified real sampling for final model validation.

5.1.5 Definition of Hypotheses

We have formulated the following hypotheses for investigation:

- **Null Hypothesis (H0):** patterns cannot be detected in established IoT environments using the defined strategies.
- **Alternative Hypothesis (H1):** patterns can be detected in established IoT environments using defined strategies.

5.2 Execution

In this section, we explain the systematic steps and techniques used to test our hypotheses and answer the research questions. We provide a clear description of how our study was conducted and demonstrate the thoroughness of our approach.

5.2.1 Exploratory Data Analysis

Analysis of sensor time series graphs, as presented in Figure 5.4, revealed that data from the luminosity, temperature, sound, PIR and CO_2 sensors provide an indication of the number of occupants in the room.

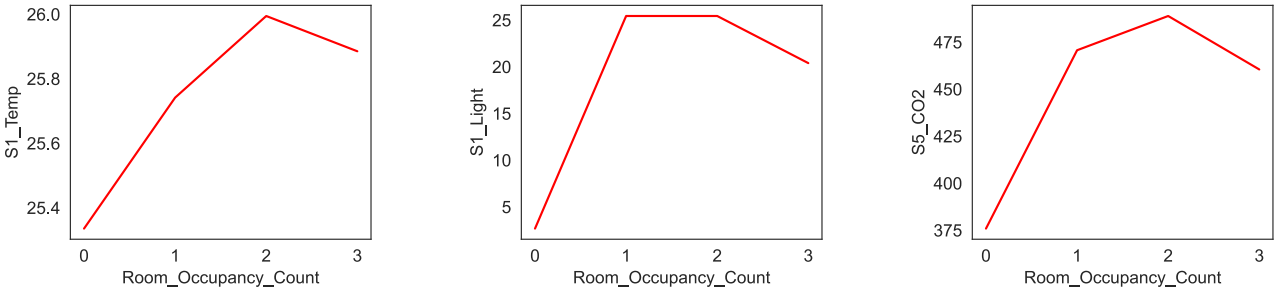

Figure 5.4 – Representation of sensor data related to the number of occupants.

Furthermore, Figure 5.5 allows us to visualize the correlation between variables in the dataset utilized. We observe that the *S1_Temp* and *S5_CO2* sensors exhibit a strong correlation, indicating that temperature changes may be related to variations in carbon dioxide CO_2 levels. On the other hand, we notice that the *S1_Temp* and *S4_Light* sensors do not exhibit a strong relationship, suggesting that temperature changes are not directly related to variations in luminosity.

A significant correlation between variables can indicate a direct relationship between them, which is useful for prediction and pattern identification. This information is particularly relevant in forensic analyzes, where the goal is to uncover specific patterns to estimate the number of people in physical environments within the heterogeneity of the IoT.

Identifying highly correlated variables allows us to take advantage of this information to enhance forensic analyzes. These variables can provide valuable information on interactions among connected devices and help identify potential evidence during forensic investigations. Understanding the correlation between variables allows one to focus analysis efforts on the most relevant aspects, facilitating the detection of patterns and suspicious behaviors.

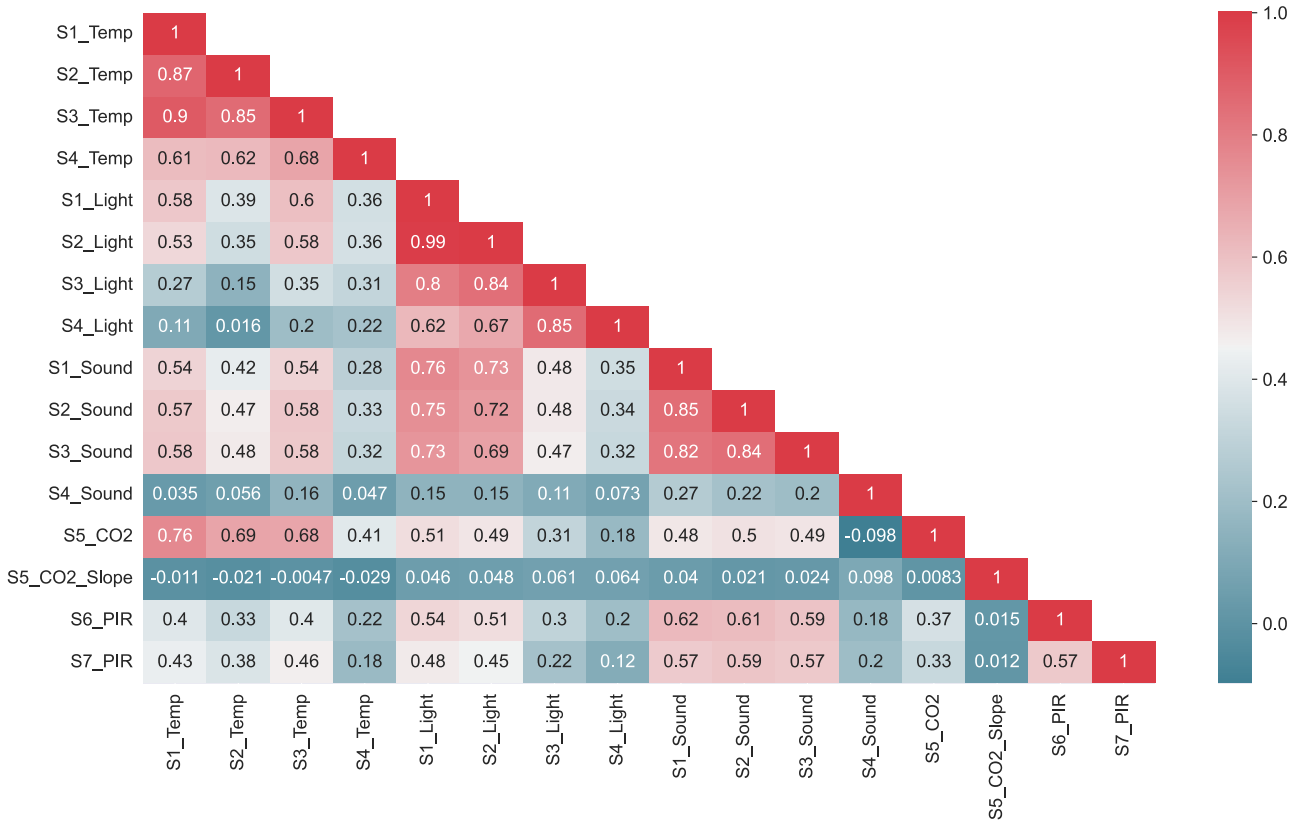

Figure 5.5 – Correlation analysis between independent and target variables.

Furthermore, understanding the dataset allows us to explore and employ techniques such as oversampling and undersampling. These techniques are widely used to address class imbalance in datasets, as significant disparities in class distribution often occur in forensic investigation problems in IoT environments. These approaches aim to balance the class distribution, allowing ML models to be trained more fairly [39].

However, it is important to note that some ML algorithms may struggle when dealing with highly correlated variables. The presence of correlation among variables can introduce bias in the results and affect the accuracy of the models. It should be mentioned that the correlation between variables is not directly related to oversampling or undersampling, but rather to how data are treated and processed before applying these techniques [54].

By understanding the nature of the data and the correlation among the variables, we can identify which oversampling or undersampling techniques are most suitable for the given dataset [39], thus enhancing the effectiveness of forensic analyzes in IoT environments. However, before applying oversampling or undersampling techniques, it is crucial to detect and address outliers in the datasets.

5.2.2 Outlier Detection and Treatment

Outlier detection and treatment are crucial steps in data analysis. Outliers are atypical values that deviate significantly from the data distribution pattern. These outlier observations may arise due to measurement errors, data flaws, or rare and relevant events. The presence of outliers can distort descriptive statistics and negatively impact the performance of ML models [22].

A commonly used method to identify outliers is the Interquartile Range (IQR), which uses the difference between the third quartile (Q3) and the first quartile (Q1) of a distribution. The IQR represents the range of central data used to define a range within which the majority of data is considered normal. Equation 5.5 defines the criteria for identifying outliers based on IQR [2].

$$x < Q1 - 1.5 \cdot IQR \quad \text{or} \quad x > Q3 + 1.5 \cdot IQR \tag{5.5}$$

Once identified, there are different approaches to dealing with outliers. One option is to remove them from the dataset, but this should be done with caution, as excluding such data may result in the loss of important information. Another alternative is to perform data transformations, such as applying logarithms or robust normalization, to reduce the impact of outliers on the results [22].

In this work, we chose to adopt the approach of removing outliers from the dataset. This choice was made because outliers can distort statistical analysis and harm the performance of ML models. By removing these outlier data points, our aim is to ensure a more accurate and reliable representation of patterns within the data, potentially leading to more consistent and reliable outcomes in the analysis and prediction of room occupancy.

Figures 5.6 and 5.7 illustrate the comparison between certain classes containing outliers and the same classes after the application of the IQR method (interquartile range). It is evident that after removing outliers using this method, the data distribution became more homogeneous and devoid of outlier values. This indicates that the IQR method was effective in identifying and removing outliers, contributing to a more robust and accurate data analysis [35].

5.2.3 Data Preprocessing

After the outlier detection and treatment step in the datasets, we can proceed with data preprocessing using undersampling and oversampling techniques. These tech-

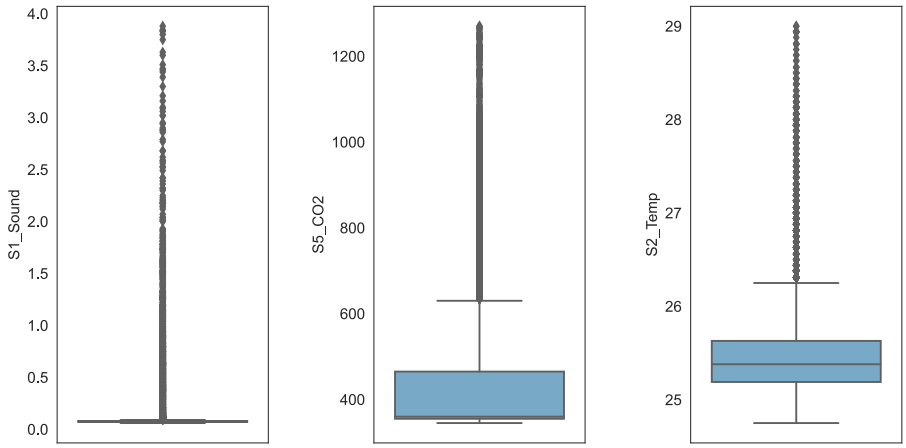

Figure 5.6 – Outlier analysis before the application of IQR.

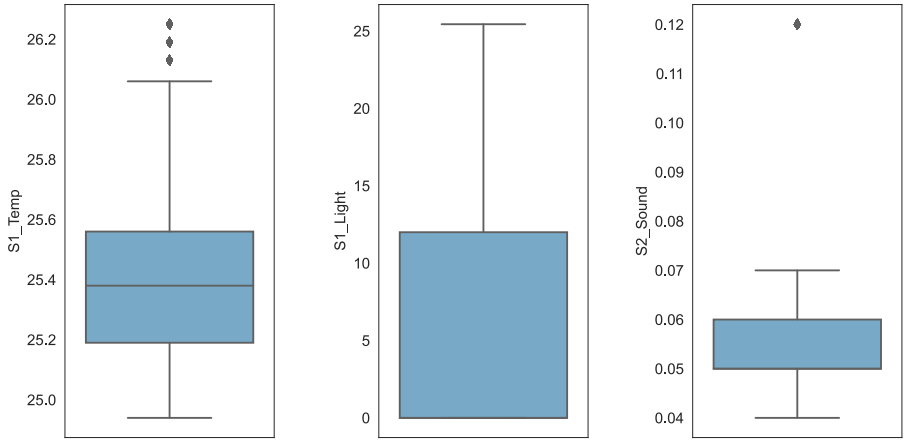

Figure 5.7 – Outlier analysis after the application of IQR.

niques aim to balance the distribution of classes, ensuring that all classes have appropriate representation for training ML models.

The dataset used consists of more than 10,000 records and 16 columns, each representing data from a specific sensor. However, the dataset was found to exhibit a significant class imbalance, as illustrated in Figure 5.8. Such an imbalance can hinder the performance of certain ML models [30]. To address this issue, we chose to employ the oversampling technique known as SMOTE (Synthetic Minority Oversampling Technique) [12].

SMOTE is a widely used technique in data pre-processing to address the class imbalance in a dataset. Class imbalance occurs when one class has significantly fewer examples compared to another class, which can lead to performance issues and bias in ML models [12].

This approach involves oversampling the minority class to improve classification performance using ML models. Instead of simply replicating existing examples, this technique creates additional synthetic examples of the minority class, increasing its represen-

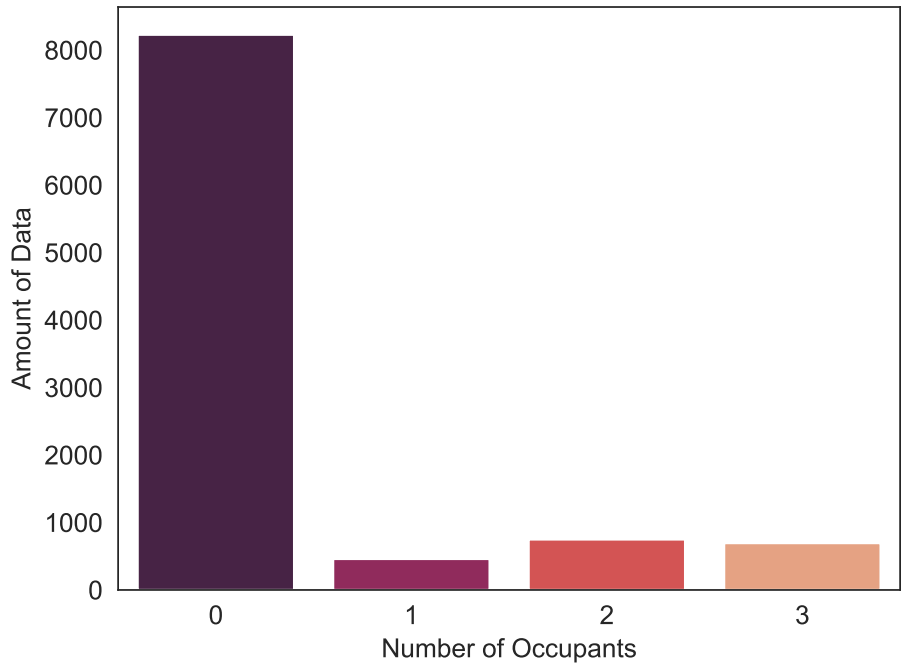

Figure 5.8 – Imbalanced dataset.

tation in the dataset. This allows models to have access to more information about the minority class, contributing to better generalization and accurate classification [12].

When SMOTE is applied to the dataset, synthetic examples are generated along the segments connecting the nearest neighbors of the minority class. The number of synthetic examples to be generated is determined by a parameter called *k*, which represents the number of nearest neighbors to consider. The choice of these neighbors is made at random to ensure diversity in the synthetic examples generated [12].

As illustrated in Figure 5.9, it can be seen that, after applying SMOTE, the training sample has been balanced so that the minority class matches the majority class in terms of the number of examples.

Another technique for balancing datasets is *undersampling*. This technique involves reducing the majority classes while keeping the minority classes intact [12]. *Undersampling* is used to decrease the disparity between classes, providing a more balanced distribution of data. By reducing the number of examples from the majority class, the aim is to mitigate bias and improve the performance of ML models when dealing with imbalanced classes [40].

An example of an *undersampling* technique can be found in the work of Estabrooks *et al.* [18]. In this study, the authors propose a nonheuristic algorithm that balances the dataset by randomly eliminating data from the majority class. However, it is important to note that this technique can remove valuable data [40], particularly in forensic analyzes where each data point is a valuable source of information. Therefore,

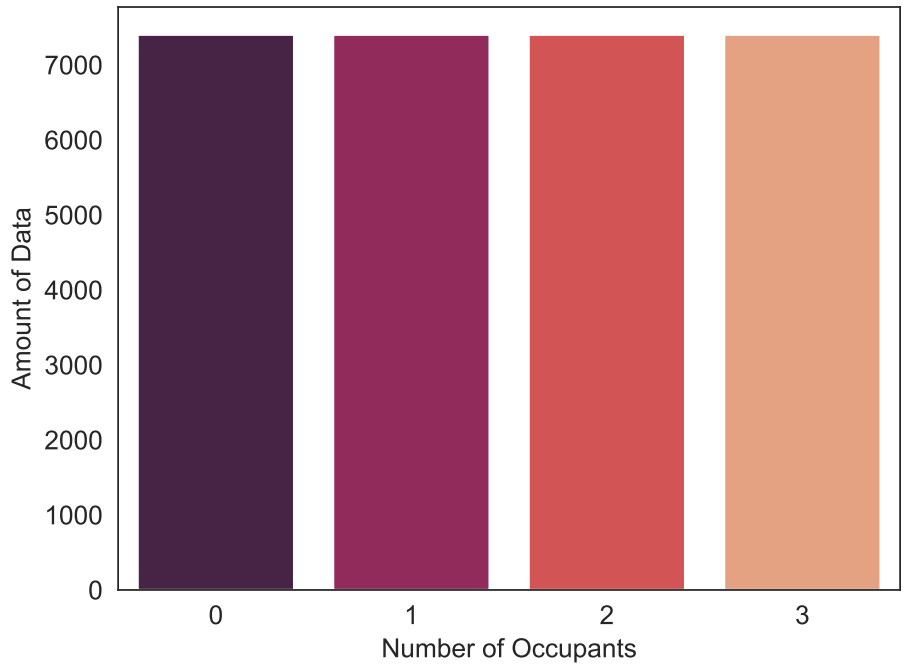

Figure 5.9 – Balanced training sample.

the application of *undersampling* in forensic scenarios should be carefully considered, to avoid the loss of crucial information for the investigation.

Despite the literature listing various works and algorithms in this area [36, 23, 33, 34], this study does not utilize any *undersampling* technique on the data to prevent the loss of important information.

5.2.4 Data Normalization

Normalization is a process in which the attributes of a dataset within a model are adjusted or rearranged to enhance the coherence and consistency of the data. This technique helps in the flexibility of the data, allowing them to be compared and related more efficiently [45].

Through data normalization, redundancy and duplication of information are reduced, avoiding contradictions in the data and rendering it more reliable. This step plays a crucial role, as it classifies and organizes the data in a standardized manner [27].

5.2.5 Results

In this section, we present the findings of our study that allow us to test the hypotheses we have formulated.

After analyzing the results presented in Table 5.3, we observed that most models showed satisfactory performance, achieving metrics that exceed 0.90. In particular, the models based on *Random Forest* [15] and *XGBoost* [13] exhibited the best results in terms of precision, precision, recall and F1 score. These results indicate that these models are better suited to infer and detect specific patterns in the IoT infrastructure based on the features analyzed. These models demonstrated a superior ability to predict and accurately identify relevant events and behaviors within the context of the IoT, thus contributing to more informed and efficient decision making.

Table 5.3 – Metrics of the analyzed models

Model	**Accuracy**	**F1-Score**	**Precision**	**Recall**
Decision Tree	0.993	0.975	0.976	0.975
Random Forest	0.994	0.977	0.976	0.979
SVM	0.979	0.922	0.923	0.922
KNN	0.973	0.901	0.899	0.903
XGBoost	0.995	0.981	0.980	0.982
Naive Bayes	0.944	0.787	0.799	0.804
AdaBoost	0.858	0.610	0.634	0.632

Regarding the Decision Tree model, a high accuracy rate can be observed across all classes, demonstrating its effectiveness in the classification task. However, there was a higher incidence of errors in predicting the class representing the presence of 3 people in the room, totaling 3 incorrect cases. Figure 5.10 shows the confusion matrix resulting from applying this model to the analyzed data, providing a visual representation of the performance of the classification algorithm. It shows the distribution of predicted classes in relation to actual classes; for example, in the second class, there were 47 correct matches between predicted and actual labels, with only one error out of a total of 48 labels.

The Random Forest ML model showed that luminosity, sound, and temperature were the most influential attributes, as seen in Figure 5.11. These variables had a major effect on the model's decision making, demonstrating their critical role in recognizing patterns and making inferences related to the IoT. These data are useful in understanding which features were the most important.

However, the AdaBoost and Naive Bayes models showed less satisfactory results. This performance difference can be attributed to various factors. In the case of *AdaBoost*, as mentioned earlier, it can be sensitive to imbalanced data or the presence of outliers. Therefore, if the dataset used possesses these characteristics, it may have adversely affected the performance of the *AdaBoost* model.

A potential explanation for the performance gap between AdaBoost and XGBoost could lie in the differences in the algorithms and strategies employed. XGBoost is an extension of AdaBoost that employs a more advanced boost approach, allowing greater capacity to learn and adapt to complex patterns in the data. This distinction may have

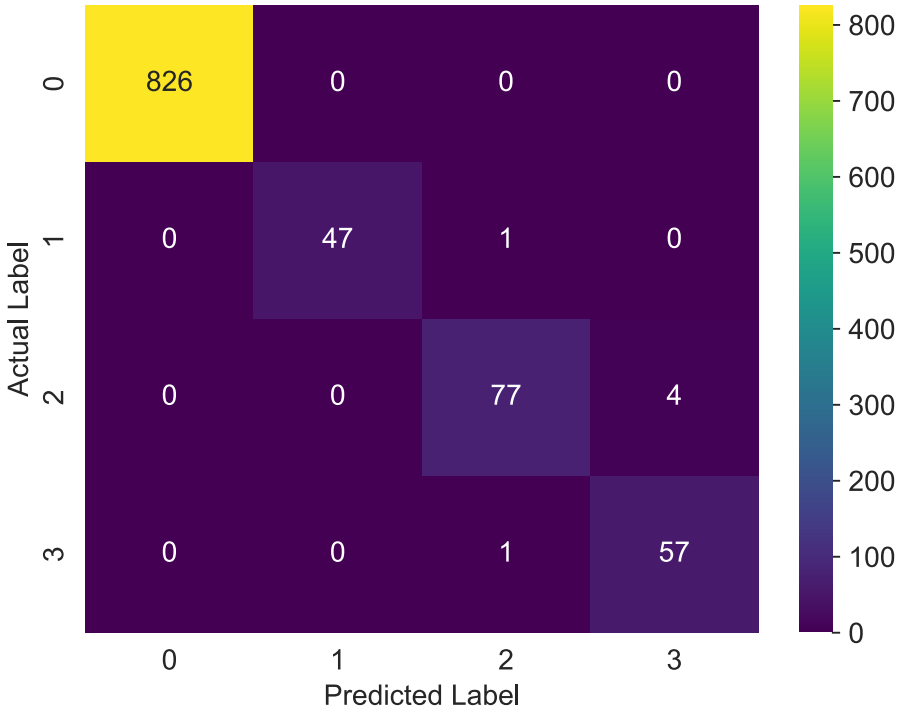

Figure 5.10 – Confusion matrix of the *Decision Tree*.

directly influenced the results, making XGBoost more effective in inferring and detecting specific patterns within the IoT infrastructure.

In summary, although the negative outcome of AdaBoost might suggest a connection to outlier handling, further investigation is necessary to confirm this assumption. Furthermore, the discrepancy in performance between AdaBoost and XGBoost underscores the importance of exploring and comparing different algorithms and techniques to achieve more accurate and reliable results in the analysis of IoT data.

Regarding Naive Bayes, it is important to highlight that this model makes a strong assumption of conditional independence between variables, which may not always hold true in all datasets. If the analyzed features possess dependencies among themselves, Naive Bayes can fail to capture these relationships, and consequently, provide less precise results.

It is evident that it is feasible to recognize patterns in existing IoT environments utilizing the ML algorithms that have been examined and described. This allows us to reject our null hypothesis H0 due to the high precision of the trained ML models investigated in this research in determining the number of people in a room.

The original dataset temperature readings, primarly consist of regular patterns with minimal variations, making it challenging to assess the Isolation Forest algorithm performance in identifying outliers. In order to comprehensively test the model's capability, synthetic temperature data is generated to include specific types of anomalies, such as prolonged outliers.

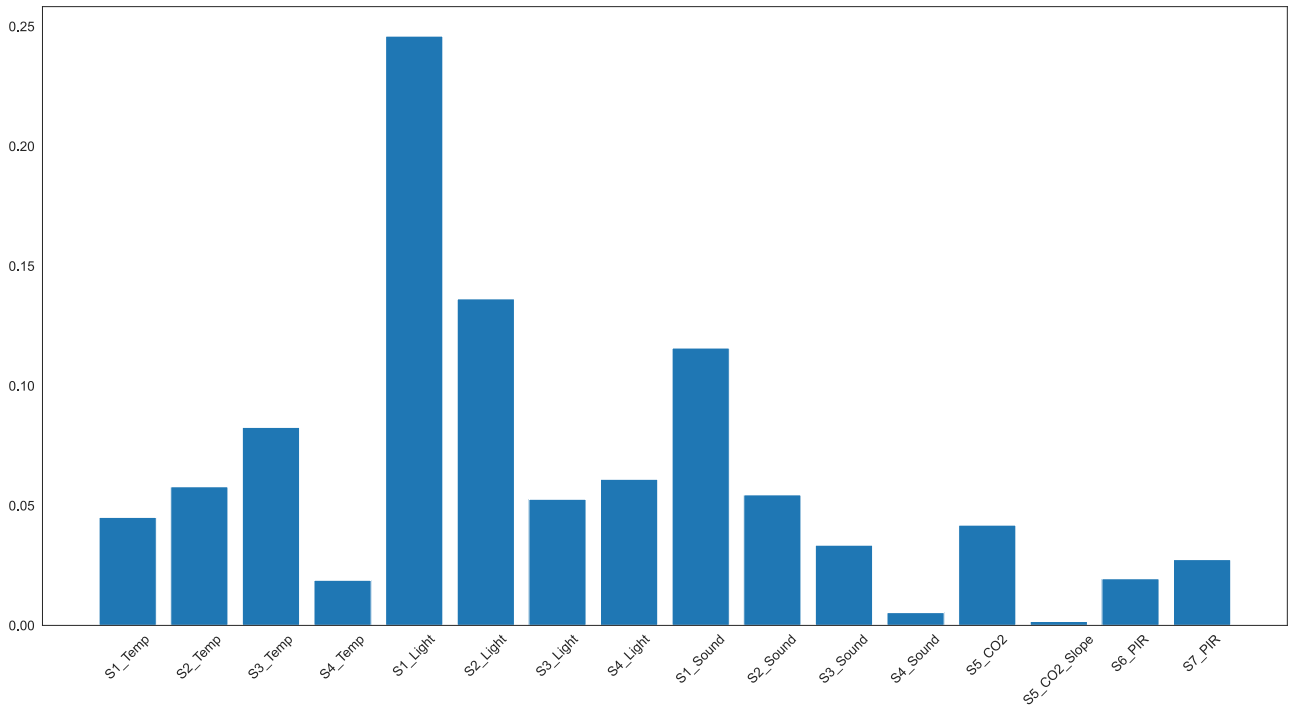

Figure 5.11 – Feature Importance for the *Random Forest* model.

Synthetic anomalies are strategically designed to mimic potential irregularities that may occur in a real-world scenario, even if they are infrequent in the original dataset. By introducing these synthetic anomalies, the evaluation becomes more rigorous, allowing for a thorough examination of the algorithm's sensitivity to different types of deviations from the norm.

This synthetic data creation is significant as it enables researchers to measure the algorithm's effectiveness in scenarios that might not be prevalent in the existing dataset but could have critical implications in practical applications. It provides a proactive approach to testing the model's adaptability to a broader range of anomalies, ultimately enhancing its robustness and reliability in anomaly detection tasks. In addition, the purpose of this was to determine whether unsupervised ML could successfully identify these data points and distinguish which are considered anomalous.

Figure 5.12 illustrates all temperature data points from the *S1_Temp* sensor, including synthetic data, along with the results of the analysis of the Isolation Forest model.

During analysis, the algorithm successfully identified the synthetic data as outliers, indicating the ability of the isolation forest model to distinguish unusual patterns in the temperature data. Additionally, the identification of synthetic outliers showcases the model's capability in handling these situations, providing valuable insights for detecting anomalies in similar datasets.

Although this may not be a highly complex real-world situation, due to the synthetic data, it is important to emphasize the potential importance of these algorithms in assisting criminal investigations.

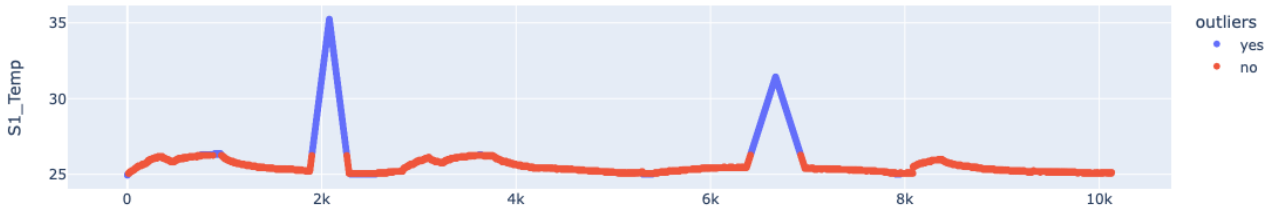

Figure 5.12 – Temperature Outliers Identified with Synthetic Data Points.

5.2.6 Discussion of Results

This section discusses the results obtained from our study and thus answers our defined research questions.

This study has demonstrated the ability of Random Forest and XGBoost to recognize patterns in data obtained from IoT devices, thus affirming their usefulness in this context. The effectiveness of these models in recognizing significant features in the IoT infrastructure shows their potential to help in criminal investigations within this environment.

Using these models in criminal investigations in IoT settings, it is possible to detect suspicious events and anomalies that could be indicative of criminal behavior. For example, irregular patterns of brightness, CO_2 concentrations, or noise could be signs of illegal or unauthorized activities. These models can be adjusted and improved over time, improving their ability to recognize relevant patterns for criminal activities. We use the models in our context to calculate the number of people in a certain room, and they achieved a success rate of more than 97% for multiple metrics.

- **RQ1:** We have conducted a thorough investigation to identify methods that are best suited to decipher the complex patterns found in large datasets created from IoT data. By comparing different techniques, we have been able to pinpoint the approaches that are most effective in deriving meaningful information from our data set.
- **RQ2:** This research has demonstrated the effectiveness of ML techniques in recognizing and interpreting patterns. Random Forest and XGBoost have been used to great effect, showcasing the potential of the right algorithms in this field.
- **RQ3:** Our research has revealed patterns that could be of great benefit to forensic analysis in the IoT environment. These patterns can help identify unusual events and potential criminal behavior, providing useful clues and information that can be of great help in investigations.
- **RQ4:** Our analysis of unsupervised ML algorithms, such as Isolation Fores illustrated in Figure 5.12, reveals their effectiveness in identifying outliers within datasets gen-

erated by IoT data. For example, Isolation Forest works by isolating outliers in the data by selecting a feature and then randomly selecting a split value between the maximum and minimum values of the selected feature. The number of splits required to isolate the outlier is used as a measure of its abnormality.

- **RQ5:** Our research has demonstrated the effectiveness of ML algorithms, particularly Random Forest and XGBoost, in accurately estimating the number of people in a smart environment. We found that these algorithms achieved a success rate of more than 97%. Furthermore, our findings highlight the potential of ML algorithms to provide valuable information on occupancy patterns, thus facilitating more informed decision-making processes in various applications.
- **RQ6:** The ML algorithms that demonstrated the highest accuracy and reliability in estimating the number of people in a smart environment were Random Forest and XGBoost. These algorithms consistently outperformed others in our research, achieving success rates that exceed 97%.

In conclusion, our study has shown that Random Forest and XGBoost are capable of accurately detecting the number of individuals in a given room. Additionally, our results indicate that these ML algorithms can be utilized to enhance and support criminal investigations in the field of IoT forensics. Furthermore, the ability of these algorithms to identify patterns and anomalies in IoT data not only highlights their immediate usefulness, but also suggests potential advancements in the application of data-driven methods for more comprehensive investigative insights and detailed forensic analyses in the coming years.

5.3 Hyperparameters Tuning

In this section, we will dive into the process of optimizing the hyperparameters of our ML models. Hyperparameters play a fundamental role in determining the performance and effectiveness of these models, and finding the right combination is essential to achieve the best possible results. Through a systematic exploration of hyperparameter settings, we aim to enhance the predictive power and generalizability of our models, ultimately fine-tuning their performance to meet the specific demands of our problem domain. This section will outline the strategies and methodologies we used to optimize hyperparameters, shedding light on the critical decisions made to improve the overall efficacy of our ML algorithms.

In the realm of ML, hyperparameter tuning is the fine-tuning of the settings of a powerful instrument that aims to infer room occupancy from data. These settings, known

as hyperparameters, are the keys to orchestrating the performance of our ML models, specifically tailored to the task of inferring room occupancy.

Our primary goal in hyperparameter tuning is to identify the ideal combination of hyperparameters that results in the highest accuracy and reliability in inferring room occupancy. This optimal configuration ensures that our model excels not only in training but, most importantly, in making accurate real-time predictions. However, the computational resources required to navigate a complex landscape of possibilities are substantial. Therefore, we must tread carefully to avoid overfitting our model to the training data.

To address these challenges, we will exclusively employ Grid Search and Random Search to fine-tune our hyperparameters. Grid Search systematically explores a predefined hyperparameter grid, while Random Search randomly samples hyperparameters from specified ranges.

Hyperparameter tuning, particularly through Grid Search and Random Search, is a dynamic process where the right combinations create a powerful model that accurately and reliably infers room occupancy. It is a fundamental part of our journey to harness the full potential of ML in the context of room occupancy inference. For the hyperparameter tuning process, we opted for Random Search, primarily because of its superior performance in efficiently exploring the hyperparameter space. Additionally, our focus for hyperparameter tuning was directed at both the Random Forest and XGBoost models.

Table 5.4 – Hypertuned models

Model	Accuracy	F1-Score	Precision	Recall
Random Forest	0.994	0.975	0.977	0.979
XGBoost	0.995	0.978	0.981	0.981

In our research, while rigorously optimizing the hyperparameters for our XGBoost model, we observed a marginal decrease in model performance, compared to the untuned baseline model, as summarized in Table 5.4. This phenomenon is not uncommon in the field of ML and can be attributed to various factors [6]. One possibility is that, in our pursuit of optimizing hyperparameters, the model may have been unintentionally fine-tuned to the validation data to such a degree that it no longer generalized as effectively to unseen data. Additionally, intricate interactions among hyperparameters can sometimes result in suboptimal combinations that appear to perform well in the validation phase, but fail to maintain their effectiveness in real-world data. It is crucial to consider that, due to inherent noise or randomness in real-world datasets, observed fluctuations in model performance can occur. Furthermore, the evaluated metric chosen for optimization, the F1 score, although carefully selected, may not be perfectly aligned with the real world goals of our room occupancy inference task. In light of this observation, our study emphasizes the importance of not only the pursuit of optimal model configurations but also the recognition of the broader challenges and trade-offs inherent in ML.

5.4 Limitations and Lessons Learned

Our research encountered a certain limitation, that our ML models were tested on only our initial dataset. To address this, we took proactive steps to reduce potential biases. We allocated 10% of the initial dataset for testing, allowing us to assess the generalizability and strength of our models beyond the data on which they were trained.

To maximize the accuracy and dependability of our results, we extended our methodology by using cross-validation techniques. This involves dividing the dataset into separate parts, training the models on one part, and then testing them on the other. This iterative process provides a thorough evaluation of the models' performance in different data sets, which helps to more accurately determine their effectiveness in recognizing patterns in various IoT environments.

Using cross-validation, our research exceeded the restrictions that come with relying on a single set of tests. It safeguarded against the possibility of overfitting and increased the model's ability to handle data that have not been seen before. Furthermore, cross-validation is in line with the construct validity framework, which guarantees that our models accurately capture the complex patterns and behaviors common in IoT environments.

At the start, our testing approach gave us some understanding, but by adding cross-validation we made our findings more reliable, providing a more thorough assessment of how the ML models worked with a wider range of data.

The use of ML models in pattern detection within IoT environments can be improved by taking into account several aspects. Through the development of this study and the analysis of the results, it was possible to identify and consider relevant factors that can help to better understand the use of these models. These lessons learned can be used to guide future research and improve investigation approaches in scenarios involving IoT devices.

- **Significance of Proper Selection of ML Models:** The choice of the right ML models is essential to achieve good performance and accurate results. By carefully comparing and evaluating different models, it was possible to determine which models were more effective in recognizing certain patterns within the IoT infrastructure. This emphasizes the importance of making a well-considered selection of models, taking into account the characteristics of the data and the objectives of the research.

- **Importance of Data Exploration and Preprocessing:** Data preprocessing, such as dealing with outliers and normalization, can have a major effect on the outcomes. Additionally, it is critical to investigate the characteristics of the data and understand

their significance in the decision-making process of the models to draw meaningful conclusions and direct the analysis.

- **Need for a Representative and Balanced Dataset:** The importance of the quality and representation of the dataset cannot be overstated when it comes to the performance of ML models. To ensure reliable and accurate results, it is essential to have a dataset that is diverse, balanced, and reflective of the context being studied. To avoid bias in the results, comprehensive data collection and consideration of potential class or event imbalances must be taken into account. In the cases of imbalanced datasets, such as those found in the IoT, techniques such as SMOTE can be used to even out minority classes.

Throughout the development of this study, another lesson we have learned is the importance of carefully considering the size and complexity of the datasets used. Eventually, especially when dealing with extensive databases, we encountered significant challenges related to the memory requirements for training these models.

Memory issues can arise due to the vast amount of data that models need to process and store during training. This situation may cause practical constraints in terms of the computational resources that are available, leading to extended training durations, system instability, and potential failures during the training phase.

Hence, it became clear to us that, apart from dealing with the technical and algorithmic aspects of machine learning, it is essential to also take into account constraints like memory when creating machine learning solutions. Consequently, we have acquired the knowledge to embrace a comprehensive strategy, integrating methods like data preprocessing, effective sampling, and deliberations on model structure to address memory-related issues and guarantee successful training of our models.

This experience has taught us that while machine learning offers incredible opportunities, it is essential to maintain a broad and balanced perspective, considering not only the technical aspects but also the practical and operational challenges that may arise when dealing with large datasets.

These lessons offer invaluable advice to those looking to use ML to detect patterns in IoT settings. By taking these elements into account, future studies can gain from better model selection, more reliable data preprocessing, and more precise and reliable results.

6. CONCLUSION

The results obtained in this study highlight the ability of Random Forest and XGBoost to identify patterns in the data collected by IoT devices, solidifying their role in this context. The efficiency of these models in detecting relevant features in the IoT infrastructure demonstrates their potential to assist in criminal investigations within this environment. By applying these models to criminal investigations in IoT environments, it is possible to identify suspicious events and anomalies that may indicate criminal or unauthorized activities. For example, abnormal patterns in brightness, CO_2 levels, or sounds can serve as indications of illicit activities. These models can be adapted and refined over time, enhancing their ability to detect relevant patterns related to crime.

While Random Forest and XGBoost have demonstrated satisfactory efficiency in estimating the number of people in an environment based on data collected from IoT devices, these models must be interpretable. This means that the results should be comprehensible and explainable, allowing investigative professionals and stakeholders to understand how decisions are made and trust the results obtained. In summary, the results underscore that the Random Forest and XGBoost algorithms show the potential to detect room occupancy in IoT environments, highlighting their efficiency for investigation purposes. In this study, both models exhibited the best performance, validating their utility in analyzing data from IoT environments.

However, it is important to note that ML models should be used as a supplementary tool in the investigation process and that the interpretation of results and the consideration of other available evidence are essential. As technology related to IoT and DF continues to evolve, new opportunities and challenges will arise for the application of ML models. With proper data collection and analysis, these models can lead to significant advancements in the ability to detect, prevent, and elucidate crimes in IoT environments. Furthermore, it is essential that this evolution is accompanied by an ethical and responsible approach, which ensures the protection of individual privacy and the appropriate use of the collected information.

In conclusion, the results suggest that the use of ML models can be a promising approach to detecting patterns in IoT environments and assisting in criminal investigations, but their use must be exercised with caution, combining them with other techniques, and considering the specifics of each individual case.

It is important to note that part of the results found in Chapter 4 of this work are based on the research presented in the publication [16]: *'Machine Learning for Forensic Occupancy Detection in IoT Environments'* at the WorldCist 2024 conference. This recognition highlights the potential impact of our methodologies and findings, further enriching the ongoing academic discourse within the field of DF.

6.1 Future Work

In light of the challenges associated with sourcing a pertinent dataset for ML applications in people detection within an IoT environment, we will create a new comprehensive dataset. This data collection will be meticulously collected from a single room within the IoT environment, incorporating a wide range of sensory inputs. The primary objective of this data collection effort is to facilitate more thorough testing and validation, providing an improved understanding of how ML models can effectively identify patterns and anomalies within the unique context of room dynamics. This specialized dataset is expected to significantly advance forensic IoT analysis, providing additional fresh insights into this dynamic field.

To ensure the completeness of the dataset, we will incorporate various scenarios of room dynamics. This will include variations in occupancy levels, various lighting conditions, and a spectrum of human activities. By integrating these diverse scenarios, the dataset will closely emulate the complexities of real-world IoT environments, providing a robust foundation for testing ML models.

An emphasis will be placed on introducing unique and challenging scenarios, commonly referred to as edge cases. These carefully chosen instances will strategically assess the adaptability and resilience of ML models to handle unexpected situations, contributing to the refinement of their overall performance.

Recognizing the sensitivity of person detection data, we commit ourselves to implementing rigorous privacy protocols. Employing anonymization techniques and secure data handling practices will ensure ethical collection and use of data, upholding the highest standards of privacy protection.

Prioritizing the exploration of real-time data acquisition methods will allow the simulation of dynamic and evolving scenarios within the room. This approach allows for the evaluation of ML models in situations where the environment undergoes continuous changes, providing a more authentic assessment of their adaptability.

To increase the complexity of the data set, we plan to integrate multimodal data sources, including audio and video inputs. This comprehensive approach seeks to offer a holistic representation of the IoT environment, enhancing the models' ability to adapt to a diverse range of sensory inputs.

A crucial component of our research will involve benchmarking the performance of ML models trained on our newly generated dataset against established datasets commonly employed in similar studies. This comparative analysis aims to underscore the distinctive qualities and effectiveness of our dataset, contributing to the advancement of research in IoT forensics.

We are actively exploring the prospect of contributing our dataset to the open research community. By openly sharing the dataset, our intention is to foster collaboration, promote experiment reproducibility, and contribute to the establishment of standardized benchmarks in the field of IoT forensics.

REFERENCES

[1] Adedayo, O. M. “Big data and digital forensics”. In: 2016 IEEE International Conference on Cybercrime and Computer Forensic (ICCCF), 2016, pp. 1–7.

[2] Aggarwal, C. C. “Probabilistic and Statistical Models for Outlier Detection”. Cham: Springer International Publishing, 2017, pp. 35–64.

[3] Alpaydin, E. “Introduction to Machine Learning, fourth edition”. MIT Press, 2020.

[4] Atlam, H. F.; El-Din Hemdan, E.; Alenezi, A.; Alassafi, M. O.; Wills, G. B. “Internet of things forensics: A review”, *Internet of Things*, vol. 11, 2020, pp. 100220.

[5] Beebe, N. “Digital forensic research: The good, the bad and the unaddressed”. In: Advances in Digital Forensics V, Peterson, G.; Shenoi, S. (Editors), 2009, pp. 17–36.

[6] Bergstra, J.; Bengio, Y. “Random search for hyper-parameter optimization”, *J. Mach. Learn. Res.*, vol. 13–null, feb 2012, pp. 281–305.

[7] Bishop, C. M. “Pattern Recognition and Machine Learning (Information Science and Statistics)”. Berlin, Heidelberg: Springer-Verlag, 2006.

[8] Breiman, L. *Machine Learning*, vol. 45–1, 2001, pp. 5–32.

[9] Breiman, L.; Friedman, J.; Stone, C.; Olshen, R. “Classification and Regression Trees”. Taylor & Francis, 1984.

[10] Campos, M.; Caldas, R.; Buarque, F. “explainable and individualizable for physiotherapeutic decision support for the elderly”. In: 2021 IEEE International Conference on Systems, Man, and Cybernetics (SMC), 2021, pp. 1876–1881.

[11] Chapelle, O.; Schlkopf, B.; Zien, A. “Semi-Supervised Learning”. The MIT Press, 2010.

[12] Chawla, N. V.; Bowyer, K. W.; Hall, L. O.; Kegelmeyer, W. P. “Smote: synthetic minority over-sampling technique”, *Journal of artificial intelligence research*, vol. 16, 2002, pp. 321–357.

[13] Chen, T.; Guestrin, C. “Xgboost: A scalable tree boosting system”. In: Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining, 2016, pp. 785–794.

[14] Cook, D. J.; Crandall, A. S.; Thomas, B. L.; Krishnan, N. C. “Casas: A smart home in a box”, *Computer*, vol. 46–7, 2013, pp. 62–69.

[15] Cutler, A.; Cutler, D. R.; Stevens, J. R. “Random Forests”. New York, NY: Springer New York, 2012, pp. 157–175.

[16] Deconto, G. D.; Zorzo, A. F.; Dalalana, D. B.; OliveiraJr, E.; Lunardi, R. C. “Machine learning for forensic occupancy detection in iot environments”. In: WorldCIST 2024 Lecture Notes in Networks and Systems, 2024.

[17] Dey, A.; Ling, X.; Syed, A.; Zheng, Y.; Landowski, B.; Anderson, D.; Stuart, K.; Tolentino, M. E. “Namatad: Inferring occupancy from building sensors using machine learning”. In: 2016 IEEE 3rd World Forum on Internet of Things (WF-IoT), 2016, pp. 478–483.

[18] Estabrooks, A.; Japkowicz, N. “A mixture-of-experts framework for learning from imbalanced data sets”. In: International Symposium on Intelligent Data Analysis, 2001, pp. 34–43.

[19] Freund, Y.; Schapire, R. E. “A decision-theoretic generalization of on-line learning and an application to boosting”, *Journal of computer and system sciences*, vol. 55–1, 1997, pp. 119–139.

[20] Garfinkel, S. L. “Digital forensics innovation: Searching a terabyte of data in 10 minutes”, *DCACM*, Jan. 2013.

[21] Guarino, A. “Digital Forensics as a Big Data Challenge”. Wiesbaden: Springer Fachmedien Wiesbaden, 2013, pp. 197–203.

[22] Hair, J. F.; Black, W. C.; Babin, B. J.; Anderson, R. E.; Tatham, R. L. “Multivariate data analysis. uppersaddle river”, *Multivariate Data Analysis (5th ed) Upper Saddle River*, vol. 5–3, 1998, pp. 207–219.

[23] Hart, P. “The condensed nearest neighbor rule (corresp.)”, *IEEE transactions on information theory*, vol. 14–3, 1968, pp. 515–516.

[24] Hastie, T.; Tibshirani, R.; Friedman, J. H.; Friedman, J. H. “The elements of statistical learning: data mining, inference, and prediction”. Springer, 2009, vol. 2.

[25] hoon Kim, T.; Ramos, C.; Mohammed, S. “Smart city and iot”, *Future Generation Computer Systems*, vol. 76, 2017, pp. 159–162.

[26] Islam, M. J.; Mahin, M.; Khatun, A.; Debnath, B. C.; Kabir, S. “Digital forensic investigation framework for internet of things (iot): A comprehensive approach”. In: 2019 1st International Conference on Advances in Science, Engineering and Robotics Technology (ICASERT), 2019, pp. 1–6.

[27] Jayalakshmi, T.; Santhakumaran, A. “Statistical normalization and back propagation for classification”, *International Journal of Computer Theory and Engineering*, vol. 3–1, 2011, pp. 1793–8201.

[28] Kebande, V. R.; Karie, N. M.; Venter, H. S. “Cloud-centric framework for isolating big data as forensic evidence from iot infrastructures”. In: 2017 1st International Conference on Next Generation Computing Applications (NextComp), 2017, pp. 54–60.

[29] Kebande, V. R.; Ray, I. “A generic digital forensic investigation framework for internet of things (iot)”. In: 2016 IEEE 4th International Conference on Future Internet of Things and Cloud (FiCloud), 2016, pp. 356–362.

[30] Kim, J.; Kim, J. “The impact of imbalanced training data on machine learning for author name disambiguation”, *Scientometrics*, vol. 117–1, 2018, pp. 511–526.

[31] Kotsiantis, S. B. “Decision trees: a recent overview”, *Artificial Intelligence Review*, vol. 39, 2013, pp. 261–283.

[32] Kotsiantis, S. B.; Zaharakis, I.; Pintelas, P.; et al.. “Supervised machine learning: A review of classification techniques”, *Emerging artificial intelligence applications in computer engineering*, vol. 160–1, 2007, pp. 3–24.

[33] Kubat, M.; Matwin, S.; et al.. “Addressing the curse of imbalanced training sets: one-sided selection”. In: Icml, 1997, pp. 179.

[34] Laurikkala, J. “Improving identification of difficult small classes by balancing class distribution”. In: Artificial Intelligence in Medicine: 8th Conference on Artificial Intelligence in Medicine in Europe, AIME 2001 Cascais, Portugal, July 1–4, 2001, Proceedings 8, 2001, pp. 63–66.

[35] Leys, C.; Ley, C.; Klein, O.; Bernard, P.; Licata, L. “Detecting outliers: Do not use standard deviation around the mean, use absolute deviation around the median”, *Journal of experimental social psychology*, vol. 49–4, 2013, pp. 764–766.

[36] Mani, I.; Zhang, I. “knn approach to unbalanced data distributions: a case study involving information extraction”. In: Proceedings of workshop on learning from imbalanced datasets, 2003, pp. 1–7.

[37] Mitchell, T. “Machine Learning”. McGraw-Hill Education, 1997.

[38] Mohammed, H. J.; Clarke, N.; Li, F. “An automated approach for digital forensic analysis of heterogeneous big data”, *The Journal of Digital Forensics, Security and Law*, Jan. 2016.

[39] Mohammed, R.; Rawashdeh, J.; Abdullah, M. “Machine learning with oversampling and undersampling techniques: Overview study and experimental results”. In: 2020 11th International Conference on Information and Communication Systems (ICICS), 2020, pp. 243–248.

[40] Mohammed, R.; Rawashdeh, J.; Abdullah, M. “Machine learning with oversampling and undersampling techniques: overview study and experimental results”. In: 2020 11th international conference on information and communication systems (ICICS), 2020, pp. 243–248.

[41] Novak, M. “Improving the collection of digital evidence”. Source: https://nij.ojp.gov/topics/articles/improving-collection-digital-evidence#1-0, Dec 2021.

[42] Oriwoh, E.; Jazani, D.; Epiphaniou, G.; Sant, P. “Internet of things forensics: Challenges and approaches”. In: 9th IEEE International Conference on Collaborative Computing: Networking, Applications and Worksharing, 2013, pp. 608–615.

[43] Park, J.; Jang, K.; Yang, S.-B. “Deep neural networks for activity recognition with multi-sensor data in a smart home”. In: 2018 IEEE 4th World Forum on Internet of Things (WF-IoT), 2018, pp. 155–160.

[44] Quick, D.; Choo, K.-K. R. “Iot device forensics and data reduction”, *IEEE Access*, vol. 6, 2018, pp. 47566–47574.

[45] Raju, V. G.; Lakshmi, K. P.; Jain, V. M.; Kalidindi, A.; Padma, V. “Study the influence of normalization/transformation process on the accuracy of supervised classification”. In: 2020 Third International Conference on Smart Systems and Inventive Technology (ICSSIT), 2020, pp. 729–735.

[46] Ray, S. “A quick review of machine learning algorithms”. In: 2019 International conference on machine learning, big data, cloud and parallel computing (COMITCon), 2019, pp. 35–39.

[47] Richard, G. G.; Roussev, V. “Next-generation digital forensics”, *Communications of the ACM*, vol. 49–2, Feb. 2006, pp. 76–80.

[48] Safavian, S. R.; Landgrebe, D. “A survey of decision tree classifier methodology”, *IEEE transactions on systems, man, and cybernetics*, vol. 21–3, 1991, pp. 660–674.

[49] Sarker, I. H. “Machine learning: Algorithms, Real-World applications and research directions”, *SN Computer Science*, vol. 2–3, Mar. 2021, pp. 160.

[50] Schneider, G. “Foca-iot: Modelo integrado para forense digital em cenários envolvendo aplicações iot”, Master’s Thesis, Pontifícia Universidade Católica do Rio Grande do Sul, Porto Alegre, Rio Grande do Sul, 2023.

[51] Shalev-Shwartz, S.; Ben-David, S. “Understanding Machine Learning: From Theory to Algorithms”. USA: Cambridge University Press, 2014.

[52] Singh, A. P.; Jain, V.; Chaudhari, S.; Kraemer, F. A.; Werner, S.; Garg, V. “Machine learning-based occupancy estimation using multivariate sensor nodes”. In: 2018 IEEE Globecom Workshops (GC Wkshps), 2018, pp. 1–6.

[53] Sutton, R. S.; Barto, A. G. “Reinforcement Learning: An Introduction”. Cambridge, MA, USA: A Bradford Book, 2018.

[54] Tan, P.; Steinbach, M.; Karpatne, A.; Kumar, V. “Introduction to Data Mining”. Pearson, 2019.

[55] Vega, M. A. “Deeppatrol: Finding illicit videos for law enforcement”. Source: https://www.ojp.gov/ncjrs/virtual-library/abstracts/deeppatrol-finding-illicit-videos-law-enforcement, Jun 2019.

[56] Xu, R.; Wunsch, D. “Survey of clustering algorithms”, *IEEE Transactions on neural networks*, vol. 16–3, 2005, pp. 645–678.

[57] Yaacoub, J.-P. A.; Noura, H. N.; Salman, O.; Chehab, A. “Advanced digital forensics and anti-digital forensics for iot systems: Techniques, limitations and recommendations”, *Internet of Things*, vol. 19, 2022, pp. 100544.

[58] Yaqoob, I.; Hashem, I. A. T.; Ahmed, A.; Kazmi, S. A.; Hong, C. S. “Internet of things forensics: Recent advances, taxonomy, requirements, and open challenges”, *Future Generation Computer Systems*, vol. 92, 2019, pp. 265–275.

[59] Zawoad, S.; Hasan, R. “Digital forensics in the age of big data: Challenges, approaches, and opportunities”. In: 2015 IEEE 17th International Conference on High Performance Computing and Communications, 2015 IEEE 7th International Symposium on Cyberspace Safety and Security, and 2015 IEEE 12th International Conference on Embedded Software and Systems, 2015, pp. 1320–1325.

[60] Zhu, X.; Ghahramani, Z.; Lafferty, J. D. “Semi-supervised learning using gaussian fields and harmonic functions”. In: Proceedings of the 20th International conference on Machine learning (ICML-03), 2003, pp. 912–919.

Pontifícia Universidade Católica do Rio Grande do Sul
Pró-Reitoria de Pesquisa e Pós-Graduação
Av. Ipiranga, 6681 – Prédio 1 – Térreo
Porto Alegre – RS – Brasil
Fone: (51) 3320-3513
E-mail: propesq@pucrs.br
Site: www.pucrs.br