

Universidade do Minho
Escola de Engenharia

Leandro Lopes Cunha

Anomaly Detection in Cryptocurrency Transactions with Active Learning

July 2023

Anomaly Detection in Cryptocurrency Transactions with Active Learning
Leandro Lopes Cunha
UMinho | 2023



Universidade do Minho
Escola de Engenharia

Leandro Lopes Cunha

Anomaly Detection in Cryptocurrency Transactions with Active Learning

Master's Dissertation
Integrated Master's in Engineering and Management of Information Systems

Work performed under the supervision of
Professor Miguel Abrunhosa de Brito

July 2023

COPYRIGHT AND TERMS OF USE OF THE WORK BY THIRD PARTIES

This is an academic work that can be used by third parties as long as the internationally accepted rules and good practices are respected, with regard to copyright and related rights. Thus, the present work can be used under the terms foreseen in the license indicated below. If the user needs permission to be able to use the work under conditions not provided for in the indicated licensing, he/she should contact the author, through the RepositóriUM of the University of Minho.

Licença concedida aos utilizadores deste trabalho



Atribuição
CC BY
https://creativecommons.org/licenses/by/4.0/

ACKNOWLEDGMENTS

Upon reaching the culmination of this work, it is crucial to pause and engage in introspection, while also expressing profound gratitude to those who have played a pivotal role in the completion of this dissertation.

First, I would like to express my sincere gratitude to my supervisors at Uphold, Paulo Marques and Diogo Pinto, as well as to Professor Miguel Abrunhosa Brito from academy. Without their unwavering support and guidance, my dissertation would not have been possible. Throughout my research journey, they consistently offered valuable insights, constructive feedback, and motivational words. I am truly grateful for their dedicated efforts and continuous encouragement, which played a pivotal role in shaping the outcome of this work.

I am also incredibly appreciative of Uphold for welcoming me into their team, supporting this project financially, and equipping me with all the necessary computational resources to successfully accomplish this endeavor.

Finally, I would like to deeply express my profound gratitude to my family and girlfriend for their unwavering support throughout this journey. Their love, encouragement, and presence have been the driving force behind my every accomplishment, including this dissertation. I am truly blessed and forever grateful for their support. Thank you for always being there for me.

STATEMENT OF INTEGRITY

I hereby declare having conducted this academic work with integrity. I confirm that I have not used plagiarism or any form of undue use of information or falsification of results along the process leading to its elaboration.

I further declare that I have fully acknowledged the Code of Ethical Conduct of the University of Minho.

Assinado por: **Leandro Lopes Cunha**
Num. de Identificação: 15010104
Data: 2023.07.30 17:39:15 +0100

ABSTRACT

Anomaly Detection in Cryptocurrency Transactions with Active Learning

The cryptocurrency market has experienced tremendous growth in recent years, providing numerous opportunities for investors, innovators, and technological advancements. However, this rapid expansion has also led to an increase in fraudulent activities. As cryptocurrencies have become more popular and widely used, dishonest individuals have taken advantage of the decentralized and anonymous nature of these digital assets for their own illegal purposes. Consequently, there is a pressing need for the implementation of effective fraud detection mechanisms to safeguard the integrity of the crypto market.

This work aims to tackle the problem of insufficient labels, which poses a challenge for training high-performance supervised classifiers. The main idea is to reduce the need to manually label an enormous volume of transactions, which can be an arduous and time-consuming task for companies due to budget constraints. Therefore, this work assumes that there are labels of some unlabeled data points that are more informative and relevant to the supervised model to learn. Additionally, this work tries to develop a robust fraud detection approach to deal with the ever-evolving nature of criminal behavior, since criminals are constantly seeking innovative methods to commit their fraudulent crimes.

By addressing a cold start scenario, where there are no initial labeled transactions, this work investigates the feasibility of the synergistic utilization of unsupervised Anomaly Detection algorithms and Active Learning techniques to create an iterative process for acquiring labeled transactions. The primary aim is to explore the capabilities of Anomaly Detection in selecting a subset of data that maximizes the supervised models' learning potential.

The results of this study demonstrated that Anomaly Detection algorithms exhibited subpar performance when it comes to identifying relevant patterns associated with cryptocurrency fraud in unlabeled data. The findings emphasize that Anomaly Detection algorithms are only necessary for addressing cold start scenarios. Therefore, it was concluded that switching as soon as possible to more sophisticated Active Learning strategies would lead to superior results and improved performance from a supervised Machine Learning model. Remarkably, Random Forest achieved an optimal F1-score performance of 83% with as few as 700 labels.

Keywords: Active Learning; Anomaly Detection; Cryptocurrencies; Fraud Detection.

RESUMO

Deteção de Anomalias em Transações de Criptomoedas com Aprendizagem Ativa

O mercado de criptomoedas cresceu muito nos últimos anos, proporcionando oportunidades para investidores, inovadores e avanços tecnológicos. No entanto, esse crescimento também resultou num aumento das atividades fraudulentas. Com o aumento da popularidade das criptomoedas, muitos estão a aproveitar-se da natureza descentralizada e anónima destes ativos digitais para fins ilegais. Portanto, é urgente implementar mecanismos eficazes de deteção de fraudes para proteger a integridade do mercado de criptomoedas.

Este trabalho tem como objetivo enfrentar o problema de rótulos insuficientes, o que representa um desafio para treinar modelos supervisionados. A ideia principal é reduzir a necessidade de rotular manualmente um enorme volume de transações, o que pode ser uma tarefa árdua e demorada. Portanto, este trabalho pressupõe que existem rótulos de alguns pontos de dados não rotulados que são mais informativos para o modelo supervisionado aprender. Além disso, este trabalho tenta desenvolver uma abordagem robusta de deteção de fraudes para lidar com a natureza em constante evolução do comportamento criminoso.

Ao abordar um cenário de partida a frio, onde inicialmente não há transações rotuladas, este trabalho investiga a viabilidade da utilização sinérgica de algoritmos de Deteção de Anomalias e Aprendizagem Ativa para criar um processo iterativo de aquisição de transações rotuladas. O objetivo principal desta abordagem é explorar as capacidades de algoritmos de Deteção de Anomalias na seleção de um subconjunto de dados que maximize o potencial de aprendizagem dos modelos supervisionados de aprendizagem de máquina.

Os resultados deste trabalho mostraram o fraco desempenho dos algoritmos de Deteção de Anomalias na identificação de padrões relevantes relacionados à fraude de criptomoedas. Os resultados deste estudo enfatizam que os métodos de Deteção de Anomalias são necessários apenas para lidar com cenários de partida a frio. Portanto, concluiu-se que a mudança o mais rápido possível para estratégias de Aprendizagem Ativa mais sofisticadas levaria a um desempenho aprimorado de um modelo supervisionado. Notavelmente, Random Forest obteve um F1-score de 83% com apenas 700 rótulos.

Palavras-chave: Aprendizagem Ativa; Criptomoedas; Deteção de Anomalias; Deteção de Fraude.

INDEX

Copyright and Terms of Use of the Work by Third Parties ii

Acknowledgments iii

Statement of Integrity iv

Abstract v

Resumo vi

List of Abbreviations and Acronyms xi

List of Figures xii

List of Tables xiii

1. Introduction 1

1.1 Problem 1

1.2 Motivation 2

1.3 Objectives 4

1.4 Research Methodology 5

1.5 Document Structure 8

2. Theoretical Framework 9

2.1 Cryptocurrency Fraud 9

2.2 Fraud Detection 10

2.3 Machine Learning 11

2.4 Anomaly Detection 12

2.4.1 Feature Selection 13

2.4.2 Main Methods 14

2.5 Active Learning 16

2.5.1 Relevance of using Anomaly Detection for Active Learning 17

2.5.2 Active Learning Policies 18

3. Literature review 20
3.1 Traditional Approaches to Fraud Detection 20
3.1.1 Supervised Methods 20
3.1.2 Anomaly Detection Techniques 21
3.2 Approaches to Active Learning with Anomaly Detection 23
3.2.1 General Domain Applications 23
3.2.2 Financial Domain Applications 24
4. Materials and Tools 27
4.1 Development Tools 27
4.2 Dataset 29
4.2.1 Data Structure 31
4.2.2 Exploratory Data Analysis 32
5. Machine Learning Algorithms Selection 36
5.1 Supervised Learning 37
5.1.1 Random Forest 37
5.1.2 Logistic Regression 39
5.1.3 Extreme Gradient Boosting 40
5.2 Unsupervised Active Learning Policies 41
5.2.1 Isolation Forest 42
5.2.2 Local Outlier Factor 43
5.2.3 Elliptic Envelope 45
5.3 Supervised Active Learning Policies 46
5.3.1 Uncertainty Sampling 47
5.3.2 Query By Committee 49
5.3.3 Expected Model Change 50

5.4 Evaluation Metrics 52
5.4.1 Precision 53
5.4.2 Recall 53
5.4.3 F1-score 54
6. Experiments 55
6.1 Data Pre-processing 55
6.2 Experimental Setup 56
6.2.1 Supervised Baseline 57
6.2.2 Active Learning 58
7. Results and Discussion 65
7.1 Supervised Baseline 65
7.2 Active Learning 68
7.2.1 Scenario 1 - Unsupervised Active Learning Policies 69
7.2.2 Scenario 2 - Combining Unsupervised and Supervised Active Learning Policies 73
7.3 Results Analysis and Key Findings 84
8. Application of The Method to Uphold's Proprietary Dataset 88
8.1 Dataset 88
8.1.1 Dataset Description 89
8.1.2 Pre-Processing 90
8.1.3 Feature Engineering 91
8.2 Dataset Applicability Discussion 96
8.3 Method Application 97
8.4 Implementation Difficulties 98
9. Conclusion and Future Work 100
References 105

Appendix I – Supervised Baseline Models' Parameters 113

Appendix II – Anomaly Detection Models' Parameters 114

Appendix III – Uncertainty Sampling Models' Parameters..... 115

Appendix IV – Query By Committee Model Parameters..... 116

Appendix V – Expected Model Change Model Parameters..... 117

List of Abbreviations and Acronyms

AD	Anomaly Detection
AL	Active Learning
AUROC	Area Under the Receiver Operating Characteristic
DT	Decision Tree
DSR	Design Science Research
EGL	Expected Gradient Length
EMC	Expected Model Change
FD	Fraud Detection
FN	False Negative
FP	False Positive
FPR	False Positive Rate
IF	Isolation Forest
KL	Kullback-Leibler
LOF	Local Outlier Factor
LR	Logistic Regression
ML	Machine Learning
ODAL	Outlier Discriminative Active Learning
OSVM	One-Class Support Vector Machine
PCA	Principal Component Analysis
PyOD	Python Outlier Detection
QBC	Query By Committee
RF	Random Forest
SVM	Support Vector Machine
t-SNE	t-Distributed Stochastic Neighbor Embedding
TP	True Positive
TPR	True Positive Rate
UTXO	Unspent Transaction Output
XGBoost	eXtreme Gradient Boosting

LIST OF FIGURES

Figure 1: Taxonomy of Anomaly Detection Methods 15
Figure 2: Simplified Structure of an Active Learning Process 17
Figure 3: Bitcoin UTXO Transaction Model 30
Figure 4: Number of Transactions by Class 33
Figure 5: Number of Transactions by Time Step 33
Figure 6: Number of Transactions for Each Class in Each Time Step 34
Figure 7: Number of Transactions for Licit and Illicit Classes in Each Time Step 34
Figure 8: t-SNE Representation of the True Labels 35
Figure 9: Simplified Structure of Random Forest 38
Figure 10: Sigmoid Function 39
Figure 11: Simplified Structure of Extreme Gradient Boosting 41
Figure 12: Isolating Data Points with Isolation Forest 43
Figure 13: Local Outlier Factor Neighborhood Reachability Distance 45
Figure 14: Simplified Structure of Elliptic Envelope 46
Figure 15: Simplified Structure of Uncertainty Sampling Policy 49
Figure 16: Simplified Structure of an Active Learning Setup 62
Figure 17: Supervised Baseline Metrics by Threshold 65
Figure 18: t-SNE Projection on the Test Set, Colored by the True Labels 66
Figure 19: t-SNE Projection on the Test Set, Colored by the Predicted Labels 67
Figure 20: F1-score by Time Step, for Each Supervised Classifier 68
Figure 21: t-SNE Projection on the Train Set, Colored by the True Labels 73
Figure 22: Learning curves for Random Forest Setups (Cut-off - 30th Iteration) 76
Figure 23: Learning curves for Extreme Gradient Boosting Setups (Cut-off - 30th Iteration) 77
Figure 24: Learning Curves for Logistic Regression Setups (Cut-off - 30th Iteration) 79
Figure 25: Learning curves for Random Forest Setups (Cut-off – 1st Iteration) 81
Figure 26: Learning Curves for Extreme Gradient Boosting Setups (Cut-off – 1st Iteration) .. 83
Figure 27: Learning Curves for Logistic Regression Setups (Cut-off – 1st Iteration) 84

LIST OF TABLES

Table 1: Supervised Baseline Results 66
Table 2: Active Learning Setups for Random Forest 69
Table 3: Active Learning Setups for Extreme Gradient Boosting 69
Table 4: Active Learning Setups for Logistic Regression 70
Table 5: Active Learning Setups at 1% Illicit Rate 72
Table 6: Active Learning Setups for Random Forest (Cut-off - 30th Iteration) 74
Table 7: Active Learning Setups for Extreme Gradient Boosting (Cut-off - 30th Iteration) 77
Table 8: Active Learning Setups for Logistic Regression (Cut-off - 30th Iteration) 78
Table 9: Active Learning Setups for Random Forest (Cut-off – 1st Iteration) 80
Table 10: Active Learning Setups for Extreme Gradient Boosting (Cut-off – 1st Iteration) 82
Table 11: Active Learning Setups for Logistic Regression (Cut-off – 1st Iteration) 83
Table 12: Uphold's Dataset Constitution 89
Table 13: Number of Transactions by Class 90
Table 14: Transactional Attributes Description 92
Table 15: Network Features Description 93
Table 16: User Behavior Features Description 94
Table 17: Supervised Classifiers Parameters 113
Table 18: Unsupervised Anomaly Detection Models' Parameters 114
Table 19: Uncertainty Sampling Models' Parameters 115
Table 20: Query by Committee Model Parameters 116
Table 21: Expected Model Change Model Parameters 117

1. INTRODUCTION

The technological evolution that it has been experienced in the past years, with the emergence of new computing technologies makes Machine Learning (ML) today different from ML of the past. Nowadays, this type of technology has been applied to almost every type of business, due to its ability to extract patterns from the data that would help organizations identify profitable business opportunities or avoid risks.

In parallel with technological development, there is a growing concern related to the detection of fraudulent behavior in financial transactions. Therefore, despite all the effort and investment put into fraud prevention, there is still a need for efficient Fraud Detection (FD) mechanisms. These approaches have been extensively used in traditional financial systems to detect unusual fraudulent behaviors.

The exponential growth of the cryptocurrency market in recent years has made it a widely accepted form of digital currency. However, this surge in popularity has also attracted a host of fraudulent activities, such as money laundering, phishing, Ponzi schemes, and insider trading, thereby creating a pressing need to safeguard the integrity of the cryptocurrency market, opening significant opportunities for exploring FD mechanisms in this field.

This dissertation is part of a work done in Uphold. Uphold is a multi-asset digital money platform offering financial services to a global market, allowing users to transact digital currencies. Therefore, this work will focus on the provision of FD mechanisms that would allow Uphold to identify novel fraudulent behaviors in cryptocurrency transactions.

1.1 Problem

There are three main problems that justify this work. The first problem revolves around the scarcity of labeled data within the FD domain, rendering it impractical to train highly effective supervised ML models using historical information. Specifically, within the financial domain, companies often lack labeled datasets, meaning that they do not possess transaction-level labels indicating whether each transaction is licit or illicit.

Furthermore, the arduous and time-consuming nature of labeling an enormous volume of transactions makes it an impractical undertaking for organizations. This happens due to the

complex nature of fraudulent behavior. As a result, the second problem becomes intricately linked to the first. Given the time and budget constraints faced by companies when reviewing transactions, there is an urgent requirement to develop mechanisms that empower analysts to concentrate their investigative efforts on transactions that are most likely to involve fraud. By doing so, valuable resources and time can be saved by bypassing the examination of low-information data instances that are unlikely to provide significant insights.

Finally, Uphold faces an ongoing challenge due to the ever-evolving nature of criminal behavior. In today's world, criminals are constantly seeking innovative methods to commit fraudulent crimes. This issue is particularly prominent in the cryptocurrency markets, where criminals exploit the vulnerabilities of cryptocurrencies. The lack of regulation and the anonymity provided by cryptocurrencies make them attractive for carrying out fraudulent activities. Uphold, as a company operating in this space, is susceptible to the risks associated with illicit activities. Such risks can have severe consequences, compromising the trust of its users and exposing the company to compliance laws and regulations. Therefore, it is crucial to adopt effective approaches that can detect these new fraudulent behaviors.

1.2 Motivation

Training high-performance supervised ML models is an essential task in the FD domain, due to their high ability to learn from known fraudulent patterns. However, the lack of labeled data to train these ML models poses a significant challenge.

To address this challenge, a combination of Anomaly Detection (AD) and Active Learning (AL) techniques can be valuable. AD algorithms are designed to identify patterns in data that deviate from normal transactional behavior (Injadat et al., 2018). By analyzing the data, AD algorithms can flag instances that exhibit unusual or anomalous behavior, indicating potential fraud. The remaining instances that conform to normal behavior are considered non-anomalous or normal. AD algorithms explore the data without prior assumptions. This allows them to uncover hidden relationships, clusters, or anomalies that may not be captured by predefined labels or known patterns. They can provide valuable insights into the data, including the identification of novel or emerging anomalies/frauds.

Therefore, in the context of limited labeled data, AD can help alleviate this issue by identifying anomalous behavior within the unlabeled data. This means that even without explicit labels,

AD can highlight instances that are likely to be fraudulent or suspicious. By leveraging AD, it is possible to narrow down the focus of labeling efforts to those instances that are flagged as most anomalous, potentially saving time and resources.

This is where AL comes into play. AL is a technique that aims to select a subset of data points from the unlabeled data for which the supervised model's performance would benefit the most if they were labeled and included in the training process (Settles, 2012). Thus, there can be immense value in incorporating AD algorithms within AL setups, when there are no initial labels available, known as cold start scenarios.

AL has the potential to combine the benefits of unsupervised methods (exploration) and supervised methods (exploitation) by iteratively selecting the most valuable data points to be labeled by an oracle (i.e., human analyst) (Lorenz et al., 2020). In this case, these data points correspond to the most anomalous transactions and are iteratively presented to an analyst for review based on rankings provided by the anomaly detector. After the feedback from the analyst, the labeled data can be used to train a supervised ML model that learns fraud.

The advantage of AL is that it allows for a human loop where a set of transactions identified by the anomaly detector is reviewed and assigned the correct class label by an analyst. These labeled instances are then added to a pool of labeled data.

This means that by combining AD and AL, the most anomalous instances, which are more likely to contain the most relevant fraud-related patterns for the supervised ML model to learn, can be prioritized for manual labeling. This approach enables more efficient utilization of labeled data, resulting in a system that performs similarly to a supervised baseline while effectively detecting fraudulent behavior with minimal labeling effort.

Therefore, since this work primarily addresses cold start scenarios, it is reasonable to start the AL process by exploring the full potential of AD algorithms (unsupervised nature). However, it is important to note that these unsupervised algorithms are typically employed only up to a certain stage in the AL process. At this point, it generally becomes more advantageous to transit to more sophisticated AL strategies, that rely on supervised algorithms predictions to find the most relevant fraudulent transactions for labeling.

This being said, and because Uphold operates in a very complex business, where many players are interacting and where there can be immense value in identifying anomalies as well as understanding how users interact with each other, it is easily understandable the importance

of using ML algorithms for mining and automating large volumes of data, as a way to help Uphold to continuously be a company with a strong focus on regulation and risk management. Thus, the main motivation for this project will be to extract the best of both unsupervised and supervised worlds and efficiently detect fraudulent patterns in cryptocurrency transactions, by identifying data points that do not conform to the normal transaction patterns and that may indicate fraudulent behavior. This would help to gain insights that can be applied in the Uphold platform to catch possible illicit transactions.

1.3 Objectives

Assuming a cold start scenario, it is intended with this work to assess the feasibility of using AD algorithms and AL techniques to uncover fraudulent patterns in cryptocurrency transactions. Thus, the main objective is to gain insights into the practical relevance of using AD approaches with AL, to iteratively gather relevant labeled transactions. The purpose of gathering these labels is to train and improve the performance of a supervised classifier, in order to achieve, optimal performance, using a minimal number of labels. Therefore, trying to assess the feasibility of using these methods to select a subset of labeled data that maximizes the supervised model's performance.

Associated with this purpose, it will be necessary to conclude some steps to achieve the main goal of this project:

- Conduct research on the key concepts related to this work, such as cryptocurrency fraud, FD, AD, and AL, to have a clear understanding of the topic of this work.
- Perform a literature review on the use of AD, supervised methods, and AL algorithms, to detect new fraudulent patterns in financial transactions. The review should include the techniques and datasets used in recent studies on these topics, as well as their conclusions and any previously discovered insights about fraudulent behavior.
- Obtain a representative dataset of cryptocurrency transactional data, that will allow the simulation of the AL process, and serve as the basis for the analysis.
- Based on the analysis of existing literature, identify an appropriate AL framework that will enable the analysis of the viability of combining AD methods with AL techniques.

- Select a set of algorithms to be implemented in the experiments and develop a comprehensive understanding of their functionality and application.
- Implement a range of supervised ML models that assume to have complete access to all available data for training purposes. These models will be used as a baseline benchmark to evaluate the performance of the AL setups that incorporate AD techniques.
- Evaluate the impact of employing AD algorithms as unsupervised AL strategies for identifying relevant fraudulent transactions within an unlabeled pool. It is intended to determine whether labeling these transactions can improve the overall performance of a supervised classifier evaluated in the test set.
- Assess if there is a certain cut-off point or threshold in the AL process up to which it is more beneficial to use AD algorithms, before switching to a more sophisticated AL policy, that relies on supervised algorithms to find the most relevant transactions to be labeled.
- Discuss the suitability and efficacy of the devised approach using Uphold's exclusive dataset. This entails identifying essential resources, delineating the necessary data pre-processing procedures, and detailing the processes involved in training and evaluating the models.

1.4 Research Methodology

DSR is the methodology employed in the development of this work. The DSR approach is a research methodology that seeks to generate and evaluate solutions to practical problems through the utilization and implementation of theoretical concepts and techniques. The primary objective of DSR is to enhance the comprehension of a specific domain and to create new artifacts, methods, and tools that can be utilized to address practical problems and enhance the welfare of individuals or organizations. Hence, this methodology is highly appropriate for this project, as the primary objective of this research is to tackle fraud in the domain of cryptocurrencies, using AD and AL algorithms. This includes addressing problems such as the scarcity of labels for training supervised ML models and the dynamic nature of fraudulent patterns. DSR is a multistage process that includes the following phases:

1. **Identifying the problem or opportunity**: This step involves identifying the problem or opportunity that the research will address. This includes defining the problem, the comprehension, and definition of the context/motivation for the research and defining the objectives.
2. **Literature review**: This phase involves the examination of existing literature pertaining to the subject matter, with the aim of identifying areas of knowledge that have yet to be explored and attaining a more comprehensive understanding of the problem or opportunity at hand.
3. **Design and development**: This stage involves the design and creation of a solution to the problem at hand. This may include the development of a prototype, in addition to outlining the procedures and techniques that shall be employed to evaluate the solution.
4. **Evaluating the solution**: Subsequently, the solution is evaluated in relation to the objectives and the problem or opportunity that was recognized. This process entails the examination of the results and discusses their implications.
5. **Communicating the results**: The final step is to disseminate the findings of the research in a dissertation, research paper, or other scholarly work, which should include an examination of the limitations, as well as potential directions for future research.

Thus, the DSR methodology can be very beneficial in multiple areas such as information systems, computer science, engineering, and management.

Follows a description correlating the objectives of this project and the phases of the DSR:

1. **Identifying the problem or opportunity**: The initial phase is vital in ensuring a clear understanding of the project's purpose and objectives. By carefully defining and analyzing the problem at hand, it is possible to ensure that the efforts are focused on addressing specific issues and achieving measurable outcomes that will bring value to the organization. In this phase, it is essential to establish the problem's significance, which serves as the driving force for the project's development. With a clear understanding of the problem and motivation, it is possible to set specific, measurable, and achievable objectives that align with the project's main goal and guide the project toward successful completion. To do so, it becomes paramount to grasp the

fundamental principles intertwined with the project, thus gaining profound insights into the realm of exploration, and staying abreast of the latest advancements in the field.

2. **Literature Review**: This phase is crucial for gaining a comprehensive understanding of the main techniques and approaches that have been previously explored by other authors in relation to the same field of experimentation. The objective of this phase is to gain a deeper understanding of the AD and AL techniques that have been tested and to gather the key issues and insights of the proposed approaches. This information will aid in the development and design of the current study.
3. **Design and development**: Building upon the literature review conducted on the topic of the project, the main gaps in existing studies will be identified as a means of proposing a novel solution that contributes to adding knowledge to the literature on AD and AL techniques. The proposed methodology aims to address identified problems and achieve the main objectives previously established. As such, this phase involves a clear definition of the AD and AL techniques that will be employed and the specifics of the AL process setup that will be evaluated. Additionally, it is necessary to select and describe a representative dataset, as well as the definition of a pipeline for the implementation. It is crucial to have a well-defined methodology to ensure that the proposed solution is sound and can effectively address the research objectives. After the methodology is established, it will be implemented.
4. **Evaluation**: The proposed methodology will be evaluated by comparing its results with those of previous studies that have employed similar techniques, specifically AD and AL. This comparison will be used to determine the effectiveness of the evaluated solutions in uncovering and detecting new patterns of fraudulent behavior in cryptocurrency transactions. In other words, this evaluation process aims to determine the feasibility of the proposed approaches and their potential to contribute new knowledge to the field of AD and AL.
5. **Communicating the results**: Finally, the main results obtained with this project will be specified and shared in a master's dissertation document. Additionally, an article will be submitted to a conference or journal.

1.5 Document Structure

In addition to this introductory chapter, this dissertation consists of eight additional chapters. Chapter 2 provides a comprehensive overview of the key concepts relevant to the dissertation. It covers topics such as cryptocurrency fraud, the FD domain, AD, and AL. Chapter 3 presents a literature review that examines the potential use of traditional ML approaches in the field of FD. It also discusses the main approaches combining AD and AL techniques.

In Chapter 4, detailed descriptions of the tools and materials used in the research are provided. This includes explanations of the employed tools, the dataset utilized, and the corresponding features and distributions.

Chapter 5 delves into the ML algorithms used in the study, offering comprehensive explanations of each algorithm and its functioning principles.

Chapter 6 offers a detailed account of the conducted experiments. It covers the pre-processing procedures and provides explanations of the experimental setup, including implementation details of each algorithm and the chosen parameters.

Chapter 7 presents the results obtained from the experiments and includes an in-depth discussion of the findings.

Chapter 8 presents a description of Uphold's proprietary dataset and discusses the potential applicability and efficacy of the proposed methodology.

Finally, Chapter 9 concludes the dissertation and discusses potential future work, highlighting areas for further research and development.

2. THEORETICAL FRAMEWORK

2.1 Cryptocurrency Fraud

With the rapid advancement of technology, the problem of financial fraud has gained greater urgency. The widespread adoption of digital transactions across numerous industries has unfortunately given rise to a corresponding increase in electronic criminal activities (Vynokurova et al., 2020). As a result, financial fraud has become increasingly prevalent, leading to significant annual financial losses (Xuan et al., 2018). One of the main contributing factors is the lack of a physical card requirement in online transactions, making it easier for fraudsters to use only stolen information to complete transactions, as it happens in traditional transactions (Xuan et al., 2018).

Financial fraud poses a significant challenge, and this issue extends to the cryptocurrency markets in which Uphold operates. Cryptocurrency markets are especially vulnerable to different types of scams and fraudulent activities. This is due to the lack of regulation stemming from the decentralized nature of the market (Trozze et al., 2022) and the anonymity that virtual currencies provide to criminals and their transactions (Sabry et al., 2020), as users can own multiple addresses and can only be identified by them (Yang et al., 2019). According to Trautman (2014), these virtual currencies provide the means to swiftly and confidently transfer illicit funds across borders while benefiting from extensive acceptance, and a reputation for reliability.

Criminals often take advantage of these specific cryptocurrencies' fragilities to commit their financial crimes, as happens with money laundering (Korejo et al., 2021). These types of fraudulent activities can take various forms, that can include Ponzi schemes, phishing scams, impersonation of legitimate crypto exchanges or wallets, and money laundering.

Cryptocurrency-related financial fraud has also been linked to funding illicit activities such as organized crime and drug trafficking (West & Bhattacharya, 2016). These crimes result in significant losses for companies, which often bear the cost of fraud.

Thus, developing effective strategies for identifying and preventing fraudulent behavior in transactional data is crucial for companies such as Uphold (Hilal et al., 2022). In other words, the lack of regulation and oversight in the cryptocurrency market, coupled with the ease with

which fraudsters can perform fraudulent activities with virtual currencies, makes it essential for investors and companies to be vigilant and take proactive measures to protect themselves from financial fraud. This means that, despite all the effort and investment put into fraud prevention, there is still a need for the adoption of FD mechanisms, making advantageous use of the available data (Jeragh & Alsulaimi, 2019).

However, it is important to note that financial fraud using cryptocurrencies can be particularly difficult to detect and prevent, due to the decentralized and anonymous nature of these currencies. Moreover, criminals today try to obfuscate their fraudulent activities by imitating normal transactional behavior, which makes this task even harder (Lorenz et al., 2020).

2.2 Fraud Detection

The field of financial FD involves identifying suspicious behaviors associated with criminal activities in various types of businesses, such as banks, credit card companies, stock markets, cryptocurrency markets, and insurance companies (Nassif et al., 2021). Therefore, there is a significant body of literature dedicated to exploring the implementation and effectiveness of FD mechanisms across various application domains.

FD mechanisms have been widely studied to identify new patterns of behavior associated with financial fraud. The discovery of these patterns is crucial for financial institutions as it helps them to prevent and limit various types of financial crimes (Sabry et al., 2020).

Financial fraud can then be defined as an intentional act of deception for personal gain, which is both a criminal and civil law violation (Zhou et al., 2021). It's then reasonable to conclude that financial fraud is a problem that affects not only the financial sector but also everyday life. As such, these crimes can undermine integrity, particularly in the financial sector, also compromising the customer's trust and leaving the companies exposed to compliance laws and regulations (D. Choi & Lee, 2018).

There are many alternative ways to fight financial fraud. Therefore, it is important to clearly define and distinguish the terms FD and fraud prevention. Fraud prevention refers to the first line of defense and encompasses measures to prevent fraud from occurring. FD, on the other hand, is the next layer of protection and aims to quickly identify fraudulent activities once they have already occurred (Abdallah et al., 2016). However, because of the dynamic and complex nature of fraudulent behavior, the development of effective FD mechanisms,

leveraging available data, will always be necessary as it is impossible to completely prevent all types of fraud (Jeragh & Alsulaimi, 2019). In other words, criminal behavior continues to evolve as criminals find new ways to carry out their fraudulent activities. Given this, it is crucial to continuously improve these methods to effectively detect and address new patterns of fraudulent behavior (Shayegan et al., 2022).

2.3 Machine Learning

Particularly, ML algorithms play a crucial role in identifying fraudulent patterns in transactional data due to their ability to perform tasks involving high-level pattern recognition (R. Y. Choi et al., 2020). There are essentially three fundamental approaches to the FD problem. The first type is related to determining fraudulent behaviors in data when there is no prior knowledge, that is, unsupervised approaches. In other words, the unsupervised ML approaches are not provided with labeled data. Instead, the algorithm is designed to find patterns or relationships within the data on its own. The second approach is known as supervised learning, which relies on pre-labeled data to train the ML model, where fraudulent and normal instances have already been identified. The goal is to learn a function that maps inputs to outputs based on the labeled examples provided in the training data. The model is then trained to predict the outcome, given the input data. Lastly, semi-supervised learning is a type of ML technique that leverages both labeled and unlabeled data during the training process. Unlike supervised learning, which relies solely on labeled data, semi-supervised learning takes advantage of a small set of labeled data along with a larger set of unlabeled data to enhance the model's performance. This approach is particularly useful in scenarios where obtaining labeled data is expensive or time-consuming (Omar et al., 2013).

Even though training supervised ML models is a crucial task in the FD domain, due to their high performance in detecting known frauds, they are more difficult to implement in real-world scenarios since companies do not possess large-scale labeled datasets that would allow these models to be trained (Khan & Hoey, 2017). Therefore, since unsupervised AD algorithms do not need any labels to be trained, they are a possible solution to identify these fraudulent patterns within data.

Particularly, k-means clustering, Isolation Forest (IF), One-class Support Vector Machine (OSVM), Principal Component Analysis (PCA), or Local Outlier Factor (LOF) are popular

algorithms among AD. On the other hand, Random Forest (RF), Decision Tree (DT), Logistic Regression (LR), Support Vector Machines (SVM), and k-nearest neighbors are popular supervised classifiers used to detect fraudulent transactions (Hilal et al., 2022).

When evaluating the effectiveness of an FD system, several important measures should be taken into consideration. These include metrics such as precision (i.e., the proportion of True Positive (TP) predictions), Recall (i.e., the proportion of actual positive instances that are correctly detected), specificity (i.e., the proportion of correctly negative classified samples), and F1-score (i.e., the harmonic mean of Precision and Recall) (Hilal et al., 2022).

Additionally, one of the main challenges in identifying fraudulent behavior in transactional data is related to the highly imbalanced nature of the datasets, as typically there are far more legitimate records than fraudulent ones (Kong et al., 2020). Therefore, in highly imbalanced datasets, a high accuracy alone may not indicate that a model is useful, as the model may only correctly predict legitimate transactions and still have high accuracy. For this reason, accuracy is not the most useful metric in this context. Instead, it is more beneficial to analyze the trade-offs between Recall and False Positive Rate (FPR) at different threshold values. However, accuracy is still one of the metrics that most authors highlight to evaluate the effectiveness of FD systems.

2.4 Anomaly Detection

AD is a crucial area of research with a wide range of applications. As defined by Injadat et al. (2018), AD is the process of identifying patterns in data that deviate from the expected normal behavior. These systems are based on the premise that abnormal behavior is distinct from typical behavior. Anomalies are deviations from the typical patterns, which are not consistent with the majority of observations (normal) and are considered rare or unique occurrences, with little to no common characteristics. Anomalies can occur in data for various reasons, depending on the application domain. The primary objective of AD approaches is to create a profile of normal behavior and classify a data instance as anomalous if it deviates from this norm (Xuan et al., 2018).

Unlike imbalanced classification, AD is an unsupervised technique that aims to identify outliers in a dataset. It is mainly used as an unsupervised technique because its main application is in scenarios where there are no labeled data to train a supervised ML model and in highly

imbalanced scenarios, as happens in the FD domain. In this specific context, these methods assume that the most deviant data instances can potentially indicate fraudulent behavior.

Nowadays, AD is a highly valuable tool for analyzing large datasets. As a result, many businesses, including insurance companies, banks, hospitals, and manufacturing industries, have implemented AD systems to assist in identifying anomalies in their daily information (Pichara & Soto, 2011). One common reason in recent times is related to system failures, where anomalous traffic patterns in a computer network can indicate a potential security breach or compromise. In healthcare, AD methods can be used to identify abnormal cells, such as cancerous ones. In transactional data, anomalies can help detect fraudulent activity by identifying abnormal patterns in the data (Hilal et al., 2022), which will be the focus of this work.

Essentially, anomalies can be categorized in three ways: point anomaly, contextual anomaly, and collective anomaly. A point anomaly or global anomaly is when a single data instance deviates significantly from the normal pattern of the entire data. These types of anomalies are the ones addressed by the majority of the AD algorithms. A contextual anomaly is when data instances behave anomalously in a specific context or when there are analyzed using only the surrounding data points. These contextual outliers could also be generalized as local outliers, which correspond to when a specific data instance's density deviates from the local area in which it occurs. Lastly, a collective anomaly is when a group of similar instances act anomalously when compared to the whole dataset (Ahmed et al., 2016).

Another issue to consider is how anomalies are represented in the output. This can be in the form of a score indicating the likelihood of a data instance being anomalous or a label assigning each data instance as either normal or anomalous (Ahmed et al., 2016).

Identifying anomalous patterns or events in data is crucial due to their potential to provide significant, actionable, and critical information (Nassif et al., 2021).

2.4.1 Feature Selection

Many authors emphasize the significance of selecting an appropriate feature set for accurate AD. Thudumu et al. (2020) highlight that this process offers several benefits, including reducing dimensionality, enhancing model performance, and improving interpretability. Adding more dimensions to a dataset increases its complexity, leading to a higher number of

False Positives (FP). Moreover, excessive dimensions can introduce complications related to noise levels, including irrelevant features or unnecessary attributes, potentially obscuring genuine instances of fraudulent data.

Additionally, some authors also state that creating certain derived features or transforming existing ones, through feature engineering, can capture underlying patterns, relationships, or anomalies indicative of fraudulent behavior that may not be apparent in the original data. This process could potentially help to improve the discriminative power of the AD models (Jeyakumar et al., 2023).

In the field of cryptocurrency markets, researchers often focus on exploring features related to price variations and trading volumes to detect anomalies and potentially fraudulent activities. Tracking price variations and analyzing measures of price volatility, such as standard deviation or average true range, can be valuable in identifying abnormal price behavior and potential market manipulation, which may indicate fraudulent patterns (Kaufman & Iaremenko, 2022). Similarly, analyzing trading volumes can help uncover abnormal liquidity patterns or unusual transactional activities that could be indicative of fraudulent practices (Kaufman & Iaremenko, 2022).

However, some authors also alert that detecting anomalies in cryptocurrency markets presents some specific considerations. For example, according to Jeyakumar et al. (2023), one of the challenges stems, exactly, from the high price volatility commonly observed in these markets, with significant fluctuations occurring within short periods. This volatility could potentially pose a difficulty in distinguishing genuine anomalies from regular market behavior. Additionally, cryptocurrency markets are susceptible to various forms of manipulation, including practices like pump-and-dump schemes, spoofing, and wash trading. These manipulative activities are carried out by individuals or groups with the intention of creating artificial anomalies specifically designed to deceive AD systems. By doing this, the manipulators aim to make their fraudulent activities appear as genuine anomalies, thus evading detection and potentially profiting from their actions.

2.4.2 Main Methods

K-means clustering, IF, OSVM, PCA, or LOF are popular algorithms among AD. It is important to note that various unsupervised AD methods employ different techniques to uncover

anomalies in data, resulting in different anomaly scores and scales (Lorenz et al., 2020). Since, it is intended in this work, to explore the feasibility of using unsupervised AD algorithms to identify the most anomalous transactions from a pool of unlabeled data, it is crucial to have a thorough understanding of how these unsupervised methods operate and how they distinguish between themselves in identifying anomalous data instances.

Unsupervised learning is a highly flexible type of ML as it does not require any labeled data for training. This makes it well-suited for situations where labels are not available. In unsupervised AD, data is scored based on its internal structure and characteristics, allowing the algorithm to identify and separate anomalous data instances from normal ones (Omar et al., 2013). Goldstein & Uchida (2016) roughly categorizes AD algorithms into the following categories illustrated in Figure 1.

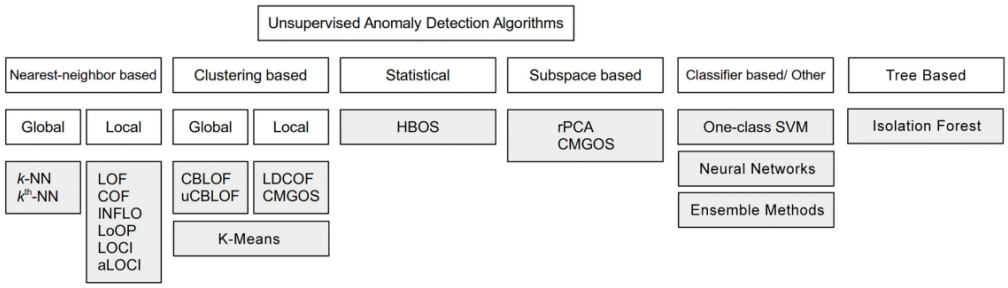

Figure 1: Taxonomy of Anomaly Detection Methods (Goldstein & Uchida, 2016)

Nearest neighbor-based AD methods rely on the premise that normal instances tend to congregate in densely populated areas, while anomalies are situated in regions that are farther away from their nearest neighbors (i.e., closest points).

On the other hand, cluster-based AD methods, usually, rely on two different assumptions. The first one, postulate that normal data instances are typically located in proximity to the centroid of their assigned cluster, while anomalous points tend to be situated farther away from the centroid of the closest cluster. The other assumption is related to the fact that non-anomalous data instances may belong to clusters that are both substantial in size and densely populated, while anomalous data is associated with clusters that are either relatively small or thinly distributed.

Subspace-based techniques such as PCA, rely on the assumption that by reducing the dimensionality of the data, the distinction between typical and abnormal instances will be more obvious.

Further, statistical approaches for AD fall under a class of AD methods that rely on the statistical properties of the data. These approaches aim to identify anomalies by comparing the observed data distribution to a model of what is considered normal.

Lastly, tree-based AD algorithms utilize a hierarchical structure to detect anomalies. These methods construct a tree-like model where each internal node represents a splitting criterion based on a specific feature or attribute. Tree-based AD methods leverage the assumption that anomalies are more likely to end up in isolated regions or as outliers, leading to the creation of distinct branches or leaf nodes dedicated to capturing these anomalous instances.

2.5 Active Learning

AL is a powerful approach in ML that involves an iterative process of selecting and labeling the most informative and relevant instances in a dataset to train a supervised model. AL allows to actively seek the assistance of an oracle, typically a human annotator, to obtain labels for specific instances that it deems most uncertain or informative. Therefore, AL improves the performance of supervised ML models by strategically selecting data for training. Instead of blindly labeling all the data, AL employs heuristics/policies to intelligently query the oracle for labels on certain unlabeled instances (Settles, 2012). This process effectively reduces the number of labels needed to train a supervised classifier by iteratively sampling the most informative data instances from a pool of unlabeled data for review (Lorenz et al., 2020).

Thus, AL can be considered a specific instance of semi-supervised learning, in a way that they both are trying to solve the same problem, learn more from unlabeled data. However, AL focuses on learning from important examples of unlabeled data because it assumes that there are labels of some unlabeled data points that are more informative than others.

AL is particularly useful in the context of FD by addressing the scarcity of labels and has been successfully applied in this field. By continuously accumulating labeled data, AL enhances its supervised model through iterative training processes (Settles, 2012). Additionally, AL can help mitigate the impact of class imbalance by actively selecting samples from under-represented classes, ensuring a more balanced and representative training set.

The AL process for collecting labels is applied only to the training set of the data. Therefore, the training set is divided into two pools: the unlabeled pool (containing instances that can be selected for query) and the labeled pool (containing instances that have been queried and

further investigated and labeled by an analyst). As shown in Figure 2, an AL setup begins with an initial pool of unlabeled data or only a few labeled instances. The most informative samples, from the unlabeled pool, are then selected and labeled by an oracle (human analyst) based on a specific AL querying strategy/policy. Further, the labeled samples are moved to the labeled pool and used to train and evaluate supervised ML models on the test set. This loop is repeated until a certain number of labels is collected, or the baseline performance of a specific supervised ML model is achieved. If the performance is still not satisfactory, the querying process continues to incrementally expand the labeled pool. Each loop execution corresponds to a new iteration of the AL process, where the model is refined on the newly labeled data.

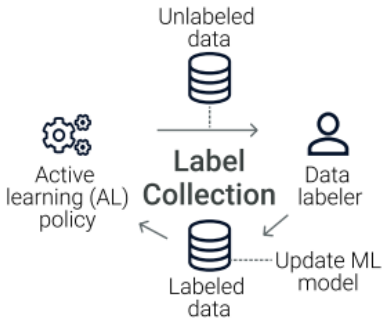

Figure 2: Simplified Structure of an Active Learning Process (Barata et al., 2021)

Regarding the evaluation strategies, the performance of the AL setups is usually compared to a supervised baseline, i.e., the performance the supervised model achieves when trained on the entire train set. This comparison holds significance because the objective of numerous AL approaches is to attain the baseline performance while minimizing the need for labeled samples. In simpler terms, the primary aim is to efficiently identify a subset of data for annotation that optimizes the model's learning capacity (Lorenz et al., 2020). Moreover, commonly used metrics for evaluation include learning curves. Thus, given a specific performance metric, it is possible to obtain a learning curve where the chosen metric would improve over the different labeled pool sizes (Barata et al., 2021).

2.5.1 Relevance of using Anomaly Detection for Active Learning

Training a high-performance supervised classifier is an essential task in the FD domain. However, labeling an entire dataset to train a supervised ML model can be challenging in real-world scenarios. This process often requires many manually reviewed transactions, which is

not feasible for companies due to the intensive and time-consuming nature of the task (Labanca et al., 2022). This is especially worse when dealing with imbalanced data scenarios because labels of the positive class take longer to accumulate (Barata et al., 2021). This problem is aggravated due to the high complexity of fraud behaviors (Kong et al., 2020).

AL studies in the literature often assume that an initial labeled pool is available to start the AL process. However, in many real-world scenarios, one may be faced with a cold start scenario with a system that has just been deployed and contains no labeled data (Settles, 2012).

Thus, the linkage between AD and AL is naturally explained because, especially, in FD, labels are not always available, and most times are very hard to get (Labanca et al., 2022). AD can help alleviate this issue by identifying anomalous behavior within the unlabeled data. This means that even without explicit labels, AD can highlight instances that are likely to be fraudulent or suspicious. By leveraging AD, it is possible to narrow down the focus of labeling efforts to those instances that are flagged as anomalous, potentially saving resources.

Therefore, there can be immense value in the combined use of AD and AL algorithms, by iteratively collecting a valuable set of labeled data, based on the most anomalous data instances (i.e., the transactions with the highest anomaly score), as efficiently as possible.

AL has the potential to combine the benefits of unsupervised (exploration) and supervised (exploitation) methods. Therefore, employing AD algorithms (as unsupervised policies) within the AL process enables the selection of the most anomalous data, from an unlabeled pool, to be reviewed and labeled by an oracle. After the feedback from the analyst, the labeled data can be used to train a supervised ML model that learns fraud. This approach assumes that the most anomalous instances are more likely to contain the most relevant fraud-related patterns for the supervised ML model to learn. This combined use of AD and AL enables more efficient utilization of labeled data, potentially resulting in a system that performs similarly to a supervised baseline with minimal labeling effort.

2.5.2 Active Learning Policies

In this dissertation, the objective is to simulate a cold start scenario. As a result, to begin the AL process, two alternatives can be considered for selecting relevant fraudulent transactions within the unlabeled pool: Random Sampling and Unsupervised AD algorithms. Random sampling involves randomly selecting transactions without any underlying pattern analysis.

Conversely, unsupervised AD algorithms aim to identify the most anomalous transactions. Given the expectation that an unsupervised AD algorithm would outperform a completely random algorithm, it is reasonable to begin the AL process by employing unsupervised AD algorithms. However, in typical scenarios of AL implementation, it is assumed that there are at least a few initial labels present to implement the AL process. Consequently, these processes commonly employ more sophisticated heuristics that require an underlying supervised ML model to perform the queries and identify the most informative instances. These supervised AL policies are trained, at each iteration, on the available labeled pool. A description of the main supervised AL policies follows.

One of the most ancient and widely explored AL techniques is Uncertainty Sampling. It relies on the classifier's confidence in its prediction, selecting the samples with the highest uncertainty. Thus, this uncertainty model works with any classifier that predicts and measures its own certainty, choosing the samples with the higher uncertainty. However, this method is known to be susceptible to noise and outliers (Sharma & Bilgic, 2017). There also exist some variants of Uncertainty Sampling, such as Epistemic Uncertainty Sampling. This heuristic represents the part of total uncertainty that can be reduced and is made up of both model uncertainty and approximation uncertainty. Typically, as the amount of labeled data used for training increases, a model's epistemic uncertainty decreases (Nguyen et al., 2019).

Another supervised querying policy that has been widely explored by researchers is Expected Model Change (EMC), which assumes that a data instance is more informative if they have a strong impact on the model. In other words, the instances that are queried are the ones that result in the maximum change of the current model (Cai et al., 2017). Another well-established AL algorithm is Query by Committee (QBC). Unlike Uncertainty Sampling, which uses a single learner, QBC employs a committee of multiple learners to handle large-scale problems. To do so, it utilizes the disagreement among the predicted probabilities of the classifiers presented in the ensemble (Kee et al., 2018).

A different supervised AL strategy was originally proposed by Houlsby et al. (2011). It is an approach based on Bayesian information theory, that expresses information gain in terms of predictive entropies and applies the method to the Gaussian Process Classifier. In other words, the examples are selected to maximally reduce the entropy of the posterior distribution over classifier parameters (Bachman et al., 2017).

3. LITERATURE REVIEW

This literature review focuses on the use of traditional ML algorithms, namely supervised and unsupervised AD for detecting fraudulent transaction behavior. Lastly, there are also presented some AL frameworks that combine the use of AD and AL techniques. The preceding concepts serve to elucidate and substantiate the provided information.

3.1 Traditional Approaches to Fraud Detection

3.1.1 Supervised Methods

Most of the cryptocurrency FD research heavily relies on ML models applied to the Elliptic Dataset (Alarab & Prakoonwit, 2022). The Elliptic Dataset, provided by cryptocurrency intelligence firm Elliptic, is a graph network of Bitcoin transactions and a valuable resource for FD and research communities, being one of the largest labeled datasets for any cryptocurrency.

Particularly, supervised ML models have shown very promising results in classifying transactions as fraudulent or not as demonstrated by Weber et al. (2019) who applied various binary classification methods, such as LR, RF, Multilayer Perceptron's, and Graph Convolutional Networks, to the Elliptic Dataset and found that RF was the best-suited model to this specific context with high-performance rates, despite the graph structure information providing an expected advantage to the graph-based approaches. Alarab et al. (2020) have also achieved successful results using ensemble learning on the Elliptic dataset, outperforming other methods, including Weber et al. (2019), by combining different supervised learning techniques.

Particularly, RF is one of the most widely explored supervised ML models in cryptocurrency FD research. For instance, both Ostapowicz & Żbikowski (2019) and Ibrahim et al. (2021) have obtained good detection results using RF to classify Ethereum blockchain transactions as fraudulent or not. However, Ostapowicz & Żbikowski (2019) have concluded that the 85% Recall rate could not be suitable for real-world anti-fraud systems. Recently, Melo-Acosta et al. (2017) have presented a Balanced Random Forest methodology that can be used in supervised and semi-supervised scenarios via co-training. This Balanced Random Forest

approach involves drawing a bootstrap sample from the minority class and building a full-sized classification tree from the data, iteratively. Overall, this innovative approach showed improvement over a standard RF.

Some authors have also contributed to the FD literature by benchmarking traditional supervised classifiers. For instance, Awoyemi et al. (2017) compared three FD techniques and found that the k-nearest neighbor method performed better than Naïve Bayes and LR with high detection rates. The highly imbalanced dataset was sampled using a hybrid approach that oversampled the positive class and undersampled the negative class. On the other hand, Ajedig et al. (2011) compared SVM and DT algorithms on a real-world dataset and found that DT outperformed SVM. However, as the dataset grew, SVM's performance improved, but it still detected fewer frauds than DT.

Lastly, Niu et al. (2019) compared the performance of supervised and unsupervised ML algorithms in a study on a credit card FD dataset. The dataset was highly imbalanced, with only 492 fraudulent transactions among over 280,000 transactions. In order to deal with the highly imbalanced scenario, the non-fraud transactions were randomly reduced, through random undersampling. The performance was evaluated using the Area Under the Receiver Operating Characteristic (AUROC), a metric combining FPR and True Positive Rate (TPR). The study found that supervised models performed slightly better than unsupervised models, with eXtreme Gradient Boosting (XGBoost) and RF being the top performers.

However, as previously stated, the use of supervised methods is limited by the need for labeled datasets, which can be costly and time-consuming to obtain through manual annotation of all historical transactions (Labanca et al., 2022).

3.1.2 Anomaly Detection Techniques

AD methods assume cases of fraud to be outliers, meaning that illicit transactions should have significantly different behavior patterns when compared to the legitimate ones, which are the majority. Overall, the application of traditional AD methods for detecting fraudulent transactions has yielded optimistic and encouraging results with high detection rates. For instance, Zengan (2009) combined distance-based unsupervised clustering techniques with LOF to design a Cluster-based Local Outlier Factor. This approach was tested in a synthetic dataset to access its applicability and effectiveness, and it showed very good results in

detecting suspicious transactions with high processing speed and a good detection rate. Another example is the approach applied to the Bitcoin network by Monamo et al. (2017), which tested both global (clustering k-means) and local (tree-based k-d trees) AD algorithms, successfully identifying most of the known fraudulent behaviors. However, even though both methods achieved good results, using k-d trees, the authors were able to detect two more fraudulent transactions.

Some studies have also investigated the usage of deep autoencoding for AD, proving its high efficiency. For example, Zamini & Montazer (2019) proposed an autoencoder-based clustering method for unsupervised FD. In their study, a 3-layer autoencoder and k-means clustering were applied to 28,000 transactions and the proposed approach outperformed most of the state-of-the-art techniques. On the other hand, Zong et al. (2022) proposed a Deep Autoencoding Gaussian Mixture Model for FD. This model uses a deep autoencoder to generate a low-dimensional representation and reconstruction error, which is then fed into a Gaussian Mixture Model. Experiments on public datasets have also shown significant improvement over existing AD techniques.

Additionally, some authors have contributed to the AD literature by benchmarking the performance of several traditional unsupervised AD techniques for FD. For instance, Ounacer et al. (2018) evaluated the standard IF algorithm against other unsupervised techniques (LOF, OSVM, and K-means) for detecting fraudulent and anomalous transactions. The results showed that IF outperformed all the other techniques, with high detection rates. Likewise, Domingues et al. (2018) also conducted a comparative evaluation of different unsupervised AD algorithms, and the results showed that IF is an excellent method to efficiently identify outliers while showing excellent scalability on large datasets. OSVM showed also good results, but the authors have concluded that is not so suitable for larger datasets. LOF was the one that reached lower performance and scalability.

Some authors have also shown that unsupervised AD techniques have the capability of uncovering fraudulent transaction patterns that were previously unknown or missed by traditional rule-based systems, which turns out to be a huge advantage (Camino et al., 2017; Zengan, 2009).

However, despite the high detection rates reported by most of the authors, some authors also question their real-world effectiveness with complex patterns and incomplete labels. For example, Lorenz et al. (2020) found high FPR when benchmarking AD algorithms on a real-

world Bitcoin transaction dataset and concluded that past studies may have relied too heavily on synthetic anomalous data points that are outliers by design, not reflecting real-world scenarios where criminals can imitate normal behavior. Because of that, even though authors such as Zhou et al. (2021), have tried to define fraudulent behavior as generally presenting various abnormal characteristics such as abnormal transaction amount, abnormal transaction time, abnormal transaction account, abnormal transaction IP, or abnormal personal credit ranking, this is not necessarily true, since nowadays a fraudulent transaction can have apparently similar characteristics to the licit ones, which makes the task of identifying fraudulent transactions even harder.

3.2 Approaches to Active Learning with Anomaly Detection

In this section, there are presented several AL and AD frameworks, both in the financial domain and general domain applications.

3.2.1 General Domain Applications

Combining AL and AD has been proven to enhance the supervised model's performance by selecting only the most relevant data instances for review. Particularly, this combination has been successfully tested in different application domains. An example is the approach proposed by Nixon et al. (2021) that combined an AD autoencoder algorithm with AL strategies to optimize labeling expenses and improve training and adaptation efficiency to discover network intrusions. This method demonstrated exceptional results with only a 20% labeling budget, thereby significantly reducing the need for human annotation. Another example is the hybrid system proposed by Vynokurova et al. (2020) to solve problems of AD. It consists of two subsystems: one uses IF to calculate anomaly scores, and the other uses RF for FD classification on the anomalous and borderline data that have been detected by the previous subsystem. The system showed also very promising results when tested on a large dataset.

Studies in this field often focus on enhancing AD methods through human feedback, by incorporating it into the anomaly detector. Therefore, suggesting that the anomaly detector should be configurable by the analyst to reduce FP efforts. For this matter, Das et al. (2017), and Siddiqui et al. (2019) both proposed similar methods to incorporate binary feedback into

tree-based anomaly detectors. Using the IF algorithm as an example, results showed significant improvement in performance, with increased detection of true anomalies, when compared to the baseline. The feedback loop fine-tunes the IF output and reweights it based on analyst feedback. Additionally, it has also been proved that ensemble-based anomaly detectors can also benefit from this automatic tuning (Das et al., 2018).

Furthermore, authors such as Pelleg & Moore (2005) and Stokes et al. (2008) have explored the construction of a model for each class label. For instance, the approach proposed by Pelleg & Moore (2005) suggested dismissing anomalies when they are labeled by experts. They maintained one Gaussian model for each class label. First, the class label is estimated by discovering the most likely class. Then, for each item, it is given an anomaly score and the items with a higher score are chosen for querying. Finally, the models are updated with the new labeled data. Similar to this previous approach, Stokes et al. (2008) proposed ALADIN, which presents a small number of potential anomalies to an oracle for labeling into different categories. These labels update both the classifier and the anomaly detector. The LR classifier determines the most uncertain items, based on Uncertainty Sampling criteria, and assigns unlabeled instances to the most likely category. The anomaly detector finds the most anomalous data based on an anomaly score. This approach has shown promising results.

Alternative techniques were proposed by Pichara & Soto (2011) and Zha et al. (2020). For instance, Pichara & Soto (2011) reduced the dataset size using a Bayesian Network to identify low-likelihood records. Then, subspace clustering with candidate anomalies was used to identify micro-clusters, and a Dirichlet distribution-based AL policy uses the feedback to find relevant anomalies. Results showed accurate detection of relevant anomalies in data with apparently normal patterns. More recently, Zha et al. (2020) introduced an active AD approach with meta-policy, a framework that learns a meta-policy for query selection. This approach leverages deep reinforcement learning to train the meta-policy and select the most accurate instance to maximize the number of detected anomalies. The results showed that this approach outperforms the state-of-the-art unsupervised baseline.

3.2.2 Financial Domain Applications

The combined use of AD and AL techniques in financial systems has not been extensively investigated. Still, most of the existing studies have concluded that RF, as a supervised

classifier, is one of the ML models that benefits the most from the AL process. For instance, Labanca et al. (2022) developed a framework that combined both supervised and unsupervised AL techniques. Their system ranks anomalous data using an anomalous score from unsupervised methods and, in parallel, trains supervised models on domain experts' feedback using two AL strategies (QBC and Uncertainty Sampling). The authors also emphasize the importance of aggregating transactions for capturing correlations. The results indicate that IF and RF were the best-performing ML models. A similar example is the AL framework proposed by Stefano et al. (2018) to find anomalies in large transactional data in an unsupervised setting. An unsupervised model assigns an anomaly score, which is then used to query an oracle for labeling. The supervised model is continuously trained with the labeled data. The study employed autoencoders for AD scoring and RF for supervised scoring, resulting in high detection rates. In addition, Lorenz et al. (2020) proposed an AL framework to capture fraudulent activities in the Bitcoin blockchain. This study combined four different query strategies trained on the labeled data. Two of these strategies were supervised (Uncertainty Sampling and EMC) and two were unsupervised (IF and elliptic envelope). As classifiers, there were used three supervised ML models: RF, XGBoost, and LR. Each tested AL setup involved a combination of an unsupervised AL policy and a supervised AL policy. Specifically, once a certain threshold was reached, the authors would transit from an unsupervised AL policy to a more sophisticated supervised AL policy. The results showed that this solution could match the performance of a supervised baseline by using as few as 5% of the original labels. This was achieved using RF as a classifier. The results from this study indicated that unsupervised AD methods had poor results which highlights that experiments conducted on synthetic data can be misleading. Thus, at some point, switching to supervised learners improved the models' performance.

Furthermore, authors such as Carcillo et al. (2017) and Barata et al. (2021) focused their work on handling the high-class imbalance issue in a streaming scenario. Carcillo et al. (2017) evaluated different AL strategies for the problem of credit card FD. Given the prevalent issue of class imbalance, they began by examining a baseline strategy that only queries high-risk transactions. The results showed that this strategy could be improved by combining it with Stochastic Semi-supervised learning, resulting in higher FD rates. This being said, an effective AL strategy for AD requires proper tuning. On the other hand, to deal with the same problem, Barata et al. (2021) proposed a 3-stage sequence of AL labeling policies using Outlier

Discriminative Active Learning (ODAL). ODAL is a novel technique that trains an AD algorithm on the labeled data and then uses the obtained model to rank the unlabeled samples relative to the labeled pool. These 3-stage AL policies design using ODAL, as a warm-up policy, turned out to be very efficient to alleviate the cold start scenario with high-class imbalance. The first stage policies consisted of the utilization of AD algorithms trained on the unlabeled pool. However, since after the first stage, many of the labels collected would be mainly of the majority class (non-fraudulent transactions), the authors, in the second stage, proposed the application of ODAL, using an IF trained in the available labeled pool to find the most relevant transactions on the unlabeled one. Finally, in the last stage, supervised AL policies can make full use of the labeled transactions.

In conclusion, the literature reviewed suggests that traditional supervised models and unsupervised AD methods have been extensively explored for FD in the financial sector. Particularly, supervised ML models have shown high performance in classifying fraudulent transactions. However, these methods have limitations in terms of the high costs to obtain manual annotation of a large number of historical transactions. On the other hand, without relying on labeled data, the majority of the literature has demonstrated the high detection rates of unsupervised AD methods. Some authors even suggest that these methods were able to discover previously unknown fraudulent patterns.

Additionally, the literature also underscores the importance of employing AL techniques to efficiently select a subset of the labeled data that maximizes the supervised model's performance. In the financial systems domain, the literature points out that switching to supervised learners (supervised AL policy) at some point in the AL process improves the overall models' performance. This means that after defining an optimal cut-off point in the AL setup (e.g., the number of iterations completed or a threshold associated with the total number of illicit transactions in the labeled pool), a change from an unsupervised AL policy to a more sophisticated supervised AL policy is required.

4. MATERIALS AND TOOLS

In this chapter, the materials and tools employed in this research study are presented. The chapter encompasses a detailed overview of the libraries utilized to implement the techniques evaluated in this study, as well as the dataset and its exploratory data analysis.

It is important to note that the dataset selection, pre-processing, and exploratory data analysis were conducted collaboratively as a team at Uphold. Additionally, there was another student on the team who worked on a fraud-related use case. Given that the supervisors were the same, there was some overlap in the materials used and the exploration of those materials.

4.1 Development Tools

The programming language chosen for the development of this project and its auxiliary scripts is Python. Python is a powerful, high-level language that is well-suited for ML projects. It offers a great deal of flexibility and a wide range of resources and comprehensive documentation. Additionally, its simplicity and ease of use can greatly enhance developer productivity, allowing them to focus on solving problems rather than getting bogged down by the intricacies of the language itself. Its open-source nature and large developer community make it an ideal choice for this project.

The programming environment used in this work was Jupyter Notebook. Jupyter Notebook stands as an interactive computing environment that facilitates the creation and dissemination of documents incorporating live code, equations, visualizations, and explanatory text. It boasts a web-based interface that supports the generation and execution of code cells, thereby simplifying the process of experimentation, iteration, and collaboration on data analysis and ML endeavors. One notable advantage of the Jupyter Notebook is its capacity to execute code in a modular and exploratory manner. This feature enables the execution of individual code snippets, affording the convenience of testing and debugging specific sections of the project without necessitating the execution of the entire script.

Furthermore, Python has a diverse set of tools and libraries specifically tailored for ML. For this project, the following libraries were utilized:

- **Scikit-learn**: Scikit-learn is a powerful Python library for ML, offering a diverse selection of unsupervised and supervised algorithms. It encompasses a broad spectrum of

classification, regression, and clustering techniques. One of the standout attributes of scikit-learn is its user-friendly nature, making it accessible to programmers of all skill levels. Moreover, scikit-learn provides an extensive range of functionalities for efficient data preprocessing, feature extraction, and feature selection. It also includes robust tools for model evaluation and validation, facilitating the assessment of ML models through various metrics. In the context of this work, the scikit-learn library played a vital role in experimenting with different supervised ML classifiers (RF and LR) and evaluating the performance of the ML models, delivering essential insights and metrics.

- **xgboost**: xgboost is a popular ML library for gradient boosting algorithms that is widely used in data science and ML communities. In the context of this work, this library played a vital role in experimenting with the XGBoost algorithm.
- **Pyod**: Python Outlier Detection (PyOD) is a highly robust and comprehensive software library that focuses on the identification of outliers in data by leveraging a variety of statistical and ML techniques. It encompasses a vast array of unsupervised algorithms, including IF, LOF, OSVM, and several others. The library facilitates convenient functionalities for model training, prediction, and evaluation, enabling users to swiftly assess the efficacy of diverse approaches. In the present study, all the AD algorithms were tested using this library. The selection of PyOD over scikit-learn for testing the unsupervised algorithms was driven by its broader range of AD algorithms and its consistent application across all these algorithms.

For auxiliary tasks, there were necessary the following libraries:

- **NumPy**: NumPy plays a pivotal role in scientific computing and data analysis. It offers a versatile array object and an expansive set of mathematical functions, making it an essential tool. With NumPy, it is possible to effortlessly perform mathematical operations on entire arrays, eliminating the necessity for explicit loops, and thereby enhancing code efficiency. In the context of this project, the primary applications of this library were for numerical computations and random number generation.
- **Pandas**: Pandas is an influential and extensively utilized open-source library for data analysis and manipulation in the Python programming language. Its robust offerings include a comprehensive suite of data structures and functions that significantly

streamline the complexities involved in handling structured data. By harnessing the capabilities of Pandas, users gain effortless access to extracting meaningful insights and conducting a wide range of operations on their data. Notably, Pandas encompasses an expansive repertoire of dedicated functions and methods, meticulously designed for data cleaning, manipulation, merging, grouping, and analysis. In the context of this study, Pandas assumed a central role in facilitating the data loading process and subsequent manipulation tasks, highlighting its indispensable role in achieving research objectives.

- **Scipy**: Scipy is a robust open-source library utilized for scientific and numerical computations within the Python programming language. It offers an extensive array of functions and tools designed to facilitate various tasks, including numerical integration, optimization, linear algebra, signal processing, image processing, and other related operations. In the present study, Scipy primarily served as a computational resource for calculating some heuristics used for certain AL strategies.
- **Matplotlib**: Matplotlib is a powerful and versatile Python library used for creating visualizations and generating various types of plots, charts, and graphs. It provides a comprehensive set of tools and functions that allow developers and data scientists to create high-quality visual representations of data in an intuitive and flexible manner. For this study, matplotlib was used as support for the exploratory data analysis.

4.2 Dataset

Due to the initial unavailability of Uphold to provide a proprietary dataset comprising real-life cryptocurrency transactional data, it was crucial to acquire a public dataset that could adequately represent Uphold's transactional data.

Thus, this work will use the Elliptic Dataset, the world's largest publicly available cryptocurrency transaction dataset. This dataset was originally created by Weber et al. (2019) with the purpose of facilitating the creation of innovative methods for detecting fraudulent activity in cryptocurrency transactions.

Four reasons led to the selection of this dataset:

1. The fact that this is a representative dataset of cryptocurrency transactional data makes it valuable for exploring the feasibility of using AD and AL algorithms to detect fraudulent patterns.
2. The existence of labeled samples (licit and illicit transactions) allows an easy simulation of the AL process by linking the class label to the queried transaction.
3. Previous exploration of unsupervised AD techniques combined with AL strategies has been performed on this specific dataset (Lorenz et al., 2020).
4. The dataset pertains to the Bitcoin transactions network, which is of great interest due to Bitcoin being the most well-known and secure cryptocurrency.

Before presenting the specifics of the Elliptic Dataset, it is important to understand how Bitcoin transactions work. Bitcoin transactions operate on a protocol known as Unspent Transaction Output (UTXO), which enforces specific rules (Cachin & Vukolić, 2017). When a user (A) intends to send a payment to another user (B), user A needs to reference the prior transaction where they received the funds. This reference to the preceding transaction is called the transaction input. The transfer of funds to user B is referred to as the transaction output. It's worth noting that a transaction can involve multiple inputs and outputs, allowing for more complex scenarios (Cachin & Vukolić, 2017).

Figure 3 illustrates the Bitcoin UTXO transaction model.

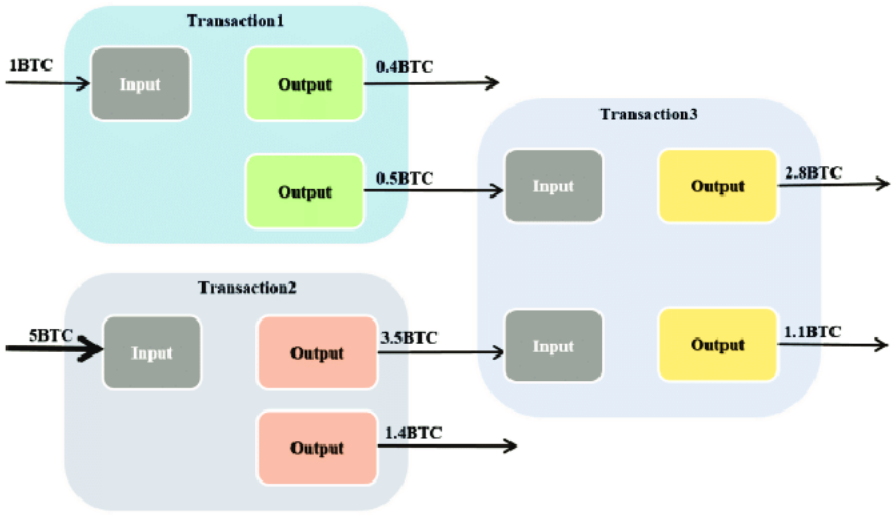

Figure 3: Bitcoin UTXO Transaction Model (Xue et al., 2021)

The Elliptic Dataset contains over 200,000 transactions from the Bitcoin blockchain and maps them to real entities in either licit, illicit, or unknown categories. Labels include "licit" for transactions made by regulated exchanges, licit services, miners, etc., and "illicit" for

transactions related to dark markets, scams, malware, terrorism organizations, Ponzi schemes, etc.

The dataset includes 49 transaction graphs sampled from the Bitcoin blockchain at different points in time. Each node in the graph represents a transaction with 166 anonymized features and is labeled as "licit", "illicit", or "unknown". The graph consists of 203,769 nodes and 234,355 edges (regarding the flow of Bitcoin transactions). A total of 4545 Bitcoin transactions are labeled as illicit, corresponding to approximately 2% of the total transaction nodes. Moreover, a total of 42019 transactions are labeled as licit ones, which corresponds to 21% of the graph nodes. The remaining 157205 transactions (77% of the total transactions) are not labeled regarding the licit vs illicit, thereby being represented as “unknown” transactions.

It is important to note that, due to intellectual property issues, the dataset is mostly anonymized. Therefore, there is no exact description of most of the features in the dataset, nor it does have the columns' names.

The transaction address, time step, and class label are the only known information in the anonymized data. The time step measures the time when a transaction was broadcast to the network and ranges from 1 to 49 with a two-week interval between each.

The first 94 features pertain to local information about the transaction such as the time step, number of inputs/outputs, transaction fee, and aggregated information like average BTC received/spent. The remaining 72 features are aggregated using one-hop backward/forward information from the center node and include maximum, minimum, standard deviation, and correlation coefficients of neighboring transactions.

4.2.1 Data Structure

The main dataset consisted of three distinct datasets, each one containing specific information about the transactions and the flow of transactions. These datasets include data pertaining to transaction classes, transaction features, and transaction edges:

1. The dataset pertaining to the classes comprises two distinct features. The initial feature, *transaction_id*, is a categorical variable denoting unique identifiers for each transaction. The subsequent feature is the target label, also categorical, encompassing three possible values: 'licit', 'illicit', or 'unknown'. Based on the dataset characteristics it helps to classify transactions as fraudulent or not.

2. The second dataset, with a focus on the edges, captures essential information pertaining to the connections or links established between transactions. It provides valuable insights into the relationships or associations among distinct transactions. As such, this dataset is characterized by two categorical features. The first feature denotes the source *transaction_id*, while the second feature corresponds to the target *transaction_id*.
3. The third and final dataset consists of the transaction ID along with a comprehensive collection of other relevant features. This dataset encompasses a wide array of transaction-related information, enabling a meticulous analysis and profound understanding of the attributes and properties associated with each transaction. Thus, the dataset is composed of a total of 168 features. These features include the *transaction_id*, *time_step*, 94 *local_features*, and 72 *aggregated_features*.

4.2.2 Exploratory Data Analysis

In this chapter, the key findings derived from a comprehensive exploratory data analysis are delved into. Undertaking this crucial step provides a better understanding of the label targets and the underlying dataset before applying ML algorithms. Through meticulous exploration and analysis of the data, valuable insights have been unearthed that shed light on the characteristics within the dataset. The primary focus of the exploratory data analysis was to gain a deep understanding of the label targets. Through careful examination, essential information about these targets, such as their distributions and potential class imbalances, has been uncovered.

The plot in Figure 4 is a bar plot that showcases the cumulative count of transactions across the class labels.

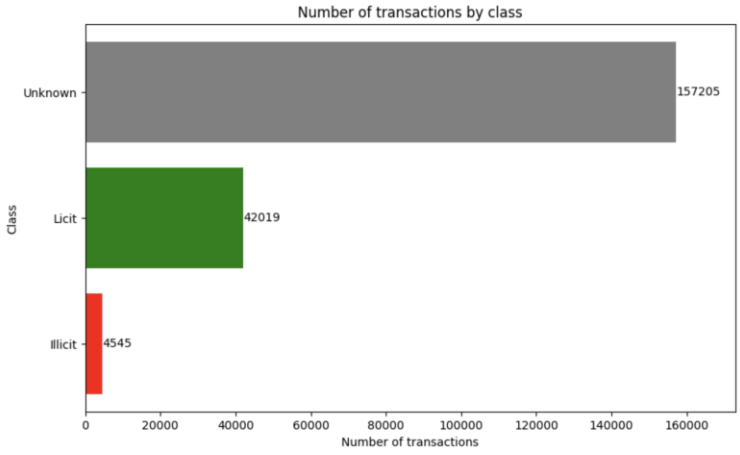

Figure 4: Number of Transactions by Class

The visualization clearly indicates that the dataset predominantly comprises transactions labeled as "unknown," which account for a substantial total of 157,205 transactions. Moreover, the plot effectively demonstrates a significant class imbalance, as the number of recorded illicit transactions stands at a mere 4,545.

Figure 5 shows a line plot that portrays the cumulative count of transactions across various time steps.

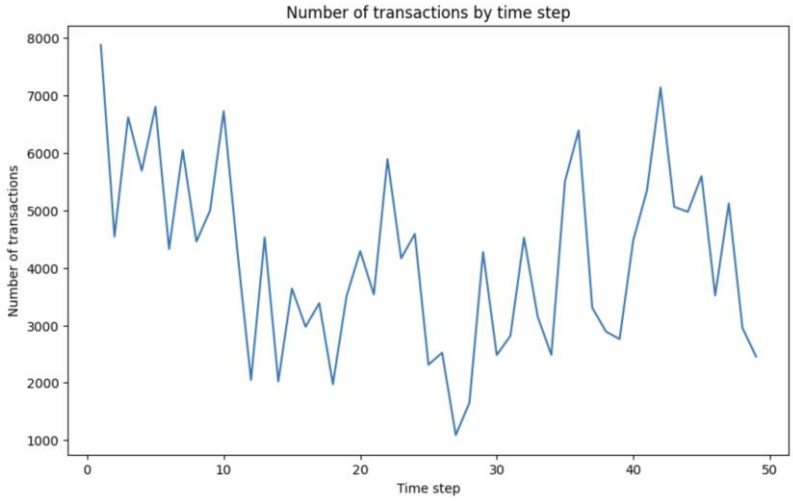

Figure 5: Number of Transactions by Time Step

The plot reveals significant variations in the total number of transactions observed over time steps. Notably, certain time steps exhibit a number of transactions that are approximately seven to eight times higher than others, indicating a substantial disparity. This considerable fluctuation underscores the dynamic nature of transaction activity over the recorded time period.

The stacked bar plot presented in Figure 6 complements the visualization depicted in Figure 4 by illustrating the distribution of class labels across different time steps.

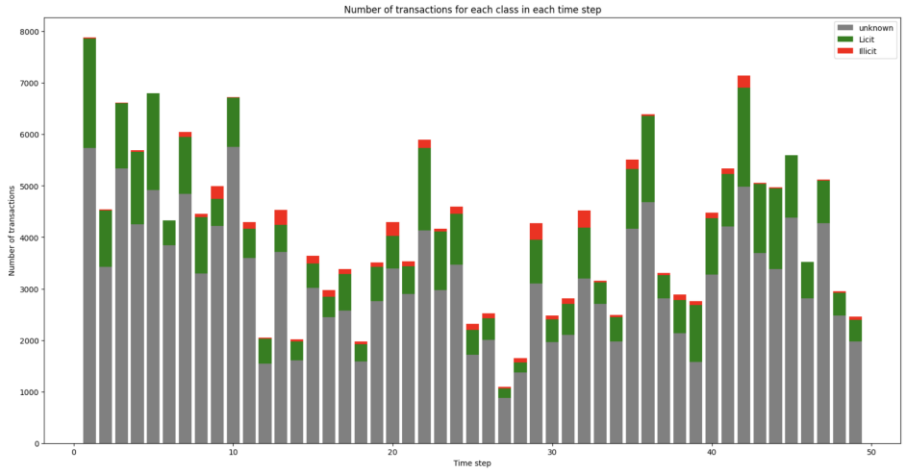

Figure 6: Number of Transactions for Each Class in Each Time Step

By analyzing this plot, it becomes evident that the overall pattern observed in the dataset remains consistent across all time steps, thereby offering significant insights. Remarkably, the dataset showcases a greater frequency of unknown transactions in comparison to labeled ones, where illicit activities constitute only a small portion.

The plot depicted in Figure 7 serves as a supplementary analysis to the plot illustrated in Figure 6.

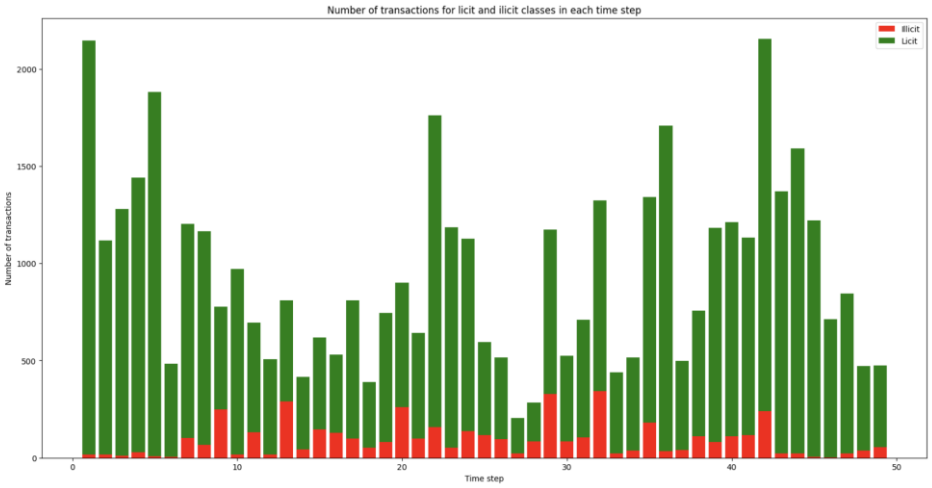

Figure 7: Number of Transactions for Licit and Illicit Classes in Each Time Step

This particular plot enables a more comprehensive examination of the proportion of illicit versus licit transactions at each time step. Consistent with the findings of Weber et al. (2019) it can be concluded that a notable decline in the number of illicit transactions occurred between time steps 43 and 46, attributed to an abrupt cessation of the dark market.

Lastly, Figure 8 employs t-Distributed Stochastic Neighbor Embedding (t-SNE), a widely used dimensionality reduction technique for visualizing high-dimensional data in a lower-dimensional space, more specifically, in this case, in a two-dimensional representation.

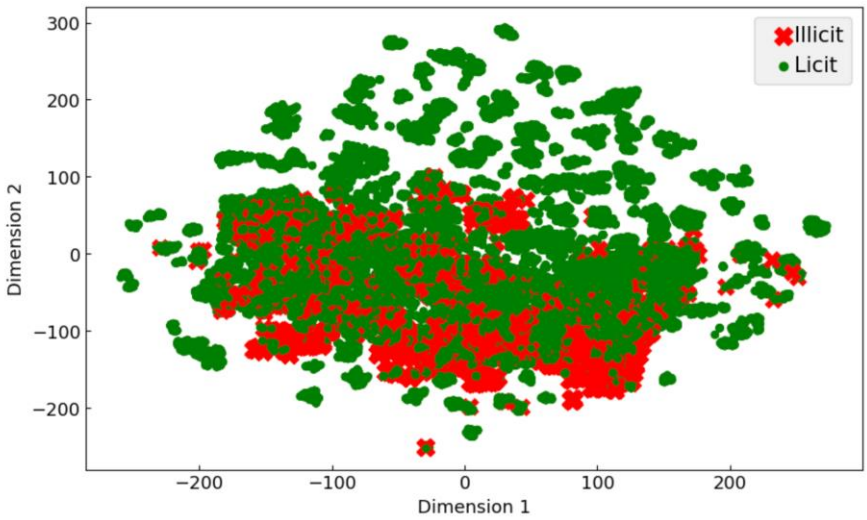

Figure 8: t-SNE Representation of the True Labels

The plot illustrates that fraudulent behavior lacks a discernible pattern, as the instances are scattered across the plot without any distinct clustering, overlapping with the normal transaction behavior. This observation potentially supports the notion that fraudsters often mimic normal transaction behavior, making their activities challenging to identify based on visual patterns alone.

5. MACHINE LEARNING ALGORITHMS SELECTION

This chapter focuses on the selection of algorithms for assessing the feasibility of utilizing AD and AL techniques to uncover fraudulent behaviors in cryptocurrency transactions. The main objective of this work is to explore the potential combined use of AD and AL algorithms, in the absence of sufficient labeled data, by comparing the performance of different AL setups against a supervised baseline and by understanding the impact of using AD algorithms. To achieve this objective, this study builds upon the work conducted by Lorenz et al. (2020), on the same dataset, which provides a suitable framework for combining AD algorithms with AL techniques.

Thus, in this chapter, it is explained all the algorithms tested in the experiments, which include:

- Three supervised techniques used to assess the performance of a supervised baseline and the performance of the AL setups.
- Three unsupervised AD techniques used as unsupervised AL policies in the AL experiments.
- Three supervised AL policies employed when a certain optimal point is reached, indicating a transition from unsupervised AL policies within the AL process.

The intention of this chapter is to present a thorough comprehension of the various algorithms chosen for this research, including supervised learning algorithms, unsupervised learning algorithms, and optimized supervised AL policies.

This study will encompass the exploration of unsupervised policies in combination with supervised policies since the literature suggests that switching to supervised learners at some point in the AL process can potentially improve the overall models' performance. Based on this, it is intended to understand the real impact of using AD on the overall models' performance and understand if there is any optimal point to change from these unsupervised learners during the AL process. In simpler terms, explore the ability of AD algorithms to identify relevant transactions within the unlabeled data, which, when labeled, can positively influence the performance of a supervised classifier, and determine the optimal point in the AL process up to which it is beneficial to prolong the use of these algorithms.

As this study builds upon the foundational work of Lorenz et al. (2020), in order to maintain consistency, it was employed the same algorithms used in the aforementioned study. The

authors utilized RF, LR, and XGBoost as supervised classifiers. Regarding the AL policies, the authors focused on four strategies. Two of them were unsupervised, IF and Elliptic Envelope algorithms, with the main goal of finding the most anomalous transactions on the unlabeled pool. The other two, Uncertainty Sampling and EMC, were supervised, requiring an underlying supervised model to define queries.

However, considering the literature review conducted in this study, it was also evaluated two additional well-established algorithms. An additional unsupervised AD technique was used as an unsupervised AL policy, namely LOF, and an additional supervised AL policy, namely QBC.

For a more in-depth understanding of how the experimental setup was built, refer to Experiments chapter, where a detailed description is provided.

5.1 Supervised Learning

In this chapter, the supervised learning algorithms utilized in this work will be explored. Supervised learning is a ML approach where an algorithm learns from a labeled dataset, comprising input data and corresponding target labels. The objective of supervised learning is to train a model capable of accurately predicting or classifying unseen data by capturing patterns and relationships present in the training dataset.

In supervised learning, the input data, also known as features or independent variables, are represented as numerical or categorical values. The target labels or dependent variables represent the desired outputs associated with the input data. By analyzing the input-output pairs, the algorithm discerns underlying patterns and constructs a predictive model.

In the subsequent sections, it will be provided a comprehensive explanation of three specific algorithms: RF, LR, and XGBoost.

5.1.1 Random Forest

RF have emerged as a highly effective ensemble learning method for classification tasks. It leverages the collective wisdom of multiple DTs to make accurate predictions.

At a high level, RF combines the predictions of multiple DTs to arrive at a final prediction. However, the key to their effectiveness lies in the introduction of randomness during both the training phase and the prediction phase.

During the training phase, each DT in the RF is constructed independently using a process called bootstrap aggregating or bagging. This involves randomly selecting a subset of the training data, with replacement. By allowing the same sample to appear multiple times or omitting some samples altogether, bagging creates diverse subsets of the training data for each tree. This diversity helps to reduce overfitting and improves the overall robustness of the ensemble. Furthermore, RF introduces randomness at each split of the DT. Instead of considering all features at every split, a random subset of features is chosen. Typically, the number of features considered is the square root of the total number of features. This random feature selection injects more diversity into the ensemble by ensuring that different trees focus on different subsets of features. Consequently, each DT in the RF becomes specialized and contributes unique information to the final prediction. Various splitting criteria, such as Gini impurity or information gain, are utilized to determine the best feature for splitting.

When it comes to making predictions, RF employs a voting or averaging mechanism. Each DT in the ensemble independently predicts the class label and the final prediction is determined by aggregating the individual tree predictions. In classification tasks, the most common approach is majority voting, where the class label that receives the most votes across the trees is selected as the final prediction.

Figure 9 illustrates the operational mechanics of the RF model through an illustrative example.

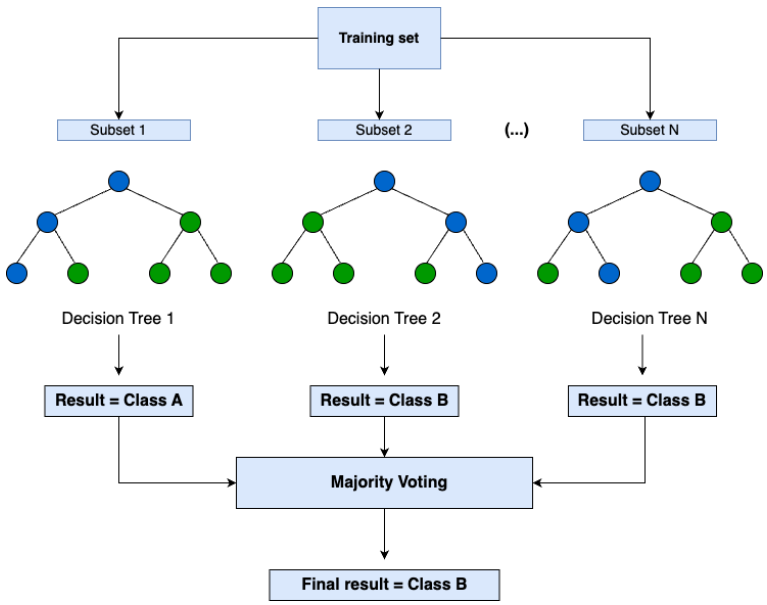

Figure 9: Simplified Structure of Random Forest

By harnessing the power of ensemble learning, RF can overcome the limitations of individual DTs. They exhibit excellent generalization capabilities, robustness against overfitting, and resilience to noisy data. The randomization techniques employed during training and prediction enhance the diversity and independence of the DTs, resulting in accurate and reliable predictions.

5.1.2 Logistic Regression

LR is a popular ML method used for binary classification problems, where the goal is to predict the probability of an event occurring. It is a type of regression analysis that models the relationship between a dependent variable and one or more independent variables. The dependent variable in LR represents the binary outcome or the event of interest, which can take only two possible values (e.g., 0 or 1, yes or no).

The LR algorithm applies the logistic function, also known as the sigmoid function, to map the linear combination of the independent variables to a value between 0 and 1. The logistic function transforms the output of the linear equation to the range of probabilities, representing the likelihood of the event occurring.

The equation for LR can be written as follows:

$$\sigma(z) = \frac{1}{1 + e^{-z}}$$

Where e is the base of the natural logarithm, and z is the linear combination of the independent variables and their corresponding coefficients. Figure 10, illustrates the sigmoid function.

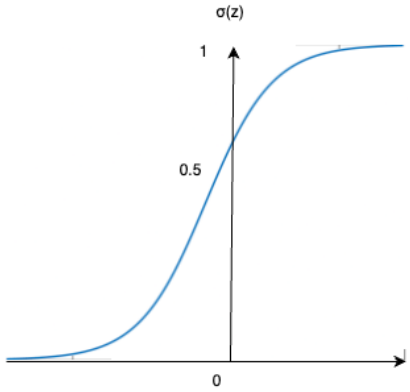

Figure 10: Sigmoid Function

To train a LR model, an optimization algorithm, such as gradient descent or maximum likelihood estimation, is used to estimate the coefficients that minimize the difference between the predicted probabilities and the actual binary outcomes in the training data (loss function). Once the model is trained, it can be used to predict the probability of the event occurring for new instances based on their independent variable values.

The main advantage of LR is its interpretability. It provides interpretable results by estimating the coefficients for each independent variable. These coefficients indicate the direction and magnitude of the relationship between the independent variables and the probability of the event occurring. This interpretability is valuable in domains where understanding the impact of individual features is crucial.

5.1.3 Extreme Gradient Boosting

XGBoost is an ML algorithm that as RF belongs to the ensemble learning family. The basic idea behind XGBoost is to create a strong predictive model by combining multiple weak models, known as DTs. As stated before, DTs are simple, tree-like structures that make predictions based on a series of if-then-else conditions. However, DTs alone often suffer from limitations such as overfitting or underfitting, resulting in suboptimal predictions.

XGBoost overcomes these limitations by iteratively adding DTs to the ensemble and optimizing the model's performance through the process of boosting. In boosting, each new tree is built to correct the mistakes made by the previous trees, focusing on the instances that were misclassified or had higher errors. This iterative process helps the model learn from its mistakes and improve its predictions.

Thus, basically, the algorithm starts with an initial DT and calculates the residuals to consider the magnitude of errors. It then constructs new trees, prunes them to prevent overfitting, and updates the ensemble based on their contributions. To do this, XGBoost applies regularization techniques during the tree construction process. It adds penalties to the loss function based on the complexity of the trees, which discourages the model from creating very complex ones. This process is repeated for a specified number of iterations. XGBoost incorporates various optimizations to enhance speed and accuracy.

Figure 11 illustrates the operational mechanics of the XGBoost model through an illustrative example. The XGBoost algorithm starts by calculating the prediction using the first learner.

Then, the residuals are computed based on the differences between the actual target values and the predicted values made by the learner. From here on, all the new weak learners (DTs) are trained to predict the residuals of their predecessors, with the goal of minimizing the errors or mistakes made by previous learners. In other words, it aims to capture the remaining errors or discrepancies in the current ensemble's predictions. The final prediction is obtained by combining the predictions of the multiple weak learners and adjusting their contributions with a learning rate.

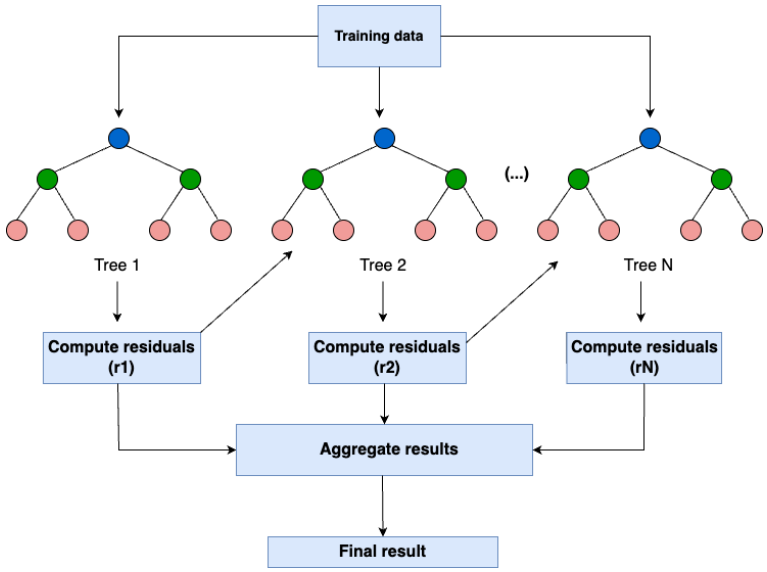

Figure 11: Simplified Structure of Extreme Gradient Boosting

XGBoost is optimized for efficiency and scalability, allowing it to train models and make predictions significantly faster than many other algorithms. It leverages parallelization techniques to effectively utilize multicore processors, leading to improved training speed and real-time prediction capabilities. This advantage is particularly valuable when working with large datasets or in time-sensitive applications where quick model training and inference are crucial.

5.2 Unsupervised Active Learning Policies

Unsupervised learning is a category of ML algorithms that focuses on analyzing unlabeled data. In contrast to supervised learning, where algorithms are trained using labeled data with known input-output pairs, unsupervised learning seeks to uncover patterns, structures, or relationships in the data without any predetermined guidance or existing labels.

Unsupervised learning algorithms can potentially be particularly useful in the early stages of the AL process when labeled data may be scarce or unavailable. These algorithms can help identify the most anomalous transactions or data points from the unlabeled pool, as they often contain the most informative fraud-related patterns that a supervised model should learn.

In the following sections, it will be provided a comprehensive explanation of three specific unsupervised algorithms: IF, LOF, and Elliptic Envelope. It will be specified how these algorithms work and their respective approaches to detecting anomalies or outliers in data.

5.2.1 Isolation Forest

The IF algorithm is a widely used traditional unsupervised AD technique in the literature. Developed by Liu et al. (2008), IF leverages the two key characteristics of anomalies to identify them within a dataset:

1. **Less**: A fundamental characteristic of anomalies is that they constitute a minority within a dataset, in comparison to the normal data points.
2. **Different**: Anomalies possess attribute values that deviate significantly from the typical characteristics exhibited by normal instances.

Taking this into account, the IF algorithm utilizes an ensemble of trees, known as Isolation Trees, and operates on the principle that anomalies within a dataset have different characteristics from non-anomalous instances, making them easier to identify and isolate. To do so, the algorithm uses a random partitioning method that recursively splits the sample by selecting a random attribute and a random value within its range, to progressively isolate the anomalous instances. Moreover, IF works under the assumption that anomalous instances are characterized by shorter average path lengths within the Isolation Trees (i.e., closer to the root of the tree), which is consistent with the principle that anomalous instances are more easily isolated, as previously stated. Thus, instances with shorter average path lengths are assigned higher anomaly scores, making them suitable for querying in AL.

Figure 12 illustrates the difference in random partitioning between a normal instance and an anomalous instance, thereby effectively demonstrating how the process of anomaly identification is done using the IF algorithm. On the left side of the figure, it can be observed that normal instances (x_i) take additional partitions to be isolated, because, in theory, they are

not that distinctive among other instances. On the other hand, due to their distinct characteristics, anomalous instances (x_0) are isolated with fewer partitions.

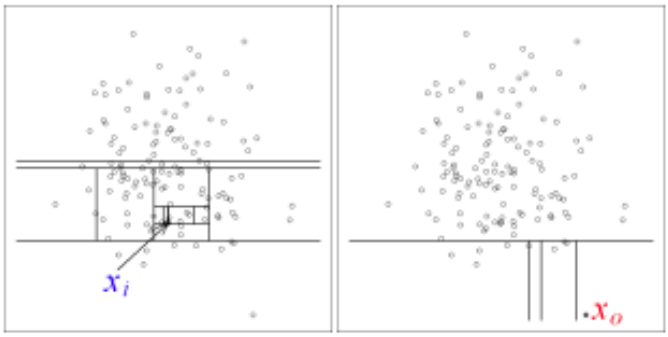

Figure 12: Isolating Data Points with Isolation Forest (Liu et al., 2008)

The IF approach in AD diverges from traditional methods by taking a different approach to identifying anomalies. Unlike conventional techniques that initially establish a profile of normal instances, IF directly utilizes a binary tree structure for AD.

Instead of relying on distance or density measures like many other methods, IF avoids the computational expense associated with calculating distances or densities. This allows for faster and more efficient AD.

5.2.2 Local Outlier Factor

LOF is another popular and well-established unsupervised algorithm used for AD. It works by assigning an anomaly score to each data point based on its local density compared to its neighboring points. LOF is particularly effective at identifying outliers in high-dimensional datasets, where traditional methods may struggle.

The LOF algorithm operates under the assumption that outliers are less densely surrounded by similar data points compared to the majority of the dataset. It measures the local density of a point by calculating the average distance between the point and its k nearest neighbors. The distance metric used, such as Euclidean distance, can vary depending on the specific implementation.

The LOF anomaly score is a ratio of the average local density of a point's neighbors to the point's own local density. A LOF score greater than 1 indicates that the point is potentially an anomaly, while a score lower than 1 suggests that the point is similar to its neighbors. Thus, in the context of AL, it can be considered for labeling the instances that would have higher LOF scores.

Taking this into account it is possible to break down LOF into the following steps:

1. For each data point, find its k nearest neighbors.
2. Calculate the reachability distance (RD) for each neighbor by taking the maximum of the distance between the data point and the neighbor and the distance between the neighbor and its kth nearest neighbor. Let x be the target and y be one of its neighbors:

$$\mathrm{RD}(\mathrm{y},\mathrm{x}) = \max(\mathrm{distance}(\mathrm{x},\mathrm{y}), distance_k(\mathrm{x}))$$

   Where distance(x,y) represents the distance between x and y, and $\mathrm{distance}_k(\mathrm{y})$ represents the distance between y and its kth nearest neighbor. The distance measure is problem specific (Euclidean, Manhattan, etc…).
3. Calculate the local reachability density (lrd) for each data point by taking the inverse of the average of the RD values for its neighbors. For each data point x, let N_k be its set of neighbors:

$$lrd_k(\mathrm{x}) = \frac{1}{\frac{\sum_{y \in N_k} RD(x,y)}{|N_k(x)|}}$$

   Where $|N_k|$ represents the number of neighbors of the data point x.
4. Lastly, the LOF score is computed for each data point by taking the average of the lrd values for its neighbors divided by its own LRD value. Let, again, for each data point x, N_k be its set of neighbors:

$$LOF_k(x) = \frac{\sum_{y \in N_k} lrd_k(y)}{|N_k(x)|} \times \frac{1}{lrd_k(x)}$$

   Where $\mathrm{lrd}_k(\mathrm{y})$ represents the local reachability density of y.

To have a better understanding, Figure 13 illustrates what could be an example of the reachability distance for each neighbor of x. It is possible to observe that, data points A and B have the same reachability distance, while C is not a k nearest neighbor.

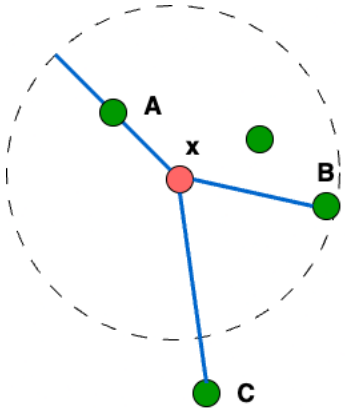

Figure 13: Local Outlier Factor Neighborhood Reachability Distance

5.2.3 Elliptic Envelope

The Elliptic Envelope algorithm is another method used for AD in ML tasks. It is specifically designed to detect outliers or anomalies in a dataset that may deviate significantly from the norm or expected behavior. Unlike some other techniques, the Elliptic Envelope algorithm assumes that the normal data points follow a multivariate Gaussian distribution. A multivariate Gaussian distribution is basically a generalization of the one-dimensional normal distribution to higher dimensions.

The algorithm works by fitting an elliptical envelope to the central observations of the dataset, which are considered to be the non-anomalous data points. This envelope represents the boundaries within which the normal data points are expected to lie. Any data point falling outside the envelope is considered an anomaly or a suspicious data point.

To determine the elliptical envelope, the algorithm needs to estimate the mean and covariance of the normal data points. The mean represents the center of the elliptical envelope, while the covariance matrix characterizes the shape and orientation. The estimation of these parameters is typically done using methods like maximum likelihood estimation.

Once the mean and covariance are estimated, the algorithm computes the Mahalanobis distance for each data point. The Mahalanobis distance measures the deviation of a point from the mean in terms of the covariance structure of the data. It takes into account the correlations between different features and scales the distances accordingly. The data points with larger Mahalanobis distances are more likely to be anomalies. In other words, the instances that would have a large Mahalanobis distance to a multivariate Gaussian

distribution fit to the unlabeled pool would be the ones to be considered for labeling in AL. The Mahalanobis distance formula can be written as follows:

$$D = \sqrt{(X-\mu)^T \times cov^{-1} \times (X-\mu)}$$

In this formula:

- D is the Mahalanobis distance.
- X is a vector of data.
- μ the mean vector of the distribution.
- $(X-\mu)^T$ denotes the transpose of the vector (X - μ).
- cov^{-1} is the inverse covariance matrix of the distribution.

Overall, the Elliptic Envelope algorithm offers a robust approach to detecting anomalies by modeling the normal data distribution and identifying deviations from it.

Figure 14 provides an intuitive illustration of the Elliptic Envelope algorithm. The blue line represents the learned decision function, which delimits the ellipse. In this context, the green points located within the ellipse are considered inliers, indicating that they exhibit behavior consistent with the majority of the dataset. On the other hand, the red points outside the ellipse are considered anomalous because they possess a larger Mahalanobis distance, indicating a significant deviation from the normal distribution.

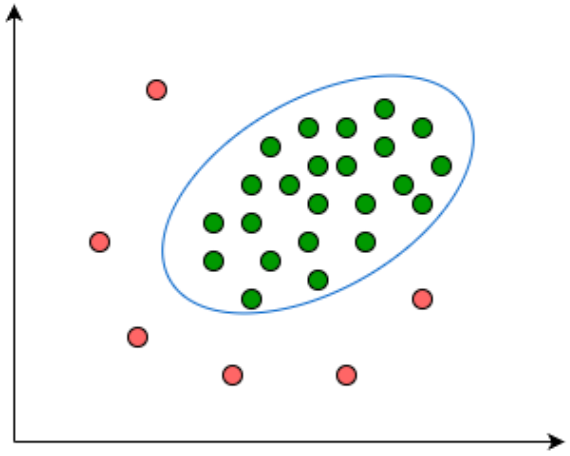

Figure 14: Simplified Structure of Elliptic Envelope

5.3 Supervised Active Learning Policies

As it was already briefly introduced in the Theoretical Framework chapter, many supervised AL policies have been proposed over the years. These optimal supervised AL policies, essentially, rely on supervised ML algorithms to define their queries.

It was concluded from the Literature review that the authors achieved very good results by applying these types of AL policies. Specifically, in the financial systems domain, the literature points out that switching to supervised learners at some point in the AL process improves the overall models' performance. This means that after defining an optimal point in the AL setup (e.g., the number of iterations completed or a threshold associated with the total number of illicit transactions in the labeled pool), a transition from an unsupervised AL policy to a more optimized supervised AL policy should be executed.

These types of AL policies are known to actively select the most informative instances based on different heuristics. They employ intelligent strategies to identify data instances that possess the highest potential for improving the model's performance, by leveraging the potential of the supervised algorithms.

In the following sections, it will be provided a comprehensive explanation of three specific supervised AL policies: Uncertainty Sampling, QBC, and EMC. It will be explained how each one of these algorithms works and how they effectively select the most informative instances to be labeled by a human analyst.

5.3.1 Uncertainty Sampling

Uncertainty sampling is a widely used AL strategy that, like most of the other strategies, seeks to meticulously choose informative instances from a set of unlabeled data for annotation. Nonetheless, this specific strategy harnesses the notion of uncertainty to pinpoint instances that pose challenges or uncertainties for a binary classification model to confidently classify. This strategy was originally discussed by Lewis & Gale (1994).

The process involves training a supervised ML model during each AL iteration utilizing the labeled pool instances. Subsequently, the unlabeled pool is evaluated, and the instances are prioritized based on a metric of uncertainty associated with their proximity to the classification boundary. Instances in closer proximity to the classification boundary are presumed to be more informative and have a higher likelihood of contributing to model improvement.

A widely adopted criterion involves the selection of instances that exhibit the highest expected entropy among the potential class labels, considering the model scores as probabilities. Therefore, in the context of a probabilistic learning model and a binary classification problem, this translates to the identification of instances whose predicted score approaches 0.5.

Let's consider a classification problem with C classes. For a given data point, if the predicted probabilities for each class are denoted as P(c), where c ∈ {1, 2, ..., C}, then the entropy of the predicted probabilities can be calculated as:

$$\mathrm{H(P)} = -\sum_{c=1}^{C} [P(c) * \log\big(P(c)\big)]$$

A minimal entropy of 0 represents a completely certain prediction (i.e., only one class has a probability of 1), and a maximum entropy represents maximum uncertainty (i.e., all classes have equal probabilities of $\frac{1}{C}$).

Therefore, in binary classification tasks, selecting the data points with probabilities closest to 0.5 (i.e., with the highest entropy) can be an effective way to maximize the information gain and reduce the uncertainty of the model. However, this method generally assumes scores that are well-calibrated, which is even more important in high-class imbalance scenarios. Calibrating the scores involves training the model to produce predicted probabilities that align closely with the true probabilities of the classes. By having well-calibrated scores, Uncertainty Sampling can more accurately identify instances that are close to the classification boundary and have high uncertainty. This enables the selection of instances that are most likely to improve the model's performance and enhance its ability to handle challenging examples in subsequent iterations of AL.

Figure 15 provides an intuitive illustration of the Uncertainty Sampling querying process. If it is assumed that three instances will be queried to be labeled by a human analyst, it would be selected the three red points that are closest to the decision boundary.

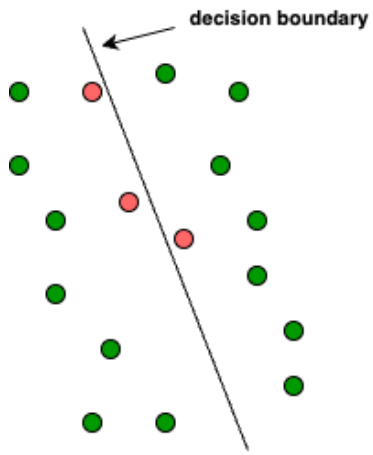

Figure 15: Simplified Structure of Uncertainty Sampling Policy

In this study, it is used the same type of supervised classifier for Uncertainty Sampling and for evaluating the performance of the AL setup in the test set. For instance, if for a certain AL setup, it is used RF as a classifier on the test set, it is also conducted Uncertainty Sampling using RF. The same applies to all the supervised classifiers explained in the Supervised Learning chapter.

5.3.2 Query By Committee

QBC is another important strategy in AL. The main goal of QBC is to improve the accuracy and performance of classification models by incorporating diverse opinions from a committee of members.

The QBC algorithm operates in an iterative fashion. Initially, a committee of classifiers, selected by the user, trains on distinct subsets of the labeled data. Subsequently, each classifier within the committee is employed to assign scores to the unlabeled instances. The algorithm proceeds to compute a measure of disagreement among the committee members, thereby identifying the instances that possess the greatest potential for being labeled by an oracle. Notably, instances with higher ranks correspond to those instances in which the committee members exhibit the most substantial disagreements. Similar to the Uncertainty Sampling algorithm, the QBC algorithm also assumes scores that are well-calibrated.

There are many different ways to measure the disagreement between the predictions of the ensemble of classifiers for each sample in the unlabeled pool. However, the most popular one is the average Kullback-Leibler (KL) divergence. The KL divergence measures the difference between two probability distributions. In the context of QBC, it provides an indication of the

dissimilarity between the predictions made by each classifier in the committee with respect to the average predicted probabilities across all committee members.

Let N be the committee members. To calculate the average KL divergence, it is used, for each data point, the following formula:

$$\frac{1}{|N|}\sum_{m\in N} \mathrm{D}(P_m||P_{avg})$$

In this formula:

- |N| represents the total number of committee members.
- P_m is the predicted probabilities distribution for the m committee member.
- P_{avg} is the average probability distribution across all committee members.
- The $\mathrm{D}(P_m||P_{avg})$ represents the Kullback-Leibler divergence between the two distribution probabilities.

Now, for a given data point, if we denote the predicted probabilities for each class as P(c), where c ∈ {1, 2, ..., C}, the KL divergence (D) between two distributions can be calculated as follows:

$$\mathrm{D}(\mathrm{P1}||\mathrm{P2}) = \sum_{c=1}^{C} \mathrm{P1(c)} * \log(\frac{P1(c)}{P2(c)})$$

Where, P1(c) and P2(c) are the probabilities of class c in distributions P1 and P2, respectively.

For QBC, in the experiments, it was used the ensemble supervised learning method RF for all the tested AL setups, regardless of whether the classifier employed to assess the performance of the AL setup on the test set is RF, LR, or XGBoost. The objective of employing this AL policy with the RF algorithm is to identify instances where the DTs within the RF ensemble exhibit the highest level of disagreement.

The rationale behind selecting RF as the AL method stems from the fact that its DTs are trained independently on distinct subsets of the training data. This independent training process allows each DT to specialize and provide unique information for the final prediction.

5.3.3 Expected Model Change

Another important AL strategy is the EMC method, which employs a decision-theoretic approach to identify instances that exert a stronger influence on the model. The EMC technique accomplishes this by querying unlabeled instances that cause the most substantial

modification in the model's parameters, quantifying the effect of labeling a particular unlabeled instance on the gradient of the model's loss function. In simpler terms, the EMC method focuses on labeling unlabeled instances that would produce the most significant alterations to the current model if their labels were known.

A widely recognized example of an EMC algorithm is the Expected Gradient Length (EGL) approach (Settles et al., 2007). In the context of this study, the experiments will employ the EGL algorithm. It is important to note that the EGL algorithm is designed specifically for gradient-based classifiers.

Assuming that the model is trained using gradient descent, the process entails identifying the unlabeled sample, denoted as $(\{x_{ij}\}, y_{ij})$, that, upon inclusion in the labeled pool (i.e., the training set), would induce the most substantial change in the gradient of the objective function. In other words, this unlabeled sample would result in the largest gradient vector used to re-estimate values for θ (parameters of the model).

If we denote $\nabla L(\theta)$ as the gradient of the loss function with respect to θ, which, essentially is a vector composed of the partial derivates of L (the loss function), with respect to each model parameter, ∇L_{ij}^{+} would represent the updated gradient when the positive tuple $(x_{ij}, 1)$ is added to the training data (the labeled samples), and, similarly, ∇L_{ij}^{-} would be the new gradient by adding the negative tuple $(x_{ij}, 0)$. Since we are dealing with unlabeled data, the label to be provided by the human analyst cannot be known in advance. Hence, EGL is computed based on the learner's current belief o_{ij} in each outcome:

$$EGL(x_{ij}) = o_{ij}\left\|\nabla L_{ij}^{+}(\theta)\right\| + (1 - o_{ij})\left\|\nabla L_{ij}^{-}\right\|$$

Where, $\|.\|$ is the Euclidean norm of each resulting gradient vector.

The EMC can be conceptualized as a weighted sum encompassing all potential labels, given that the true labels of instances remain unknown until queried. Consequently, in each iteration, are prioritized for querying the labels of instances that exhibit the highest expected gradients.

During the querying phase, it is anticipated that the gradient of the loss function will approach zero, assuming convergence of the loss function in the previous training round. Therefore, for computational efficiency, it is possible to calculate the new gradients only for the positive and negative instances, not considering the remaining training data (i.e., the labeled samples), assuming that training instances are independent.

In the context of this work, for EMC, it was manually built a binary LR trained with gradient descent. In LR, the goal is to find the optimal set of model parameters that maximize the likelihood of the observed data. This is typically achieved by minimizing the binary cross-entropy loss, which measures the dissimilarity between the predicted probabilities and the true binary labels.

The binary cross entropy loss can be given by the following formula:

$$LCE(P, y) = -\frac{1}{m}\sum_{i=1}^{m} y_i \log(P_i) + (1 - y_i)\log(1 - P_i)$$

In this formula:

- m is the number of samples.
- P_i refers to the prediction of sample i.
- y_i the true label of sample i.

To find the optimal parameters, gradient descent is employed as an optimization algorithm. Gradient descent is an iterative optimization algorithm that adjusts the model parameters based on the gradients of the loss function with respect to those parameters. It aims to find the global minimum of the loss function by iteratively taking steps in the direction of steepest descent. In this case, it is computed the gradients of the cross-entropy loss, as the partial derivative regarding each one of the model parameters (i.e., the weights/coefficients and bias/intercept).

The partial derivative of the loss function, with respect to the bias can be obtain as follows:

$$\frac{\partial LCE(P, y)}{\partial b} = \frac{1}{m}(P - y)$$

On the other hand, the partial derivative of the binary cross entropy loss, with respect to the weighs can be written as follows:

$$\frac{\partial LCE(P, y)}{\partial w} = \frac{1}{m}(P - y)X^T$$

Where, X^T refers to the transposed of the input feature matrix X.

5.4 Evaluation Metrics

In order to enhance comprehension and facilitate the evaluation of prediction quality across the various methods to be examined in the experimental section, this chapter presents an

overview of the metrics to be employed. These metrics are intended to provide a standardized means of assessing and quantifying the effectiveness of the generated predictions.

5.4.1 Precision

Precision is an ML metric that measures the accuracy of positive predictions made by a model. In other words, Precision evaluates how well a model performs in correctly identifying positive instances out of all instances it predicted as positive.

A high Precision value indicates that the model has a low rate of FP, meaning that when it predicts a positive result, it is more likely to be correct. On the other hand, a low Precision value suggests a high rate of FP, indicating that the model is incorrectly classifying negative instances as positive. Precision is particularly useful when the cost of FP is high.

The Precision metric can be given by the following formula:

$$Precision = \frac{TP}{(TP + FP)}$$

In this formula:

- TP is the number of positive instances correctly identified or classified as positive by an ML model.
- FP is the number of negative instances incorrectly identified or classified as positive by an ML model.

5.4.2 Recall

The Recall metric is a performance measure used in ML to evaluate the effectiveness of a classification model, particularly in scenarios where identifying TP instances is of high importance. It is also known as the TPR or sensitivity.

Recall measures the proportion of actual positive instances that are correctly identified by the model. It quantifies the model's ability to identify all relevant positive instances in the dataset. A high Recall value indicates that the model is proficient at capturing positive instances, minimizing the number of False Negatives (FN). Conversely, a low Recall value implies that the model is missing a significant number of positive instances, resulting in a higher number of FN.

The Recall metric can be given by the following formula:

$$Recall = \frac{TP}{(TP + FN)}$$

Where FN refers to the number of instances that are actually positive but are incorrectly classified as negative by the model.

5.4.3 F1-score

The F1-score is a commonly used evaluation metric in ML. It combines Precision and Recall into a single value, providing a balanced measure of the model's performance.

The F1-score is the harmonic mean of Precision and Recall, combining both metrics to provide a single evaluation value. It balances Precision and Recall by taking their harmonic mean, which gives more weight to lower values. This makes the F1-score useful in scenarios where both FP and FN need to be minimized.

A higher F1-score reflects an improved balance between Precision and Recall, indicating the overall effectiveness of a model in accurately predicting positive instances while minimizing FP and FN.

The formula for calculating the F1-score is as follows:

$$F1_score = 2 \times \frac{(Precision \times Recall)}{(Precision + Recall)}$$

The F1-score holds great importance as an evaluative measure in gauging the overall effectiveness of classification models and serves as a standard for comparison when analyzing various algorithms.

6. EXPERIMENTS

This section aims to present a comprehensive overview of the proposed solutions and the experimental setup. It will provide a detailed description of the experiments designed to address the objectives and challenges outlined in the introduction of this document. By doing so, it aims to offer a clear understanding of the methodologies employed.

6.1 Data Pre-processing

Data pre-processing is an indispensable aspect of any research undertaking, particularly when it pertains to refining data for the precise problem at hand and aligning it with the needs of ML models. This section investigates the fundamental data preparation process, examining the necessary modifications to ensure the dataset's suitability for the specific problem and its smooth integration with the chosen ML models. By implementing these vital procedures, the primary aim is to improve the correctness of the dataset and enhance its efficacy during the modeling stages.

The main changes made to the dataset are described as follows:

- **Convert the categorical labels to integers**: The original target labels in the dataset were categorical, represented as '1', '2', and 'unknown'. In order to enhance comprehensibility, the target labels were transformed from categorical values into integers. Specifically, '2' was converted to 0 to indicate licit transactions, '1' was changed to 1 to represent illicit transactions, and 'unknown' was transformed to -1.
- **Consider only the labeled samples**: This step involved filtering the dataset to focus solely on labeled samples. As stated, the dataset comprised three primary target categories: 0 (licit transactions), 1 (illicit transactions), and -1 (unknown transactions). However, to test the supervised algorithms and to simulate the AL setup effectively, it was only considered the labeled samples (licit and illicit). This approach enabled this work to associate the original labels with the transactions chosen for querying. Consequently, it eliminated the need for a human analyst to review all selected transactions.

- **Rename the transaction features:** In order to improve the comprehensibility of the feature datasets, it was assigned more representative names for the transaction features. For example, it was assigned column names such as 'local_feature_1', or ‘aggregated_feature_1'. Note that there are 93 local features and 72 aggregated features. This renaming step was essential because the original dataset lacked descriptive names for all the transaction features. By undertaking this process, the aim was to make the dataset more interpretable and provide a clearer understanding of the underlying information each feature represents.
- **Merge classes and features datasets**: This step involved merging the features and class datasets, thereby combining the label and feature information for each transaction. As explained in Chapter 4.2, initially, the classes and feature information for each transaction were stored in separate datasets. To perform this merge operation, the transaction IDs from both datasets were mapped together. Since the scope of this work was not to explore the Bitcoin transaction graph structure, the dataset containing the edge information was discarded.

After the data preparation steps, the refined dataset consisted of 46,564 labeled transactions.

6.2 Experimental Setup

In this chapter, it is presented a comprehensive overview of the experiments conducted in this study, specifically focusing on the supervised baseline and AL setups.

To maintain consistency with the work of Lorenz et al. (2020) on the same dataset, it is followed a similar approach in dividing the data into sequential train and test datasets for all experiments. For all the experiments, the training dataset encompasses 29,894 transactions, comprising all labeled samples until the 34th time step, while the testing dataset consists of 16,670 transactions, including labeled samples from the 35th time step onwards. It’s important to note that the train set comprises 10% of illicit cases.

Additionally, to guarantee reproducibility and consistency in the experiments, it was implemented a systematic approach using a predetermined seed value (e.g., 8347658) at the start of the code execution. This seed value was utilized to initialize the random number generator, which determined the sequence of random operations throughout the

experiments. By following this methodology, it was possible to ensure that the experiments could be reliably replicated, enabling consistent evaluation and comparison of the models.

6.2.1 Supervised Baseline

To establish a supervised baseline for benchmarking various AL setups, the initial step of the experiments involved training a set of supervised classifiers on the 166 features of the train set and then evaluating them on the test set. This involves training the supervised classifiers on the entire available training set (29894 transactions).

By testing different supervised classifiers as a baseline, it becomes simpler to grasp the real-world significance of the AL setups. Thus, comparing the performance of the AL setups (evaluated using a supervised classifier trained on a sample of the total training set) against this baseline enables the understanding of whether specific AL setups can match or even outperform the performance of the supervised baseline, by using only a small amount of labeled transactions, obtained through the AL process. By doing so, the necessity of labeling the entire dataset can be minimized.

As already mentioned in Supervised Learning chapter, three supervised algorithms were tested: RF, XGBoost, and LR. It was used the scikit-learn implementation of RF, the scikit-learn implementation of LR, and the Python implementation of XGBoost. To ensure consistency with the work of Lorenz et al. (2020) the results are achieved using the default parameters of the models. The option for testing the algorithms with the default parameters was due to replicability concerns. Since this is mainly a research study, setting the parameters as default, will ensure easier replicability of this study for further investigation.

A comprehensive summary of the model parameters used for each supervised classifier can be found in Table 17.

For evaluating the performance, the maximum F1-score is reported. This score is determined by finding the threshold that maximizes the F1-score based on the predicted probabilities from each supervised classifier. Furthermore, the Precision and Recall values are also reported at this specific threshold, providing a comprehensive evaluation of the classifier's performance. Note that, all these metrics were reported using the scikit-learn library. By adopting this customized threshold selection strategy, it achieved a more refined evaluation of the model's ability to identify and classify relevant transactions. The main objective is to

optimize the F1-score, aiming for higher accuracy and effectiveness in detecting fraudulent activities within the dataset.

In addition to the maximum F1-score, the performance of the supervised classifiers over time is measured by reporting the F1-score per time step on the test set. This enables tracking the classifiers' effectiveness and identifying any variations in performance throughout the duration of the evaluation. By analyzing the F1-score over time, it is possible to gain insights into the classifiers' consistency and adaptability in handling the test data.

Lastly, to introduce additional randomness and variability, it was implemented a unique random state for each one of the supervised classifiers tested. The random state was generated using a random integer function. Subsequently, this newly generated random state value was passed as a parameter ("random_state=random_state*") during the model initialization phase.

6.2.2 Active Learning

Since the aim of this dissertation is to explore the feasibility of using AD algorithms in combination with AL techniques, in this section, a detailed explanation of the structure of the AL setups employed and tested in this work is provided. Additionally, all the parameters used for each one of the AL policies algorithms are also detailed.

As was concluded in Chapter 3, the literature, on this subject matter, points out to the benefits of the combined use of AD techniques with more sophisticated supervised AL policies, when structuring the AL setup. In other words, at some point based on a certain threshold, switching to a more optimized supervised hot learner (i.e., such as Uncertainty Sampling, EMC, and QBC) would improve the overall models' performance.

Particularly, Lorenz et al. (2020) provided a suitable framework for combining AD algorithms with AL techniques. This specific framework will be used in this work, however, with some differences that will be explained further, in this section.

To delve deeper into the framework outlined by the authors, they specifically combined four different query strategies in their AL experiments. Two of them were supervised (EMC and Uncertainty Sampling) and two of them were unsupervised (Elliptic Envelope and IF). All these AL policies were specifically trained on the labeled pool and tested in the unlabeled pool to find the most relevant transactions for querying. The main objective of using the unsupervised

AD techniques trained on the labeled pool was to find the most outlying instances on the unlabeled pool with regards to the labeled pool, in order to have a more diverse labeled pool at the end of the AL process.

In their experiments, the authors combined unsupervised and supervised AL policies, depending on the number of fraudulent instances on the labeled pool. More specifically, when the labeled pool possessed at least one illicit transaction, they would switch to a supervised policy. However, since they train all the AL policies in the labeled pool and at the beginning of the process there are no labeled transactions, they started all the tested setups by randomly sampling the first batch of transactions to be labeled. However, this raises some questions about their work, because since there are 10% of illicit transactions in the Elliptic Dataset (i.e., only considering the labeled samples from the original dataset, thereby not considering the unknown ones), the initial random sampling of one batch of 50 transactions would most likely, include on average 5 illicit transactions. Because of that since the labeled pool after the first iteration would already include one illicit transaction, they would already switch to the supervised hot learners on the second iteration of the AL setup, thereby not using or exploring the potential of unsupervised AD algorithms as unsupervised policies.

Note that all these aspects were acknowledged after replicating the work of Lorenz et al. (2020). Thus, based on all these conclusions acknowledged after replicating the work of these authors, some questions emerged:

- Why start with Random Sampling, when it is possible to start immediately with an unsupervised AL policy trained on the unlabeled pool?
- What is the real feasibility and practical relevance of constructing AL setups where AD algorithms are used as unsupervised policies?
- Is there any optimal point (e.g., the number of completed iterations in the AL setup) to change from an unsupervised AL policy to a more sophisticated supervised AL policy?

Based on this, in order to really explore the feasibility of using AD algorithms for AL, it was needed to partially adapt the AL framework provided by the authors, to address the objectives of this work. Follows, a comprehensive overview of all the AL setups tested in the experiments. As was explained in Chapter 5, in the AL experiments, there were combined six different query strategies. Among these, three were unsupervised, namely Elliptic Envelope, IF, and LOF. The

remaining three strategies employed were supervised in nature, including EMC, QBC, and Uncertainty Sampling.

It is used, for all the experiments, a batch size of 50 transactions sampled at each iteration of the AL process, until a total of 3000 labeled transactions (60 iterations) are collected. This means that at each iteration only the top 50 most informative/relevant transactions will be selected for labeling. This corresponds to executing each AL setup till the labeled pool comprises approximately 10% of the total transactions originally available in the training set.

It's important to note that at the beginning of the execution of each AL setup, the training set is divided into two pools: labeled and unlabeled pools. The labeled pool initially contains no labeled transactions (cold start scenario), but as the AL process progress, it gradually accumulates labeled transactions through iterative increments of 50 transactions. The transactions chosen for labeling during each iteration are subsequently removed from the unlabeled pool for further iterations.

As supervised algorithms, there were utilized the three supervised classifiers evaluated in the supervised baseline: RF, LR, and XGBoost. The performance of each AL setup was assessed through the maximum F1-score at each labeled pool size. Meaning that at each iteration, a supervised classifier will be trained on the available labeled pool and assessed its performance on the test set. The performance of each supervised classifier is compared against its respective supervised baseline described in Supervised Baseline chapter. Additionally, there will be provided the respective learning curves of each AL setup execution showing the evolution of the F1-score through the different labeled pool sizes.

The AL experiments were divided into two main steps/scenarios:

- **Scenario 1 – Unsupervised AL policies**: In this scenario, the focus is on utilizing only unsupervised query strategies. The aim is to assess the effectiveness of unsupervised AD algorithms as AL policies to iteratively (at each iteration) query the most relevant fraud-related patterns within the unlabeled pool (the most anomalous transactions). Instead of transitioning to a supervised AL policy, this approach solely relies on an unsupervised AD algorithm used as an AL policy throughout the entire AL process. To provide an additional baseline for comparison, there will also be evaluated setups where a Random Sampling algorithm is used instead of an unsupervised learner. Setting this baseline, where Random Sampling is used instead of an AD algorithm, is

crucial for comprehending the true influence of utilizing an AD algorithm as an AL policy to address the cold start scenarios. It is anticipated that an AD algorithm would outperform and identify more meaningful illicit transactions than an algorithm that randomly samples transactions without analyzing any underlying patterns in the data.

- **Scenario 2 – Combine unsupervised and supervised AL policies:** This particular scenario aims to determine the optimal point at which to switch from an unsupervised to a supervised AL policy. In this scenario, the AL setups will start by querying the most anomalous transactions but only until a certain threshold occurs. To define this optimal cut-off point, it was considered the number of iterations (AL loop executions) as a basis. Specifically, once a predefined number of iterations is completed, the setup will automatically transit from an unsupervised learner to a supervised learner. This structure was evaluated by testing them for all the possible cut-off points in terms of the number of iterations, ranging from 1 to 59 iterations. By doing this, it is possible to assess if there is a certain cut-off point in the AL setup up to which it is beneficial to use AD algorithms.

Given these previous considerations, Figure 16 depicts the operational mechanisms of the AL experiments. This visualization represents a simplified structure of the execution of each AL setup. The condition "iteration <= threshold" determines whether an unsupervised or supervised AL policy is employed. When the condition is true, it indicates that the current iteration of the AL process is still below the predetermined threshold (cut-off point), indicating the execution of an unsupervised AL policy. Conversely, when the condition is false, it means that the current iteration has surpassed the threshold, triggering the execution of a supervised AL policy. Each AL setup executed in the experiments comprises a combination of one unsupervised policy, one supervised policy (depending on the scenario being executed), and one supervised classifier. It is worth noting that each loop execution corresponds to a new iteration of the AL process and each AL setup is executed for a total of 60 iterations. At each iteration of the AL process, 50 transactions will be queried for labeling.

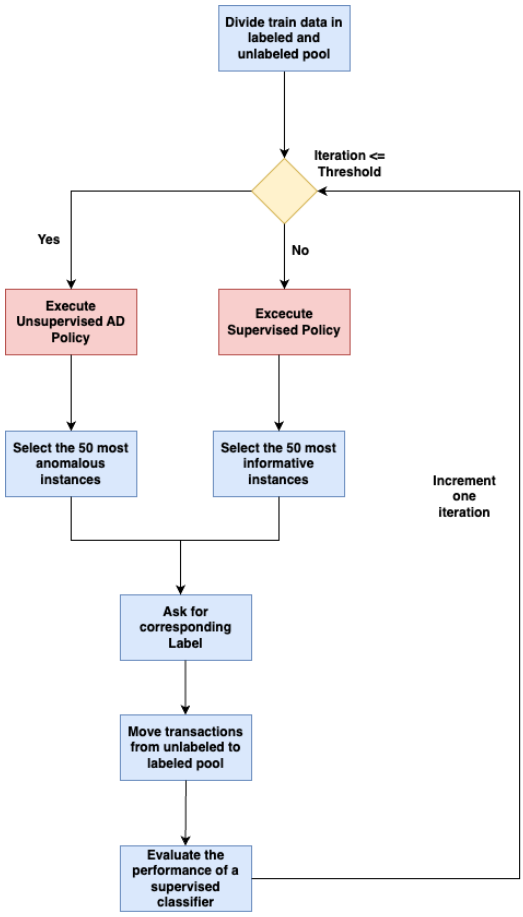

Figure 16: Simplified Structure of an Active Learning Setup

It is possible to observe that all the AL setups will start with an unsupervised AL policy since the condition “iteration<=threshold” will never be false at the first iteration. It is imperative to start the AL process with an unsupervised AD algorithm since there are no labeled transactions at the beginning. As the threshold criterion is met a supervised AL policy is implemented. Particularly, Scenario 1 will be executed when the predefined total number of AL iterations per setup (60 iterations) is equal to the predefined threshold. Therefore, in that case, it will never be executed the supervised AL policy since the condition “iteration<=threshold” will always be true for every iteration in the AL process. The supervised policies are trained on the available labeled pool to identify the most informative instances within the unlabeled data. On the other hand, the unsupervised policies are trained on the unlabeled pool and the anomaly scores are also computed on the same unlabeled pool.

Each AL setup will be run one time, with the exception of the baseline setups that use the Random Sampling algorithm. Since it is expected, with a completely random algorithm, more variability, each of the setups that use the Random Sampling algorithm were executed five times. In this case, it was reported the median F1-score at each labeled pool size.

Moreover, it is important to note that the human feedback loop from the AL process was simulated, eliminating the need, in the experiments, for a human analyst to manually review each selected transaction for querying. Instead, the human feedback loop was emulated by associating the true class labels with the transactions chosen for annotation. Initially, the true labels remained concealed, and as a set number of transactions were selected for labeling, the true labels were disclosed.

Below is a comprehensive overview of the models and parameters utilized for the AL experiments:

- A summary of the model parameters used for each unsupervised AL policy (IF, LOF, and Elliptic Envelope) implemented using the PyOD library, can be found in Table 18 from Appendix II. There were used the default parameters.
- For Uncertainty Sampling, it is employed the same supervised classifier that is used to evaluate the performance of the AL setup on the test. For example, if an RF classifier was used to evaluate the AL setup's performance, RF was also used for Uncertainty Sampling. Table 19 in Appendix III provides a comprehensive overview of the parameters used for RF, LR, and XGBoost for Uncertainty Sampling. To ensure well-calibrated scores, the class_weight was set to 'balanced'. All other parameters followed the default settings for each model.
- Regarding QBC, it was used the scikit-learn implementation of RF with n_estimators=100, max_depht=3, and class_weight='balanced'. For each unlabeled instance, this approach measures the disagreement between the predicted probabilities of 100 DTs. For a detailed list of all parameters used, refer to Table 20 in Appendix IV.
- Considering EMC, a custom binary LR model trained with gradient descent was built, as explained in Chapter 5.3.3. The loss function and gradient computations are described in that chapter of the dissertation document. Table 21 from Appendix V provides the list of parameters used for EMC.

Further, to introduce additional randomness and variability, a unique random state is generated at each iteration. This random state was obtained using a random integer function. During the model initialization phase, the newly generated random state value was passed as a parameter ("random_state=random_state*").

It is worth emphasizing that additional approaches were tested to explore the unsupervised AD policies. These additional approaches were developed with the goal of enhancing the performance of unsupervised policies. The following points provide a description of the main approaches that were tested:

- **Feature subset selection for training AD algorithms**: This was an exploratory approach that involved training AD algorithms using different subsets of features from the dataset. This was considered a valuable approach as it potentially would help to identify the features that may introduce noise and negatively impact the accuracy of prediction results. By determining the most significant features, it becomes possible to train the anomaly detectors more effectively and enhance their prediction accuracy.
- **Examining Bitcoin transaction network properties**: The second approach involved considering the inherent characteristics of the Bitcoin transaction network. Since illicit transactions often have connections to other illicit transactions within the network, it becomes possible to utilize this information by integrating it with the anomaly scores generated by the anomaly detector. This policy incorporated the anomaly scores calculated for the transactions within the unlabeled pool along with the transactions associated with previously identified illicit activities (illicit transactions that were already discovered in previous iterations). As a result, only transactions with higher anomaly scores, which were also connected to other known illicit transactions in the network were selected for labeling.
- **Consider the unknown transactions**: Approximately 77% of the dataset consists of unknown transactions. Exploring this portion of the data could prove valuable as it has the potential to provide a broader range of information that could potentially enhance the AD algorithms. Thus, the main idea was to train the anomaly detector in each iteration using all transactions up to the 34th timestep. This includes considering the transactions labeled as unknown as well as the remaining transactions in the unlabeled pool. The anomalous score would then be computed solely for the unlabeled pool.

Although these specific approaches did not yield significant results, it is important to acknowledge and document them as they were conducted as part of this dissertation. However, the findings of these endeavors will not be included in the Results and Discussion chapter due to their lack of significance.

7. Results and Discussion

In this chapter, a thorough examination is presented regarding the outcomes attained through the experimentation of the methodologies elucidated in the preceding chapters. Additionally, it delves into the primary discoveries derived from these outcomes.

7.1 Supervised Baseline

The research findings suggest that the supervised baseline models outperformed Lorenz et al. (2020) study in terms of F1-score performance. This improvement can be credited to the study's emphasis on determining the ideal thresholds for maximizing the F1-Score, instead of relying on the default threshold offered by the scikit-learn library. By specifically analyzing the thresholds that produced the highest F1-Scores, this approach significantly enhanced the supervised model's performance within the scope of this investigation.

Figure 17 presents the evolution of Precision, Recall, and F1-score across different thresholds.

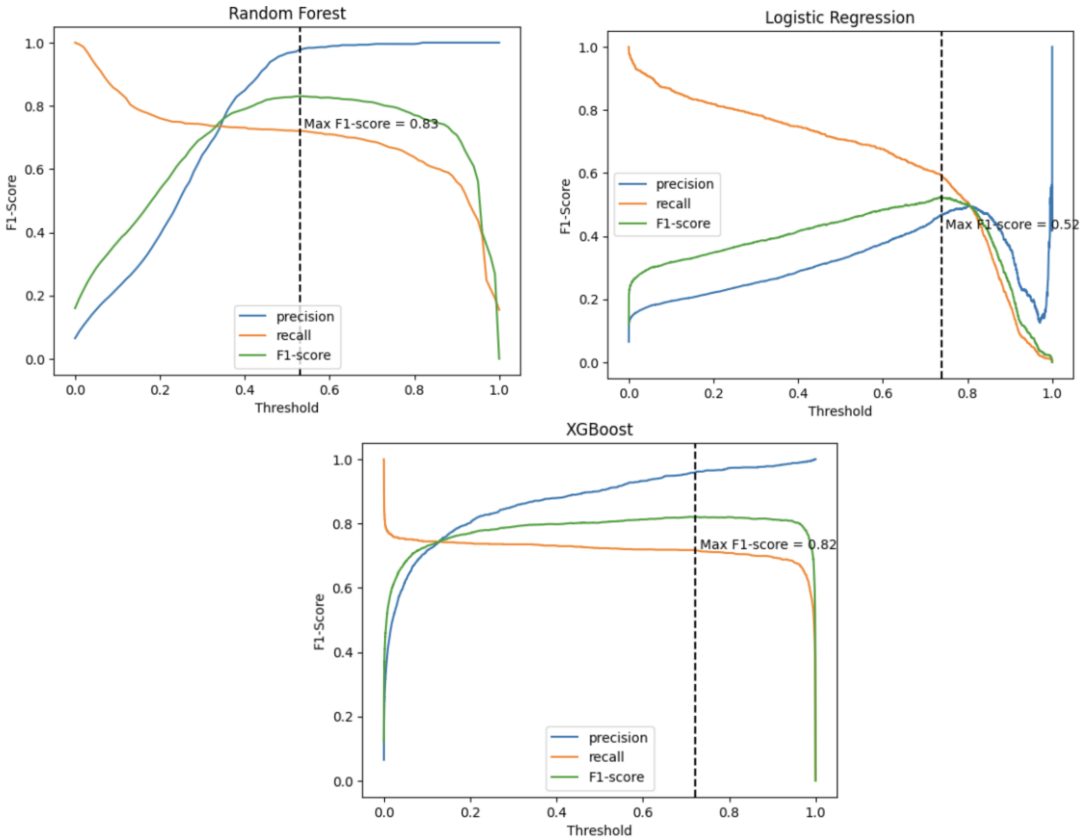

Figure 17: Supervised Baseline Metrics by Threshold

After locating the peak F1-score, the optimal threshold is established, followed by the extraction of the Precision and Recall values associated with this particular threshold. A summary of the results for each model can be found in Table 1.

Table 1: Supervised Baseline Results

Model	F1-score	Precision	Recall
RF	0.83	0.98	0.72
XGBoost	0.82	0.96	0.72
LR	0.52	0.47	0.59

The results obtained from the experiments indicate that both the RF and XGBoost models exhibit commendable performance in terms of F1-score, Precision, and Recall. With F1-scores of 0.83 and 0.82 respectively, these models demonstrate good overall performance. Furthermore, both models achieve high Precision scores, with RF at 0.98 and XGBoost at 0.96, suggesting that the majority of positive predictions made by these models are accurate.

On the other hand, the LR model performs comparatively less, as evidenced by its lower F1-score of 0.52. This suggests a poorer overall performance compared to RF and XGBoost models. The LR model also exhibits a lower Precision score of 0.47, indicating a higher number of FP predictions. This model showed a trade-off between identifying more TP instances and generating more FP, with a higher Recall but a lower Precision rate.

Figure 18 displays a t-SNE projection that visualizes the distribution of the true labels within the test set, providing a realistic depiction of their spatial arrangement.

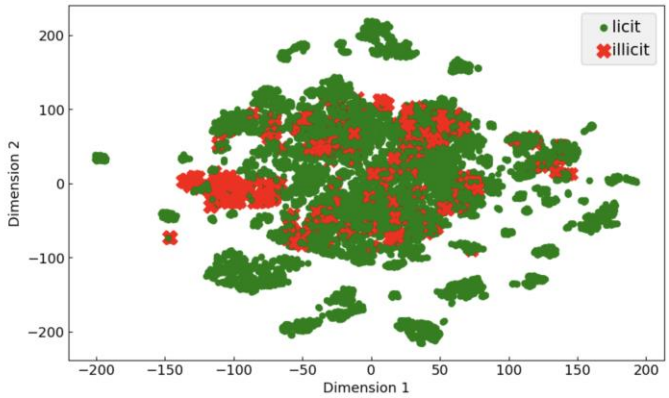

Figure 18: t-SNE Projection on the Test Set, Colored by the True Labels

Based on Figure 18, it becomes evident that a considerable number of illicit transactions are concentrated on the left side of the plot. Furthermore, the remaining illicit transactions are distributed across the entire plot.

Figure 19 presents a t-SNE projection that provides a visual representation of how the predicted labels are distributed within the test set for each of the supervised classifiers evaluated in the supervised baseline.

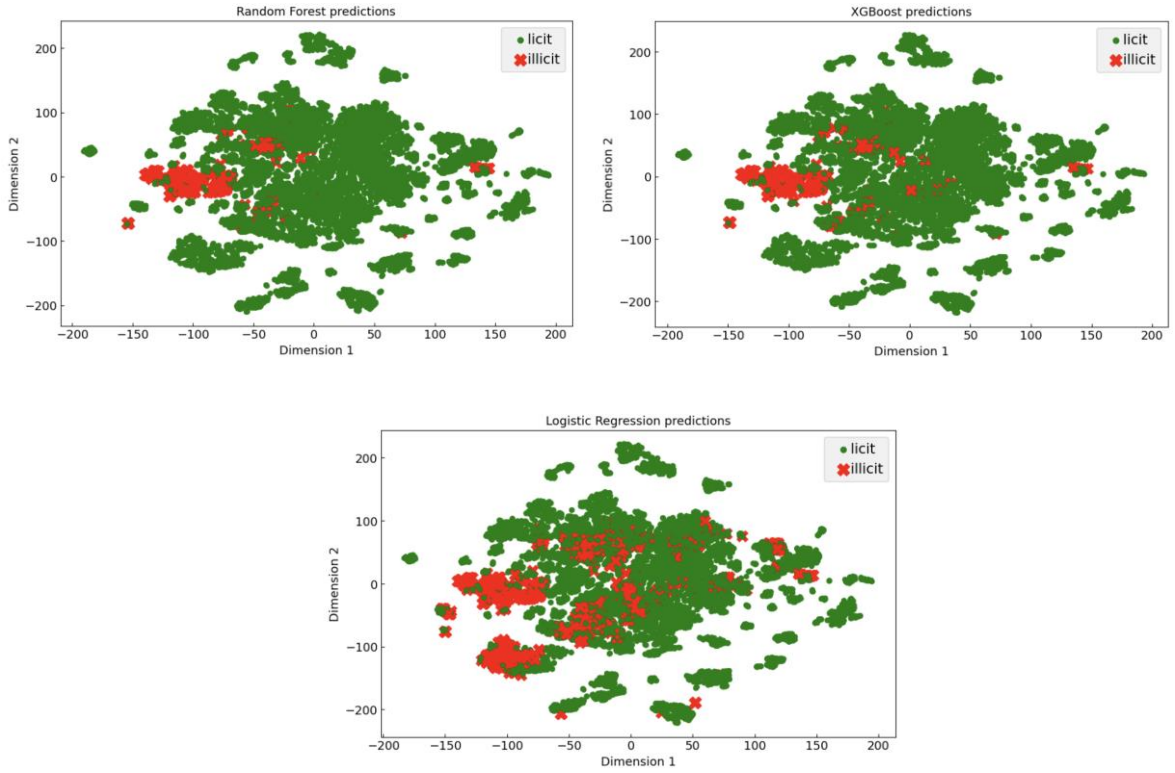

Figure 19: t-SNE Projection on the Test Set, Colored by the Predicted Labels

Based on the analysis of these plots, it becomes evident that both the RF and XGBoost models exhibit remarkable similarities in their performance. Notably, they effectively identify the significant portion of illicit transactions, that are prominently clustered on the left side of the plot. These visualizations align with previous considerations, indicating that the majority of positive predictions made by these models are indeed accurate, resulting in high Precision values. However, these methods, still, fail to capture a vast number of illicit transactions (recall is substantially lower). In contrast, the LR model displays inadequate results, indicating its poor performance in correctly identifying the positive/fraudulent instances.

Figure 20 depicts the F1-score performance of each supervised classifier across the different time steps.

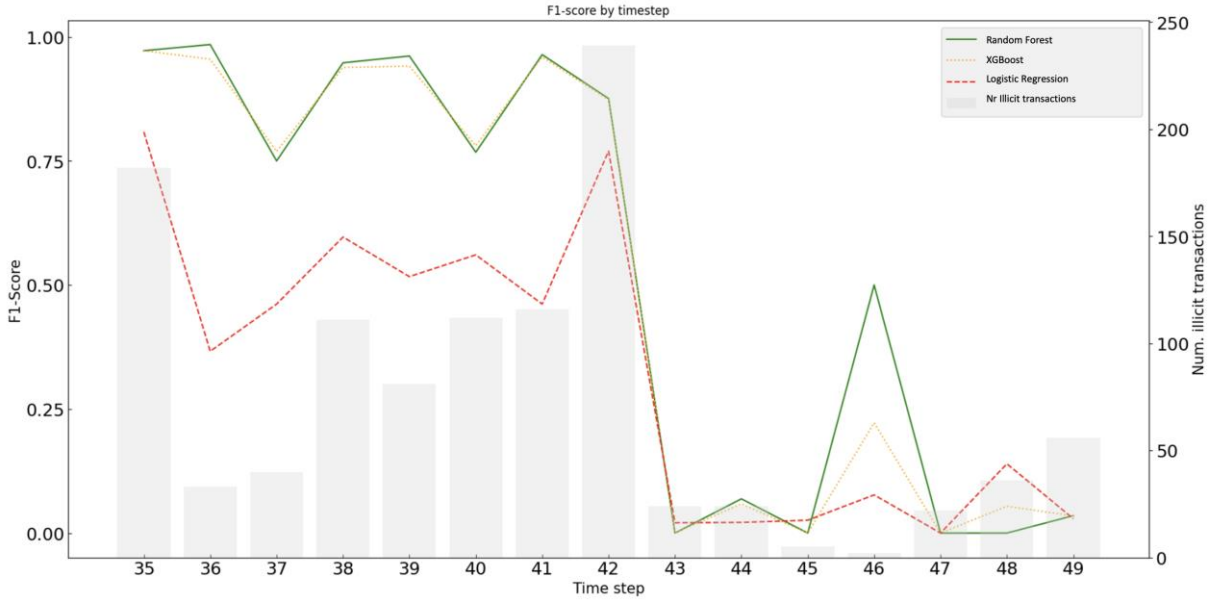

Figure 20: F1-score by Time Step, for Each Supervised Classifier

Based on the findings presented in Figure 20, it becomes evident that the performance of each supervised classifier is significantly impacted by the abrupt cessation of the dark market, which occurred subsequent to time step 43. This decline, specifically in terms of the volume of illicit transactions, has already been duly acknowledged in Chapter 4.2.2.

7.2 Active Learning

This section presents and discusses the results of the various approaches tested for AL with AD. The outcomes from these experiments will be consolidated in a table, which will include both the unsupervised and supervised AL policies that were employed. For consistency with Lorenz et al. (2020) work, the unsupervised AL policies will be referred to as AL warmup learners, while the supervised ones will be referred to as AL hot learners. Additionally, the table will also provide information about the classifier that is going to be trained on the available labeled data, for each setup. As an example, the results will be presented for labeled pool sizes of 200, 500, 1000, 1500, and 3000 transactions. Meaning that the maximum F1-score will be presented for the supervised classifiers when trained on 200, 500, 100, 1500, and 3000 labeled transactions. Additionally, each classifier used in the AL setups will be compared against its baseline, i.e., the performance they achieve when trained on the entire train set.

7.2.1 Scenario 1 - Unsupervised Active Learning Policies

First, there will be presented the results of Scenario 1, thereby, not switching from an unsupervised policy to a more sophisticated supervised policy, at any point in the AL process. As a baseline, there will also be shown the results of using the Random Sampling algorithm as a warmup policy. The results are shown in Table 2, Table 3, and Table 4, regarding each one of the supervised classifiers: RF, XGBoost, and LR, respectively.

Table 2 shows the performance of the AL setups executed for the RF classifier.

Table 2: Active Learning Setups for Random Forest

Classifier	**Warmup Learner**	**Hot Learner**	**Labeled Pool Size**					**Baseline F1-score**
			200 (0.7%)	**500 (1.7%)**	**1000 (3.3%)**	**1500 (5%)**	**3000 (10%)**	
RF	LOF	-	0.76	0.77	0.77	0.78	0.79	0.83
	Elliptic Envelope	-	0.49	0.53	0.57	0.55	0.54	
	IF	-	-	-	-	-	0.04	
	Random Sampling	-	0.77	0.77	0.79	0.79	0.8	

Table 3 shows the performance of the AL setups executed for the XGBoost classifier.

Table 3: Active Learning Setups for Extreme Gradient Boosting

Classifier	**Warmup Learner**	**Hot Learner**	**Labeled Pool Size**					**Baseline F1-score**
			200 (0.7%)	**500 (1.7%)**	**1000 (3.3%)**	**1500 (5%)**	**3000 (10%)**	
XGBoost	LOF	-	0.75	0.75	0.76	0.76	0.78	0.82
	Elliptic Envelope	-	0.62	0.68	0.7	0.68	0.65	
	IF	-	-	-	-	-	0.17	
	Random Sampling	-	0.75	0.75	0.78	0.79	0.8	

Table 4 shows the performance of the AL setups executed for the LR classifier.

Table 4: Active Learning Setups for Logistic Regression

Classifier	Warmup Learner	Hot Learner	Labeled Pool Size					Baseline F1-score
			200 (0.7%)	500 (1.7%)	1000 (3.3%)	1500 (5%)	3000 (10%)	
LR	LOF	-	0.43	0.28	0.29	0.34	0.36	0.52
	Elliptic Envelope	-	0.19	0.25	0.29	0.3	0.31	
	IF	-	-	-	-	-	0.14	
	Random Sampling	-	0.41	0.41	0.44	0.47	0.45	

Based on the available information, it can be observed from the provided tables that, regardless of the classifiers used, employing a Random Sampling algorithm as a warm-up learner yields comparable or even superior performance compared to more sophisticated AL setups that utilize unsupervised AD algorithms. These results are unexpected since it would be anticipated that AD algorithms, despite their susceptibility to FP, would identify more meaningful fraud-related patterns than a completely random algorithm, thereby exerting a greater influence on the overall performance of the supervised classifiers. However, what was observed is exactly the opposite. It is even possible to understand that the transaction queries by the Random Sampling algorithm provoke a faster improvement in the overall performance of the three supervised ML models.

It's also important to note that still, none of the setups tested, were able to reach the supervised baseline of each supervised classifier, thereby, suggesting, as it was stated in the literature, that a possible transition to a supervised learner at some point in the AL process would be potentially beneficial.

Specifically, IF yielded unsatisfactory results across all classifiers, with the first illicit transaction only being detected in the 31st iteration. This finding contradicts the prevailing literature, which strongly suggested IF as the most effective AD method. In the literature review on AL pertaining to this topic, IF was employed as an unsupervised policy by most of the studies. However, based on these observations, IF emerged as the poorest-performing AD method by a significant margin. On the other hand, LOF was the AD algorithm that managed

to identify more meaningful transactions, but it still failed to outperform the performance achieved by the Random Sampling algorithm.

It is important to note that the overall poor performance of AD algorithms can be attributed to their limitations in effectively detecting illicit transactions based on the specific set of features that were used. The anomalies flagged by these algorithms are often FP and are not indicative of fraudulent cryptocurrency activity. In simple words, transactions identified as deviating significantly from the expected behavior or normal patterns, as determined by the AD algorithms, do not necessarily turn out to be fraudulent. This could potentially be justified, due to the intrinsic characteristics of the cryptocurrency market.

However, it is essential to acknowledge that the conclusions drawn in this study may vary when considering different subsets of features. As mentioned in Chapter 6.2.2, exploratory research was conducted in order to reduce the dataset's dimensionality and train AD algorithms using various subsets of features. During this process, certain features were identified as highly redundant, and efforts were made to determine the most significant ones. Unfortunately, this approach did not yield significant results, and will not be discussed here. Therefore, further research should undertake an in-depth investigation into the influence of different features on the performance of AD algorithms.

One possible explanation that could justify the comparable results obtained from using the Random Sampling algorithm is that the classifier would start approaching relatively good performance when the labeled pool includes a sufficient number of illicit transactions. Given that the training set consists of approximately 10% illicit cases, Random Sampling can quickly reach the desired number of illicit transactions. Since there are 50 transactions sampled at each iteration, on average, there will be sampled five illicit transactions per iteration with Random Sampling.

However, as previously stated in the Theoretical Framework chapter, instances of cryptocurrency crime are exceedingly rare among licit ones. To account for this, the same setups were tested with a more realistic class imbalance. Therefore, a random undersampling technique was employed on the minority class to achieve an illicit rate of 1%.

A comprehensive overview of the outcomes obtained from this experiment can be found in Table 5. Note that the baseline score of each supervised classifier was also updated based on the new illicit rate of 1%.

Table 5: Active Learning Setups at 1% Illicit Rate

Classifier	Warmup Learner	Hot Learner	Labeled Pool Size					Baseline F1-score
			200 (0.8%)	500 (1.8%)	1000 (3.7%)	1500 (5,6%)	3000 (11%)	
RF	LOF	-	0.15	0.32	0.36	0.47	0.35	0.79
	Random Sampling	-	0.32	0.54	0.55	0.65	0.73	
XGBoost	LOF	-	0.21	0.36	0.39	0.4	0.3	0.79
	Random Sampling	-	0.36	0.44	0.48	0.51	0.66	
LR	LOF	-	0.2	0.21	0.21	0.21	0.22	0.55
	Random Sampling	-	0.26	0.3	0.27	0.28	0.43	

The results presented in Table 5 focus solely on LOF and Random Sampling, as neither the Elliptic Envelope nor the IF algorithms were able to detect any illicit transactions within the unlabeled pool throughout the entire AL process. From what can be observed, it becomes evident that, for this specific set of features, even at lower illicit rates, Random Sampling still surprisingly outperforms AD algorithms. These findings underscore even more the subpar performance of AD algorithms in detecting illicit transactions and the limited feasibility of utilizing them to uncover meaningful fraudulent patterns within the unlabeled pool, that can be used to improve the overall performance of the supervised classifiers. Therefore, it can be concluded that, for this specific set of features, using a Random Sampling algorithm would cause a greater impact on the performance of the supervised classifiers in comparison to the use of AD algorithms.

Therefore, it has been demonstrated that the utilization of a Random Sampling algorithm offers a more efficient resolution to the challenges posed by cold start scenarios when contrasted with AD algorithms. These further highlight previous considerations, as the AD algorithms proved ineffective in accurately identifying fraudulent transactions based on this specific set of features. Even the most anomalous transactional patterns did not correspond to fraudulent activity.

The fact that, overall, neither of the AD algorithms finds relevant fraudulent transactions in the unlabeled pool that can benefit the overall model's performance, can be explained, and visualized in Figure 21.

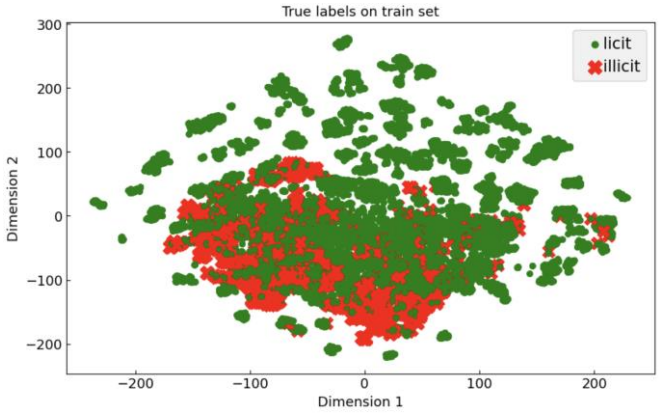

Figure 21: t-SNE Projection on the Train Set, Colored by the True Labels

From Figure 21, it becomes evident that, for this set of features, the illicit transactions within the training set are not merely outliers or deviant patterns. This poses a significant challenge for AD algorithms in identifying fraudulent transactions, as illicit transactions exhibit similar patterns to regular transactions.

7.2.2 Scenario 2 - Combining Unsupervised and Supervised Active Learning Policies

In this section, there will be presented the results of Scenario 2, thereby, combining both unsupervised and supervised policies. Therefore, all the setups will start with an unsupervised AL policy that queries the most anomalous transactions at each iteration, until a certain cut-off point is reached. After that, a more sophisticated supervised AL policy will be employed to select the most informative instances. This transition from an unsupervised to a supervised query strategy was tested for all the possible thresholds (iterations). However, there will only be shown the results obtained for the 30th and 1st iteration cut-off points, as they are representative of the overall behavior of the AL setups across all thresholds. This means that in this section there will be presented and discussed the results for the AL setups that use unsupervised policies (AD algorithms) until the AL setup completes 30 or 1 iteration(s), respectively. After that, a supervised query policy (QBC, EMC, or Uncertainty Sampling) is applied. It's worth noting that since the batch size for each iteration is set to 50 transactions,

this means that by the 30th and 1st iterations, a total of 1500 and 50 labeled transactions will have been collected by using unsupervised policies, respectively.

7.2.2.1 Cut-off point: 30th iteration

Firstly, the obtained results from employing unsupervised AL policies up to the 30th iteration (until the labeled pool contains 1500 transactions) are presented. The results are illustrated in Table 6, Table 7 and, Table 8, regarding each one of the supervised classifiers: RF, XGBoost, and LR.

Table 6 depicts the results from the AL setups executed for the RF classifier.

Table 6: Active Learning Setups for Random Forest (Cut-off - 30th Iteration)

Classifier	**Warmup Learner**	**Hot Learner**	**Labeled Pool Size**					**Baseline F1-score**
			200 (0.7%)	**500 (1.7%)**	**1000 (3.3%)**	**1500 (5%)**	**3000 (10%)**	
RF	LOF	Uncertainty Sampling	0.76	0.77	0.77	0.78	0.83	0.83
		QBC	0.76	0.77	0.77	0.78	0.81	
		EMC	0.76	0.77	0.77	0.78	0.82	
	Elliptic Envelope	Uncertainty Sampling	0.49	0.53	0.57	0.55	0.83	
		QBC	0.49	0.53	0.57	0.55	0.82	
		EMC	0.49	0.53	0.57	0.55	0.74	
	IF	Uncertainty Sampling	-	-	-	-	0.83	
		QBC	-	-	-	-	0.75	
		EMC	-	-	-	-	0.82	

The results presented in Table 6 demonstrate a significant improvement in the performance of the RF classifier when transitioning to a supervised AL policy. By comparing the classifier's performance with 1500 labeled transactions to that with 3000 labeled transactions, it becomes evident how quickly the F1-score improves. In most cases, the classifier achieved a near-optimal solution, matching the baseline F1-score, with as few as 3000 labeled

transactions. This accounts for only 10% of the original labels in the training set. These findings highlight the superior performance of supervised policies in comparison to unsupervised ones. Moreover, these results even suggest that by lowering the cut-off point to switch from unsupervised policies earlier in the AL process, it is likely to benefit the overall performance of the classifier. This adjustment would lead to a potentially faster increase in its performance. These findings also underscore even more the limitations of AD algorithms in identifying relevant illicit transactions within the unlabeled pool, particularly when contrasted with policies that rely on supervised classifiers. As a result, these findings suggest that there is no need to prolong the use of unsupervised AD policies throughout the AL setup.

Furthermore, the optimal performance of these supervised AL policies can also be inferred by examining the results obtained from setups utilizing IF as a warmup learner. In this context, IF detects an illicit transaction only at the 31st iteration. However, remarkably, even after applying the supervised AL policies to the remaining 29 iterations, it is still possible to attain the supervised baseline for the setup that employs IF and Uncertainty Sampling.

Overall, these findings emphasize the effectiveness of supervised AL policies and the potential for optimizing the classifier's performance by adjusting the threshold for transitioning from unsupervised policies earlier in the AL process.

The best-performing setups utilize LOF and Uncertainty Sampling, as well as Elliptic Envelope and Uncertainty Sampling or IF and Uncertainty Sampling. Figure 22 illustrates the progression of the F1-score across various labeled pool sizes, showcasing the evolution of the results.

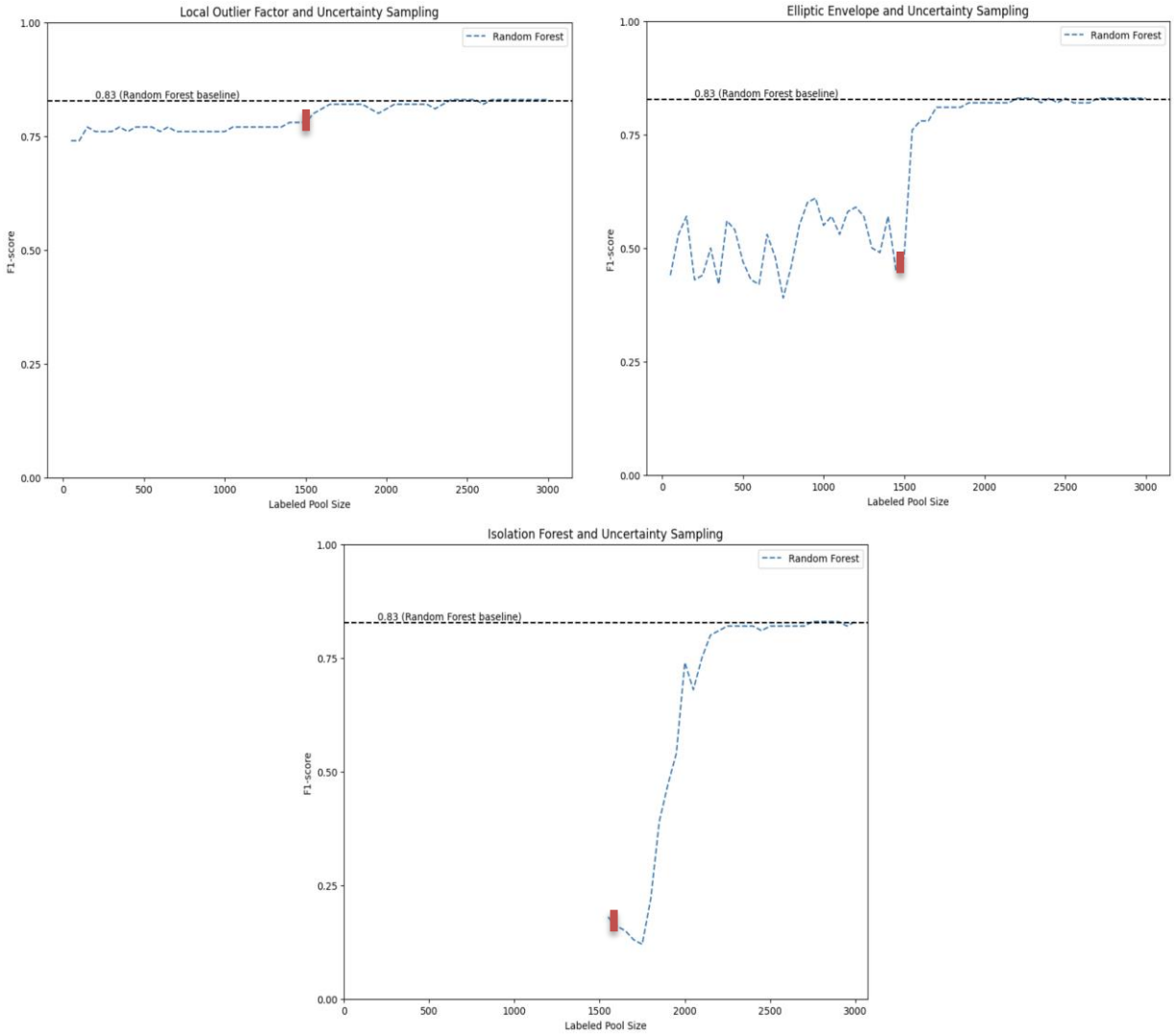

Figure 22: Learning curves for Random Forest Setups (Cut-off - 30th Iteration)

By analyzing Figure 22, it becomes evident that the RF classifier exhibits accelerated improvement when an AL policy, such as Uncertainty Sampling, is applied. These plots visually confirm the conclusion that once the cut-off point is reached and a supervised AL policy is implemented, there is a substantial enhancement in the model's performance. Remarkably, it is possible to observe that the baseline F1-score can be achieved with as few as 2000 labels, indicating the efficiency of these approaches.

Table 7 illustrates the results from the AL setups executed for the XGBoost classifier.

Table 7: Active Learning Setups for Extreme Gradient Boosting (Cut-off - 30th Iteration)

Classifier	Warmup learner	Hot learner	Labeled Pool Size					Baseline F1-score
			200 (0.7%)	500 (1.7%)	1000 (3.3%)	1500 (5%)	3000 (10%)	
XGBoost	LOF	Uncertainty Sampling	0.75	0.75	0.76	0.76	0.82	0.82
		QBC	0.75	0.75	0.76	0.76	0.81	
		EMC	0.75	0.75	0.76	0.76	0.82	
	Elliptic Envelope	Uncertainty Sampling	0.62	0.68	0.7	0.68	0.82	
		QBC	0.62	0.68	0.7	0.68	0.82	
		EMC	0.62	0.68	0.7	0.68	0.77	
	IF	Uncertainty Sampling	-	-	-	-	0.82	
		QBC	-	-	-	-	0.71	
		EMC	-	-	-	-	0.67	

Based on the findings presented in Table 7, it is evident that the XGBoost classifier exhibits a similar trend to the RF classifier. When a supervised AL policy is implemented in the 31st iteration, the performance of the XGBoost classifier shows a rapid enhancement, with the F1-score baseline being achieved using just 3000 labels.

Figure 23 illustrates the evolution of the F1-score across different labeled pool sizes for the best-performing AL setups that utilized XGBoost as a classifier.

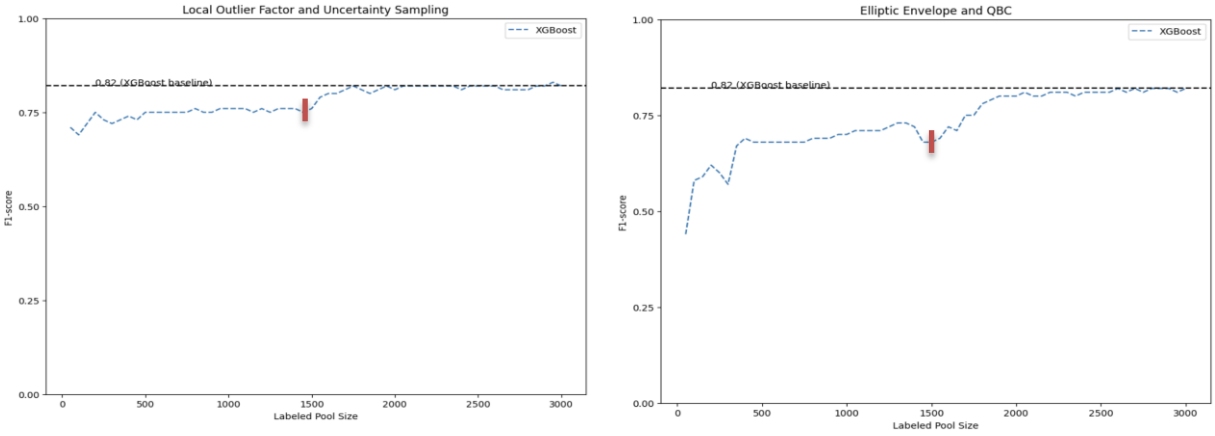

Figure 23: Learning curves for Extreme Gradient Boosting Setups (Cut-off - 30th Iteration)

Similarly, to what was observed in Figure 22, Figure 23 also visually highlights the accelerated improvement when a supervised AL policy is applied. From what can be visualized the baseline F1-score from XGBoost can be achieved with as few as 2000 labels.

Table 8 presents the results from the AL setups executed for the LR classifier.

Table 8: Active Learning Setups for Logistic Regression (Cut-off - 30th Iteration)

Classifier	**Warmup Learner**	**Hot Learner**	**Labeled Pool Size**					**Baseline F1-score**
			200 (0.7%)	**500 (1.7%)**	**1000 (3.3%)**	**1500 (5%)**	**3000 (10%)**	
LR	LOF	Uncertainty Sampling	0.43	0.28	0.29	0.34	0.45	0.52
		QBC	0.43	0.28	0.29	0.34	0.40	
		EMC	0.43	0.28	0.29	0.34	0.58	
	Elliptic Envelope	Uncertainty Sampling	0.19	0.25	0.29	0.3	0.48	
		QBC	0.19	0.25	0.29	0.3	0.56	
		EMC	0.19	0.25	0.29	0.3	0.44	
	IF	Uncertainty Sampling	-	-	-	-	0.53	
		QBC	-	-	-	-	0.37	
		EMC	-	-	-	-	0.44	

Based on the results presented in Table 8, the application of a supervised AL policy also led to a significant enhancement in the performance of the LR classifier. In specific setups that used LOF and EMC, Elliptic Envelope and QBC, or IF and Uncertainty Sampling, this improvement resulted in an F1-score that even surpassed the baseline. Remarkably, achieving this level of performance required only 3000 labeled data points, which corresponded to a mere 10% of the original labels.

Figure 24 depicts the evolution of the LR classifier performance over the different labeled pool sizes for the best-performing setups.

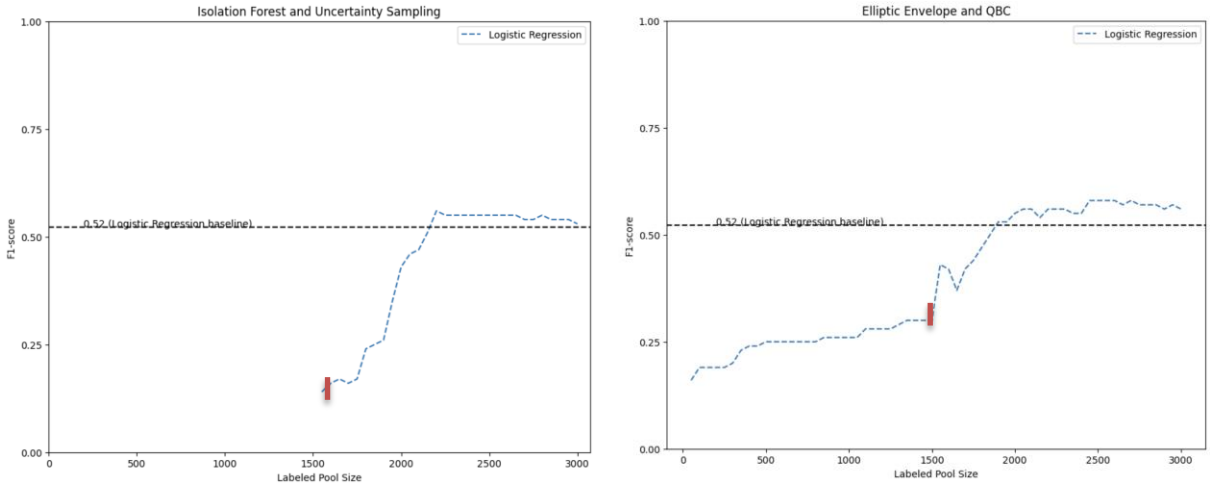

Figure 24: Learning Curves for Logistic Regression Setups (Cut-off - 30th Iteration)

One valuable observation that can be taken from Figure 24 is that the LR classifier exhibit improved performance when trained on a subset of labeled data, but eventually, tend to converge to its supervised baseline. This finding suggests that the labeled pool primarily consists of the most relevant samples in the earlier stages of the AL process, and at the same time, the class imbalance increases. Laws et al. (2008), recognized that in specific scenarios, stopping the AL process early could prevent potential degradation of the model. These findings align with the conclusions drawn in the study conducted by Lorenz et al. (2020).

7.2.2.2 Cut-off Point: 1st Iteration

In this section, there are presented the results of the AL setups that were tested considering the usage of unsupervised AL policies exclusively for the initial iteration or until the labeled pool included at least one illicit transaction (i.e., it is necessary to have at least one illicit transaction in the labeled pool to train a supervised classifier). This means, that the unsupervised AL policies will only be used for the minimum time possible throughout the AL process. The results can be found in Table 9, Table 10, and Table 11, regarding each one of the supervised classifiers.

Table 9 illustrates the results from the AL setups executed for the RF classifier.

Table 9: Active Learning Setups for Random Forest (Cut-off – 1st Iteration)

Classifier	Warmup Learner	Hot Learner	Labeled Pool Size					Baseline F1-score
			200 (0.7%)	500 (1.7%)	1000 (3.3%)	1500 (5%)	3000 (10%)	
RF	LOF	Uncertainty Sampling	0.77	0.8	0.82	0.83	0.83	0.83
		QBC	0.77	0.8	0.8	0.82	0.82	
		EMC	0.78	0.79	0.82	0.81	0.83	
	Elliptic Envelope	Uncertainty Sampling	0.67	0.8	0.82	0.83	0.83	
		QBC	0.6	0.62	0.79	0.81	0.82	
		EMC	0.52	0.56	0.81	0.82	0.83	
	IF	Uncertainty Sampling	-	-	-	-	0.83	
		QBC	-	-	-	-	0.75	
		EMC	-	-	-	-	0.82	

Table 9 illustrates the improvement in performance achieved by utilizing only AD algorithms as warm-up learners for a single iteration throughout the AL process. In comparison to extending the use of AD algorithms for a larger number of iterations, as shown in Table 2 or Table 6, it becomes evident that the supervised baseline of RF, in this case, can be achieved faster, thereby, with minimal labeling. In fact, a near-optimal solution is reached with as few as 1000 labeled transactions, which corresponds to only 3.3% of the original labels in the training set. This earlier transition from an unsupervised warm-up learner to a supervised hot learner leads to improved performance of the RF classifier. This emphasizes, even more, the limited effectiveness of AD algorithms in identifying relevant fraud-related patterns for being queried, while demonstrating the optimal performance of supervised AL policies. This observation aligns with previous findings, as the AD algorithms, given the specific set of features, are unable to accurately identify fraudulent transactions within the unlabeled pool, as these transactions are not, in fact, anomalous. Therefore, there is no need to extend the use of these techniques throughout the AL process. Once a supervised AL policy is employed, the impact on the performance of the supervised classifier becomes more pronounced.

The findings presented in Table 9 underscore the practical significance of utilizing AL techniques as a cohesive framework. Within this specific context, it becomes apparent that AL allows for attaining performance levels similar to those of a supervised baseline while utilizing a mere 3% of the original labels from the training set. Consequently, the necessity to label an entire dataset for training a supervised classifier is greatly reduced. These results clearly demonstrate that optimal solutions can be achieved by training a supervised classifier exclusively with the most informative transactions obtained through the AL process.

The best-performing setups use LOF and Uncertainty Sampling, and Elliptic Envelope and Uncertainty Sampling. Figure 25 presents the evolution of the RF classifier F1-score over the different labeled pool sizes.

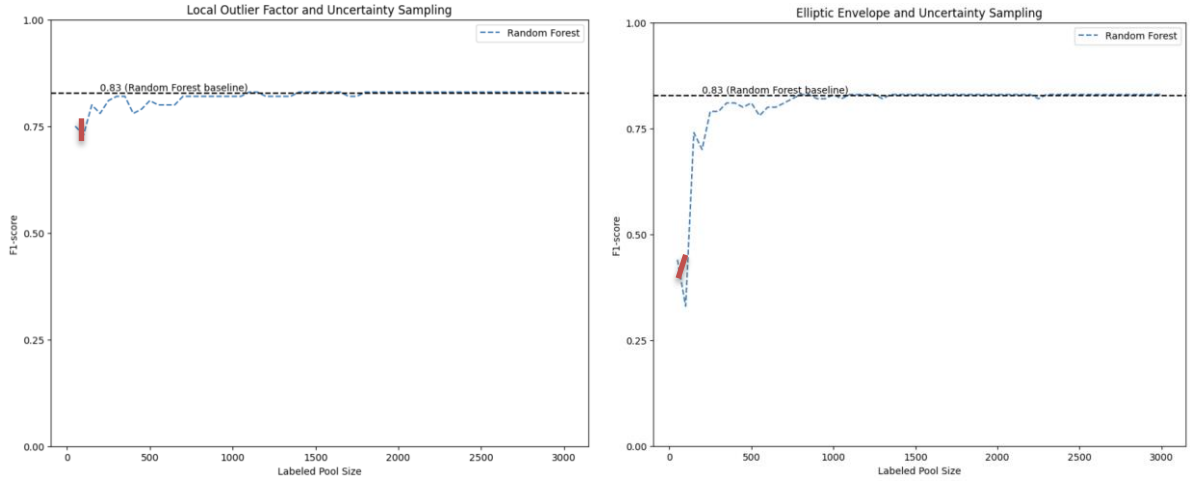

Figure 25: Learning curves for Random Forest Setups (Cut-off – 1st Iteration)

By comparing Figure 25 to Figure 22, it becomes evident that a swift transition to a supervised AL policy results in a significantly accelerated improvement in the overall performance of the RF classifier. In Figure 22, unsupervised AL policies were employed until the 30th iteration, whereas in Figure 25, a faster transition to a supervised AL policy is observed. As it can be observed, this accelerated transition enables the baseline performance of the RF classifier to be reached faster, requiring only 700 to 800 labels.

Table 10 illustrates the results from the AL setups execution for the XGBoost classifier.

Table 10: Active Learning Setups for Extreme Gradient Boosting (Cut-off – 1st Iteration)

Classifier	Warmup Learner	Hot Learner	Labeled Pool Size					Baseline F1-score
			200 (0.7%)	500 (1.7%)	1000 (3.3%)	1500 (5%)	3000 (10%)	
XGBoost	LOF	Uncertainty Sampling	0.63	0.79	0.82	0.81	0.82	0.82
		QBC	0.74	0.76	0.76	0.79	0.82	
		EMC	0.73	0.77	0.81	0.82	0.82	
	Elliptic Envelope	Uncertainty Sampling	0.64	0.81	0.81	0.8	0.82	
		QBC	0.54	0.69	0.72	0.74	0.81	
		EMC	0.62	0.74	0.8	0.8	0.82	
	IF	Uncertainty Sampling	-	-	-	-	0.82	
		QBC	-	-	-	-	0.71	
		EMC	-	-	-	-	0.67	

Based on the results presented in Table 10, it becomes evident that the F1-score baseline of the XGBoost classifier is achieved more quickly for the majority of the setups compared to those that continued utilizing an unsupervised AL policy until the 30th iteration. However, it should be noted that when the labeled pool size is limited to only 200 labeled transactions, certain setups that extended the use of unsupervised AL policies (up to the 30th iteration) demonstrated superior performance for that specific pool size (consult Table 7). This could be justified by the fact that supervised policies after being implemented provoke higher variability in the model performance before significantly improving and reaching its respective baseline.

The best-performing setups use LOF and Uncertainty Sampling, and Elliptic Envelope and Uncertainty Sampling. Figure 26 depicts the evolution of the XGBoost classifier F1-score over the different labeled pool sizes.

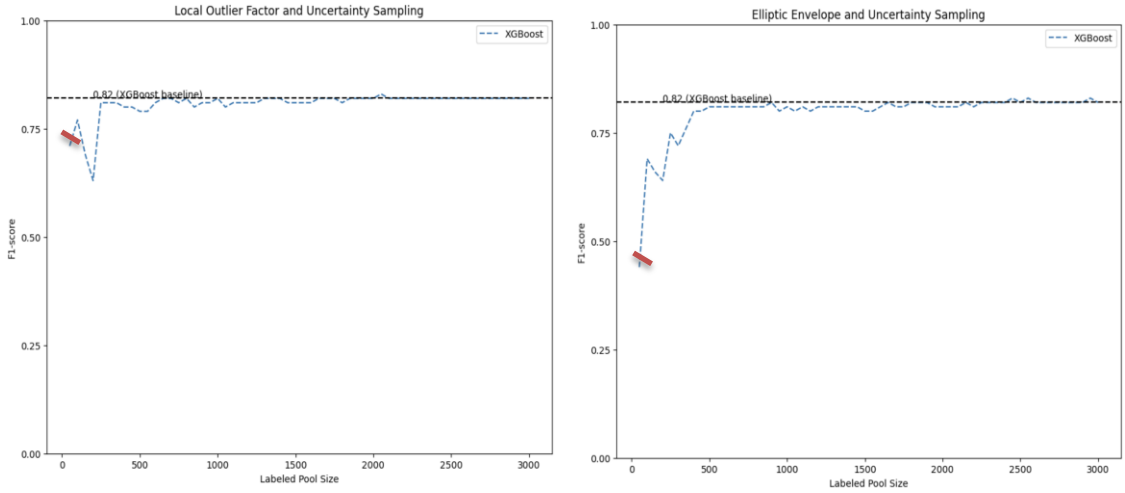

Figure 26: Learning Curves for Extreme Gradient Boosting Setups (Cut-off – 1st Iteration)

Table 11 illustrates the results from the AL setups executed for the LR classifier.

Table 11: Active Learning Setups for Logistic Regression (Cut-off – 1st Iteration)

Classifier	Warmup Learner	Hot Learner	Labeled Pool Size					Baseline F1-score
			200 (0.7%)	500 (1.7%)	1000 (3.3%)	1500 (5%)	3000 (10%)	
LR	LOF	Uncertainty Sampling	0.56	0.61	0.6	0.57	0.58	0.52
		QBC	0.43	0.47	0.55	0.54	0.49	
		EMC	0.52	0.63	0.67	0.69	0.68	
	Elliptic Envelope	Uncertainty Sampling	0.31	0.5	0.57	0.56	0.62	
		QBC	0.24	0.3	0.34	0.37	0.52	
		EMC	0.22	0.27	0.55	0.57	0.7	
	IF	Uncertainty Sampling	-	-	-	-	0.53	
		QBC	-	-	-	-	0.37	
		EMC	-	-	-	-	0.44	

Based on the findings presented in Table 11, several interesting considerations can be made. It is possible to infer, that the performance of the LR classifier was also significantly affected when a supervised AL policy was applied immediately after the labeled pool contained one illicit transaction. By minimizing the use of unsupervised AL policy throughout the AL setups, some setups were even able to achieve or even surpass the performance of the supervised

baseline using as few as 200 labels, which accounts for only 0.7% of the training set. Notably, for certain setups, LR even outperformed its baseline by 0.15 considering the entire AL process. Consequently, it can be concluded that this particular structure greatly enhanced the LR classifier's performance. Note that, even though, LR was able to surpass its own baseline F1-score, it was not able to surpass the best baseline F1-score, which was achieved by RF.

In certain setups, particularly those starting with LOF, it is worth noting that the LR model tends, again, to exhibit improved performance when trained on a sample of the labeled data, but eventually tends to converge to its supervised baseline. This observation further emphasizes the need for implementing early stopping in the AL process.

The best-performing setups used LOF and EMC, as well as Elliptic Envelope and EMC. Figure 27 provides a visual representation of the LR F1-score's progression across the different labeled pool sizes, showcasing its evolution.

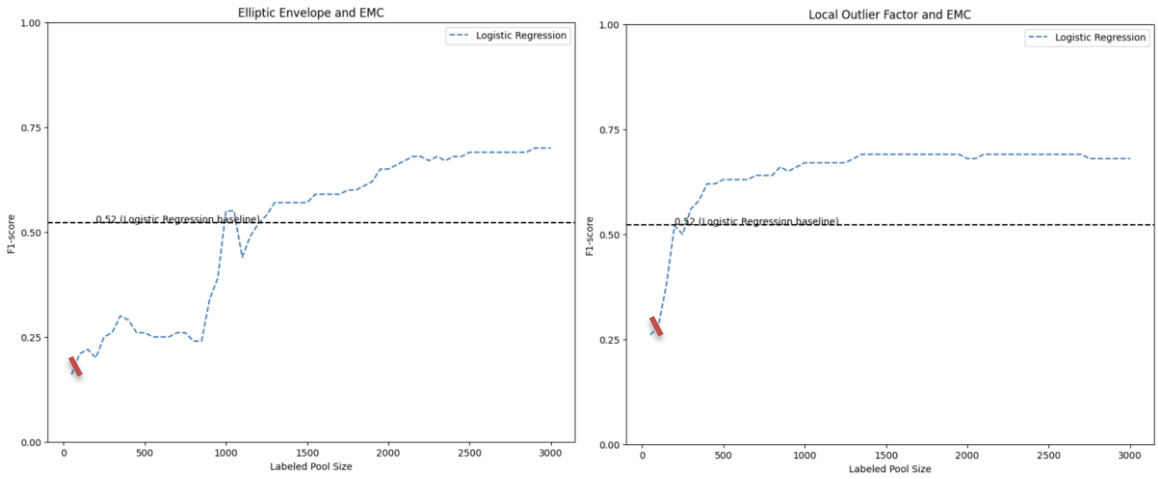

Figure 27: Learning Curves for Logistic Regression Setups (Cut-off – 1st Iteration)

Based on Figure 27, it is evident that the previous considerations can be visually validated, especially for the setup that combines LOF and EMC, where the sudden improvement is more notorious. The enhanced performance of the LR classifier is clearly observed after the optimized supervised AL policy is implemented. These approaches exhibit superior performance compared to the LR baseline.

7.3 Results Analysis and Key Findings

All these findings show the subpar performance of AD algorithms in identifying relevant cryptocurrency fraud-related patterns within the unlabeled pool. The findings even suggest

that using a totally random algorithm would give similar or superior results compared to AD algorithms, for different illicit rate levels. Since AD methods, for this specific set of features, fail to correctly detect and identify illicit instances, they are highly ineffective at querying illicit transactions to be added to the labeled pool and improving the performance of a supervised classifier quickly. Hence, it has been demonstrated that incorporating AD algorithms in AL configurations fails to yield favorable outcomes in terms of the model's overall performance.

Although the AL literature pertaining to this topic has already indicated that employing AD algorithms throughout the entire AL setup may not result in optimal performance for a supervised classifier, it would still be anticipated that utilizing AD algorithms would surpass the use of Random Sampling. These findings highlight the main limitations of AD algorithms, that often struggle when faced with the complexity and evolving nature of the illicit activities. Therefore, these types of methods, generally, have inherent limitations in scenarios where the boundary between normal and illicit behavior is not well-defined. As it was proven, for these specific set of features, in the Elliptic Dataset, there is no clear distinction between normal and illicit behavior, they have similar patterns. In other words, evidence that anomalies/outliers in the feature space are not indicative of fraudulent behavior was presented. Therefore, the fact that illicit transactions can exhibit various patterns and characteristics that overlap with normal transactions, makes it challenging for AD algorithms to accurately distinguish between the two.

Although the insights from the majority of the AD literature review indicate that the AD methods are often able to detect relevant illicit transactions and even previous unknown patterns, the results of this research did not align with those expectations. These findings suggest that studies conducted on synthetic datasets where fraudulent transactions were intentionally designed to be anomalous, may lead to unreliable results. Therefore, this study highlights the necessity and importance of conducting research on real-world datasets to obtain applicable results. It is noteworthy that the findings from this research are consistent with the study conducted by Lorenz et al. (2020).

Moreover, the fact that this is a Bitcoin transactional dataset may influence and justify the results that were obtained. In other words, the bad performance of AD algorithms can be related to the fact that, in the cryptocurrency context, the most deviant transactions are not in fact, fraudulent ones. The cryptocurrency market is highly volatile, with prices fluctuating rapidly. This volatility can lead to legitimate transactions that appear anomalous from a static

perspective. For instance, large transactions made during periods of high market volatility might seem suspicious, but they could be driven by genuine investment or trading activities. In the cryptocurrency world, large transactions are often a normal part of trading activities or investments. Investors might move significant amounts of funds between different exchanges or wallets as part of their trading strategies. These activities, while they might stand out as anomalous based on traditional financial transaction patterns, are not necessarily indicative of fraudulent behavior. They are more reflective of the unique dynamics and emerging usage patterns within the cryptocurrency ecosystem.

Additionally, the lack of anomalous or deviant patterns in Bitcoin transactions associated with illicit activities could also be attributed to fraudsters intentionally imitating normal transaction behavior. This deliberate mimicry poses a significant challenge for anti-fraud algorithms, as it becomes increasingly difficult to accurately identify illicit transactions within the Bitcoin cryptocurrency network.

Hence, it is imperative to conduct further investigations in this realm to validate these findings and gain a more profound understanding of their applicability to other real-world cryptocurrency transaction datasets. It is crucial to establish a parallel between the results obtained from this dataset and those from diverse real-life cryptocurrency datasets, which may exhibit different class distributions and illicit activity rates.

Based on the unsatisfactory performance observed in the unsupervised AL policies, it has been demonstrated that transitioning to supervised AL policies as quickly as possible can significantly enhance the performance of the supervised classifier. This transition allows for the attainment of optimal solutions with as little as 3% of the original training set. In essence, these findings suggest that by adopting a supervised AL policy as soon as the initial iteration of the AL setup is completed, or when at least one illicit transaction is identified and labeled, the performance of the supervised classifier can be markedly improved. These findings emphasize that AD methods are only effective in addressing the cold start scenario, where there are no initial labels available for the AL process to start with. However, from previous considerations, these methods were proven to perform similarly or even worse than the Random Sampling algorithm in resolving this problem. This dispels the notion of their feasibility. This means that, despite not requiring the analysis of any underlying pattern in the data, the transactions sampled by the Random Sampling algorithm have a more pronounced

impact on the performance of a supervised classifier compared to the transactions queried by the AD algorithms.

Even so, the prolonged use of unsupervised AD algorithms in the AL process has been found to be ineffective, the findings obtained from this study also suggest the practical relevance of using AL techniques as a way to gather the minimum number of labeled instances necessary to achieve a performance close to the supervised baseline, thereby, reducing the time and effort required for labeling a large-scale dataset to train a high performance supervised classifier. Specifically, the use of AL as a cohesive framework was proven to be very effective at selecting a subset of the data for annotation that maximized the model's learning potential.

Even so, neither of the AD algorithms performed adequately, it is possible to conclude, that, still, LOF was the only unsupervised algorithm capable of identifying more relevant illicit transactions within the unlabeled pool, at different illicit rates. This proves that, in fact, LOF works better than the remaining traditional AD methods at identifying outliers in high-dimensional datasets.

Additionally, it is also possible to conclude that among the supervised AL policies that were employed, there is no clear best policy. These supervised AL policies performed very adequately for most of the tested setups.

8. Application of The Method to Uphold's Proprietary Dataset

The preceding chapter shed light on the imperative need for conducting research on real-world datasets to obtain applicable results. Specifically, the findings indicated that for the specific set of features utilized in the Elliptic Dataset, AD methods exhibited poor performance in detecting relevant fraud-related patterns within the unlabeled pool. Consequently, the potential to enhance the performance of supervised classifiers using such patterns remained limited. Additionally, the results revealed that even a Random Sampling algorithm outperformed AD methods in terms of performance. Therefore, it is crucial to validate these outcomes on other real-life datasets, such as the one obtained from Uphold, where the features are not anonymized. This dataset will potentially facilitate the selection and extraction of relevant features.

Moreover, the elimination of a considerable volume of transactions within the Elliptic Dataset led to a notable decrease in the overall dataset size. This reduction in data size could potentially have had adverse consequences on the effectiveness of the models, constraining their capacity to discern valuable insights and detect patterns within the data.

Given these considerations, the examination and analysis of the proposed methodology using Uphold's proprietary dataset hold significant importance and deserve thorough exploration and discussion. This investigation will help determine if the implementation of these techniques can bolster Uphold's capacity to identify fraudulent activities.

It is important to note that the dataset selection and pre-processing were conducted collaboratively as a team at Uphold. Additionally, there was another student on the team who worked on a fraud-related use case. Given that the supervisors were the same, there was some overlap in the materials used and the exploration of those materials.

8.1 Dataset

Uphold is a leading multi-asset digital money platform that caters to a global market, providing financial services to individuals across over 184 countries. With support for over 200 currencies, both traditional and digital, as well as commodities, Uphold offers seamless foreign exchange and cross-border remittance solutions to its members worldwide. One of Uphold's standout features is its innovative 'Anything-to-Anything' trading experience, which

allows customers to trade directly between different asset classes. Uphold's platform is built upon a foundation of proprietary technologies and e-money apps, ensuring robustness, security, and efficiency in every transaction. By leveraging these cutting-edge tools, Uphold aims to redefine the way individuals interact with and manage their financial assets in a rapidly evolving digital landscape.

Uphold's proprietary dataset is a comprehensive collection of 21,656,252 transactions, showcasing the transfer of funds among various users within the Uphold ecosystem. These transactions span from January 1, 2021, to January 1, 2023, encompassing exchanges involving diverse types of currencies. These currencies encompass both conventional ones like the Euro and Dollar, as well as cryptocurrencies such as Bitcoin, Ethereum, Dogecoin, and others. The dataset provided represents a collection of transactions between Uphold users, where each line corresponds to a single transaction.

Given these previous considerations, it is worth noting that this particular dataset differs from the Elliptic Dataset in a significant manner: it encompasses not only Bitcoin transactions but also a broader range of currencies.

8.1.1 Dataset Description

Table 12 depicts the main structure of Uphold’s dataset.

Table 12: Uphold's Dataset Constitution

Feature	Description	Data Type
origin_user	The user that sent the funds	categorical
destination_user	The user who received the funds	categorical
created_at	Timestamp of the transaction	categorical
origin_currency	Currency of the funds in the origin	categorical
destination_currency	Currency of the funds in the destination	categorical
amount_usd	Amount transacted in United States Dollars	float
application_id	ID of the application that triggered the transaction	categorical
resulted_in_sar	Boolean indicating if the transaction resulted in a Suspicious Activity Report (Anti Money Laundering)	Boolean

The “resulted_in_sar” feature is the target label of the dataset. It corresponds to an alert of a money laundering fraud. The remaining features of the dataset correspond to transactional features.

Table 13 illustrates the distribution of transactions by class label.

Table 13: Number of Transactions by Class

Label	Number of Transactions
Licit	21477866
Illicit	178385

As it can be observed, Uphold's dataset exhibits a significant class imbalance between Licit and Illicit transactions. Specifically, the dataset has an illicit rate of 0.82%, which represents a substantial difference compared to the Elliptic Dataset. However, it is worth noting that this dataset offers a more realistic representation of the actual fraud rate.

Nevertheless, it is important to acknowledge certain limitations regarding the identification of fraudulent behaviors through the money laundering label. These labels are constrained by predefined rules, which may restrict the scope of detecting all possible fraudulent activities accurately.

8.1.2 Pre-Processing

This section examines the important process of data preparation, focusing on the necessary adjustments made to ensure the dataset is appropriate for the specific problem at hand and can be smoothly integrated with the chosen ML models.

The main changes made to the dataset are described as follows:

- **Convert labels from boolean to integer**: During this step, the labels represented as Booleans were converted to integers. To be precise, the illicit class label was transformed from "True" to 1, while the licit label was converted from "False" to 0.
- **Convert “created_at” to a DateTime format**: The “created_at” field contained the timestamp information, represented as a string. Converting it to a DateTime format involved parsing the string and representing it in a standardized DateTime format. This

conversion enables easier manipulation of the timestamp data, such as extracting specific time components or performing time-based calculations.

- **Create a transaction ID**: Generating a transaction ID involved creating a unique identifier for each transaction in the dataset. This makes it easier to track, analyze, or join with other related data.
- **Convert IDs from string to integers**: The user IDs, application IDs, and origin and destination currencies, which were originally represented as strings, have been converted to integers. To ensure accurate correspondence between IDs in the source and destination systems, a mapping indexing system was implemented.

The obtained dataset after the pre-processing step is a clean, transformed, and well-prepared dataset that is ready for further analysis, modeling, and ML tasks.

8.1.3 Feature Engineering

Feature engineering emerges as a powerful technique, enabling the extraction of meaningful insights from raw data and creating new variables that can augment the predictive capabilities of FD models.

Within this chapter, a comprehensive exploration of the feature engineering process will be undertaken, focusing on the creation of multiple novel columns derived from domain knowledge. These new features have been crafted with the intention of potentially augmenting the detection of fraudulent transactions. The subsequent tables will elucidate the details of the features that were meticulously engineered, unveiling their inherent characteristics. By delving into these intricacies, the aim is to deepen the understanding of the underlying patterns and insights that can be harnessed to enhance the FD capabilities of the developed system.

The data set's characteristics have led to the classification of the features into three primary categories: transactional attributes, network features, and user behavior features. This categorization allows for a better understanding of the different types of features and for further assessment of their potential impact on the performance of ML algorithms in detecting relevant fraud-related patterns.

Each new feature is computed at the transactional level (each line of the dataset), exclusively considering the preceding transactions. This procedural approach ensures that for every

individual entry in the dataset, the feature computations solely account for the historical transactions associated with it. Notably, when undertaking calculations for features such as the in-degree of a user (origin or destination), exclusive consideration is given to the antecedent transactions in which the user was actively involved.

Table 14 depicts the description of the transactional features.

Table 14: Transactional Attributes Description

Category	Feature	Description	Data Type
Transactional Attributes	transaction_ID	ID of each transaction	integer
	origin_user	User who sent the funds	integer
	destination_user	User who received the funds	integer
	origin_urrency	Currency of the funds in the origin	integer
	destination_currency	Currency of the funds at the destination	integer
	amount_usd	The amount transacted in USD	float
	application_id	ID of the application that triggered the transaction	integer
	year	Year in what the transaction occurred	integer
	month	Month in what the transaction occurred	integer
	day	Day in what the transaction occurred	integer
	resulted_in_sar	Integer indicating if the transaction resulted in a Suspicious Activity Report (Anti Money-Laundering)	integer

Transactional attribute features refer to the intrinsically characteristics and properties of transactions in the cryptocurrency market. They include information such as transaction amounts, timestamps, and sender and recipient addresses. By incorporating transactional attribute features into ML models, it becomes possible to analyze the specific attributes of each transaction and identify suspicious patterns or behaviors.

Table 15 illustrates the description of the network features.

Table 15: Network Features Description

Category	Feature	Description	Data Type
Network Features	in_degree (Origin User)	Indicates how many times other users have sent transactions to the origin user	integer
	out_degree (Origin User)	Indicates how many times other users have received transactions from the origin user	integer
	in_degree (Destination User)	Indicates how many times other users have sent transactions to the destination user	integer
	out_degree (Destination User)	Indicates how many times other users have received transactions from the destination user	integer
	total_amount_sent (Origin User)	Refers to the sum of the amount of all transactions initiated by the origin user.	float
	total_amount_received (Origin User)	Refers to the sum of the amount of all transactions received by the origin user.	float
	total_amount_sent (Destination User)	Refers to the sum of the amount of all transactions initiated by the destination user.	float
	total_amount_received (Destination User)	Refers to the sum of the amount of all transactions received by the destination user.	float
	average_amount_sent (Origin User)	Refers to the average amount of each transaction initiated by the origin user	float
	average_amount_received (Origin User)	Refers to the average amount of each transaction received by the origin user	float
	average_amount_sent (Destination User)	Refers to the average amount of each transaction initiated by the destination user	float
	average_amount_received (Destination User)	Refers to the average amount of each transaction received by the destination user	float

Category	Feature	Description	Data Type
	total_transactions (Origin User)	Number of transactions where the origin user was either the sender or the recipient	integer
	total_transactions (Destination User)	Number of transactions where the destination user was either the sender or the recipient	integer
	ratio_sent_received (Origin User)	Balance between the origin user outgoing and incoming transactions	float
	ratio_sent_received (Destination User)	Balance between the destination user outgoing and incoming transactions	float

Network features focus on the interactions and relationships between different entities within the cryptocurrency network. These features capture the network structure, connectivity, and flow of transactions. By considering network features in ML models, it becomes possible to detect anomalies that emerge from unusual network behaviors. For instance, if a specific address is involved in a large number of transactions or exhibits an unusually high number of connections, it might indicate a fraudulent scheme such as money laundering.
Table 16 provides a description of the user behavior features.

Table 16: User Behavior Features Description

Category	Feature	Description	Data Type
User Behavior Features	days_since_last_transaction (Origin User)	Number of days elapsed since the last transaction made by the origin user	integer
	days_since_last_transaction (Destination User)	Number of days elapsed since the last transaction made by the destination user	integer
	days_since_first_transaction (Origin User)	Number of days elapsed since the first transaction made by the origin user	integer
	days_since_first_transaction (Destination User)	Number of days elapsed since the first transaction made by the destination user	integer

Category	Feature	Description	Data Type
	user_frequency (Origin User)	Frequency in which the origin user transacts. This corresponds to the ratio of the total number of transactions and the days since the first transaction	float
	user_frequency (Destination User)	Frequency in which the destination user transacts. This corresponds to the ratio of the total number of transactions and the days since the first transaction	float
	active_days (Origin User)	Number of days that the origin user was actively involved in transactions	int
	active_days (Destination User)	Number of days that the destination user was actively involved in transactions	int
	amount_difference_last _ transaction (Origin User)	Difference of amount from last and current transaction made by the origin user	float
	amount_difference_last _ transaction (Destination User)	Difference of amount from last and current transaction made by the destination user	float
	range_amount_sent (Origin User)	Difference of the maximum and minimum amount sent by the origin user	float
	range_amount_sent (Destination User)	Difference of the maximum and minimum amount sent by the destination user	float
	range_amount_received (Origin User)	Difference of the maximum and minimum amount received by the origin user	float
	range_amount_received (Destination User)	Difference of the maximum and minimum amount received by the destination user	float

User behavior features pertain to the actions and patterns exhibited by users participating in cryptocurrency transactions. These features encompass various factors, including transaction frequency, transaction volumes, or temporal patterns. By analyzing user behavior features using ML models, it becomes possible to identify deviations from normal behavior, which can signify potentially fraudulent activities.

8.2 Dataset Applicability Discussion

Before the application of the ML models, there are some issues that must be discussed.

The Uphold Dataset offers a unique perspective on the application of ML in the context of Anti Money Laundering. Its primary focus is on tackling money laundering tasks specifically, which sets it apart from the Elliptic Dataset which covers a broader range of fraudulent activities.

One notable advantage of the Uphold Dataset is its substantial scale compared to the Elliptic Dataset. Its larger size enables a more extensive collection of fraudulent data points, facilitating a more comprehensive analysis and potentially leading to more robust and accurate ML models. With a larger dataset, there is greater room for exploring underlying patterns and relationships, which can enhance the effectiveness of ML algorithms in detecting fraudulent activities.

However, it is important to consider the computational challenges that come with the increased size and complexity of the Uphold Dataset. Therefore, effectively applying the proposed methodology to this dataset can potentially require substantial processing computational capacity. The training and optimization of ML models on such a large and intricate dataset can be resource-intensive, often necessitating specialized infrastructure to handle the processing power requirements effectively.

Uphold's dataset being specifically focused on money laundering offers an additional advantage, particularly in terms of offering the possibility to transfer the use case from a transaction level to a user level. This means that, depending on the objective, it becomes feasible to classify instances of money laundering at the user level rather than just at the transaction level. This advantage stems from the inherent nature of money laundering, where both parties involved in a fraudulent transaction are often aware of the illegality of their activities and can be categorized as fraudsters. Therefore, both users involved in such transactions are generally, fraudulent ones.

However, it is important to note that the primary emphasis in the initial phase will still be on the transaction level (same as the Elliptic Dataset). Put simply, the primary goal will be to uncover and detect fraudulent crypto transactions within Uphold's ecosystem.

As previously stated in the Dataset Description chapter, this label is accompanied by certain limitations due to its reliance on predefined rules. These limitations can be problematic as they may categorize certain activities as fraudulent based solely on these predetermined criteria, without considering the broader context. Consequently, some behaviors that are flagged as potential fraud may not actually be fraudulent at all. This issue arises because after an alert is triggered, the case is often presented before a tribunal for judgment. However, the outcome of these judgments is not known beforehand. This means that, upon further judgment in a tribunal, these transactions might be found to be non-fraudulent. This limitation can compromise the reliability of the ML models applied to Uphold's dataset.

8.3 Method Application

Overall, the same method applied to the Elliptic Dataset will also be applied to Uphold's proprietary dataset. Applying the same method to both datasets will be important to gain more insights into the nature of AD algorithms and whether they are capable of identifying relevant fraud-related patterns within an unlabeled pool, that can improve the performance of a supervised classifier. By comparing and establishing parallels between the results obtained from using the two datasets, it will be possible to determine if the patterns observed in the Elliptic Dataset hold true in Uphold's proprietary dataset as well. By doing so, it will be possible to compare and contrast the findings to gain a deeper understanding of anomalous transactions and their potential indications of fraudulent behavior in cryptocurrency transactions, when incorporated in AL setups.

From the Elliptic Dataset, it was concluded, that the anomalies identified by the AD algorithms, considering the specific set of features used, were not indicative of cryptocurrency fraudulent behavior (high FP), thereby, these methods were not capable of identifying relevant fraud-related patterns that a supervised model should learn to improve its performance. However, it is important to confirm if these findings account for other real-life cryptocurrency datasets such as Uphold's dataset.

Analyzing Uphold's dataset would potentially offer a broader and more comprehensive understanding of the efficacy of training AD algorithms using diverse subsets of features. Furthermore, it would be intriguing to explore the specific features associated with transactional attributes, network characteristics, or user behavior that contribute most significantly to the performance of AD algorithms in detecting fraudulent behavior. Such investigation could enhance the knowledge in the cryptocurrency domain and shed light on which set of anomalous values have the greatest impact on identifying relevant crypto fraudulent activities.

Additionally, the comparison of results obtained from both datasets will allow to establish a parallel between the performance of the models for different illicit rates. It's worth noting that the illicit rate in Uphold's dataset is substantially lower (0.82%) compared to the Elliptic Dataset (10%).

Furthermore, it would be valuable to confirm if transitioning from an unsupervised AL strategy to a more sophisticated supervised AL policy as soon as possible significantly enhances, in fact, the performance of a supervised classifier faster. Therefore, it is important to continue investigating whether it is beneficial or not to continue using AD algorithms throughout a larger number of iterations of the AL process. These insights will help to understand whether AD algorithms are only essential to address the cold start scenario.

However, a challenge arises when it comes to splitting the data into train and test sets. When dividing the dataset into train and test sets for transaction-level prediction, it is important to ensure that transactions involving the same users do not appear in both sets. This prevents data leakage and ensures that the models are tested on unseen users. This is important because, generally, the behavior of users is of extreme importance to identify a transaction as fraudulent or not. Therefore, if the same users are on the train and test set, it is possible that the evaluation does not reflect the real performance of the model.

8.4 Implementation Difficulties

As mentioned earlier in this document, this dissertation was conducted as part of a work done at Uphold. The initial objectives of this work were to assess the feasibility of the combined use of AD and AL algorithms using the proposed methodology on a publicly available dataset, and subsequently apply the same methodology to Uphold's proprietary dataset.

However, since the dataset provided by Uphold was received very close to the internship end date, there was insufficient time to apply the proposed methodology to Uphold's dataset.

Moreover, as stated, the dataset was of enormous size, making it impractical to apply the methodology within the available timeframe. Consequently, the remaining time was dedicated to understanding the intricacies of the dataset and performing feature engineering and pre-processing steps.

The size of Uphold's dataset presented a significant challenge as it required substantial processing time to conduct the experiments. For instance, executing an LR (one of the supervised models tested) took approximately two days to complete.

Unfortunately, the internship's completion marked the end of the implementation process due to the strict privacy concerns and stringent data security protocols enforced by Uphold. Consequently, this phase of the project yielded no definitive outcomes or findings.

9. CONCLUSION AND FUTURE WORK

The exponential growth witnessed in the cryptocurrency market over the past years has undoubtedly brought forth numerous opportunities, but it has also inadvertently paved the way for a surge in fraudulent activities. These illicit practices, including money laundering, phishing, Ponzi schemes, and insider trading, pose a significant threat to the integrity and stability of the cryptocurrency market. As a result, there is an urgent requirement for the implementation of robust and effective Fraud Detection (FD) mechanisms to safeguard the market from these nefarious activities.

This research investigated the synergistic utilization of unsupervised AD algorithms and AL techniques to create an iterative process for acquiring labeled transactions, which in turn facilitates training a supervised classifier. The primary aim of this approach is to harness the capabilities of AD algorithms in selecting a subset of data that maximizes the learning potential of the subsequent supervised model.

Initially, a comprehensive investigation was conducted, shedding light on several crucial concepts relevant to this research. Specifically, it was explored the evolving nature of cryptocurrency fraud, the utilization of ML techniques in the FD domain, the principles of AD methods, and the potential combination of AL with AD algorithms. Further, it was conducted a comprehensive literature review to explore the existing approaches in the field. The findings from this review were instrumental in gaining a profound understanding of the strengths and weaknesses of traditional ML methods for FD. Moreover, the review shed light on the main approaches to AL with AD and how they could effectively explore the potentialities associated with both unsupervised (exploration) and supervised (exploitation) methods. The aim of this comprehensive review was to identify the most effective methods and those that have been extensively investigated in the field. To achieve this, an exhaustive analysis of 68 scientific articles was conducted, allowing for a robust and comprehensive examination of the topic.

The next phase of the project involved adopting a more practical approach. However, due to the initial unavailability of Uphold to provide a proprietary dataset comprising real-life cryptocurrency transactional data, it was crucial to acquire a public dataset that could adequately represent Uphold's transactional data. After careful consideration, the Elliptic dataset was chosen, as it had been extensively studied in the literature review. Moreover, this dataset possessed the necessary characteristics to facilitate the simulation of the AL process.

Further, to gain a deeper understanding of the main features and the distribution of transactions, exploratory data analysis was conducted on the dataset.

In the literature review, a study conducted by Lorenz et al. (2020) was selected due to its provision of a suitable framework for combining AD algorithms with AL. The purpose of selecting this particular study was to establish a foundation from which valuable insights could be gained and subsequently applied to Uphold's data.

Based on Lorenz et al. (2020) work, several key algorithms were selected and described comprehensively. The initial phase involved the selection of different supervised ML models such as RF, LR, and XGBoost, which served as a baseline for performance comparison with the AL approaches. As part of the unsupervised AL policies, three algorithms were employed: LOF, IF, and Elliptic Envelope. These algorithms aim to identify the most anomalous transactions within the unlabeled pool to be further investigated and labeled. Additionally, three supervised AL policies were implemented, which rely on supervised models to identify the most informative instances in the unlabeled pool. These supervised AL strategies include Uncertainty Sampling, EMC, and QBC.

Concerning the supervised baseline several conclusions were drawn. Among the models tested, RF emerged as the top performer in terms of the F1-score, achieving a score of 0.83. Additionally, RF also exhibited exceptional Precision indicating that the positive predictions made by the RF model were highly accurate. It is important to highlight that while RF exhibited the best performance, XGBoost also demonstrated commendable results. On the other hand, LR showed poorer overall performance, in comparison to the RF and XGBoost models. These findings underscored the effectiveness of ensemble methods like RF and XGBoost in handling the complexities and nuances of cryptocurrency FD.

The experiments conducted to assess the feasibility of using AD algorithms within AL setups were divided into two scenarios. In the first scenario, AD algorithms were utilized as unsupervised AL strategies throughout the entire AL process. In the second scenario, a combination of unsupervised and supervised AL strategies was employed, with the transition to a supervised AL policy being made after reaching a specific cut-off point. The results of these scenarios shed light on the impact of incorporating AD algorithms into the AL process and provided insights into the optimal cut-off point for transitioning to a more sophisticated supervised AL policy.

With regard to the AL experiments scenario where it was used only unsupervised AL policies throughout the entire AL process, some conclusions can be taken. It was observed a subpar performance of the AD algorithms in correctly identifying and detecting a sufficient number of illicit transactions. Consequently, these methods proved to be highly ineffective in iteratively querying relevant fraud-related patterns within the unlabeled pool to be further labeled and enhance the performance of a supervised classifier. Surprisingly, even an algorithm that randomly selects instances from the unlabeled data exhibited similar or even superior results compared to the AD algorithms. These findings shed light on the limitations of AD algorithms, particularly when the boundary between normal and illicit behavior is not well-defined. In the case of the Elliptic dataset, there were no distinct outlying instances that could be unequivocally classified as illicit.

This raises two important conclusions, which were subsequently discussed in detail. Firstly, it was hypothesized that the absence of clearly anomalous transactions corresponding to fraudulent activities could be attributed to the nature of the dataset, which primarily consists of Bitcoin cryptocurrency transactions. The inherent volatility and dynamics of cryptocurrency markets may result in legitimate transactions appearing anomalous from a static perspective. However, these transactions are driven by genuine investments and strategic trading activities rather than fraudulent intentions. This suggests that the apparent anomaly of certain transactions should be carefully evaluated within the broader context of cryptocurrency market behavior. Secondly, the findings imply the potential existence of deliberate mimicry of normal transaction behavior by fraudsters when conducting cryptocurrency transactions. This observation raises important questions about the strategies employed by criminals to mask their illicit activities by adopting patterns similar to legitimate transactions.

Based on the bad performance of the unsupervised AD algorithms, the results from the second scenario shed light on some additional conclusions. This scenario prompted a crucial realization: transitioning from an unsupervised AL strategy to a more sophisticated supervised AL policy as early as possible significantly enhances the performance of a supervised classifier. Remarkably, optimal solutions were achieved using a minimal number of labeled transactions, typically around 700 to 800 labels. This further highlighted the poor performance of unsupervised AL policies in finding relevant transactions within the unlabeled pool. By adopting a supervised AL policy after the completion of a single iteration of the AL process, it was even possible to surpass the baseline performance of the LR classifier by a substantial

margin of 0.15 in terms of the F1-score. This great performance demonstrates the practical significance of utilizing AL techniques as a cohesive framework, as a way to reduce the time and effort required for labeling a large-scale dataset to train a supervised classifier.

These findings also emphasized that AD methods are only effective in addressing the cold start scenario, where there are no initial labels available for the AL process to start with. However, from previous considerations, these methods were proven to perform similarly or even worse than the Random Sampling algorithm in resolving this problem. This dispels the notion of their feasibility.

It's worth noting that all the implemented code is available on GitHub at https://github.com/leandrocunha13/Anomaly-Detection-in-Cryptocurrency-Transactions-with-Active-Learning.

Finally, the opportunity to work with real data from Uphold has emerged, offering a unique chance to evaluate the practicality of the proposed methodology using Uphold's proprietary dataset. The subsequent discussions encompassed a broad spectrum of topics, including an examination of the dataset itself, the process of feature engineering, and a comprehensive overview of the dataset's strengths and limitations. These limitations were further deliberated, and potential solutions were thoroughly explored.

The biggest challenge of this research was related to the significant computational time required to execute the AL setups. The nature of this research involved conducting numerous AL setups, which encompassed various combinations of unsupervised and supervised AL strategies across different cut-off points and supervised classifiers. Consequently, this extensive experimentation demanded considerable computational resources and proved to be highly time-consuming. To address the aforementioned challenge, Uphold provided a personal computer equipped with enhanced computational resources. Even so, it still, took considerable time to obtain reliable results.

Finally, this research contributes to the expanding knowledge base within the field of cryptocurrency FD, offering insights for future studies seeking to enhance the efficacy of AD and AL techniques in this domain. Through careful analysis and evaluation, it was gained a deeper understanding of the underlying mechanisms and factors that may influence their effectiveness. This knowledge is crucial for researchers, practitioners, and stakeholders involved in the development and implementation of AD and AL strategies in the cryptocurrency realm.

As future work emerges:

- Investigate and evaluate additional unsupervised and supervised labeling strategies, and analyze their outcomes, considering the balance between implementation complexity and achieved results.
- Explore other real-life cryptocurrency datasets. Exploring other cryptocurrency datasets can provide a more comprehensive understanding of the performance and limitations of AD and AL techniques. Different datasets may exhibit varying characteristics, transaction patterns, and levels of fraud, which can impact the effectiveness of the proposed approaches. It is important to evaluate the generalizability of the findings to ensure the robustness of the tested methods. Additionally, analyzing these findings in relation to other real-life cryptocurrency datasets will enable to draw meaningful comparisons. By establishing parallels, it would be possible to gain insights into various fraud rates and class distributions and determine if the most deviant crypto transactions are still not indicative of fraudulent activity.
- Conduct an in-depth investigation on the impact of different feature sets on the performance of AD algorithms. Gaining a deeper understanding of the relevance and impact of different features in AD algorithms can enhance the accuracy of detecting illicit transactions. Future research should explore various feature sets, including transactional attributes, network characteristics, and user behavior, to determine which features contribute most significantly to the performance of AD algorithms.
- Investigate graph embedding techniques that utilize graph information and evaluate their potential to enhance the prediction accuracy of AD algorithms. These techniques are designed to capture the structural and relational details of the cryptocurrency transactional graph and represent them as lower-dimensional vectors known as embeddings. By leveraging these embeddings, it becomes possible to apply AD algorithms and compute anomaly scores for individual nodes, potentially leading to improved performance of AD methods.

By addressing these areas, one can advance the understanding and make more strides in the field of AD and AL.

REFERENCES

Abdallah, A., Maarof, M. A., & Zainal, A. (2016). Fraud detection system: A survey. *Journal of Network and Computer Applications*, *68*, 90–113. https://doi.org/10.1016/J.JNCA.2016.04.007

Ahmed, M., Naser Mahmood, A., & Hu, J. (2016). A survey of network anomaly detection techniques. *Journal of Network and Computer Applications*, *60*, 19–31. https://doi.org/10.1016/J.JNCA.2015.11.016

Ajedig, M. A., Li, F., & Ur Rehman, A. (2011). Detecting credit card fraud by decision trees and support vector machines. *IMECS 2011 - International MultiConference of Engineers and Computer Scientists 2011*, *1*, 283–286. https://openaccess.dogus.edu.tr/xmlui/handle/11376/2366

Alarab, I., & Prakoonwit, S. (2022). Graph-Based LSTM for Anti-money Laundering: Experimenting Temporal Graph Convolutional Network with Bitcoin Data. *Neural Processing Letters*, 1–19. https://doi.org/10.1007/S11063-022-10904-8/TABLES/5

Alarab, I., Prakoonwit, S., & Nacer, M. I. (2020). Comparative Analysis Using Supervised Learning Methods for Anti-Money Laundering in Bitcoin. *ACM International Conference Proceeding Series*, 11–17. https://doi.org/10.1145/3409073.3409078

Awoyemi, J. O., Adetunmbi, A. O., & Oluwadare, S. A. (2017). Credit card fraud detection using machine learning techniques: A comparative analysis. *Proceedings of the IEEE International Conference on Computing, Networking and Informatics, ICCNI 2017*, *2017-Janua*, 1–9. https://doi.org/10.1109/ICCNI.2017.8123782

Bachman, P., Sordoni, A., & Trischler, A. (2017). Learning Algorithms for Active Learning. *34th International Conference on Machine Learning, ICML 2017*, *1*, 477–486. https://doi.org/10.48550/arxiv.1708.00088

Barata, R., Leite, M., Ricardo Pacheco, F., Marco P Sampaio, F. O., João Tiago Ascensão, F., Pedro Bizarro, F., Pacheco, R., P Sampaio, M. O., Tiago Ascensão, J., & Bizarro, P. (2021). Active learning for imbalanced data under cold start. *ICAIF 2021 - 2nd ACM International Conference on AI in Finance*. https://doi.org/10.48550/arxiv.2107.07724

Cachin, C., & Vukolić, M. (2017). Blockchain consensus protocols in the wild. *Leibniz International Proceedings in Informatics, LIPIcs*, *91*. https://doi.org/10.4230/LIPIcs.DISC.2017.1

Cai, W., Zhang, M., & Zhang, Y. (2017). Batch mode active learning for regression with expected model change. *IEEE Transactions on Neural Networks and Learning Systems*, *28*(7), 1668–1681. https://doi.org/10.1109/TNNLS.2016.2542184

Camino, R. D., State, R., Montero, L., & Valtchev, P. (2017). *Finding Suspicious Activities in Financial Transactions and Distributed Ledgers*. https://orbilu.uni.lu/handle/10993/33743

Carcillo, F., le Borgne, Y. A., Caelen, O., & Bontempi, G. (2017). An assessment of streaming active learning strategies for real-Life credit card fraud detection. *Proceedings - 2017 International Conference on Data Science and Advanced Analytics, DSAA 2017*, *2018-Janua*, 631–639. https://doi.org/10.1109/DSAA.2017.10

Choi, D., & Lee, K. (2018). An Artificial Intelligence Approach to Financial Fraud Detection under IoT Environment: A Survey and Implementation. *Security and Communication Networks*, *2018*. https://doi.org/10.1155/2018/5483472

Choi, R. Y., Coyner, A. S., Kalpathy-Cramer, J., Chiang, M. F., & Peter Campbell, J. (2020). Introduction to Machine Learning, Neural Networks, and Deep Learning. *Translational Vision Science & Technology*, *9*(2), 14–14. https://doi.org/10.1167/TVST.9.2.14

Das, S., Islam, M. R., Jayakodi, N. K., & Doppa, J. R. (2018). *Active Anomaly Detection via Ensembles*. https://arxiv.org/abs/1809.06477v1

Das, S., Wong, W.-K., Fern, A., Dietterich, T. G., & Siddiqui, M. A. (2017). *Incorporating Feedback into Tree-based Anomaly Detection*. https://doi.org/10.48550/arxiv.1708.09441

di Milano, P., Stefano, R. :, Correlatore, Z., Carminati, D. M., Di, T., Magistrale Di:, L., Dignani, A., & 837471, M. (2018). *FraudsDigger : an active learning tool for online banking fraud detection*. https://www.politesi.polimi.it/handle/10589/144814

Domingues, R., Filippone, M., Michiardi, P., & Zouaoui, J. (2018). A comparative evaluation of outlier detection algorithms: Experiments and analyses. *Pattern Recognition*, *74*, 406–421. https://doi.org/10.1016/J.PATCOG.2017.09.037

Goldstein, M., & Uchida, S. (2016). A Comparative Evaluation of Unsupervised Anomaly Detection Algorithms for Multivariate Data. *PLOS ONE*, *11*(4), e0152173. https://doi.org/10.1371/JOURNAL.PONE.0152173

Hilal, W., Gadsden, S. A., & Yawney, J. (2022). Financial Fraud: A Review of Anomaly Detection Techniques and Recent Advances. *Expert Systems with Applications*, *193*, 116429. https://doi.org/10.1016/J.ESWA.2021.116429

Houlsby, N., Huszár, F., Ghahramani, Z., & Lengyel, M. (2011). *Bayesian Active Learning for Classification and Preference Learning*. https://doi.org/10.48550/arxiv.1112.5745

Ibrahim, R. F., Elian, A. M., & Ababneh, M. (2021). Illicit Account Detection in the Ethereum Blockchain Using Machine Learning. *2021 International Conference on Information Technology (ICIT)*, 488–493. https://doi.org/10.1109/ICIT52682.2021.9491653

Injadat, M., Salo, F., Nassif, A. B., Essex, A., & Shami, A. (2018). Bayesian Optimization with Machine Learning Algorithms Towards Anomaly Detection. *2018 IEEE Global Communications Conference, GLOBECOM 2018 - Proceedings*. https://doi.org/10.1109/GLOCOM.2018.8647714

Jeragh, M., & Alsulaimi, M. (2019). Combining Auto Encoders and One Class Support Vectors Machine for Fraudulant Credit Card Transactions Detection. *Proceedings of the 2nd World Conference on Smart Trends in Systems, Security and Sustainability, WorldS4 2018*, 57–64. https://doi.org/10.1109/WORLDS4.2018.8611624

Jeyakumar, S., Andrew Charles, E. Y. , Rathore, P., PALANISWAMI, M., Muthukkumarasamy, V., & Hóu, Z. (2023). *Feature Engineering for Anomaly Detection and Classification of Blockchain Transactions*. https://doi.org/10.36227/TECHRXIV.22329805.V1

Kaufman, E., & Iaremenko, A. (2022). *Anomaly Detection for Fraud in Cryptocurrency Time Series*. https://arxiv.org/abs/2207.11466v1

Kee, S., del Castillo, E., & Runger, G. (2018). Query-by-committee improvement with diversity and density in batch active learning. *Information Sciences*, *454–455*, 401–418. https://doi.org/10.1016/J.INS.2018.05.014

Khan, S. S., & Hoey, J. (2017). Review of fall detection techniques: A data availability perspective. *Medical Engineering and Physics*, *39*, 12–22. https://doi.org/10.1016/J.MEDENGPHY.2016.10.014

Kong, J., Kowalczyk, W., Menzel, S., & Bäck, T. (2020). Improving imbalanced classification by anomaly detection. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, *12269 LNCS*, 512–523. https://doi.org/10.1007/978-3-030-58112-1_35/TABLES/4

Korejo, M. S., Rajamanickam, R., & Muhamad, M. H. (2021). The concept of money laundering: a quest for legal definition. *Journal of Money Laundering Control*, *24*(4), 725–736. https://doi.org/10.1108/JMLC-05-2020-0045/FULL/PDF

Labanca, D., Primerano, L., Markland-Montgomery, M., Polino, M., Carminati, M., & Zanero, S. (2022). Amaretto: An Active Learning Framework for Money Laundering Detection. *IEEE Access*, *10*, 41720–41739. https://doi.org/10.1109/ACCESS.2022.3167699

Laws, F., Schütze, H., & Schütze, S. (2008). *Stopping Criteria for Active Learning of Named Entity Recognition*. 465–472. https://doi.org/10.5555/1599081.1599140

Lewis, D. D., & Gale, W. A. (1994). A Sequential Algorithm for Training Text Classifiers. *Proceedings of the 17th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 1994*, 3–12. https://doi.org/10.1007/978-1-4471-2099-5_1

Liu, F. T., Ting, K. M., & Zhou, Z. H. (2008). Isolation forest. *Proceedings - IEEE International Conference on Data Mining, ICDM*, 413–422. https://doi.org/10.1109/ICDM.2008.17

Lorenz, J., Silva, M. I., Aparício, D., Ascensão, J. T., & Bizarro, P. (2020). Machine learning methods to detect money laundering in the bitcoin blockchain in the presence of label scarcity. *ICAIF 2020 - 1st ACM International Conference on AI in Finance*. https://doi.org/10.1145/3383455.3422549

Melo-Acosta, G. E., Duitama-Munoz, F., & Arias-Londono, J. D. (2017). Fraud detection in big data using supervised and semi-supervised learning techniques. *2017 IEEE Colombian Conference on Communications and Computing, COLCOM 2017 - Proceedings*. https://doi.org/10.1109/COLCOMCON.2017.8088206

Monamo, P. M., Marivate, V., & Twala, B. (2017). *A Multifaceted Approach to Bitcoin Fraud Detection: Global and Local Outliers*. 188–194. https://doi.org/10.1109/ICMLA.2016.0039

Nassif, A. B., Talib, M. A., Nasir, Q., & Dakalbab, F. M. (2021). Machine Learning for Anomaly Detection: A Systematic Review. *IEEE Access*, *9*, 78658–78700. https://doi.org/10.1109/ACCESS.2021.3083060

Nguyen, V. L., Destercke, S., & Hüllermeier, E. (2019). Epistemic Uncertainty Sampling. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, *11828 LNAI*, 72–86. https://doi.org/10.48550/arxiv.1909.00218

Niu, X., Wang, L., & Yang, X. (2019). *A Comparison Study of Credit Card Fraud Detection: Supervised versus Unsupervised*. https://doi.org/10.48550/arxiv.1904.10604

Nixon, C., Sedky, M., & Hassan, M. (2021). *SALAD: An Exploration of Split Active Learning based Unsupervised Network Data Stream Anomaly Detection using Autoencoders*. https://doi.org/10.36227/TECHRXIV.14896773.V1

Omar, S., Ngadi, A., & H. Jebur, H. (2013). Machine Learning Techniques for Anomaly Detection: An Overview. *International Journal of Computer Applications*, *79*(2), 33–41. https://doi.org/10.5120/13715-1478

Ostapowicz, M., & Żbikowski, K. (2019). Detecting Fraudulent Accounts on Blockchain: A Supervised Approach. *ArXiv*, *11881 LNCS*, 18–31. https://doi.org/10.1007/978-3-030-34223-4_2

Ounacer, S., el Bour, H. A., Oubrahim, Y., Ghoumari, M. Y., & Azzouazi, M. (2018). Using Isolation Forest in anomaly detection: The case of credit card transactions. *Periodicals of Engineering and Natural Sciences*, *6*(2), 394–400. https://doi.org/10.21533/PEN.V6I2.533

Pan, D., Nie, L., Kang, W., & Song, Z. (2020). UAV Anomaly Detection Using Active Learning and Improved S3VM Model. *International Conference on Sensing, Measurement and Data Analytics in the Era of Artificial Intelligence, ICSMD 2020 - Proceedings*, 253–258. https://doi.org/10.1109/ICSMD50554.2020.9261709

Pelleg, D., & Moore, A. (2005). Active learning for anomaly and rare-category detection. *Advances in Neural Information Processing Systems*.

Pichara, K., & Soto, A. (2011). Active learning and subspace clustering for anomaly detection. *Intelligent Data Analysis*, *15*(2), 151–171. https://doi.org/10.3233/IDA-2010-0461

Sabry, F., Labda, W., Erbad, A., & Malluhi, Q. (2020). Cryptocurrencies and Artificial Intelligence: Challenges and Opportunities. *Undefined*, *8*, 175840–175858. https://doi.org/10.1109/ACCESS.2020.3025211

Settles, B. (2012). Active Learning. *Synthesis Lectures on Artificial Intelligence and Machine Learning*, *18*, 1–111. https://doi.org/10.2200/S00429ED1V01Y201207AIM018

Settles, B., Craven, M., & Ray, S. (2007). Multiple-Instance Active Learning. *Advances in Neural Information Processing Systems*, *20*.

Sharma, M., & Bilgic, M. (2017). Evidence-based uncertainty sampling for active learning. *Data Mining and Knowledge Discovery*, *31*(1), 164–202. https://doi.org/10.1007/S10618-016-0460-3/TABLES/9

Shayegan, M. J., Sabor, H. R., Uddin, M., & Chen, C. L. (2022). A Collective Anomaly Detection Technique to Detect Crypto Wallet Frauds on Bitcoin Network. *Symmetry 2022, Vol. 14, Page 328*, *14*(2), 328. https://doi.org/10.3390/SYM14020328

Siddiqui, M. A., Stokes, J. W., Seifert, C., Argyle, E., McCann, R., Neil, J., & Carroll, J. (2019). Detecting Cyber Attacks Using Anomaly Detection with Explanations and Expert Feedback. *ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings*, *2019-May*, 2872–2876. https://doi.org/10.1109/ICASSP.2019.8683212

Stokes, J. W., Platt, J. C., Kravis, J., & Shilman, M. (2008). *ALADIN: Active Learning of Anomalies to Detect Intrusions*. https://www.microsoft.com/en-us/research/publication/aladin-active-learning-of-anomalies-to-detect-intrusions/

Thudumu, S., Branch, P., Jin, J., & Singh, J. (Jack). (2020). A comprehensive survey of anomaly detection techniques for high dimensional big data. *Journal of Big Data*, *7*(1). https://doi.org/10.1186/S40537-020-00320-X

Trautman, L. (2014). Virtual Currencies Bitcoin & What Now After Liberty Reserve, Silk Road, and Mt. Gox? *Richmond Journal of Law & Technology*, *20*(4). https://scholarship.richmond.edu/jolt/vol20/iss4/3

Trozze, A., Kamps, J., Akartuna, E. A., Hetzel, F. J., Kleinberg, B., Davies, T., & Johnson, S. D. (2022). Cryptocurrencies and future financial crime. *Crime Science*, *11*(1), 1–35. https://doi.org/10.1186/S40163-021-00163-8/TABLES/8

Vynokurova, O., Peleshko, D., Bondarenko, O., Ilyasov, V., Serzhantov, V., & Peleshko, M. (2020). Hybrid machine learning system for solving fraud detection tasks. *Proceedings of the 2020 IEEE 3rd International Conference on Data Stream Mining and Processing, DSMP 2020*, 1–5. https://doi.org/10.1109/DSMP47368.2020.9204244

Weber, M., Domeniconi, G., Chen, J., Weidele, D. K. I., Bellei, C., Robinson, T., & Leiserson, C. E. (2019). *Anti-Money Laundering in Bitcoin: Experimenting with Graph Convolutional Networks for Financial Forensics*. https://doi.org/10.48550/arxiv.1908.02591

West, J., & Bhattacharya, M. (2016). Intelligent financial fraud detection: A comprehensive review. *Computers & Security*, *57*, 47–66. https://doi.org/10.1016/J.COSE.2015.09.005

Xuan, S., Liu, G., Li, Z., Zheng, L., Wang, S., & Jiang, C. (2018). Random forest for credit card fraud detection. *ICNSC 2018 - 15th IEEE International Conference on Networking, Sensing and Control*, 1–6. https://doi.org/10.1109/ICNSC.2018.8361343

Xue, Z., Wang, M., Zhang, Q., Zhang, Y., & Liu, P. (2021). A regulatable blockchain transaction model with privacy protection. *International Journal of Computational Intelligence Systems*, *14*(1), 1642–1652. https://doi.org/10.2991/IJCIS.D.210528.001

Yang, L., Dong, X., Xing, S., Zheng, J., Gu, X., & Song, X. (2019). An abnormal transaction detection mechanim on bitcoin. *Proceedings - 2019 International Conference on Networking and Network Applications, NaNA 2019*, 452–457. https://doi.org/10.1109/NANA.2019.00083

Zamini, M., & Montazer, G. (2019). Credit Card Fraud Detection using autoencoder based clustering. *9th International Symposium on Telecommunication: With Emphasis on Information and Communication Technology, IST 2018*, 486–491. https://doi.org/10.1109/ISTEL.2018.8661129

Zengan, G. (2009). Application of cluster-based local outlier factor algorithm in anti-money laundering. *Proceedings - International Conference on Management and Service Science, MASS 2009*. https://doi.org/10.1109/ICMSS.2009.5302396

Zha, D., Lai, K. H., Wan, M., & Hu, X. (2020). Meta-AAD: Active anomaly detection with deep reinforcement learning. *Proceedings - IEEE International Conference on Data Mining, ICDM*, *2020-November*, 771–780. https://doi.org/10.1109/ICDM50108.2020.00086

Zhou, H., Sun, G., Fu, S., Wang, L., Hu, J., & Gao, Y. (2021). Internet Financial Fraud Detection Based on a Distributed Big Data Approach with Node2vec. *IEEE Access*, *9*, 43378–43386. https://doi.org/10.1109/ACCESS.2021.3062467

Zong, B., Song, Q., Min, M. R., Cheng, W., Lumezanu, C., Cho, D., & Chen, H. (2022). Deep Autoencoding Gaussian Mixture Model for Unsupervised Anomaly Detection. *31st International Conference on Machine Learning, ICML 2014*, *5*, 3800–3809.

APPENDIX I – SUPERVISED BASELINE MODELS' PARAMETERS

Table 17: Supervised Classifiers Parameters

Model	Parameters
RF	n_estimators=100, criterion='gini', max_depth=None, min_samples_split=2, min_samples_leaf=1, min_weight_fraction_leaf=0.0, max_features='sqrt', max_leaf_nodes=None, min_impurity_decrease=0.0, bootstrap=True, oob_score=False, n_jobs=None, random_state=None*[1], verbose=0, warm_start=False, class_weight=None, ccp_alpha=0.0, max_samples=None
LR	penalty='l2', dual=False, tol=0.0001, C=1.0, fit_intercept=True, intercept_scaling=1, class_weight=None, random_state=None*[1], solver='lbfgs', max_iter=100, multi_class='auto', verbose=0, warm_start=False, n_jobs=None, l1_ratio=None
XGBoost	booster='gbtree', verbosity=1, validade_parameter=false, disable_default_eval_metric=false, eta=0.3, gamma=0, max_depth=6, min_child_weight=1, max_delta_step=0, subsample=1, sampling_method='uniform', colsample_bytree=1, colsample_bylevel=1, colsample_bynode=1, lambda=1, alpha=0, tree_method=auto, class_weight=None, refresh_leaf=1, process_type='default', grow_policy='depthwise', max_leaves=0, max_bin=256, predictor='auto', random_state=None*[1]

[1] The 'random_state' parameter was initially set as 'None*'. The asterisk (*) denotes that 'None' serves as the default value, indicating that the random state was not explicitly specified in the table. However, in the actual code implementation, a specific random state was generated instead.

APPENDIX II – ANOMALY DETECTION MODELS' PARAMETERS

Table 18: Unsupervised Anomaly Detection Models' Parameters

Model	Parameters
IF	n_estimators=100, max_samples='auto', contamination=0.1, max_features=1.0, behavior='old', bootstrap=False, n_jobs=1, random_state=None*[1], verbose=0, warm_start=False
LOF	n_neighbors=20, algorithm='auto', leaf_size=30, metric='minkowski', p=2, metric_params=None, contamination=0.1, novelty=False, n_jobs=1
Elliptic Envelope	store_precision=True, assume_centered=False, support_fraction=None, contamination=0.1, random_state=None*[1]

APPENDIX III – UNCERTAINTY SAMPLING MODELS' PARAMETERS

Table 19: Uncertainty Sampling Models' Parameters

Model	Parameters
RF	n_estimators=100, criterion='gini', max_depth=None, min_samples_split=2, min_samples_leaf=1, min_weight_fraction_leaf=0.0, max_features='sqrt', max_leaf_nodes=None, min_impurity_decrease=0.0, bootstrap=True, oob_score=False, n_jobs=None, random_state=None*[1], verbose=0, warm_start=False, class_weight='balanced' , ccp_alpha=0.0, max_samples=None
LR	penalty='l2', dual=False, tol=0.0001, C=1.0, fit_intercept=True, intercept_scaling=1, class_weight='balanced', random_state=None*[1], solver='lbfgs', max_iter=100, multi_class='auto', verbose=0, warm_start=False, n_jobs=None, l1_ratio=None
XGBoost	booster='gbtree', verbosity=1, validade_parameter=false, disable_default_eval_metric=false, eta=0.3, gamma=0, max_depth=6, min_child_weight=1, max_delta_step=0, subsample=1, sampling_method='uniform', colsample_bytree=1, colsample_bylevel=1, colsample_bynode=1, lambda=1, alpha=0, tree_method=auto, class_weight='balanced' , refresh_leaf=1, process_type='default', grow_policy='depthwise', max_leaves=0, max_bin=256, predictor='auto', random_state=None*[1]

APPENDIX IV – QUERY BY COMMITTEE MODEL PARAMETERS

Table 20: Query by Committee Model Parameters

Model	Parameters
RF	n_estimators=100, criterion='gini', max_depth=3, min_samples_split=2, min_samples_leaf=1, min_weight_fraction_leaf=0.0, max_features='sqrt', max_leaf_nodes=None, min_impurity_decrease=0.0, bootstrap=True, oob_score=False, n_jobs=None, random_state=None*[1], verbose=0, warm_start=False, class_weight='balanced' , ccp_alpha=0.0, max_samples=None

APPENDIX V – EXPECTED MODEL CHANGE MODEL PARAMETERS

Table 21: Expected Model Change Model Parameters

Model	Number of Iterations	Tolerance	Learning rate
LR	2500	0.0001	0.01