NOVA IMS Information Management School

MAA

Mestrado em Métodos Analíticos Avançados
Master Program in Advanced Analytics

Machine learning methods to detect money laundering in the Bitcoin blockchain in the presence of label scarcity

Joana Susan Lorenz

Internship Report presented as the partial requirement for obtaining a Master's degree in Data Science and Advanced Analytics

NOVA Information Management School
Instituto Superior de Estatística e Gestão de Informação
Universidade Nova de Lisboa

NOVA Information Management School
Instituto Superior de Estatística e Gestão de Informação
Universidade Nova de Lisboa

MACHINE LEARNING METHODS TO DETECT MONEY LAUNDERING IN THE BITCOIN BLOCKCHAIN IN THE PRESENCE OF LABEL SCARCITY

by

Joana Susan Lorenz

Internship Report presented as the partial requirement for obtaining a Master's degree in Data Science and Advanced Analytics

Advisor: : Flávio Luís Portas Pinheiro
External advisors: Maria Inês Silva, David Aparício

March 2021

The following thesis project resulted in the publication of a scientific research article. The article was accepted as a spotlight presentation to the KDD'20 Workshop on Machine Learning in Finance. It was also accepted to the 2020 ACM International Conference on AI in Finance and will be published in the proceedings of the conference. The paper is currently available on arXiv (Lorenz et al., 2020) and can be found in Appendix A.

Machine learning methods to detect money laundering in the Bitcoin blockchain in the presence of label scarcity

Acknowledgements

First, I would like to thank my advisors at Feedzai, Maria Inês Silva and David Aparício. They let this work be my own, yet they gave invaluable support and guidance through constructive discussions, relevant input and thorough reviews of my work. I also thank João Ascensão and Pedro Bizarro for their exceptional leadership, availability and advice. Throughout this project, the entire Feedzai Research team was a remarkable support through their culture of team spirit, collaboration and open knowledge exchange. I am proud to have become a part of this extraordinary team.

I would also like to thank my academic thesis advisor Flávio Pinheiro of NOVA IMS. He gave me the trust to pursue this project independently, together with my advisors at Feedzai, but was always available to give useful feedback. Additionally, he was incredibly helpful in overcoming any bureaucratic hurdles in the process.

Abstract

Every year, criminals launder billions of dollars acquired from serious felonies (e.g. terrorism, drug smuggling, or human trafficking), harming countless people and economies. Cryptocurrencies, in particular, have developed as a haven for money laundering activity. Machine Learning can be used to detect these illicit patterns. However, labels are so scarce that traditional supervised algorithms are inapplicable. This research addresses money laundering detection assuming minimal access to labels. The results show that existing state-of-the-art solutions using unsupervised anomaly detection methods are inadequate to detect the illicit patterns in a real Bitcoin transaction dataset. The proposed active learning solution, however, is capable of matching the performance of a fully supervised baseline by using just 5% of the labels. This solution mimics a typical real-life situation in which a limited number of labels can be acquired through manual annotation by experts.

Keywords: Anti-money laundering, Applied machine learning, Supervised learning by classification, Anomaly detection, Active learning

Contents

List of Figures

List of Tables

Acronyms

ABOD Angle-Based Outlier Detection

AL Active learning

AML Anti-money laundering

CBLOF Cluster-Based Outlier Factor

DT Decision Tree

FPR False positive rate

GBM Gradient Boosting Machine

IF Isolation Forest

KNN K-Nearest Neighbours

LOF Local Outlier Factor

LR Logistic Regression

ML Machine learning

OCSVM One-Class Support Vector Machine

PCA Principal Component Analysis

RF Random Forest

RS Random sampling

TPR True positive rate

XGB XGBoost

Chapter 1

Introduction

This thesis project was conducted within the scope of an internship in the machine learning research team at Feedzai, a technology company focused on financial crime detection using machine learning (ML). To motivate the thesis project in the context of the internship, the following chapter first gives an overview of the company's operations as well as the organization and responsibilities of the machine learning research team, in particular. Next, the research problem and its relevance for the company are discussed. Finally, the specific goals and the scope of the thesis project are defined.

1.1 Company and Team Overview

Feedzai develops, implements and maintains ML-based solutions for financial crime detection in transaction data. Its main clients are financial institutions, such as banks and payment processors, as well as retailers. Feedzai covers multiple financial crime use-cases, such as account opening fraud, transaction fraud and money laundering. The company has been recognized for its artificial intelligence capabilities (*AI 50: America's Most Promising Artificial Intelligence Companies* 2019) and best-in-class fraud detection solution (*AIM Evaluation: Fraud and AML Machine Learning Platform Vendors* 2019).

For each of its clients, Feedzai's team trains its in-house, state-of-the-art ML solution on client data and deploys it to production in order to monitor transactions in real-time. The different financial crime use-cases are combined in a platform solution, which allows clients to track cases and facilitates manual reviews by adding a human-readable layer on top of the artificial intelligence. Transactions deemed suspicious by the algorithm are then handled based on each client's preferences. Often, transactions are either approved, blocked or alerted. Alerted transactions can then be further

reviewed by human domain experts.

Feedzai consistently works on improving, re-inventing and expanding both the core ML algorithms as well as all the features and capabilities of its platform solution. Feedzai's research team is at the core of the companies constant efforts to innovate within its field and to broaden its business model. The research team consists of subgroups which are dedicated to particular research areas. The machine learning research group, where this thesis project was conducted, focuses on enabling the company to deliver state-of-the-art ML technology and explore new areas of ML application. Additionally, the team aims to constantly improve the product by integrating ML into all aspects of Feedzai's financial crime detection platform, facilitating processes and making reviews and data analyses increasingly more efficient. The content and scope of the projects in the machine learning research team as well as the immediate relevance of the project outcome for the product vary widely because projects are aligned with different company goals. Some projects aim directly at solving specific client needs, while other projects are of an exploratory nature. Projects are conducted using client data, publicly available datasets or both.

1.2 Problem Statement

Money laundering is a global and high impact problem. Criminals obtain money illegally from serious crimes and inject it into the financial system as seemingly legitimate funds (Soltani et al., 2016). Economies are increasingly acknowledging the negative implications of money laundering, such as the funding of criminal activity, tax evasion and asset bubbles in markets where the illicit funds are re-invested, such as real-estate. The resulting tightening of regulations implies large fines for financial institutions who fail to report cases of money laundering through their system (European Union, 2018). Recent examples are the 1MDB (Noonan et al., 2020) and the Danske Bank scandals (Monroe, 2020). Thus, appropriate anti-money laundering (AML) procedures are increasingly required and requested by financial institutions to comply with regulations.

In recent years, regulations have also been broadened to include cryptocurrencies (European Union, 2018). Cryptocurrencies are digital, distributed cryptographic currencies. While gaining increasing popularity as a payment method, cryptocurrencies have also developed as a haven for illegal money laundering activity due to their decentralized nature, the protection of identities through pseudonymity as well as transactional differences compared to traditional financial instruments (Brenig et al., 2015).

For Feedzai, money laundering is a relatively new, yet highly relevant use-case for financial crime detection. The company's clients (mainly financial institutions) are the market actors that are potentially subject to high fines in case they fail to appropriately detect and report money laundering cases and are thus increasingly in need of an

effective AML system. Feedzai has already extended its product portfolio to serve the market with a rule-based AML solution, which is currently the most common approach to AML efforts in the financial sector (Li et al., 2017). However, vulnerabilities derive from the relative simplicity of the rule-sets, leading to high false-positive rates (FPR) and low detection rates (Wang et al., 2008). Additionally, the public availability of regulatory rule sets makes them target to bypassing.

ML techniques could overcome the rigidity of rule-based systems by inferring complex patterns from historical data, and can potentially increase detection rates and decrease FPRs. However, labeled real-life datasets for AML are scarce, due to the sensitivity of the related information as well as the difficulty to achieve a clear, timely and complete labeling of money laundering cases. Given the complexity and constantly evolving nature of money laundering schemes, it is highly unlikely that all or even most of the entities involved in money laundering can be identified, resulting in incomplete labels. Furthermore, labels resulting from law enforcement investigations can be delayed by years, while manual labeling efforts are extremely costly and time-consuming. The resulting lack of real-life labeled datasets not only leads to current research being inconclusive, but also implies that in order to achieve a practical feasibility of ML for AML, strategies that require no labels (unsupervised learning) or just a few labels (active learning (AL)) are paramount.

For Feedzai, exploring such ML methods, specifically in a cryptocurrency setting, is highly interesting. On one hand, the findings could also give insights for AML in the financial transaction setting, where Feedzai's AML solution requires state-of-the-art techniques in order to stay ahead of the competition. On the other hand, testing AML methods in a cryptocurrency setting represents an exploration of possible AML business opportunities in the ever-growing cryptocurrency market.

1.3 Project Goals

Elliptic, a company dedicated to detecting financial crime in cryptocurrencies, recently released a dataset, which consists of a sample of 200k Bitcoin transactions, including labels according to whether the entity that initiated a transaction is licit or illicit. Bitcoin is one of the most popular cryptocurrencies. The dataset enables researchers to benchmark ML methods for AML on a real-life cryptocurrency dataset. In the paper associated with the release of the dataset, Weber et al. (2019) argued that ML approaches can boost AML efforts considerably and they evaluated various supervised ML models with good results. However, as discussed in Section 1.2, supervised methods are not an option in most real-life cases since financial institutions and companies will rarely have large-scale labelled data sets available for training.

Thus, the goal of this thesis project is to address the following problem: *how can we detect money laundering in a dataset with no labels or only very few labels?* Both unsupervised and AL approaches are compared against a supervised baseline in the real-life

cryptocurrency dataset.

The main contributions to existing research are:

- A thorough review of ML approaches for anti-money laundering (Section 2).
- A reproduction of the supervised baseline proposed by Weber et al. (2019), obtaining comparable results (Section 5.1).
- A benchmark of anomaly detection algorithms and exploration of the results (Section 5.2).
- An AL setting which obtains similar results to a supervised baseline while using only a small fraction ($\approx 5\%$) of the labels (Section 5.3).

At Feedzai, the findings of this research directly contribute to the development of an effective AML solution in a realistic transaction setting as well as to the evaluation of the cryptocurrency crime detection market as a potential business use-case.

Chapter 2

Related Work

Given the lack of appropriate benchmark datasets, the ambiguity of money laundering patterns, and the relative novelty of the problem, research on ML for AML is very scattered and heterogeneous. The lack of datasets in particular makes a fair comparison of the approaches in the literature impossible. Researchers mainly use real-world datasets with simulated illicit transactions (Gao, 2009; Keyan and Tingting, 2011; Lv et al., 2008; Tang and Yin, 2005; Wang and Dong, 2009) labeled by analysts (Camino et al., 2017; Zhang and Trubey, 2019) or through external proxies, such as criminal records (Hu et al., 2019; Weber et al., 2019). Others use fully synthetic data (Luo, 2014; Michalak and Korczak, 2011; Soltani et al., 2016) or no data at all (Bershtein and Tselykh, 2013; Gao and Ye, 2007; Liu et al., 2011; Yunkai et al., 2006).

This section provides a review of previous ML approaches proposed for AML. The approaches are hereby divided into *categories*, by adjusting the taxonomy proposed in (Li et al., 2017) to a smaller set of broader categories: behavioural modeling, AML typologies, community detection, and feature-based risk modeling. An overview of the reviewed approaches, grouped by their category as well as type (supervised, unsupervised or semi-supervised) is presented in Table 1.

Behavioural modeling assumes money laundering cases to be outliers, meaning that the illicit instances (a minority) should exhibit significantly different behaviour from legitimate ones (the majority). Typically, these approaches use unsupervised anomaly detection methods to model licit behaviour and find the instances that deviate from it. Anomalies can be defined on an individual level (i.e. anomalous transactions with respect to an entity's usual activities) (Hu et al., 2019; Li et al., 2020) or concerning group behaviour (i.e. entities that exhibit different characteristics from others in the population or in their peer group) (Gao, 2009; Larik and Haider, 2011; Liu and Zhang, 2010; Liu et al., 2008b; Raza and Haider, 2011; Wang and Dong, 2009). Peer groups

can be defined manually, based on categorical attributes such as industry, account category, or company size (Liu et al., 2008b), or based on the output of a clustering algorithm (Chen et al., 2014; Gao, 2009; Larik and Haider, 2011; Raza and Haider, 2011; Wang and Dong, 2009).

AML typologies aim to detect suspicious entities or transactions based on specific money laundering patterns. These patterns can either be encoded by experts (Michalak and Korczak, 2011) or defined leveraging frequent pattern mining (Luo, 2014). Alternatively, the money laundering patterns can be learned by a supervised classification algorithm applied to a labeled dataset (Keyan and Tingting, 2011; Lv et al., 2008; Weber et al., 2019; Zhang and Trubey, 2019).

Community detection treats groups of strongly-connected entities as potential criminal money laundering networks. These methods build the transaction graph (with entities as nodes and transactions as edges) to find communities based on network statistics on the entities' ego-networks (Savage et al., 2017), distances (Yunkai et al., 2006), or node embeddings (Wagner, 2019). Some methods apply filtering beforehand, focusing on areas appearing in many suspicious topological patterns (Li et al., 2017; Soltani et al., 2016).

Finally, *feature-based risk modeling* intends to separate the suspicious activity based on specific feature values. It includes finding noisy feature values for each feature (e.g. univariate anomaly detection methods) to flag transactions with many noisy features (Yang et al., 2014) and link analysis of the semantics to discover specific behaviour patterns associated with illicit transactions (Cao et al., 2017), e.g. shared addresses or locations of the transactions.

This study focuses on behavioral modeling approaches. This decision is based on their unsupervised nature as well as their popularity in the context of AML and the encouraging results presented in the literature. Previous studies report low FPRs (Chen et al., 2014; Liu and Zhang, 2010; Liu et al., 2008b) and good detection rates (Chen et al., 2014; Gao, 2009; Liu et al., 2008b; Wang and Dong, 2009). Some studies even report that the ML approaches were able to detect money laundering patterns that were previously unknown (Tang and Yin, 2005) or not caught by rule-based systems (Camino et al., 2017). However, a fair comparison between methods is impossible, given the heterogeneity of the evaluation setups. Generally, authors are openly doubtful about real-world reproducibility of good results, in the face of intricate patterns and incomplete labels (Chen et al., 2014; Gao, 2009; Wang and Dong, 2009). The question arises on whether reliable anomaly detection is possible in non-synthetic data, as criminals could intentionally mimic normal behaviour. This research contributes to assess the reproducibility of such results by conducting the first in-depth benchmark of anomaly detection methods in a large, labeled real-world cryptocurrency dataset and comparing their performance against a supervised baseline (AML typologies).

Table 1: Summary of research on ML methods for AML, grouped by category and learning class.

Category	Learning class	References
Behavioural modeling	unsupervised	Gao, 2009; Liu et al., 2008b Larik and Haider, 2011; Liu and Zhang, 2010; Wang and Dong, 2009
	semi-supervised	Camino et al., 2017; Chen et al., 2014; Raza and Haider, 2011
	supervised	Hu et al., 2019 Li et al., 2020
AML typologies	unsupervised	Tang and Yin, 2005
	supervised	Keyan and Tingting, 2011; Luo, 2014; Lv et al., 2008; Michalak and Korczak, 2011; Savage et al., 2016; Zhang and Trubey, 2019
Community detection	unsupervised	Li et al., 2017; Soltani et al., 2016; Wagner, 2019; Yunkai et al., 2006
Feature-based risk modeling	unsupervised	Yang et al., 2014
	supervised	Cao et al., 2017

Previous studies on money laundering in cryptocurrencies are scarce and inconclusive due to a lack of labels for evaluation. Some conclude that supervised models perform well (Bartoletti et al., 2018; Hu et al., 2019; Monamo et al., 2016b) in highly imbalanced data with very few positive cases (Bartoletti et al., 2018; Monamo et al., 2016b). Others report low detection rates for unsupervised methods in similar conditions (Monamo et al., 2016a; Monamo et al., 2016b; Pham and Lee, 2016a; Pham and Lee, 2016b; Wu et al., 2020). Often, the evaluation of anomaly detection methods consists of checking whether the detected anomalies indeed represent extreme cases (Pham and Lee, 2016a; Pham and Lee, 2016b) or behaviour deemed suspicious by human analysts (Hirshman et al., 2013).

AL has been proposed as a method to reduce the number of labels needed for the training of an effective classifier by iteratively sampling the most informative samples for labeling from an initially unlabeled pool (Settles, 2009). Given the apparent label scarcity in money laundering data, it is a highly relevant setting for the practical implementation of ML-based AML systems. Previously, Deng et al. (2009) applied AL to detect money laundering in financial transactions. In an account-level classification of 92 real-life accounts, they report that their method can accurately estimate the threshold hyperplane with only 22% of the labels. AL has also successfully been applied in other fraud-related use-cases such as credit card fraud (Carcillo et al., 2018) and network intrusion detection (Almgren and Jonsson, 2004; Görnitz et al., 2009; Stokes et al., 2008), reporting the sufficiency of as few as 1.5% of the original labels to

achieve near-optimal performance (Deng et al., 2009).

In this study, experiments with AL are conducted, assuming an unlabeled dataset and the capacity to acquire labels progressively to train a supervised classifier. This extends the study by Deng et al. (2009) to a transaction-level analysis in a much larger cryptocurrency dataset.

Chapter 3

Methods

This section provides relevant theoretical background on the ML methods applied in this research. ML defines a broad class of algorithms that learn from data and is based on concepts from computer science, statistics and information theory (Alpaydin, 2010). Supervised learning, unsupervised learning and AL are all sub-fields of ML and further outlined in the following sections.

3.1 Supervised Learning

Supervised learning describes all ML algorithms that are trained on data points which have been labeled with their target value (labeled data). In the context of money laundering detection, the label of each transaction could indicate whether it was identified as money laundering or not. Labels can be obtained through the judgement of a human (usually a domain expert) about the unlabeled data point. They can also be obtained or approximated through the consolidation of other information sources. In the case of money laundering, if the true label of a transaction is unavailable, the label could for example be approximated through information on previous criminal conduct of the initiating entity of each transaction (e.g. criminal records).

While some studies exist that apply supervised ML algorithms in the AML context, as discussed in Chapter 2, the availability of large-scale, labeled datasets to train a supervised classifier is unrealistic in a real-life setting. However, the inclusion of supervised baselines in this research allows to determine the trade-off between the number of available labels (equivalent to labeling efforts in a real-life setting) and the classifiers' performance.

In the following, the three supervised classification algorithms used in the underlying study, namely Logistic Regression (LR), Random Forest (RF) and XGBoost are

described. LR and RF were chosen because they were used in the paper presented by Elliptic and thus allow for a comparison with other results on the dataset, and a seamless extension of previous research. XGBoost is an implementation of the Gradient Boosting Machine algorithm (GBM) , which is frequently applied in client scenarios at Feedzai and thus represents a relevant benchmark algorithm for the company.

3.1.1 Logistic Regression

In the following, LR is described based on Hilbe (2009). LR is a multiple regression suitable for binary classification, which assesses the relationship between the binary dependent variable (target) and a set of independent categorical or continuous variables (predictors). LR calculates the odds for an event, which are defined as the ratio between the probability that an event will occur and the probability that the event will not occur. Hence, the odds are equal to $\frac{p}{1-p}$, where p is the probability of the event based on the predictor values. In LR for binary classification, the event represents the belonging of an instance to class 1.

However, in order for the target value to fall between 0 and 1 and thus being suitable for binary classification, the odds are transformed into log odds, using the natural logarithm. LR then models the natural log odds of belonging to class 1 as a linear function of the independent variables. The log of the odds of an event x are called $logit(x)$. Thus, LR is an ordinary regression where the logit of the target is defined as the response variable. In this way, by assuming a linear relationship between the independent variables and the logit of belonging to class 1, LR is able to model a previously nonlinear relationship between independent variables and the target in a linear way.

$$logit(p) = ln(odds) = ln(\frac{p}{1-p}) = \alpha + \beta_1 x_1 + \ldots + \beta_k x_k \tag{1}$$

From equation 1, the equation for the prediction of the probability of an instance to belong to class 1, based on the predictors, is given as

$$p = \frac{e^{\alpha+\beta_1 x_1+\ldots+\beta_k x_k}}{1+e^{\alpha+\beta_1 x_1+\ldots+\beta_k x_k}} = \frac{1}{1+e^{-\alpha+\beta_1 x_1+\ldots+\beta_k x_k}} \tag{2}$$

The coefficients of the regression are estimated using Maximum Likelihood Estimation. With this technique, the model coefficients are iteratively altered in such a way that the probability of achieving the observed target value given the observed predictors is maximized over all observations. The likelihood L can be written as

$$L = \prod_{i=1}^{n} p_i^{y_i}(1-p_i)^{1-y_i} \tag{3}$$

Through Maximum Likelihood Estimation, LR will estimate values for α and β that maximize L.

3.1.2 Random Forest

RF is an ensemble learning technique, combining a (usually large) number of simple Decision Trees (DT) to optimize the overall prediction capabilities of the model. DT's are classifiers represented as acrylic graphs with a root node and successive child nodes, which are connected through directed edges (Shalev-Shwartz and Ben-David, 2014, pp. 250–255). At each node a decision is made based on the attribute values of an instance. In this way, an instance traverses down the a hierarchical decision scheme within the tree until reaching a leaf node. Then, a final classification for the instance is returned.

A DT is trained by choosing, at each node, the feature constraint that best splits the instances in the train set into further subsets. The quality of each possible split is determined based on a certain metric, the splitting criterion. Different splitting criteria have been proposed. Usually, they aim at an increasing homogenity of the target value within the subsets that are created at the respective split. Thus, in a classification task, the aim is to choose splits that increasingly separate the distinct classes from each other. Based on the RF implementation used in this research, the gini impurity (or gini index), proposed by Breiman et al. (1984), is used as a splitting criterion. The gini impurity measures the probability of misclassifying an arbitrary instance if it was randomly classified based on the class distribution in the dataset. The quality of a feature constraint as a split is determined by weighting the impurities of each branch by the number of elements that are part of the resulting subset. The best split is then chosen by maximinizing the gini gain, which is calculated by substracting the weighted impurities from the impurity of the dataset before the split.

The gini impurity G can be written as

$$G = \sum_{i=1}^{n} p_i(1 - p_i) = 1 - \sum_{i=1}^{n} p_i^2 \tag{4}$$

where n is the number of classes and p_i is the probability of randomly selecting an element of class i, thus the fraction of records of class i in the subset. A gini impurity of 0 indicates that the feature constraint leads to pure, i.e. one-class-only subsets of instances and is thus the perfect case (Breiman et al., 1984).

In RF, as introduced by Breiman (2001), the ensemble method bootstrap aggregation (bagging) is applied to sample random subsets of the training data for each DT. Ensemble methods aim at improving predictive accuracy of a classifier by combining multiple individual weak learners. According to the bagging principle, for each individual DT, the train set composes of N' data points that are uniformly sampled from the train set with replacement, where N is the size of the original train set and $N' < N$. In a classification task, the final prediction is then determined by merging the individual DTs in a voting scheme to achieve a more accurate and stable prediction. Results are thus more robust.

Besides bagging, RF also selects a random subset of features to be considered at each node in order to lower the risk of overfitting on specific features.

3.1.3 XGBoost

XGBoost, short for Extreme Gradient Boosting, is a distributed implementation of GBM with optimized performance, designed to be efficient, flexible and portable (Chen and Guestrin, 2016). GBM was originally proposed by Friedman (2001) and is based on the boosting concept. Boosting, like bagging, is an ensemble method, introduced by Valiant (1984). Contrary to bagging, where individual models are combined in a voting scheme, in boosting the K individual weak base learners are trained in a gradual, additive and sequential manner. Each individual classifier is trained with the goal to improve on the weaknesses of the previous models by focusing on areas where the preceding base learners did not classify appropriately. The individual models are subsequently added to determine the final prediction. Thus,

$$\hat{y}_i = \sum_{k=1}^{K} f_k(x_i) \tag{5}$$

where $\hat{y}_i$ is the predicted value for the $i-th$ observation and f_k represents the $k-th$ DT in the ensemble.

GBM uses DT's, which are described in section 3.1.2 as a base learner. In GBM, the DT base learners are commonly constrained, e.g. by restricting the maximum number of layers, nodes, splits or leaf nodes.

GBM reduces the loss by following the (negative) gradient of the loss function and thus represents a gradient descent procedure. While in the traditional gradient descent algorithm, *model parameters* are optimized based on the loss function gradient, in GBM, a set of *sequential models* are trained in order to follow the current gradient of the loss function.

The GBM algorithm, as described by Hastie et al. (2009), works as follows. First, an initial DT is trained, based on the target value y_i, such that

$$f_0(x) = argmin \sum_{i=1}^{n} L(y_i, \gamma) \tag{6}$$

where L is the loss function and γ represents an initial function.

Then, at each iteration, so called pseudo residuals r_{im} are computed, representing the negative gradient of the loss function (the direction of the steepest descent) of the previous model.

$$r_{im} = -[\frac{\delta L(y_i, f(x_i))}{\delta f(x_i)}]_{f=f_{m-1}} \tag{7}$$

The new DT model is then trained on the pseudo residuals, instead of the true target value of the classification task, as a target. In this way, the shortcomings of the

previous models can be improved. Finally, the model is updated by adding the output of the new DT to the sum of the outputs of the subsequent DTs in order to correct and improve on the final prediction of the model and reduce the overall loss:

$$f_M(x) = f_{m-1}(x) + f_m(x) \tag{8}$$

This procedure is repeated until a fixed number of base learners is reached, the loss reaches a desired level or there are no further improvements on a validation dataset.

In comparison to traditional GBM, XGB uses additional techniques to improve the performance of the algorithm, such as weight shrinkage, increased regularization and parallel execution (Chen and Guestrin, 2016).

3.2 Unsupervised Learning

In contrast to supervised learning, unsupervised learning describes all ML algorithms which are trained on data points where the target value is unknown (unlabeled data) (Alpaydin, 2010).

Anomaly detection methods are unsupervised learning techniques to detect outliers in a dataset. Literature suggests their effectiveness in the AML context (Chapter 2). While this section focuses on the anomaly detection methods applied in this research, Chandola et al. (2009) and Domingues et al. (2018) provide a broader review of anomaly detection.

The standard definition of outliers refers to instances that are unlikely to be drawn from the same distribution as the train data or instances that are far from other data points in the feature space. Although this research focuses mainly on unsupervised anomaly detection, some methods are semi-supervised discriminators trained to learn a boundary around normal instances. In that context, outliers are instances that fall outside of the boundary (Gao, 2009).

Aiming at a diversity of strategies, in this research, seven common anomaly detection algorithms with available Python implementations are tested: Local Outlier Factor (LOF), K-Nearest Neighbours (KNN), Principal Component Analysis (PCA), One-Class Support Vector Machine (OCSVM), Cluster-Based Outlier Factor (CBLOF), Angle-Based Outlier Detection (ABOD), and Isolation Forest (IF).

LOF (Breunig et al., 2000) and KNN (Upadhyaya and Singh, 2012) start by computing the distance (in this study the Euclidean distance) of each instance to its k nearest-neighbour. Then, KNN defines that distance as the outlier score. LOF uses the distance as the instance's density, and if the density is substantially lower than the average density of its k nearest-neighbours, the instance is declared anomalous. ABOD (Kriegel et al., 2008) computes the pairwise cosine similarities between each point and its k nearest neighbours. Then, those data points with a low average radius and variance, and thus a farther distance to other data points, are classified as anomalies.

PCA (Shyu et al., 2003) and OCSVM (Schölkopf et al., 2001) define anomalies as observations that deviate from normal behaviour. OCSVM computes the maximal margin hyperplane that best separates the non-anomalous training data from the origin in the feature space, given that the feature values are standardized. Anomalies are data points that fall outside the boundaries of the hyperplane. PCA detects anomalous instances as observations with a large distance to the principal components of non-anomalous observations.

CBLOF (He et al., 2003) uses the outcome of a clustering algorithm on the instances and classifies each cluster as either *small* or *large*. It calculates an anomaly score for each instance, marking instances that belong to small clusters or that are far from big clusters as anomalous. In this research, K-Means is used as the clustering algorithm.

Lastly, IF (Liu et al., 2008a) isolates anomalies by performing recursive random splits on attribute values. Based on the resulting tree structure, anomalies are instances that are easy to isolate, i.e. have shorter paths from the root node.

3.3 Active Learning

AL is an incremental learning approach that iteratively queries instances for labeling (e.g. by human analysts) and uses the increasing number of labeled instances to retrain a supervised model. It fits the AML context by addressing label scarcity and has previously been successfully applied to detect money laundering accounts based on financial transaction history, as presented in Chapter 2. An extensive survey on AL can be found in Settles (2009).

The goal of AL is to minimize the number of labels necessary to achieve adequate classifier performance. The process is depicted in Figure 1. It starts with an entirely unlabeled train set, the *unlabeled pool*, although sometimes there is a residual number of labels. At each iteration, a *query strategy* queries a batch of instances for manual labeling by human analysts. After labeling, the instances go into the *labeled pool*. Finally, a supervised algorithm (the *classifier*) is trained on the labeled pool and evaluated on a test set. If the performance is not satisfactory, the querying process continues to enrich the labeled pool incrementally. To mimic the manual labeling process in this study, the labels are appended to the queried instances.

In the literature, query strategies build on various models and uncertainty criteria. This study focuses on four query strategies trained on the labeled pool to find informative instances in the unlabeled pool. Two of them, Uncertainty Sampling and Expected Model Change, are supervised, requiring an underlying supervised model to define queries. The other two, Elliptic Envelope and Isolation Forest (IF), are unsupervised and find outlying instances with regards to the labeled pool, aiming at a heterogeneous set of instances in the labeled pool for classifier training.

Expected Model Change (Settles, 2009; Settles et al., 2008) assumes that instances

are more informative if they influence the model more strongly. It queries the unlabeled instances that lead to the most significant change in the model parameters by measuring the impact of labeling one unlabeled instance on the gradient of the model's loss function. Thus, this strategy applies only to gradient-based classifiers. In this study, LR is used. The expected model change is a weighted sum over all possible labels since the labels of the instances are unknown before querying. Then, at each iteration, the labels of the instances with the largest expected gradients are queried.

Uncertainty Sampling is one of the most commonly used query strategies (Lewis and Catlett, 1994; Settles, 2009). It queries the instances about which a model is most uncertain. Assuming a probabilistic learning model and a binary classification problem, this translates to querying the instances with predicted scores that are closest to 0.5. In this study, the same type of classifier is used for Uncertainty Sampling and for the evaluation on the test set; for instance, if the classifier is RF, Uncertainty Sampling is also conducted using RF.

The two unsupervised query strategies, IF and Elliptic Envelope represent anomaly detection techniques. Outliers are hereby transactions with high anomaly scores (IF) or a large Mahalanobis distance to a multivariate Gaussian distribution fit on the labeled pool (Elliptic Envelope).

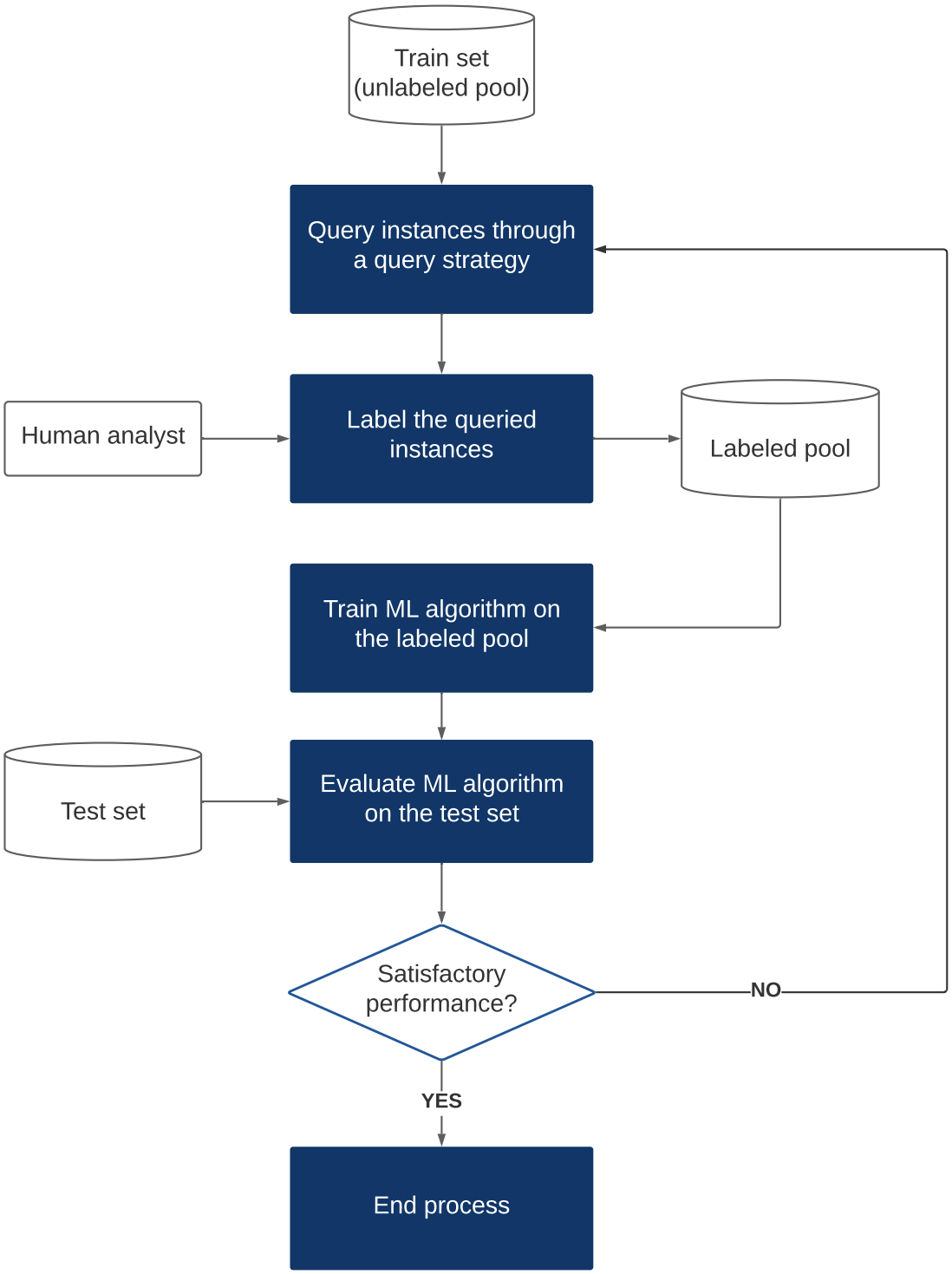

Figure 1: Flowchart of the AL process.

Chapter 4

Experimental Setup

The following chapter outlines the experimental setup of the study. First, the dataset used in the experiments is described. Then, the evaluation method and implementations of the supervised baselines, anomaly detection methods and AL are specified.

4.1 Data Description

For this study, the Bitcoin dataset[1] released by Elliptic (Weber et al., 2019) is used. It includes 49 graphs sampled from the Bitcoin blockchain at different sequential moments in time (time-steps), as presented in Figure 2. Each graph is a directed acyclic graph, starting from one transaction, and including subsequent related transactions on the blockchain, containing approximately two weeks of data. There are no connections between the graphs of different time-steps.

Bitcoin transactions are transfers from one Bitcoin address (e.g. a person or company) to another, represented as nodes in the graph. Each transaction consumes the output of past transactions and generates outputs that can be spent by future transactions. The edges in the graph represent the flow of Bitcoins between transactions. The dataset consists of 203,769 transactions, of which 46,564 are labeled - 90% of them as licit and 10% as illicit. The remaining transactions are unlabeled and not considered in this research. The labels are based on the *category* that the Bitcoin address that created the transaction is associated with. Illicit categories include scams, malware, terrorist organizations, and Ponzi schemes. Licit categories include exchanges, wallet providers, miners, and licit services. Each transaction has 166 features, 94 of which represent information about the transaction itself. The remaining features were constructed

[1] Available at https://www.kaggle.com/ellipticco/elliptic-data-set

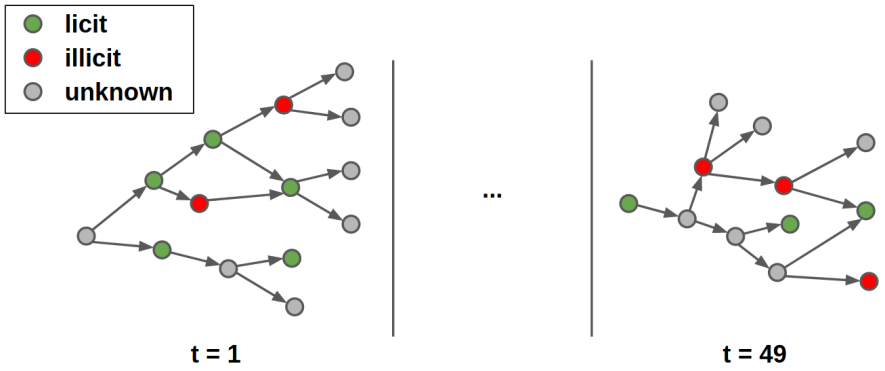

Figure 2: Structure of the dataset over timesteps t, taken from Bellei (2019). Each node represents a Bitcoin transaction and each edge represents a flow of Bitcoins. The node color indicates the class of the respective data point.

by Weber et al. (2019) using information one-hop backward/forward from the transaction, such as the minimum, maximum, and standard deviation of each transaction feature. All features, except for the time-step, are fully anonymized and standardized with zero mean and unit variance.

4.2 Methods

In this section, the experimental setup of the methods previously detailed in Chapter 3 is described. As in Weber et al. (2019), the data is split into sequential train and test datasets for all experiments. The train set includes all labeled samples up to the 34th time-step (29,894 transactions), and the test set includes all labeled samples from the 35th time-step, inclusive, onward (16,670 transactions).

All methods are evaluated using the F1-score for the illicit class, hereafter referred to as illicit F1-score. The illicit F1-score represents the harmonic mean of Recall and Precision and is thus calculated as

$$\text{F1} = 2 * \frac{(\text{Precision} * \text{Recall})}{(\text{Precision} + \text{Recall})} \tag{9}$$

where

$$\text{Recall} = \frac{\text{True positives}}{(\text{True positives} + \text{False negatives})} \tag{10}$$

and

$$\text{Precision} = \frac{\text{True positives}}{(\text{True positive} + \text{False positives})} \tag{11}$$

The choice of the illicit F1-score as the performance metric is motivated by multiple factors. First, the metric takes into account both false positives and false negatives and is thus suitable for Feedzai's AML business case, in which both cases of misclassified instances are highly relevant because each implies a significant cost for the company. While false negatives could result in the non-reporting of a money laundering case

and thus potentially lead to high fines for Feedzai's client, each false positive usually implies an unnecessary, yet very costly review of the respective transaction by a human analyst. Second, the F1-score addresses the strong class imbalance towards the negative class in the data set, as it considers both Precision and Recall. Last, the illicit F1-score was used by Weber et al. (2019) as well, which facilitates a comparison with previous results on the dataset.

4.2.1 Supervised Baselines

For the supervised baselines, the scikit-learn (Pedregosa et al., 2011) implementation of LR and RF as well as the Python implementation of XGBoost (Chen and Guestrin, 2016) are used. The presented results are achieved using default hyperparameters. This ensures the fairest comparison with the unsupervised anomaly detection methods and AL, where hyperparameter tuning is either impossible (anomaly detection methods) or computationally extremely costly, as hyperparameter tuning would need to be re-done after each querying step (AL). Each supervised model is trained on the train set using all 166 features and then evaluated on the test set. To measure performance over time, and following Weber et al. (2019), the illicit F1-score per time-step in the test set is reported.

4.2.2 Anomaly Detection Methods

For the experiments with anomaly detection methods, the PyOD package implementations (Zhao et al., 2019) of LOF, KNN, PCA, OCSVM, CBLOF, ABOD and IF with default hyperparameters are used. Using default hyperparameters makes sense because hyperparameters of unsupervised ML methods can hardly be tuned, given the inavailability of labels to optimize on. The introduced methods use different notions of anomaly scores and different scales, although generally, the anomaly score should be higher for more anomalous instances. Thus, in order to fairly compare the methods, they are evaluated at different *contamination levels*. The contamination level defines the expected proportion of outliers in the dataset, and is used to set a threshold on the decision function. Whereas the original PyOD implementation applies the contamination level on the scores of the train set, in this study it is applied on the test set scores to guarantee that the desired percentage of positive cases (anomalies) in the test set is the same across methods. The contamination level here is analogous to a fixed alert rate in real AML systems, i.e. the percentage of cases flagged for further investigation by an analyst. The illicit F1-score for each model is evaluated at contamination levels between 0 and 1, with increments of 0.05. For comparison the illicit F1-score of the RF supervised baseline, where the model threshold is defined by setting the contamination level as the predicted positive rate, is also presented.

Given the inability to optimize hyperparameters, unsupervised anomaly detection methods that are more robust to their hyperparameter settings will be able to perform

more reliably and consistenly when applied with default hyperparameters to different client datasets in a real-life setting. The labels that are available in the Bitcoin dataset used in this project allow to study the effects of different hyperparameter settings on each classifier's performance and thus, to evaluate their suitability for a real-life AML system. Thus, for all methods except PCA, which does not have any hyperparameters, the illicit F1-score on the test set at different hyperparameter settings is studied. The contamination level is hereby fixed at 0.1 (10%), which is equal to the overall percentage of outliers in the dataset. Table 2 presents the hyperparameter values used for experimentation for each anomaly detection method. All possible combinations of the presented values were tested.

Table 2: Hyperparameter values tested during hyperparameter tuning for each anomaly detection method. Default values in **bold**.

Method	Hyperparameter	Values
ABOD	n_neighbors	3, **5**, 7, 20, 50, 100, 200
CBLOF	n_clusters	3, 4, 5, **8**, 10, 15, 20
	use_weights	True, **False**
	alpha	0.5, 0.7, **0.9**
IF	n_estimators	5, 50, **100**, 300, 800
	max_samples	**'auto'**, 100, 300
	max_features	0.1, 0.2, 0.4, 0.7, **1**
	bootstrap	True, **False**
KNN	n_neighbors	3, **5**, 7, 20, 50, 100, 200
	method	**'largest'**, 'mean', 'median'
LOF	n_neighbors	3, 5, 7, **20**, 50, 100, 200
OCSVM	nu	0.01, 0.1, **0.5**
	gamma	**'auto'**, 0.01, 0.1, 1
	shrinking	**True**, False
	tol	**0.001**, 0.01, 0.1

4.2.3 Active Learning

In this study, the unsupervised and supervised query strategies described in Chapter 3 are combined, depending on the number of illicit instances in the labeled pool. After an initial random sample of one batch of instances, an unsupervised *warm-up learner* is applied, sampling instances until the labeled pool includes at least one illicit instance. When this threshold is reached, the process either switches to a supervised *hot learner* or continues to use the warm-up learner. The unsupervised warm-up learners are Elliptic Envelope and IF. The supervised hot learners are Expected Model Change and Uncertainty Sampling. As the classifier, the three supervised models RF, XGBoost, and LR are used, enabling a direct comparison to the supervised baselines. All AL setups are compared against a baseline that queries random instances at each iteration, called random sampling (RS). The batch size is set to 50 instances sampled at each iteration

for all experiments. This batch size is large enough to be computationally efficient, yet small enough to provide a detailed view on the performance difference at increasing labeled pool sizes. Each AL setup is run five times with different random seeds to ensure the robustness of the results. The performance of each AL setup is assessed through the median illicit F1-score and the confidence intervals at each labeled pool size. The setup and parameter choices follow the work by Barata et al. (2020) which was developed at Feedzai concurrently.

Chapter 5

Results

This section describes and discusses the results of the supervised baselines, the anomaly detection methods and AL.

5.1 Supervised Baselines

The supervised baselines closely match the results presented by Weber et al. (2019). Over five runs with different random seeds, the illicit F1-score on the test set is 0.45 for LR, 0.76 for XGBoost, and 0.83 for RF. Thus, the best supervised baseline is achieved with the RF model. As recognized by Weber et al. (2019), model performance is profoundly affected by a sudden dark market shutdown at time-step 43.

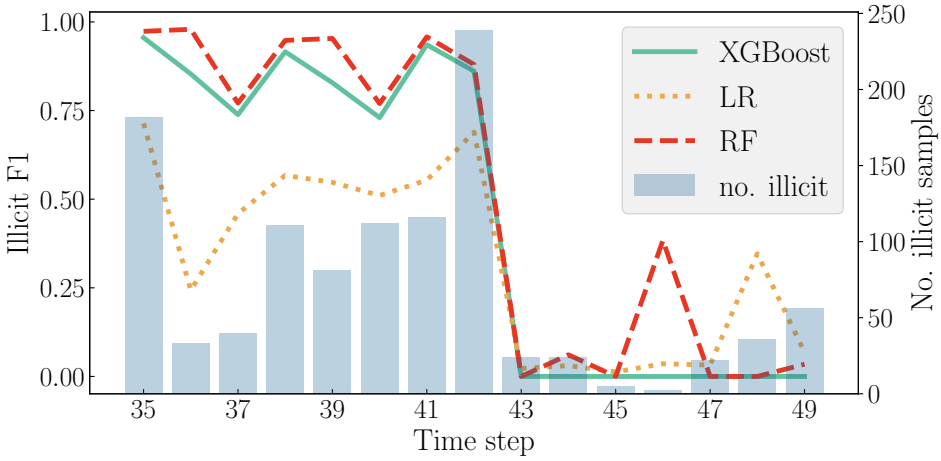

Figure 3: Illicit F1-score for each supervised baseline and number (no.) of illicit samples across time-steps in the test set.

5.2 Anomaly Detection Methods

Table 3 presents the illicit F1-score of the anomaly detection methods. The results are compared against the RF supervised baseline at different contamination levels.

Table 3: Illicit F1-score for all anomaly detection methods by contamination level (RF supervised baseline for reference).

Model	**Contamination level**			
	0.05	**0.1**	**0.15**	**0.2**
RF supervised baseline	0.82	0.58	0.46	0.39
LOF	0.11	0.15	0.19	0.18
ABOD	0.07	0.07	0.07	0.07
KNN	0.03	0.04	0.05	0.06
OCSVM	0.01	0.03	0.03	0.04
CBLOF	0.01	0.02	0.03	0.04
PCA	0.01	0.01	0.02	0.02
IF	0.00	0.00	0.00	0.01

Anomaly detection methods perform significantly below the RF supervised baseline across all contamination levels. Figure 4 allows to understand the low illicit F1-scores of the anomaly detection methods better, by comparing the true positive rate (TPR) with the false positive rate (FPR) at increasing contamination levels.

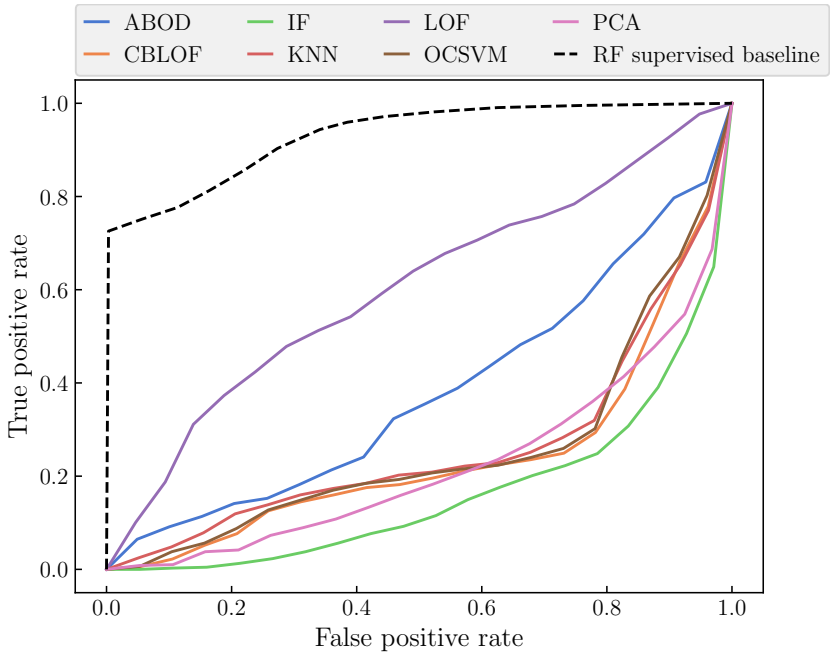

Figure 4: TPR and FPR per contamination level (increasing from left to right) per anomaly detection method (RF supervised baseline for reference).

Figure 4 shows that, with an increase in the contamination level, and thus a decreasing classification threshold on the anomaly scores returned by the models, the

FPR for most anomaly detection methods increases strongly, while the TPR only increases very slowly for a long time. Only at the highest contamination levels, the TPR increases strongly as well. This indicates that the anomaly scores are assigned the wrong way around, with the higher scores mainly assigned to the licit, rather than the illicit instances. However, as mentioned in Section 4.2.2, more anomalous instances (which were assumed to be the illicit instances) should receive higher anomaly scores.

As presented in Figure 5, the hyperparameter robustness test reveals that the low classifier performance does not seem to be due to a potentially unsuitable default hyperparameter setting, as no other setting in the study leads to a significant performance increase. The methods generally react to changes in hyperparameter settings to differing degrees, with the performance of LOF and OCSVM being much more volatile than that of the other methods. However, given the overall very low illicit F1-score ranges in the experiment, the validity of the results is questionable.

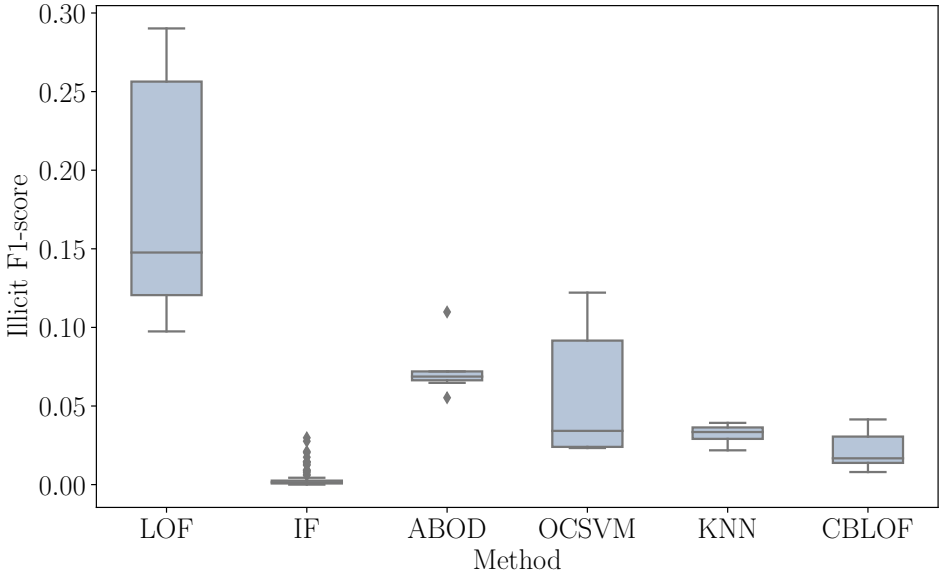

Figure 5: Illicit F1-scores for the tested hyperparameter settings per anomaly detection method.

The low classification performance of the anomaly detection methods is not consistent with past studies, where anomaly detection methods perform adequately for AML (Section 2). However, these studies mostly use synthetically generated anomalous data points that are outliers by design. Furthermore, there could be differences between money laundering patterns in financial transactions and Bitcoin transfers. In the real-life Bitcoin transaction dataset, we see that illicit cases are indeed not outlying.

To illustrate this, Uniform Manifold Approximation and Projection (UMAP) (McInnes et al., 2018) is used. UMAP is a dimension reduction technique which can be used to visualize data in a lower dimensional space. It models data points by searching for a low dimensional data projection that most closely resembles the fuzzy topological

structure of the original data. UMAP is applied to the test set. Then, two plots are constructed based on the resulting two-dimensional projection of the data points. In the first (Figure 6), each observation is colored based on the predicted label of the worst-performing method (IF), while in the second (Figure 7), each observation is colored based on the true label.

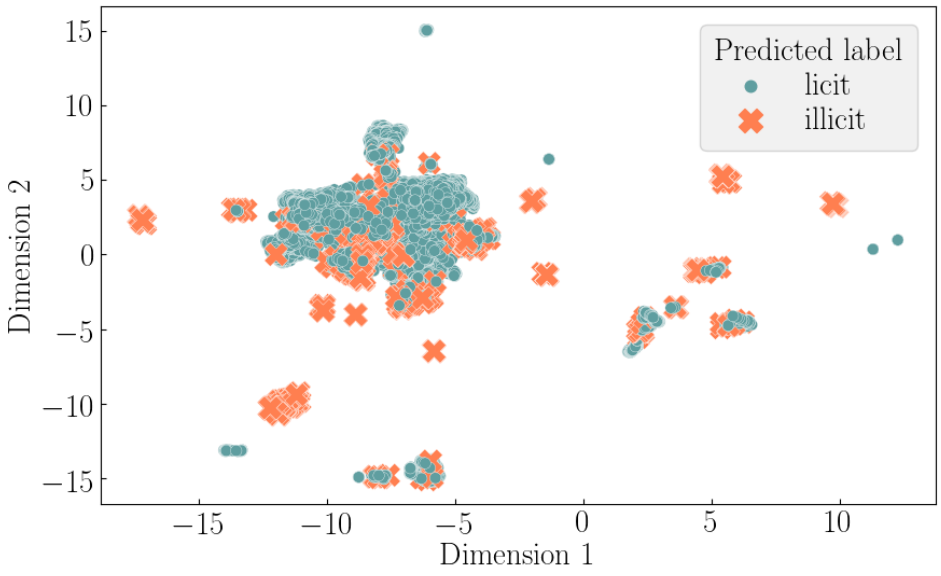

Figure 6: UMAP projection of the test set into a two-dimensional space. Instances are colored by the labels predicted by IF.

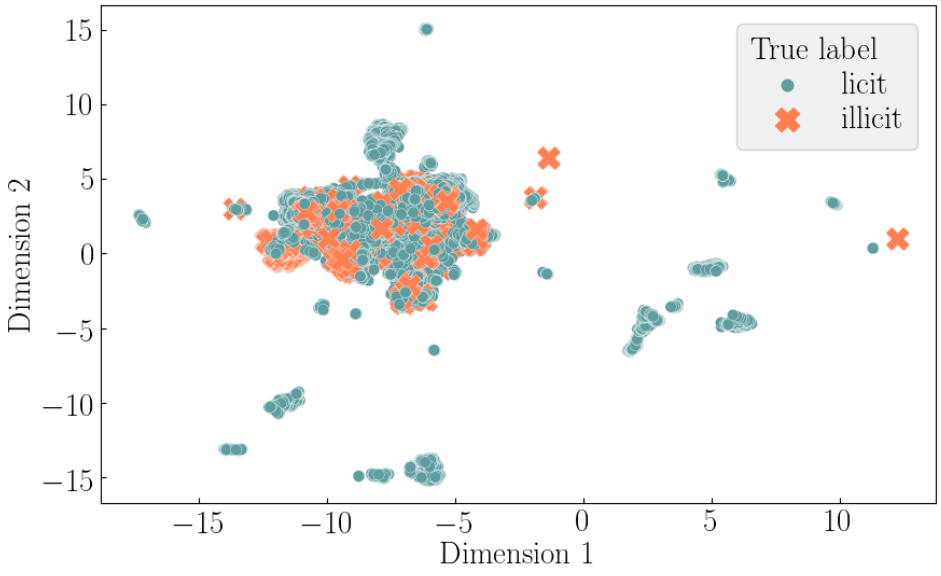

Figure 7: UMAP projection of the test set into a two-dimensional space. Instances are colored by their true labels.

The IF classifies most outlying instances as illicit, as expected from an anomaly

detection method. However, the true labels presented in Figure 7 reveal that the illicit instances in the dataset are not actually outlying, but instead hidden among licit transactions.

The observation that not all outliers are illicit and that not all illicit transactions are outliers is reasonable in AML, as sophisticated criminals obfuscate their activity by mimicking normal behaviour, hiding in regions of high licit density. This problem was previously acknowledged by Das et al. (2016). Thus, it can be concluded that anomaly detection methods are ineffective for the unsupervised classification task in this real-life Bitcoin dataset.

5.3 Active Learning

Table 4 summarises the results of the AL benchmark for each of the three different classifiers.

Table 4: Average illicit F1-score over five runs for each AL setup. The results are compared to each classifier's respective supervised baseline (Section 5.1). Results are ordered by the illicit F1-score with 3000 labels. Best values for each labeled pool size across classifiers are highlighted in **bold**.

Query strategies		Classifier	Labeled pool size					Supervised baseline
Warm-up learner	Hot learner		200 (0.7%)	500 (1.7%)	1000 (3.3%)	1500 (5%)	3000 (10%)	
Isolation Forest	Uncertainty Sampling	RF	0.75	0.75	0.80	**0.82**	**0.83**	0.83
Random sampling	Uncertainty Sampling		0.73	0.75	**0.81**	**0.82**	0.82	
Elliptic Envelope	Uncertainty Sampling		0.65	**0.77**	0.80	**0.82**	0.82	
Isolation Forest	Expected Model Change		0.56	0.61	0.77	0.79	0.81	
Random sampling	Expected Model Change		**0.76**	**0.77**	0.78	0.78	0.81	
Elliptic Envelope	Expected Model Change		0.60	0.72	0.76	0.77	0.81	
Random sampling	–		0.74	0.76	0.76	0.78	0.80	
Elliptic Envelope	–		0.50	0.53	0.56	0.65	0.70	
Isolation Forest	–		0.67	0.65	0.59	0.63	0.62	
Isolation Forest	Uncertainty Sampling	XGBoost	0.67	**0.77**	0.80	0.79	0.80	0.76
Elliptic Envelope	Expected Model Change		0.65	0.75	0.77	0.75	0.79	
Random sampling	Expected Model Change		0.70	0.75	0.79	0.80	0.78	
Isolation Forest	Expected Model Change		0.60	0.75	0.77	0.76	0.75	
Elliptic Envelope	–		0.53	0.64	0.53	0.61	0.68	
Elliptic Envelope	Uncertainty Sampling		0.62	0.62	0.64	0.80	0.64	
Random sampling	Uncertainty Sampling		0.72	0.76	0.64	0.60	0.64	
Random sampling	–		0.66	0.58	0.75	0.74	0.59	
Isolation Forest	–		0.38	0.38	0.46	0.44	0.57	
Isolation Forest	Expected Model Change	LR	0.22	0.59	0.63	0.66	0.62	0.45
Elliptic Envelope	Expected Model Change		0.20	0.48	0.61	0.61	0.61	
Random sampling	Expected Model Change		0.44	0.54	0.58	0.64	0.60	
Elliptic Envelope	Uncertainty Sampling		0.41	0.52	0.63	0.63	0.60	
Isolation Forest	Uncertainty Sampling		0.37	0.53	0.61	0.60	0.58	
Random sampling	Uncertainty Sampling		0.40	0.50	0.57	0.58	0.55	
Random sampling	–		0.36	0.36	0.36	0.37	0.39	
Elliptic Envelope	–		0.28	0.25	0.24	0.24	0.22	
Isolation Forest	–		0.25	0.24	0.29	0.21	0.02	

Switching to a supervised hot-learner significantly improves performance over the continued use of an unsupervised warm-up learner. Among hot-learners, however, there is no clear best policy. Furthermore, RS as the warm-up learner leads to a faster improvement in model performance (i.e. better performance for smaller labeled pool

sizes) compared to using the unsupervised anomaly detection methods IF or Elliptic Envelope as the warm-up learner. This observation aligns with previous considerations on the poor performance of anomaly detection methods (Section 5.2). Since these methods fail to detect illicit instances, they are ineffective at querying illicit instances to be added to the labeled pool to improve the performance of a supervised classifier quickly. Elliptic Envelope, however, performs above IF, which is also consistent with previous results using anomaly detection methods (Section 5.2), where IF proved to be the worst-performing method.

Figure 8 shows the performance over time of the best AL setup for the three classifiers tested. For comparison, it also includes the performance achieved by the best supervised baseline, the RF supervised baseline, trained on the entire train set. With the presented AL setup, all classifiers stabilize after 1000 labels, with RF and XGBoost exhibiting faster performance increase. RF reaches its baseline's performance with only 5% of the original labels, or 1500 out of the original 30000 labels (Figure 8). Near-optimal performance can be achieved with as few as 500 labels.

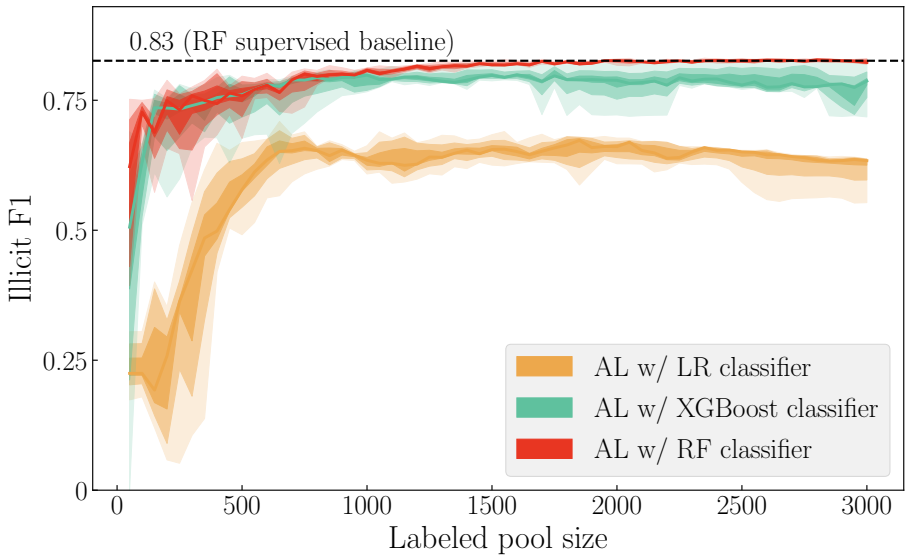

Figure 8: Median illicit F1-score of the best AL setups for each classifier over five seeds across small labeled pool sizes. Bands indicate the percentile range 0-100. RF supervised baseline for reference.

Table 4 additionally shows that XGBoost and LR temporarily surpass their supervised baseline of 0.76 and 0.45, respectively, i.e. the performance they achieved when trained on the entire train set (Section 5.1). Figure 9 highlights that the classifiers perform better when trained only on a sample of the labeled data but eventually converge to their supervised baseline as the labeled pool increases over time.

This can be due to the fact that the labeled pool consists of the most relevant samples in the beginning of the AL process, while later on, increasingly less representative instances are available to be queried and added to the labeled pool. At the same time,

the class imbalance is lower in the early stages of the AL process and increases over time until the labeled pool ultimately reaches the distribution of the train set, as all instances are included. Laws and Schätze (2008) acknowledge that, in some cases, early stopping of an AL process might prevent this model degradation.

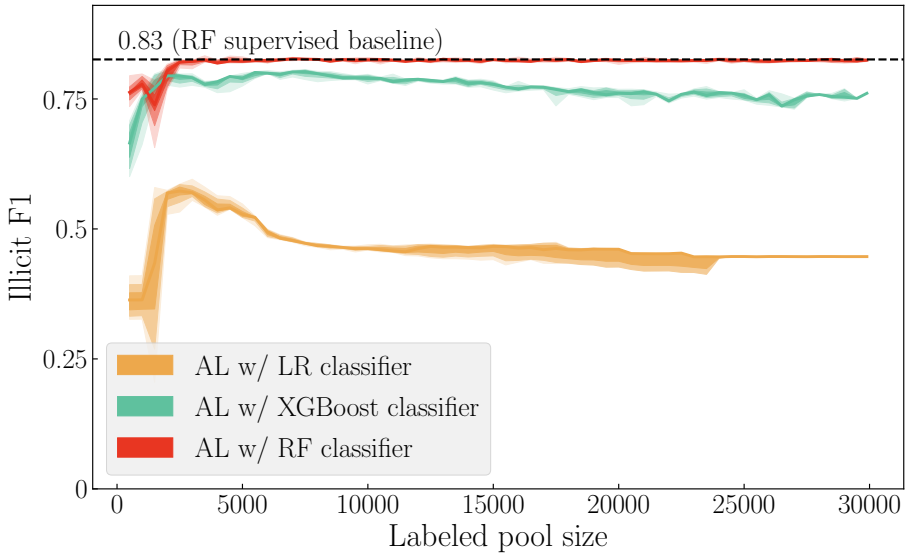

Figure 9: Median illicit F1-score of the best AL setups for each classifier over five seeds across all labeled pool sizes. Bands indicate the percentile range 0-100. RF supervised baseline for reference.

While the AL results seem promising, Table 4 shows that the RS baseline achieves a similar performance to the more sophisticated AL strategies. An intuitive explanation is that the classifier will start approaching good performances when the labeled pool includes a sufficient number of illicit instances and, with approximately 10% of illicit cases in the dataset, even RS can quickly reach that sufficient number.

In reality, financial crime is extremely rare among licit transactions, and thus datasets are much more imbalanced (Sudjianto et al., 2010). Since this research is focused on the practical relevance of AL, the best performing AL setup is compared against RS in a dataset with a higher, more realistic class imbalance. Specifically, random undersampling is applied to the minority class of the dataset to achieve illicit rates of 2% and 0.5%. The results are plotted in Figures 10 and 11, respectively. For comparison, the RF supervised baseline, trained and evaluated on the dataset with the respective reduced fraud rates, is indicated.

As expected, the AL query strategies increasingly outperform RS as the dataset imbalance increases. This proves the superiority of sophisticated AL approaches over RS in a real-life setting with highly imbalanced datasets. For both highly imbalanced datasets, the best setup uses RS (warm-up) followed by Uncertainty Sampling (hot learner).

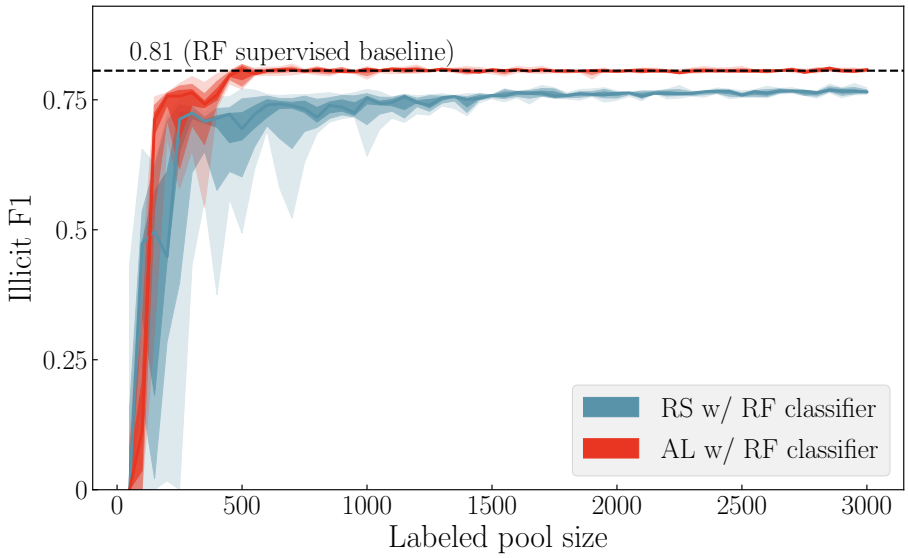

Figure 10: Median illicit F1-score of the best AL setup compared to RS over five seeds across small labeled pool sizes at an illicit rate of 2%. Bands indicate the percentile range 0-100. RF supervised baseline (at 2% illicit rate) for comparison.

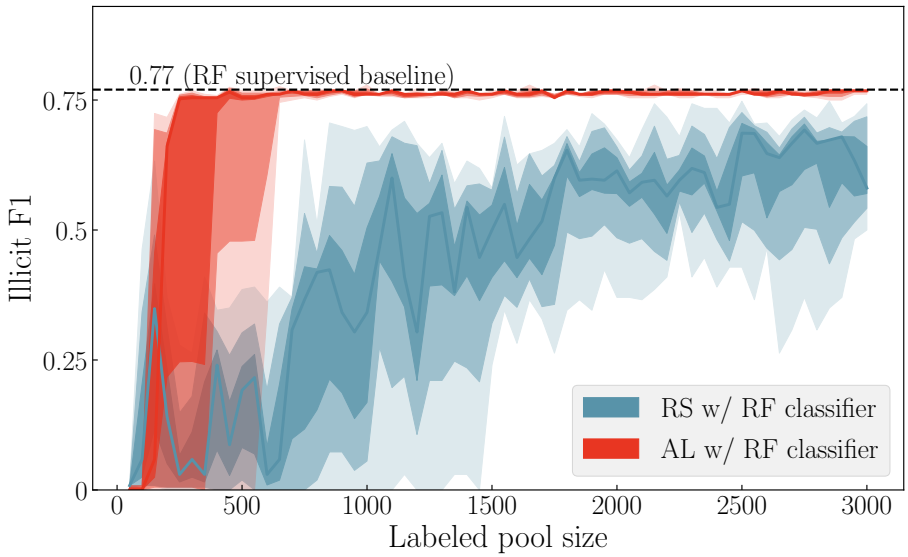

Figure 11: Median illicit F1-score of the best AL setup compared to RS over five seeds across small labeled pool sizes at an illicit rate of 0.5%. Bands indicate the percentile range 0-100. RF supervised baseline (at 0.5% illicit rate) for comparison.

Chapter 6

Conclusions

This study addressed the real-world challenge of detecting money laundering in a dataset with few or no labels by conducting experiments to detect illicit activity in a real-life Bitcoin transaction dataset. Using a supervised setting similar to Weber et al. (2019) as the baseline, the detection ability of machine learning models was studied in a more realistic setting with restricted access to labels, using unsupervised methods, and Active Learning (AL).

The results of the study show that detecting money laundering cases in the dataset without any labels using anomaly detection methods is impossible because anomalies in the feature space are not indicative of illicit behaviour. This finding highlights that experiments conducted on (partially) synthetic data can be misleading and emphasizes the importance of conducting experiments on real-life datasets to draw reliable conclusions.

Using AL, however, one can match the results of a supervised baseline with just a few labels (approximately 5% of the total). This setting is realistic and akin to asking money laundering analysts to review cases that an AL model indicates as informative.

This study extends existing research on unsupervised illicit activity detection in cryptocurrency and financial transactions by benchmarking different methods on a real-world dataset with a relatively large number of positive cases. In this way, it overcomes the typical limitation of evaluating on synthetic data or real data with few positive samples. Furthermore, this is the first research to apply AL to a transaction-level analysis in a large-scale dataset and to the specific case of cryptocurrencies.

It remains to explore if the distribution of classes that was found in the Bitcoin dataset holds for other real-life datasets and different labeling strategies. In order to transfer findings to a financial transaction setting, the differences and similarities of money laundering patterns in cryptocurrencies and financial transactions data need

to be studied carefully. Finally, due to the full anonymization and the removal of any entity information from the graph used in this research, graph structures could not be studied. Further research on other real-life cryptocurrency datasets should, however, study whether the exploitation of graph information could alter the feature space in such a way that illicit and licit behaviour become distinguishable in an unsupervised setting, since graph structures have frequently proven useful in previous research (Chapter 2).

Bibliography

AI 50: America's Most Promising Artificial Intelligence Companies (2019). URL: https://www.forbes.com/sites/jilliandonfro/2019/09/17/ai-50-americas-most-promising-artificial-intelligence-companies.

AIM Evaluation: Fraud and AML Machine Learning Platform Vendors (2019). URL: https://www.aitegroup.com/report/aim-evaluation-fraud-and-aml-machine-learning-platform-vendors.

Almgren, M. and E. Jonsson (2004). "Using active learning in intrusion detection." In: *Proceedings of the Computer Security Foundations Workshop*. Vol. 17, pp. 88–98.

Alpaydin, E. (2010). *Introduction to machine learning*. 2. MIT press.

Barata, R., M. Leite, R. Pacheco, M. O. P. Sampaio, J. Ascensão, and P. Bizarro (2020). "Active learning for online training in imbalanced data streams under cold start." To appear.

Bartoletti, M., B. Pes, and S. Serusi (2018). "Data mining for detecting Bitcoin Ponzi schemes." In: *Crypto Valley Conference on Blockchain Technology (CVCBT)*. IEEE, pp. 75–84.

Bellei, C. (Aug. 2019). "The Elliptic Data Set: opening up machine learning on the blockchain." en. In: *Medium*. URL: https://medium.com/elliptic/the-elliptic-data-set-opening-up-machine-learning-on-the-blockchain-e0a343d99a14.

Bershtein, L. S. and A. Tselykh (2013). "A clique-based method for mining fuzzy graph patterns in anti-money laundering systems." In: *Proceedings of the 6th International Conference on Security of Information and Networks*, pp. 384–387.

Breiman, L. (2001). "Random forests." In: *Machine learning* 45.1, pp. 5–32.

Breiman, L., J. Friedman, C. J. Stone, and R. A. Olshen (1984). *Classification and regression trees*. CRC press.

Brenig, C., R. Accorsi, and G. Müller (2015). "Economic analysis of cryptocurrency backed money laundering." In: *European Conference on Information Systems*.

Breunig, M. M., H.-P. Kriegel, R. T. Ng, and J. Sander (2000). "LOF: identifying density-based local outliers." In: *Proceedings of the ACM SIGMOD International Conference on Management of Data*, pp. 93–104.

Camino, R. D., R. State, L. Montero, and P. Valtchev (2017). "Finding Suspicious Activities in Financial Transactions and Distributed Ledgers." In: *IEEE International Conference on Data Mining Workshops (ICDMW)*. IEEE, pp. 787–796.

Cao, B., M. Mao, S. Viidu, and S. Y. Philip (2017). "HitFraud: a broad learning approach for collective fraud detection in heterogeneous information networks." In: *IEEE International Conference on Data Mining (ICDM)*. IEEE, pp. 769–774.

Carcillo, F., Y.-A. Le Borgne, O. Caelen, and G. Bontempi (2018). "Streaming active learning strategies for real-life credit card fraud detection: assessment and visualization." In: *International Journal of Data Science and Analytics* 5.4, pp. 285–300.

Chandola, V., A. Banerjee, and V. Kumar (2009). "Anomaly detection: A survey." In: *ACM computing surveys (CSUR)* 41.3, pp. 1–58.

Chen, T. and C. Guestrin (2016). "Xgboost: A scalable tree boosting system." In: *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 785–794.

Chen, Z., A. Nazir, E. N. Teoh, E. K. Karupiah, et al. (2014). "Exploration of the effectiveness of expectation maximization algorithm for suspicious transaction detection in anti-money laundering." In: *IEEE Conference on Open Systems (ICOS)*. IEEE, pp. 145–149.

Das, S., W.-K. Wong, T. Dietterich, A. Fern, and A. Emmott (2016). "Incorporating expert feedback into active anomaly discovery." In: *16th IEEE International Conference on Data Mining (ICDM)*. IEEE, pp. 853–858.

Deng, X., V. R. Joseph, A. Sudjianto, and C. J. Wu (2009). "Active learning through sequential design, with applications to detection of money laundering." In: *Journal of the American Statistical Association* 104.487, pp. 969–981.

Domingues, R., M. Filippone, P. Michiardi, and J. Zouaoui (2018). "A comparative evaluation of outlier detection algorithms: Experiments and analyses." In: *Pattern Recognition* 74, pp. 406–421.

European Union (June 2018). "Directive (EU) 2018/843 of the European Parliament and of the Council of 30 May 2018 amending Directive (EU) 2015/849 on the prevention of the use of the financial system for the purposes of money laundering or terrorist financing, and amending Directives 2009/138/EC and 2013/36/EU." en. In: *Official Journal of the European Union* L 156, pp. 43–74. (Visited on 04/22/2020).

Friedman, J. H. (2001). "Greedy function approximation: a gradient boosting machine." In: *Annals of statistics*, pp. 1189–1232.

Gao, Z. (2009). "Application of cluster-based local outlier factor algorithm in anti-money laundering." In: *International Conference on Management and Service Science*. IEEE, pp. 1–4.

Gao, Z. and M. Ye (2007). "A framework for data mining-based anti-money laundering research." In: *Journal of Money Laundering Control* 10.2, pp. 170–179.

Görnitz, N., M. Kloft, K. Rieck, and U. Brefeld (2009). "Active learning for network intrusion detection." In: *Proceedings of the 2nd ACM Workshop on Security and Artificial Intelligence*, pp. 47–54.

Hastie, T., R. Tibshirani, and J. Friedman (2009). *The elements of statistical learning: data mining, inference, and prediction*. Springer Science & Business Media.

He, Z., X. Xu, and S. Deng (2003). "Discovering cluster-based local outliers." In: *Pattern Recognition Letters* 24.9-10, pp. 1641–1650.

Hilbe, J. M. (2009). *Logistic regression models*. CRC press.

Hirshman, J., Y. Huang, and S. Macke (2013). *Unsupervised approaches to detecting anomalous behavior in the bitcoin transaction network*. Tech. rep. Stanford University.

Hu, Y., S. Seneviratne, K. Thilakarathna, K. Fukuda, and A. Seneviratne (2019). "Characterizing and Detecting Money Laundering Activities on the Bitcoin Network." In: *ArXiv* 1912.12060.

Keyan, L. and Y. Tingting (2011). "An improved support-vector network model for anti-money laundering." In: *Fifth International Conference on Management of e-Commerce and e-Government*. IEEE, pp. 193–196.

Kriegel, H.-P., M. Schubert, and A. Zimek (2008). "Angle-based outlier detection in high-dimensional data." In: *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 444–452.

Larik, A. S. and S. Haider (2011). "Clustering based anomalous transaction reporting." In: *Procedia Computer Science* 3, pp. 606–610.

Laws, F. and H. Schätze (2008). "Stopping criteria for active learning of named entity recognition." In: *Proceedings of the 22nd International Conference on Computational Linguistics*. Association for Computational Linguistics, pp. 465–472.

Lewis, D. D. and J. Catlett (1994). "Heterogeneous uncertainty sampling for supervised learning." In: *Proceedings of the Eleventh International Conference on Machine Learning*. Elsevier, pp. 148–156.

Li, X., S. Liu, Z. Li, X. Han, C. Shi, B. Hooi, H. Huang, and X. Cheng (2020). "FlowScope : Spotting Money Laundering Based on Graphs." In: *Proceedings of the AAAI Conference on Artificial Intelligence*.

Li, X., X. Cao, X. Qiu, J. Zhao, and J. Zheng (2017). "Intelligent anti-money laundering solution based upon novel community detection in massive transaction networks on spark." In: *Fifth International Conference on Advanced Cloud and Big Data (CBD)*. IEEE, pp. 176–181.

Liu, F. T., K. M. Ting, and Z.-H. Zhou (2008a). "Isolation forest." In: *Eighth IEEE International Conference on Data Mining*. IEEE, pp. 413–422.

Liu, R., X.-l. Qian, S. Mao, and S.-z. Zhu (2011). "Research on anti-money laundering based on core decision tree algorithm." In: *Chinese Control and Decision Conference (CCDC)*. IEEE, pp. 4322–4325.

Liu, X. and P. Zhang (2010). "A scan statistics based Suspicious transactions detection model for Anti-Money Laundering (AML) in financial institutions." In: *International Conference on Multimedia Communications*. IEEE, pp. 210–213.

Liu, X., P. Zhang, and D. Zeng (2008b). "Sequence matching for suspicious activity detection in anti-money laundering." In: *International Conference on Intelligence and Security Informatics*. Springer, pp. 50–61.

Lorenz, J., M. I. Silva, D. Aparício, J. T. Ascensão, and P. Bizarro (2020). "Machine learning methods to detect money laundering in the Bitcoin blockchain in the presence of label scarcity." In: *ArXiv* 2005.14635.

Luo, X. (2014). "Suspicious transaction detection for anti-money laundering." In: *International Journal of Security and Its Applications* 8.2, pp. 157–166.

Lv, L.-T., N. Ji, and J.-L. Zhang (2008). "A RBF neural network model for anti-money laundering." In: *International Conference on Wavelet Analysis and Pattern Recognition*. Vol. 1. IEEE, pp. 209–215.

McInnes, L., J. Healy, and J. Melville (2018). "UMAP: Uniform Manifold Approximation and Projection for Dimension Reduction." In: *arXiv* 1802.03426.

Michalak, K. and J. Korczak (2011). "Graph mining approach to suspicious transaction detection." In: *Federated Conference on Computer Science and Information Systems (FedCSIS)*. IEEE, pp. 69–75.

Monamo, P., V. Marivate, and B. Twala (2016a). "Unsupervised learning for robust Bitcoin fraud detection." In: *Information Security for South Africa (ISSA)*. IEEE, pp. 129–134.

Monamo, P. M., V. Marivate, and B. Twala (2016b). "A multifaceted approach to bitcoin fraud detection: Global and local outliers." In: *15th IEEE International Conference on Machine Learning and Applications (ICMLA)*. IEEE, pp. 188–194.

Monroe, B. (2020). *Ousted Danske Bank CEO faces nearly $400 million lawsuit tied to historic money laundering scandal*. URL: https://www.acfcs.org/ousted-danske-bank-ceo-faces-nearly-400-million-lawsuit-tied-to-historic-money-laundering-scandal/.

Noonan, L., S. Palma, and K. Shubber (2020). "The 1MDB scandal: what does it mean for Goldman Sachs?" In: *Financial Times*. URL: https://www.ft.com/content/3f161eda-3306-11ea-9703-eea0cae3f0de.

Pedregosa, F., G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay (2011). "Scikit-learn: Machine Learning in Python." In: *Journal of Machine Learning Research* 12, pp. 2825–2830.

Pham, T. and S. Lee (2016a). "Anomaly detection in bitcoin network using unsupervised learning methods." In: *ArXiv* 1611.03941.

— (2016b). "Anomaly detection in the bitcoin system-a network perspective." In: *ArXiv* 1611.03942.

Raza, S. and S. Haider (2011). "Suspicious activity reporting using dynamic bayesian networks." In: *Procedia Computer Science* 3, pp. 987–991.

Savage, D., Q. Wang, P. L. Chou, X. Zhang, and X. Yu (2016). "Detection of money laundering groups using supervised learning in networks." In: *ArXiv* 1608.00708.

Savage, D., Q. Wang, X. Zhang, P. Chou, and X. Yu (2017). "Detection of Money Laundering Groups: Supervised Learning on Small Networks." In: *Workshops at the Thirty-First AAAI Conference on Artificial Intelligence*.

Schölkopf, B., J. C. Platt, J. Shawe-Taylor, A. J. Smola, and R. C. Williamson (2001). "Estimating the support of a high-dimensional distribution." In: *Neural Computation* 13.7, pp. 1443–1471.

Settles, B. (2009). *Active learning literature survey*. Tech. rep. University of Wisconsin-Madison Department of Computer Sciences.

Settles, B., M. Craven, and S. Ray (2008). "Multiple-instance active learning." In: *Advances in Neural Information Processing Systems*. Vol. 20, pp. 1289–1296.

Shalev-Shwartz, S. and S. Ben-David (2014). *Understanding machine learning: From theory to algorithms*. Cambridge university press.

Shyu, M. L., S. C. Chen, K. Sarinnapakorn, and L. Chang (2003). "A Novel Anomaly Detection Scheme Based on Principal Component Classifier." In: *3rd IEEE International Conference on Data Mining*, pp. 353–365.

Soltani, R., U. T. Nguyen, Y. Yang, M. Faghani, A. Yagoub, and A. An (2016). "A new algorithm for money laundering detection based on structural similarity." In: *7th IEEE Annual Ubiquitous Computing, Electronics & Mobile Communication Conference (UEMCON)*. IEEE, pp. 1–7.

Stokes, J. W., J. Platt, J. Kravis, and M. Shilman (2008). *Aladin: Active learning of anomalies to detect intrusions*. Tech. rep. Microsoft.

Sudjianto, A., S. Nair, M. Yuan, A. Zhang, D. Kern, and F. Cela-Díaz (2010). "Statistical methods for fighting financial crimes." In: *Technometrics* 52.1, pp. 5–19.

Tang, J. and J. Yin (2005). "Developing an intelligent data discriminating system of anti-money laundering based on SVM." In: *International Conference on Machine Learning and Cybernetics*. Vol. 6. IEEE, pp. 3453–3457.

Upadhyaya, S. and K. Singh (2012). "Classification based outlier detection techniques." In: *International Journal of Computer Trends and Technology* 3.2, pp. 294–298.

Valiant, L. G. (1984). "A theory of the learnable." In: *Communications of the ACM* 27.11, pp. 1134–1142.

Wagner, D. (2019). "Latent representations of transaction network graphs in continuous vector spaces as features for money laundering detection." In: *SKILL - Studierendenkonferenz Informatik*, pp. 1–12.

Wang, X. and G. Dong (2009). "Research on money laundering detection based on improved minimum spanning tree clustering and its application." In: *Second International Symposium on Knowledge Acquisition and Modeling*. Vol. 2. IEEE, pp. 62–64.

Wang, Y., H. Wang, C. S. J. Gao, and D. Xu (2008). "Intelligent money laundering monitoring and detecting system." In: *European, Mediterranean and Middle Eastern Conference on Information Systems*. Brunel University, pp. 1–11.

Weber, M., G. Domeniconi, J. Chen, D. K. I. Weidele, C. Bellei, T. Robinson, and C. E. Leiserson (2019). "Anti-money laundering in bitcoin: Experimenting with graph convolutional networks for financial forensics." In: *ArXiv* 1908.02591.

Wu, Z., S. Pan, F. Chen, G. Long, C. Zhang, and S. Y. Philip (2020). "A comprehensive survey on graph neural networks." In: *IEEE Transactions on Neural Networks and Learning Systems*.

Yang, Y., B. Lian, L. Li, C. Chen, and P. Li (2014). "DBSCAN clustering algorithm applied to identify suspicious financial transactions." In: *International Conference on Cyber-Enabled Distributed Computing and Knowledge Discovery*. IEEE, pp. 60–65.

Yunkai, C., M. Quanwen, and L. Zhengding (2006). "Using link analysis technique with a modified shortest-path algorithm to fight money laundering." In: *Wuhan University Journal of Natural Sciences* 11.5, pp. 1352–1356.

Zhang, Y. and P. Trubey (2019). "Machine learning and sampling scheme: An empirical study of money laundering detection." In: *Computational Economics* 54.3, pp. 1043–1063.

Zhao, Y., Z. Nasrullah, and Z. Li (2019). "PyOD: A Python Toolbox for Scalable Outlier Detection." In: *Journal of Machine Learning Research* 20.96, pp. 1–7.

Appendix

Published research paper

Machine learning methods to detect money laundering in the Bitcoin blockchain in the presence of label scarcity

Joana Lorenz
joana.lorenz@outlook.de
NOVA IMS & Feedzai

Maria Inês Silva
maria.silva@feedzai.com
Feedzai

David Aparício
david.aparicio@feedzai.com
Feedzai

João Tiago Ascensão
joao.ascensao@feedzai.com
Feedzai

Pedro Bizarro
pedro.bizarro@feedzai.com
Feedzai

ABSTRACT

Every year, criminals launder billions of dollars acquired from serious felonies (e.g., terrorism, drug smuggling, or human trafficking) harming countless people and economies. Cryptocurrencies, in particular, have developed as a haven for money laundering activity. Machine Learning can be used to detect these illicit patterns. However, labels are so scarce that traditional supervised algorithms are inapplicable. Here, we address money laundering detection assuming minimal access to labels. First, we show that existing state-of-the-art solutions using unsupervised anomaly detection methods are inadequate to detect the illicit patterns in a real Bitcoin transaction dataset. Then, we show that our proposed active learning solution is capable of matching the performance of a fully supervised baseline by using just 5% of the labels. This solution mimics a typical real-life situation in which a limited number of labels can be acquired through manual annotation by experts.

CCS CONCEPTS

• **Computing methodologies** → *Supervised learning by classification*; *Anomaly detection*; *Active learning settings*; • **Applied computing** → **Economics**.

KEYWORDS

anti money laundering, cryptocurrency, supervised classification, anomaly detection, active learning

ACM Reference Format:
Joana Lorenz, Maria Inês Silva, David Aparício, João Tiago Ascensão, and Pedro Bizarro. 2020. Machine learning methods to detect money laundering in the Bitcoin blockchain in the presence of label scarcity. In *2020 ACM International Conference on AI in Finance, October 15–16, 2020, New York, NY*. ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/1122445.1122456

1 INTRODUCTION

Money laundering is a high-impact problem on a global scale. Criminals obtain money illegally from serious crimes and inject it into the financial system as seemingly legitimate funds. Money laundering schemes usually involve large amounts of money and, when caught, typically result in large fines for financial institutions. Recent examples are the 1MDB [28] and the Danske Bank scandals [26].

Governments and international organizations are building tighter regulations around money laundering and are broadening them to include cryptocurrencies [27, 38], where criminals benefit from pseudonymity.

In the financial sector, Anti-Money Laundering (AML) efforts often rely on rule-based systems [20]. However, vulnerabilities derive from the relative simplicity of publicly available rule-sets, leading to high false-positive rates (FPR) and low detection rates [40]. Machine learning (ML) techniques overcome the rigidity of rule-based systems by inferring complex patterns from historical data, and can potentially increase detection rates and decrease FPRs.

Recently, Weber et al. [41] released a dataset, consisting of a sample of 200k labeled Bitcoin transactions, and evaluated various supervised models on it. Unfortunately, supervised methods are often unfeasible as institutions do not possess large-scale labeled datasets. This lack of labels is due to two main reasons. First, given the evolving complexity of money laundering schemes, it is unlikely to be possible to identify all (or even most) of the entities involved in money laundering. Second, labels resulting from law enforcement investigations are not immediate, and manual annotation is costly. Thus, in order to properly evaluate the practical feasibility of ML for AML, strategies that require no labels (unsupervised learning) or just a few labels (active learning) are paramount.

We address the real-world challenge of *how to detect money laundering in a dataset with few labels*. Particularly, we show that:

(1) Detecting money laundering cases in the Bitcoin network without any labels is impossible since illicit transactions hide within clusters of licit behaviour (Section 4.2).
(2) With just a few labels (approximately 5% of the total), one can match the results of a supervised baseline by using Active Learning (AL) (Section 4.3). This setting mimics a real-world scenario with limited availability of human analysts for manual labeling.

We extend the existing research on unsupervised illicit activity detection in cryptocurrency and financial transactions by benchmarking different methods on a real-world dataset with a relatively large number of positive cases. In this way, we overcome the typical limitation of evaluating on synthetic data or real data with few positive samples. Besides, to the best of our knowledge, we are the

ICAIF 2020, October 15–16, 2020, New York, NY

ACM ISBN 978-1-4503-XXXX-X/18/06…$15.00
https://doi.org/10.1145/1122445.1122456

first work to apply AL to AML on a large transaction dataset and in the cryptocurrency setting, specifically.

We organize the remainder of the paper as follows. Section 2 presents the related work. Section 3 details the experimental setup and introduces the relevant anomaly detection methods as well as AL concepts. In Section 4 we present our results. Finally, the main conclusions are discussed in Section 5.

2 RELATED WORK

In this section, we present previous research on ML for AML in the context of both financial transactions and, more specifically, cryptocurrencies. For a thorough survey of ML approaches for AML, we refer the reader to Chen et al. [9].

Although approaches greatly vary, many methods assume money laundering cases to be outliers, i.e., illicit instances (a minority) should exhibit significantly different behaviours from legitimate ones (the majority). Typically, these approaches use unsupervised anomaly detection methods to model licit behaviour and find the instances that deviate from it [4, 8, 13, 17, 21, 22, 32, 37, 39].

Overall, the results of these studies are encouraging, reporting low FPRs [8, 21, 22] and good detection rates [8, 13, 22, 39]. Some studies even report that the ML approaches were able to detect money laundering patterns that were previously unknown [37] or not caught by rule-based systems [4]. However, a fair comparison between methods is impossible, given the heterogeneity of the evaluation setups. In these studies, researchers use real-world datasets labeled by analysts [4, 21, 22], with simulated illicit transactions [13, 37, 39], or no labels at all [17, 32]

Generally, authors are openly doubtful about real-world reproducibility of good results, in the face of intricate patterns and incomplete labels [8, 13, 39]. The question arises on whether reliable anomaly detection is possible in non-synthetic data, as criminals could intentionally mimic normal behaviour. In our research, we contribute to assess the reproducibility of such results by conducting the first in-depth benchmark of anomaly detection methods in a labeled real-world cryptocurrency dataset and comparing their performance against a supervised baseline.

Previous studies on money laundering in cryptocurrencies in particular are scarce and inconclusive due to a lack of labels for evaluation. Some conclude that supervised models perform well [2, 16, 25]. Others report low detection rates for unsupervised methods in extremely imbalanced data [24, 25, 30, 31, 42]. Often, the evaluation of anomaly detection methods consists of checking whether the anomalies represent extreme cases [30, 31] or behaviour deemed suspicious by human analysis [15].

Active Learning has been proposed as a method to reduce the number of labels needed for the training of an effective classifier by iteratively sampling the most informative samples for labeling from an initially unlabeled pool [33]. Given the apparent label scarcity in money laundering data, it is a highly relevant setting for the practical implementation of ML-based AML systems. Previously, Deng et al. [11] applied AL to detect money laundering in financial transactions. In an account-level classification of 92 real-life accounts, they report that their method can accurately estimate the threshold hyperplane with only 22% of the labels. AL has also successfully been applied in other fraud-related use-cases such as credit card fraud [5] and network intrusion detection [1, 14, 35], reporting the sufficiency of as few as 1.5% of the original labels to achieve near-optimal performance [14].

We conduct experiments with AL, assuming an unlabeled dataset and the capacity to acquire labels progressively to train a supervised classifier. We hereby extend the study by Deng et al. [11] to a transaction-level analysis in a much larger cryptocurrency dataset.

3 EXPERIMENTAL SETUP

3.1 Data

We use the Bitcoin dataset[1] released by Elliptic, a company dedicated to detecting financial crime in cryptocurrencies [41]. It includes 49 graphs sampled from the Bitcoin blockchain at different sequential moments in time (time-steps), as presented in Figure 1. Each graph is a directed acyclic graph, starting from one transaction, and including subsequent related transactions on the blockchain, containing approximately two weeks of data.

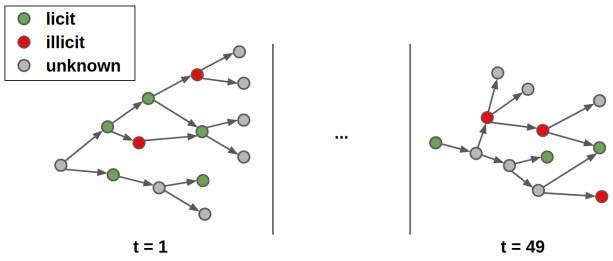

Figure 1: Structure of the dataset (taken from Bellei [3]).

Bitcoins transactions are transfers from one Bitcoin address (e.g., a person or company) to another, represented as nodes in the graph. Each transaction consumes the output of past transactions and generates outputs that can be spent by future transactions. The edges in the graph represent the flow of Bitcoins between transactions.

The dataset consists of 203,769 transactions, of which 21% are labeled as licit, and 2% as illicit, based on the *category* of the bitcoin address that created the transaction. The remaining transactions are unlabeled. Illicit categories include scams, malware, terrorist organizations, and Ponzi schemes. Licit categories include exchanges, wallet providers, miners, and licit services. Each transaction has 166 features, 94 of which represent information about the transaction itself. The remaining features were constructed by Weber et al. [41] using information one-hop backward/forward from the transaction, such as the minimum, maximum, and standard deviation of each transaction feature. All features, except for the time-step, are fully anonymized and standardized with zero mean and unit variance.

3.2 Methods

In this section, we give an overview of the methods used in our experiments and discuss our experimental setup. Following Weber et al. [41], we split the data into sequential train and test datasets for all experiments. The train set includes all labeled samples up to the 34th time-step (16670 transactions), and the test set includes all

[1] Available at https://www.kaggle.com/ellipticco/elliptic-data-set

labeled samples from the 35th time-step, inclusive, onward (29894 transactions). Like Weber et al. [41] we evaluate all methods using the F1-score for the illicit class, hereafter referred to as illicit F1-score.

3.2.1 Supervised Learning. In order to benchmark unsupervised methods and AL, we first reproduce the results of Weber et al. [41] as our baseline.

We train each supervised model on the train set using all 166 features and then evaluate them on the entire test set. To measure performance over time, and following Weber et al. [41], we also report the illicit F1-score per time-step in the test set. We use the scikit-learn [29] implementation of logistic regression (LR) and random forest (RF) as well as the Python implementation of XGBoost [7]. We present the results achieved using default parameters, as in Weber et al. [41].

3.2.2 Unsupervised Learning. Anomaly detection methods are unsupervised learning techniques to detect outliers in a dataset. Literature suggests their effectiveness in the AML context (Section 2). For a thorough review of anomaly detection, we refer the reader to the surveys by Chandola et al. [6] and Domingues et al. [12].

The standard definition of outliers refers to instances that are unlikely to be drawn from the same distribution as the train data or instances that are far from other data points in the feature space. Although we focus mainly on unsupervised anomaly detection, some methods are semi-supervised discriminators trained to learn a boundary around normal instances. In that context, outliers are instances that fall outside of the boundary [13].

We test seven common anomaly detection algorithms with readily available Python implementations: Local Outlier Factor (LOF), K-Nearest Neighbours (KNN), Principal Component Analysis (PCA), One-Class Support Vector Machine (OCSVM), Cluster-based Outlier Factor (CBLOF), Angle-based Outlier Detection (ABOD), and Isolation Forest (IF). We aim at a diversity of strategies. We use the PyOD package implementations [43] with default parameters.

LOF and KNN start by computing the distance of each instance to its k nearest-neighbour. Then, KNN defines that distance as the outlier score. LOF uses the distance to compute the instance's density, and if the density is substantially lower than the average density of its k nearest-neighbours, the instance is declared anomalous.

PCA and OCSVM define anomalies as observations that deviate from normal behaviour. They detect anomalous instances as observations with a large distance to the principal components (PCA) of non-anomalous observations or instances that lay outside of the decision boundary (OCSVM) learned around them.

CBLOF uses the outcome of a clustering algorithm on the instances (in our case k-means) and classifies each cluster as either *small* or *large*. It calculates an anomaly score for each instance, marking instances that belong to small clusters or that are far from big clusters as anomalous. ABOD computes the pairwise cosine similarities between all points and classifies those with a low average radius and variance as anomalies. Lastly, IF isolates anomalies by performing recursive random splits on attribute values. Based on the resulting tree structure, anomalies are instances that are easy to isolate, i.e., have shorter paths.

The introduced methods use different anomaly scores and scales. Thus, a fair comparison requires evaluation at different *contamination levels*, defined as the expected proportion of outliers in the dataset, and used to set a threshold for the decision function. Whereas the original PyOD implementation applies the contamination level on the scores of the train set, we apply it on the test set scores to guarantee that the desired percentage of positive cases (anomalies) in the test set is the same across methods. The contamination level here is analogous to a fixed alert rate in real AML systems, i.e., the percentage of cases flagged for further investigation by an analyst. We evaluate the illicit F1-score for each model at contamination levels between 0 and 1, with increments of 0.05. We also present the illicit F1-score of the RF supervised baseline, where we define the model threshold by setting the contamination level as the predicted positive rate (or alert rate), for comparison.

3.2.3 Active Learning. AL is an incremental learning approach that interactively queries instances for labeling (e.g., by human analysts) and uses the increasing number of labeled instances to (re-)train a supervised model. It fits the AML context by addressing label scarcity and has previously been successfully applied to detect money laundering accounts based on financial transaction history. For an extensive survey on AL, we refer the reader to Settles [33].

The goal of AL is to minimize the number of labels necessary to achieve adequate classifier performance. The process starts with a pool of unlabeled instances (the *unlabeled pool*), although sometimes there is a residual number of labels. At each iteration, a *query strategy* queries a batch of instances for manual labeling. After labeling, the instances go into the *labeled pool*. Finally, a supervised algorithm (the *classifier*) is trained on the labeled pool and evaluated on a test set. If the performance is not satisfactory, the querying process continues to enrich the labeled pool incrementally. To mimic the manual labeling process in our experiments, we append the labels to the queried instances.

In the literature, query strategies build on various models and uncertainty criteria. In this study, we focus on four query strategies trained on the labeled pool to find informative instances in the unlabeled pool. Two of them, uncertainty sampling and expected model change, are supervised, requiring an underlying supervised model to define queries. The other two, elliptic envelope and Isolation Forest (IF), are unsupervised and find outlying instances with regards to the labeled pool. We use random sampling as a baseline. This setup was based on previous work done on Feedzai's active learning annotation tool, which was used to run the experiments.

Expected model change [33, 34] assumes that instances are more informative if they influence the model more strongly. It queries the unlabeled instances that lead to the most significant change in the model parameters by measuring the impact of labeling one unlabeled instance on the gradient of the model's loss function. Thus, this strategy applies only to gradient-based classifiers. In our experiment, we use LR. The expected model change is a weighted sum over all possible labels since the labels of the instances are unknown before querying. Then, at each iteration, we query the labels of the instances with the largest expected gradients.

Uncertainty sampling is one of the most commonly used query strategies [19, 33]. It queries the instances about which a model is most uncertain. Assuming a probabilistic learning model and a

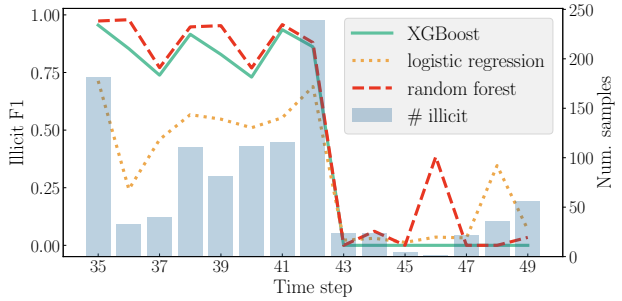

Figure 2: Illicit F1-score for each supervised baseline, across time-steps.

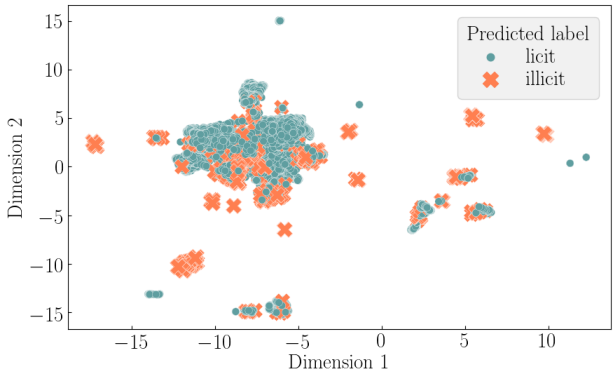

Figure 3: UMAP projection of the test set, colored by the labels predicted by IF.

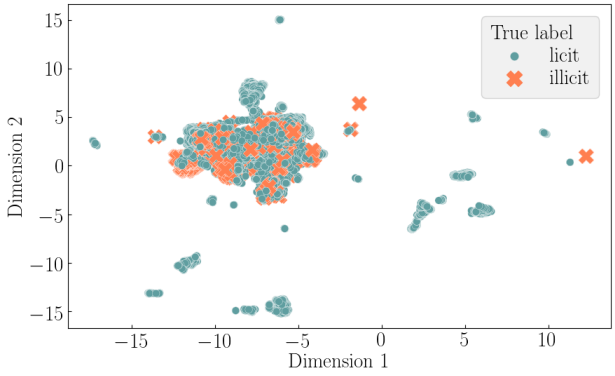

Figure 4: UMAP projection of the test set, colored by the true labels.

binary classification problem, this translates to querying the instances with the predicted score closest to 0.5. In our study, we use the same type of classifier for uncertainty sampling and evaluating on the test set; for instance, if the classifier is RF, we also conduct uncertainty sampling using RF.

The two unsupervised query strategies used are IF and elliptic envelope. Outliers are transactions with high anomaly scores (IF) or a large Mahalanobis distance to a multivariate Gaussian distribution fit on the labeled pool (elliptic envelope).

We combine unsupervised and supervised query strategies in our experiments, depending on the number of illicit instances in the labeled pool. After an initial random sample of one batch of instances, we use an unsupervised *warm-up learner* that samples instances until the labeled pool includes at least one illicit instance. When we reach this threshold, we either switch to a supervised *hot learner* or continue to use the warm-up learner. As the classifier, we use the three supervised models evaluated in the supervised baselines: RF, XGBoost, and LR. We compare all AL setups against a baseline that queries random instances at each iteration (also used as a warm-up learner). We use a batch size of 50 instances sampled at each iteration for all experiments. Each AL setup is run five times with different random seeds to ensure the robustness of the results. We assess the performance of each AL setup through the median illicit F1-score and the confidence intervals at each labeled pool size.

4 RESULTS

In this section, we present the experimental results for the supervised baseline, followed by the anomaly detection and the AL benchmarks.

4.1 Supervised baselines

We are able to reproduce the results presented by Weber et al. [41] closely. Over five runs (with different seeds), we achieve an illicit F1-score on the test set of 0.76 for XGBoost, 0.45 for LR, and 0.83 for RF. Thus, the best supervised baseline is achieved with the RF model. As Weber et al. [41], we observe that model performance is profoundly affected by a sudden dark market shutdown at time-step 43.

4.2 Anomaly detection

In Table 1, we present the illicit F1-score of the explored anomaly detection methods as well as the RF supervised baseline at different contamination levels. Recall that, at each contamination level, we define the threshold of the RF model so that it leads to an alert rate equal to that contamination level.

Table 1: Anomaly detection methods illicit F1-score by contamination level (RF supervised baseline for reference).

Model	Contamination level			
	0.05	**0.1**	**0.15**	**0.2**
RF supervised baseline	0.82	0.58	0.46	0.39
LOF	0.11	0.15	0.19	0.18
ABOD	0.07	0.07	0.07	0.07
KNN	0.03	0.04	0.05	0.06
OCSVM	0.01	0.03	0.03	0.04
CBLOF	0.01	0.02	0.03	0.04
PCA	0.01	0.01	0.02	0.02
IF	0.00	0.00	0.00	0.01

Anomaly detection methods perform significantly below the RF supervised baseline across all contamination levels. These results are not consistent with past studies, where anomaly detection methods perform adequately for AML (Section 2). However, we note that these studies often use synthetically generated anomalous data points that are outliers by design. Furthermore, there could be differences between money laundering patterns in financial transactions and Bitcoin transfers. In the real-life Bitcoin transaction dataset, we see that illicit cases are indeed not outlying.

To illustrate this, we apply the Uniform Manifold Approximation and Projection (UMAP) [23] to the test set and build two plots with the resulting projection. In the first (Figure 3), we color each observation based on the predicted label of the worst-performing method (IF), while in the second (Figure 4), we color each observation based on the true label. We can then see that the IF classifies most outlying instances as illicit, as intended. Still, the true labels presented in Figure 4 reveal that the illicit instances in the dataset are not actually outlying, but instead hiding among licit transactions.

The observation that not all outliers are illicit and that not all illicit transactions are outliers is reasonable in AML as sophisticated criminals obfuscate their activity by mimicking normal behaviour, hiding in regions of high nominal density. This problem was previously acknowledged by Das et al. [10]. Thus, we conclude that anomaly detection methods are ineffective for the unsupervised classification task in this real-life Bitcoin dataset.

4.3 Active learning

Table 2 summarises the results of the AL benchmark for each of the three different classifiers used for the supervised baselines. We conclude that switching to a supervised hot-learner significantly improves performance over the continued use of an unsupervised warm-up learners. Among hot-learners, however, there is no clear best policy.

Furthermore, we can see that random sampling as the warm-up learner leads to a faster improvement in model performance (i.e., better performance for smaller labeled pool sizes) compared to anomaly detection methods. This observation aligns with previous considerations on the poor performance of anomaly detection methods (Section 4.2). Since these methods fail to detect illicit instances, they are ineffective at querying illicit instances to be added to the labeled pool to improve the performance of a supervised classifier quickly. We observe that elliptic envelope performs above IF (also consistent with previous results).

Table 2 additionally shows that XGBoost and LR temporarily surpass their supervised baseline, i.e., the performance they achieved when trained on the entire train set (Section 4.1. The classifiers perform better when trained only on a sample of the labeled data but eventually converge to their supervised baseline as the labeled pool increases over time. This result can be because the labeled pool consists of the most relevant samples at the beginning of the AL process and, at the same time, the class imbalance increases over time. Laws and Schätze [18] acknowledge that, in some cases, early stopping of an AL process might prevent this model degradation. Note, however, that even if XGBoost and LR surpass their own supervised baselines, they do not surpass the best supervised baseline, which was achieved with the RF model.

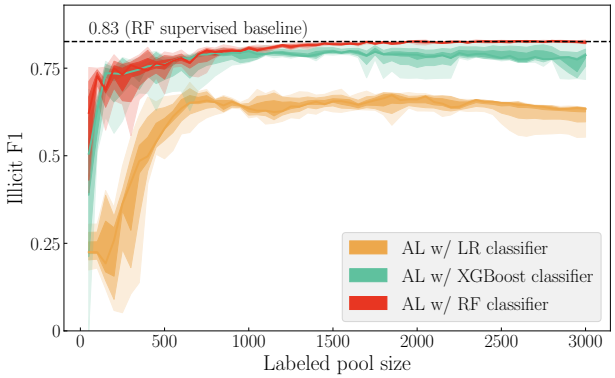

Figure 5: Best AL setups for each classifier and the RF supervised baseline.

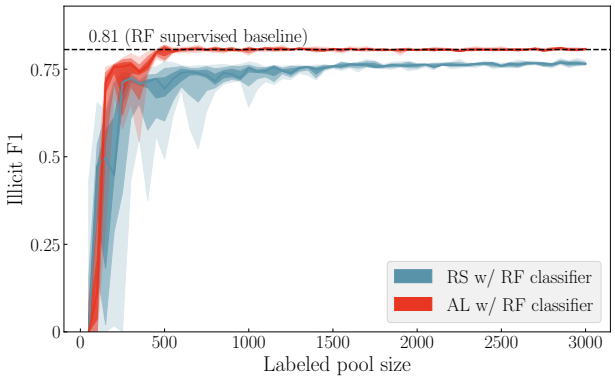

Figure 6: AL versus Random Sampling (RS) with a RF classifier and the RF supervised baseline at 2% illicit rate.

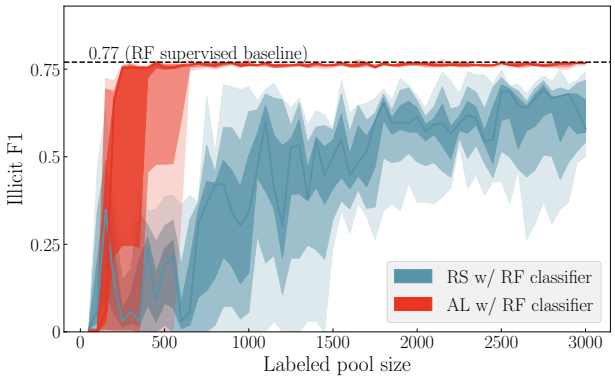

Figure 7: AL versus Random Sampling (RS) with a RF classifier and the RF supervised baseline at 0.5% illicit rate.

Figure 5 shows the performance over time of the best AL setup for the three classifiers tested. For comparison, it also includes the performance achieved by the best supervised baseline, the RF supervised baseline. With the presented AL setup, all classifiers

Table 2: Average illicit F1-score over five runs for each AL setup, consisting of an unsupervised warm-up learner, an optional supervised hot learner and the classifier that is evaluated on the test set. We compare the results to each classifier's respective supervised baseline (Section 4.1). Results are ordered by the illicit F1-score with 3000 labels. Best values for each labeled pool size across classifiers are highlighted in bold.

Query strategies		Classifier	Labeled pool size					Supervised baseline
Warm-up learner	Hot learner		200 (0.7%)	500 (1.7%)	1000 (3.3%)	1500 (5%)	3000 (10%)	
isolation forest	uncertainty sampling	random forest	0.75	0.75	0.80	**0.82**	**0.83**	0.83
random sampling	uncertainty sampling		0.73	0.75	**0.81**	**0.82**	0.82	
elliptic envelope	uncertainty sampling		0.65	**0.77**	0.80	**0.82**	0.82	
isolation forest	expected model change		0.56	0.61	0.77	0.79	0.81	
random sampling	expected model change		**0.76**	**0.77**	0.78	0.78	0.81	
elliptic envelope	expected model change		0.60	0.72	0.76	0.77	0.81	
random sampling	–		0.74	0.76	0.76	0.78	0.80	
elliptic envelope	–		0.50	0.53	0.56	0.65	0.70	
isolation forest	–		0.67	0.65	0.59	0.63	0.62	
isolation forest	uncertainty sampling	XGBoost	0.67	**0.77**	0.80	0.79	0.80	0.76
elliptic envelope	expected model change		0.65	0.75	0.77	0.75	0.79	
random sampling	expected model change		0.70	0.75	0.79	0.80	0.78	
isolation forest	expected model change		0.60	0.75	0.77	0.76	0.75	
elliptic envelope	–		0.53	0.64	0.53	0.61	0.68	
elliptic envelope	uncertainty sampling		0.62	0.62	0.64	0.80	0.64	
random sampling	uncertainty sampling		0.72	0.76	0.64	0.60	0.64	
random sampling	–		0.66	0.58	0.75	0.74	0.59	
isolation forest	–		0.38	0.38	0.46	0.44	0.57	
isolation forest	expected model change	logistic regression	0.22	0.59	0.63	0.66	0.62	0.45
elliptic envelope	expected model change		0.20	0.48	0.61	0.61	0.61	
random sampling	expected model change		0.44	0.54	0.58	0.64	0.60	
elliptic envelope	uncertainty sampling		0.41	0.52	0.63	0.63	0.60	
isolation forest	uncertainty sampling		0.37	0.53	0.61	0.60	0.58	
random sampling	uncertainty sampling		0.40	0.50	0.57	0.58	0.55	
random sampling	–		0.36	0.36	0.36	0.37	0.39	
elliptic envelope	–		0.28	0.25	0.24	0.24	0.22	
isolation forest	–		0.25	0.24	0.29	0.21	0.02	

stabilize after 1000 labels, with RF and XGBoost exhibiting faster performance increase. RF reaches its baseline's performance with only 5% of the original labels, or 1500 out of the original 30000 labels (Figure 5). We can even see a near-optimal performance with as few as 500 labels.

From Table 2, we can observe that the random sampling baseline achieves a similar performance to the more sophisticated AL strategies. Our intuition is that the classifier will start approaching good performances when the labeled pool includes a sufficient number of illicit instances and, because the dataset has approximately 10% of illicit cases, random sampling can quickly reach that sufficient number.

In reality, financial crime is extremely rare among licit transactions, and thus datasets are highly imbalanced Sudjianto et al. [36]. Since we are interested in the practical relevance of AL, we compare the best performing AL setup against random sampling in a dataset with a higher, more realistic class imbalance. Specifically, we apply a random undersampling of the minority class of the Ellipic dataset to achieve illicit rates of 2% and 0.5%. The results are plotted in Figures 6 and 7, respectively. For comparison, we indicate the RF supervised baseline performance at the respective reduced fraud rates.

As expected, the AL query strategies increasingly outperform random sampling as imbalance increases. For highly imbalanced datasets, the best setup uses random sampling (warm-up) followed by uncertainty sampling (hot learner).

5 CONCLUSION

In this study, we conducted experiments to detect illicit activity on the Bitcoin transaction dataset released by Elliptic. Using a supervised setting similar to Weber et al. [41] as our baseline, we studied the detection ability of machine learning models in a more realistic setting with restricted access to labels, using unsupervised methods, and Active Learning (AL).

Our results indicate that unsupervised anomaly detection methods have poor performance, and we present evidence that anomalies in the feature-space are not indicative of illicit behaviour. This finding highlights that experiments conducted on (partially) synthetic data can be misleading and emphasizes the importance of conducting experiments on real-life datasets to draw reliable conclusions.

To improve upon the unsupervised performance, we studied the case where few labels can be obtained by using AL and determined the minimum amount of labeled instances necessary to achieve a performance close to the best supervised baseline. This setting is realistic and akin to asking money laundering analysts to review cases that an AL model indicates as informative. We obtained similar performance to the best supervised baseline by using just a few hundred labels (5% of the total).

It remains to explore if the distribution of classes that we found in the Bitcoin dataset holds for other real-life datasets and different labeling strategies. Furthermore, given the need for proper AML processes in the entire financial system, it is crucial to conduct similar benchmarks on other verticals such as bank transfers, deposits or loans, using real datasets with proper labels.

ACKNOWLEDGMENTS

We want to thank the entire Feedzai Research team, who always gave insightful suggestions. In particular, we want to give special thanks to Marco Sampaio, Ricardo Barata, and Miguel Leite for their support with the AL experiments.

Note on reproducibility

The code to reproduce the results presented in Sections 4.1 and 4.2 is available online at https://github.com/feedzai/research-aml-elliptic. We do not include the code for the AL experiments for intellectual property purposes.

REFERENCES

[1] Magnus Almgren and Erland Jonsson. 2004. Using active learning in intrusion detection. *Proceedings of the Computer Security Foundations Workshop* 17, 88–98.

[2] Massimo Bartoletti, Barbara Pes, and Sergio Serusi. 2018. Data mining for detecting Bitcoin Ponzi schemes. In *2018 Crypto Valley Conference on Blockchain Technology (CVCBT)*. IEEE, 75–84.

[3] Claudio Bellei. 2019. The Elliptic Data Set: opening up machine learning on the blockchain. *Medium* (Aug. 2019). https://medium.com/elliptic/the-elliptic-data-set-opening-up-machine-learning-on-the-blockchain-e0a343d99a14

[4] Ramiro Daniel Camino, Radu State, Leandro Montero, and Petko Valtchev. 2017. Finding Suspicious Activities in Financial Transactions and Distributed Ledgers. In *2017 IEEE International Conference on Data Mining Workshops (ICDMW)*. IEEE, 787–796.

[5] Fabrizio Carcillo, Yann-Aël Le Borgne, Olivier Caelen, and Gianluca Bontempi. 2018. Streaming active learning strategies for real-life credit card fraud detection: assessment and visualization. *International Journal of Data Science and Analytics* 5, 4 (2018), 285–300.

[6] Varun Chandola, Arindam Banerjee, and Vipin Kumar. 2009. Anomaly detection: A survey. *ACM computing surveys (CSUR)* 41, 3 (2009), 1–58.

[7] Tianqi Chen and Carlos Guestrin. 2016. Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*. 785–794.

[8] Zhiyuan Chen, Amril Nazir, Ee Na Teoh, Ettikan Kandasamy Karupiah, et al. 2014. Exploration of the effectiveness of expectation maximization algorithm for suspicious transaction detection in anti-money laundering. In *2014 IEEE Conference on Open Systems (ICOS)*. IEEE, 145–149.

[9] Zhiyuan Chen, Le Dinh Van Khoa, Ee Na Teoh, Amril Nazir, Ettikan Kandasamy Karuppiah, and Kim Sim Lam. 2018. Machine learning techniques for anti-money laundering (AML) solutions in suspicious transaction detection: a review. *Knowledge and Information Systems* 57, 2 (2018), 245–285.

[10] Shubhomoy Das, Weng-Keen Wong, Thomas Dietterich, Alan Fern, and Andrew Emmott. 2016. Incorporating expert feedback into active anomaly discovery. In *2016 IEEE 16th International Conference on Data Mining (ICDM)*. IEEE, 853–858.

[11] Xinwei Deng, V Roshan Joseph, Agus Sudjianto, and CF Jeff Wu. 2009. Active learning through sequential design, with applications to detection of money laundering. *J. Amer. Statist. Assoc.* 104, 487 (2009), 969–981.

[12] Rémi Domingues, Maurizio Filippone, Pietro Michiardi, and Jihane Zouaoui. 2018. A comparative evaluation of outlier detection algorithms: Experiments and analyses. *Pattern Recognition* 74 (2018), 406–421.

[13] Zengan Gao. 2009. Application of cluster-based local outlier factor algorithm in anti-money laundering. In *2009 International Conference on Management and Service Science*. IEEE, 1–4.

[14] Nico Görnitz, Marius Kloft, Konrad Rieck, and Ulf Brefeld. 2009. Active learning for network intrusion detection. In *Proceedings of the 2nd ACM workshop on Security and artificial intelligence*. 47–54.

[15] Jason Hirshman, Yifei Huang, and Stephen Macke. 2013. Unsupervised approaches to detecting anomalous behavior in the bitcoin transaction network. *3rd ed. Technical report, Stanford University* (2013).

[16] Yining Hu, Suranga Seneviratne, Kanchana Thilakarathna, Kensuke Fukuda, and Aruna Seneviratne. 2019. Characterizing and Detecting Money Laundering Activities on the Bitcoin Network. *arXiv preprint arXiv:1912.12060* (2019).

[17] Asma S Larik and Sajjad Haider. 2011. Clustering based anomalous transaction reporting. *Procedia Computer Science* 3 (2011), 606–610.

[18] Florian Laws and Hinrich Schätze. 2008. Stopping criteria for active learning of named entity recognition. In *Proceedings of the 22nd International Conference on Computational Linguistics-Volume 1*. Association for Computational Linguistics, 465–472.

[19] David D Lewis and Jason Catlett. 1994. Heterogeneous uncertainty sampling for supervised learning. In *Machine learning proceedings 1994*. Elsevier, 148–156.

[20] Xurui Li, Xiang Cao, Xuetao Qiu, Jintao Zhao, and Jianbin Zheng. 2017. Intelligent anti-money laundering solution based upon novel community detection in massive transaction networks on spark. In *2017 fifth international conference on advanced cloud and big data (CBD)*. IEEE, 176–181.

[21] Xuan Liu and Pengzhu Zhang. 2010. A scan statistics based Suspicious transactions detection model for Anti-Money Laundering (AML) in financial institutions. In *2010 International Conference on Multimedia Communications*. IEEE, 210–213.

[22] Xuan Liu, Pengzhu Zhang, and Dajun Zeng. 2008. Sequence matching for suspicious activity detection in anti-money laundering. In *International Conference on Intelligence and Security Informatics*. Springer, 50–61.

[23] Leland McInnes, John Healy, and James Melville. 2018. UMAP: Uniform Manifold Approximation and Projection for Dimension Reduction. (2018).

[24] Patrick Monamo, Vukosi Marivate, and Bheki Twala. 2016. Unsupervised learning for robust Bitcoin fraud detection. In *2016 Information Security for South Africa (ISSA)*. IEEE, 129–134.

[25] Patrick M Monamo, Vukosi Marivate, and Bhesipho Twala. 2016. A multifaceted approach to bitcoin fraud detection: Global and local outliers. In *2016 15th IEEE International Conference on Machine Learning and Applications (ICMLA)*. IEEE, 188–194.

[26] Brian Monroe. 2020. Ousted Danske Bank CEO faces nearly $400 million lawsuit tied to historic money laundering scandal. https://www.acfcs.org/ousted-danske-bank-ceo-faces-nearly-400-million-lawsuit-tied-to-historic-money-laundering-scandal/ Library Catalog: www.acfcs.org Section: Uncategorized.

[27] Financial Crimes Enforcement Network. 2019. Application of FinCENâĂŹs Regulations to Certain Business Models Involving Convertible Virtual Currencies | FinCEN.gov. https://www.fincen.gov/resources/statutes-regulations/guidance/application-fincens-regulations-certain-business-models

[28] Laura Noonan, Stefania Palma, and Kadhim Shubber. 2020. The 1MDB scandal: what does it mean for Goldman Sachs? *Financial Times* (Jan 2020). https://www.ft.com/content/3f161eda-3306-11ea-9703-eea0cae3f0de

[29] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research* 12 (2011), 2825–2830.

[30] Thai Pham and Steven Lee. 2016. Anomaly detection in bitcoin network using unsupervised learning methods. *arXiv preprint arXiv:1611.03941* (2016).

[31] Thai Pham and Steven Lee. 2016. Anomaly detection in the bitcoin system-a network perspective. *arXiv preprint arXiv:1611.03942* (2016).

[32] Saleha Raza and Sajjad Haider. 2011. Suspicious activity reporting using dynamic bayesian networks. *Procedia Computer Science* 3 (2011), 987–991.

[33] Burr Settles. 2009. *Active learning literature survey*. Technical Report. University of Wisconsin-Madison Department of Computer Sciences.

[34] Burr Settles, Mark Craven, and Soumya Ray. 2008. Multiple-instance active learning. In *Advances in neural information processing systems*, Vol. 20. 1289–1296.

[35] Jack W Stokes, John Platt, Joseph Kravis, and Michael Shilman. 2008. Aladin: Active learning of anomalies to detect intrusions. (2008).

[36] Agus Sudjianto, Sheela Nair, Ming Yuan, Aijun Zhang, Daniel Kern, and Fernando Cela-Díaz. 2010. Statistical methods for fighting financial crimes. *Technometrics* 52, 1 (2010), 5–19.

[37] Jun Tang and Jian Yin. 2005. Developing an intelligent data discriminating system of anti-money laundering based on SVM. In *2005 International conference on machine learning and cybernetics*, Vol. 6. IEEE, 3453–3457.

[38] European Union. 2018. Directive (EU) 2018/843 of the European Parliament and of the Council of 30 May 2018 amending Directive (EU) 2015/849 on the prevention of the use of the financial system for the purposes of money laundering or terrorist financing, and amending Directives 2009/138/EC and 2013/36/EU.

Official Journal of the European Union L 156 (June 2018), 43–74.

[39] Xingqi Wang and Guang Dong. 2009. Research on money laundering detection based on improved minimum spanning tree clustering and its application. In *2009 Second international symposium on knowledge acquisition and modeling*, Vol. 2. IEEE, 62–64.

[40] Yingfeng Wang, Huaiquing Wang, Caddie Shi Jia Gao, and Dongming Xu. 2008. Intelligent money laundering monitoring and detecting system. In *European, Mediterranean and Middle Eastern Conference on Information Systems 2008*. Brunel University, 1–11.

[41] Mark Weber, Giacomo Domeniconi, Jie Chen, Daniel Karl I Weidele, Claudio Bellei, Tom Robinson, and Charles E Leiserson. 2019. Anti-money laundering in bitcoin: Experimenting with graph convolutional networks for financial forensics. *arXiv preprint arXiv:1908.02591* (2019).

[42] Zonghan Wu, Shirui Pan, Fengwen Chen, Guodong Long, Chengqi Zhang, and S Yu Philip. 2020. A comprehensive survey on graph neural networks. *IEEE Transactions on Neural Networks and Learning Systems* (2020).

[43] Yue Zhao, Zain Nasrullah, and Zheng Li. 2019. PyOD: A Python Toolbox for Scalable Outlier Detection. *Journal of Machine Learning Research* 20, 96 (2019), 1–7.