A Study of Users, Real Cash Flows And Temporal Activity In The Bitcoin Ecosystem

Rafael Ramos Tubino

THESE de DOCTORAT DE
L'UNIVERSITE CLAUDE BERNARD LYON 1

Ecole Doctorale N° 512
Ecole Doctorale InfoMaths

Discipline : Informatique

Soutenue publiquement le 12/12/2023, par:
Rafael Ramos Tubino

A Study of Users, Real Cash Flows And Temporal Activity In The Bitcoin Ecosystem

---

Devant le jury composé de :

Mme Laurent, Anne	Professeure, Université de Montpellier	Rapporteure
M. Labatut, Vincent	Maître de conférences, Université d'Avignon	Rapporteur
Mme. Bouakaz, Saïda	Professeure Université Lyon 1	Présidente
M. Legout, Arnaud	Directeur de recherche, INRIA	Examinateur
Mme. Robardet, Céline	Professeure, INSA Lyon	Directrice de thèse
M. Cazabet, Rémy	Maître de conférences, Université Lyon 1	Co-Encadrant

Acknowledgements

In the next few lines, I would like to thank the people who have helped me directly or indirectly in carrying out this work.

First of all, I would like to thank my thesis director, Céline Robardet, who was always there to support me and who, with her scientific rigour, ensured that my work reached the level it did.

I would also like to thank Rémy Cazabet, my co-supervisor, with whom I worked most closely during the preparation of this thesis. Rémy helped me constantly and encouraged me at every stage of this work. His support was fundamental to the completion of my thesis. I'm grateful for his patience and I'm happy for the bonds of friendship that have been forged over the last few years.

And lastly, I would like to thank my family. My wife Kátia, for her understanding throughout these years of hard work, and especially my sons, Daniel and Nicolas, who have brought me daily joy, even during the most difficult times.

Abstract

Bitcoin is the oldest cryptocurrency, and among the most active ones. All its transaction data is stored in a decentralized ledger – the Bitcoin blockchain – freely accessible to anyone willing to analyze it. Analyzing the content of this data is the purpose of this thesis. The manuscript focuses on two main research questions: the identification of Bitcoin users, and the characterisation of the activity of those users. In the first part, we propose a method for improving the construction of aggregates of Bitcoin addresses belonging to the same user, by identifying the change output of a transaction using supervised machine learning. The quality of the result is evaluated using a ground truth based on on-chain and off-chain data. We show that the results outperform previous work, but also that identifying the change output of a single user might be a better strategy than the usual objective of considering the whole blockchain as a single problem.

The second part of the work focus on interpreting the users' activity in the Bitcoin blockchain. It particularly focuses on defining the *real economic activity* present in the Bitcoin blockchain, as opposed to artificial transactions driven by the protocol or by users moving money from address to address for technical reasons. Heuristics are proposed aiming to classify users in three categories: Frequent Receivers (FR), Neighbors of FR, and Others. The work shows that FR (being a proxy for commercial entities) represent a small fraction of entities, but concentrate most of the payments, showing a centralization in the bitcoin ecosystem. A temporal study is also conducted, allowing us to estimate the geographical location of users. We notably use this information to quantify the bias of a dataset commonly used in the literature for entity tagging.

Résumé

Le bitcoin est la plus ancienne crypto-monnaie et l'une des plus actives. Toutes ses données de transaction sont stockées dans un registre décentralisé – la blockchain Bitcoin – librement accessible à toute personne désireuse de l'analyser. L'analyse du contenu de ces données est l'objet de cette thèse. Le manuscrit se concentre sur deux questions de recherche principales : l'identification des utilisateurs de Bitcoin, et la caractérisation de l'activité de ces utilisateurs. Dans la première partie, nous proposons une méthode pour améliorer la construction d'agrégats d'adresses Bitcoin appartenant à un même utilisateur, en identifiant la sortie de changement d'une transaction à l'aide de l'apprentissage automatique supervisé.

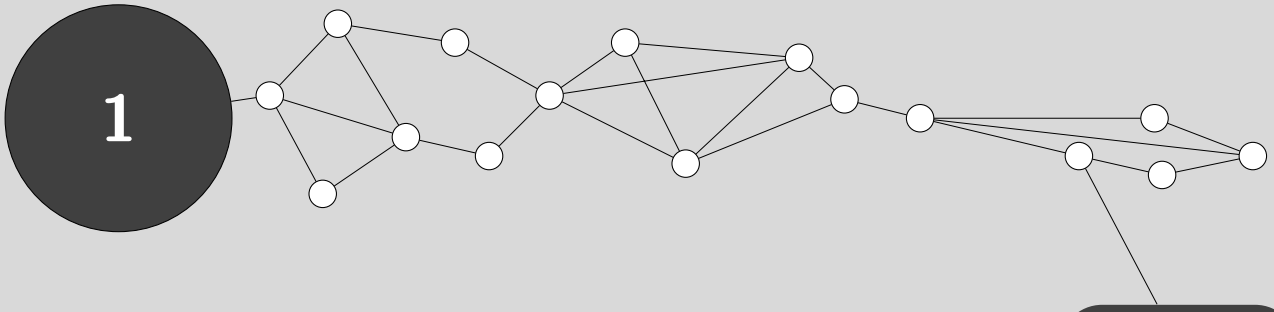
La qualité du résultat est évaluée à l'aide d'une vérité de terrain basée sur des données on-chain et off-chain. Nous montrons que les résultats sont plus performants que les travaux précédents, mais aussi que l'identification des modifications apportées par un seul utilisateur pourrait être une meilleure stratégie que l'objectif habituel consistant à considérer l'ensemble de la blockchain comme un seul problème.

La deuxième partie du travail se concentre sur l'interprétation de l'activité des utilisateurs dans la blockchain Bitcoin. Nous visons définir *l'activité économique réelle* présente dans la blockchain Bitcoin, par opposition aux transactions artificielles induites par le protocole ou par des utilisateurs déplaçant de l'argent d'une adresse à l'autre pour des raisons techniques. Une heuristique est proposée pour classer les utilisateurs en trois catégories : Les récepteurs fréquents (FR), les voisins des FR et les autres. Les travaux montrent que les FR (qui sont une approximation des entités commerciales) représentent une petite fraction des entités, mais concentrent la plupart des paiements, ce qui témoigne d'une centralisation dans l'écosystème du bitcoin. Une étude temporelle est également menée, nous permettant d'estimer la localisation géographique des utilisateurs. Nous utilisons notamment ces informations pour quantifier le biais d'un ensemble de données couramment utilisé dans la littérature pour le marquage des entités.

Contents

List of Figures

List of Tables



1 Introduction

1.1 Thesis context

Over the past decade, Bitcoin has become increasingly prominent in global exchanges, making frequent appearances in news headlines and serving as the focal point of many friendly conversations. It is remarkable how this digital currency has permeated our collective consciousness, even among those who may not fully grasp its intricacies. Throughout its journey, Bitcoin has found itself in the sights of both criticism and admiration. Detractors often argue that it facilitates cybercrimes, money laundering, or speculative trading, citing its relative anonymity and decentralized nature as potential breeding grounds for illicit activities. On the other hand, its proponents eulogize its virtues, seeing it as a potential solution to the limitations of centralized banking systems. They consider that Bitcoin enhances financial privacy and provide global accessibility to a decentralized monetary system.

We can say that the criticisms and appreciations surrounding Bitcoin all have their validity, reflecting the multifaceted nature of this revolutionary digital currency. Bitcoin, first introduced in a 2008 paper attributed to the pseudonymous author Satoshi Nakamoto [Nakamoto, 2008], made its official launch in the subsequent year. Bitcoin operates on a complex but elegant system of decentralized technology that allows for secure peer-to-peer transactions without the need for a central authority. At the heart of Bitcoin is the blockchain, a public ledger that records all transactions ever made in the Bitcoin network. It is a chain of blocks, where each block contains a set of transactions. This blockchain is maintained and updated by a network of computers called *nodes*, spread across the globe. It is public and can be read by anyone. Unlike traditional currencies, Bitcoin is not controlled by any central authority, such as a government or a central bank. Instead, it relies on a decentralized network of nodes, each with a copy of the entire blockchain. These nodes work together to validate and record transactions. A transaction records the transfer of money between users. This transaction includes the sender's public key (known as an address), the

recipient's public key, the amount of Bitcoin being transferred, and a digital signature for security. Transactions are broadcast to the Bitcoin network. Nodes in the network collect and validate the transaction to ensure that the sender has sufficient funds to make the transfer and that the digital signature is valid. This validation process helps prevent double-spending, where the same Bitcoin is spent more than once. Validated transactions are grouped together into a block. Miners, special nodes in the network, compete to solve complex cryptographic hash computation based on the transactions in the block. They make possible the verification of new transactions against the Bitcoin network. Because anyone may become an active validating node, it is not easily controllable by a unique person or company, making it a decentralized environment. Other nodes in the network verify the solution and, if correct, add the new block to their copies of the blockchain. This is how consensus is reached in the Bitcoin network. Miners are rewarded with new Bitcoins and transaction fees for their efforts. Once added to the blockchain, the transaction is considered confirmed. It becomes a permanent part of the public ledger and cannot —in principle— be altered. Bitcoin's security is maintained through the immense computational power required for mining. The decentralized nature of the network, combined with the proof-of-work consensus mechanism, makes it highly resistant to censorship and fraud.

One of the core attributes that sets Bitcoin apart is its accessibility. Since it operates over the internet, anyone with an internet connection can participate in the Bitcoin network, irrespective of geographic location or affiliation with a specific institution. Within this network, Bitcoin users can freely generate new addresses at any time. This aspect enhances privacy, as these addresses are not directly tied to physical individuals or entities, fostering a level of anonymity. However, this enhanced privacy feature has also attracted criticism. Bitcoin's potential for anonymity and pseudonymity can be exploited by cybercriminals for illegal activities like money laundering and tax evasion. This has sparked concerns among regulators and governments worldwide, leading to efforts to implement anti-money laundering.

The Bitcoin network is said to be *pseudonymous,* because the addresses are not directly linked to a person. On the other hand, as all validated transactions are freely accessible by everyone at any time, it is possible to explore the transactions and their details (sender addresses, recipient

addresses, amounts, fees, date, block number) and thus potentially break part of the anonymity.

1.2 Statement of the Problem

In this thesis, our objective is to analyze the uses of the Bitcoin currency, from the analysis of the data available in the Bitcoin blockchain. We address two problems: the identification of the multiple addresses belonging to the same user, and the study and categorization of users and money flows based on a temporal study of transactions.

1.2.1 Aggregation of addresses of a single user

Even though all transaction data is freely accessible, one cannot link bitcoin addresses to the identity of their owner, from the blockchain information. An important step in analyzing Bitcoin transactions is to group Bitcoin addresses belonging to the same user. This allows access to exchanges between users, while reducing the complexity of further analysis and making it easier to interpret the transaction flow. Indeed, a single user can have several hundred thousand addresses, and if we want to understand the activity in Bitcoin, it is necessary to identify the users beyond the addresses. Many previous works have proposed different methods to create these clusters. These methods rely on the use of complex network algorithms, heuristics and/or machine learning models. However, it is important to note that there is no perfect method to date and the generated clusters may be incomplete or have addresses belonging to multiple users.

The approach that we propose to improve the identification of clusters of addresses of the same user takes advantage of the specificities of bitcoin transactions: 1) In Bitcoin transactions, each entry corresponds to an address and contains the entire amount associated with a previous payment to that address. 2) Another fundamental property of Bitcoin transactions is that the sum of the inputs of a transaction is equal to the sum of the outputs plus transaction fees. These properties form the basis of our analysis. If there is a surplus in a transaction – that is to say if the sender's Bitcoin amount is greater than what the recipient should get – the surplus is typically redirected to an address belonging to the sender. This surplus is what we refer to as a "change output". Thus, it is possible to make hypotheses as

to whether the addresses in input and output belong to the same users. In our research, we have developed a machine learning process and method to improve the identification of these change outputs.

1.2.2 Real flow analysis

Due to the Bitcoin protocol and users' behaviors, many transactions present in the blockchain do not represent an actual transfer of money from an user to another, but simply internal management of a user's funds, or technical operations. It is not a trivial task to conduct a study of the real flow of money in the Bitcoin environment due to all these spurious transactions. For this reason, many studies attempting to analyze the flow of bitcoins may result in false or biased results, which do not represent reality. In a second contribution, we study the *real economic activity* in the Bitcoin blockchain, i.e., transactions involving retail users and/or retail companies. We first introduce a heuristic method to classify Bitcoin players into three main categories: Frequent Receivers (FR), Neighbors of FR, and Others. We show that most real transactions involve Frequent Receivers, representing a small fraction of the total of addresses according to the blockchain, but a significant fraction of all payments, raising concerns about the centralization of the Bitcoin ecosystem. We also conduct a weekly pattern analysis of activity, providing insights into the geographical location of Bitcoin users and allowing us to quantify the bias of a well-known dataset for actor identification.

1.3 Contributions

In more details, our contributions are as follows.

- **Automatic identification of change output:** A method for identifying change output is proposed. It uses existing supervised learning methods but introduces three new ideas:
  - **More accurate ground truth:** In previous work, the ground truths could be misleading, due to incomplete or biased information. We propose a new ground truth whose importance is confirmed experimentally by the improved identification of ex-

change addresses leading to a more reliable interpretation of the results.

  - **Data filtered for training:** Not all previous data may be relevant or reliable for the training phase. Although previous works used the training data indiscriminately, filtering the training dataset is crucial. We propose a method that allows obtaining more reliable data for this step.
  - **Heuristics for filtering predicted data:** Predicted data is not completely reliable and requires post-processing. Three heuristics are proposed to improve the results and are evaluated experimentally.

- **Output identification for a single user:** Existing work focuses on identifying all change outputs simultaneously for all actors, making the implicit hypothesis that they have the same behavior. We propose on the contrary to use a single-user-oriented study to better control the quality of address clusters, and use feature analysis to show the relevance of this approach.

- **Frequent Receiver analysis:** In order to be able to examine the actual flow of Bitcoin, we proposed a method to identify companies and their customers. We proposed to group users into three groups based on their activity: Frequent Receivers (FR), representing users who sell a product or service, First Neighbors of FR (N1), representing customers, and The Others (TO), representing users who do not interact with FR. Our goal is to monitor the use of Bitcoin consistent with being a currency used in a real economy, as opposed to financial trading activities, technical transactions, and transactions not involving an actual exchange of value with a user. We conducted a study to show the relevance of each of these types of users, excluding transactions that do not represent a real Bitcoin flow. The work shows that a relatively small number of actors are responsible for numerous transactions and are involved in a large proportion of the amounts exchanged.

- **Temporal Analysis:** Since Frequent Receivers are supposed to have frequent interactions with clients, we have studied their behavior in more detail from the rich information about payments they are in-

volved in. For each entity, we computed the payment volume for every weekly hour over a year. Assuming that daily activity follows a similar pattern everywhere on average, we propose finding an optimal alignment that minimizes the sum of absolute differences between weekly patterns. The alignment process allows us to estimate the time zone of FR entities. Although it provides an estimate, and does not distinguish between countries with similar time zones, it makes it possible to estimate the geographic distribution of the main Bitcoin players and to monitor the evolution of this distribution over time.

1.4 Structure of the thesis

- **Chapter 2 - Bitcoin:** The mode of operation and the Bitcoin network are presented in this chapter. Many features are explained so that the reader has the necessary knowledge for the following chapters. What is a block, what miners do, how miners are rewarded, are some of the topics covered.

- **Chapter 3 - State-of-the-art:** Previous work is described in this chapter which presents the methods developed in the literature to study Bitcoin data.

- **Chapter 4 - Construction of supervised datasets for change output prediction:** To train predictive models to recognize that a transaction output corresponds to a change address, we need a supervised dataset. This chapter explains how such a dataset is constructed based on reliable heuristics and external information.

- **Chapter 5 - Automatic discovery of change augmented Aggregates:** In this chapter, we present a method for identifying the change output of a transaction. This method is then used for aggregating addresses belonging to the same user.

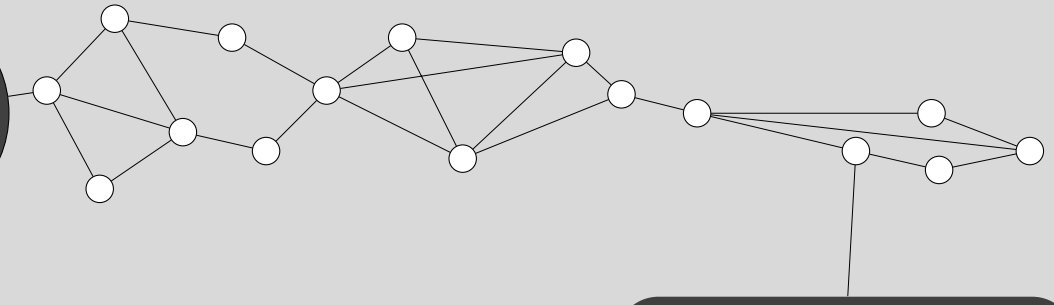
- **Chapter 6 - User specialized change detection:** Because a global study of change output identification may be very time and resources consuming and very rarely of interest, we study in this chapter a way to improve the address aggregate of a target actor by specializing the

training for a single actor. Several users and models are used in experiments in order to test the proposed method.

- **Chapter 7 - Frequent receivers and temporal analysis:** Due to the way Bitcoin works and the particular practices users may adopt, many transactions do not represent an actual flow of currency. In this chapter, we propose a study to analyze the real flow of Bitcoin. The chapter also includes a temporal study leading to the geographic identification of users.

- **Chapter 8 - Conclusion:** This chapter concludes and gives future directions for this work.

2



The Bitcoin cryptocurrency

2.1 Bitcoin principle

The main idea behind Bitcoin's creation [Nakamoto, 2008] was to introduce a decentralized, trustless, and secure digital currency that could empower individuals, resist censorship, and offer an alternative to traditional financial systems. It aimed to address issues related to centralization, inflation, and financial accessibility while leveraging cryptographic technology to ensure the integrity and security of transactions. Bitcoin was designed to operate without a central authority, such as a government or central bank. It aimed to eliminate the need for intermediaries in financial transactions, giving users more control over their money. As a trustless system, its participants can transact with one another without needing to trust a third party. Trust is instead placed in the cryptographic algorithms and the transparent, immutable nature of the blockchain. Bitcoin sought to provide a means of conducting financial transactions that would be resistant to censorship and government control. By decentralizing control, it aimed to prevent governments or other entities from freezing or confiscating funds. The security of the money is ensured by cryptographic mechanisms. This technology made it extremely difficult for unauthorized parties to access and manipulate transactions. The Bitcoin protocol offers a limited supply of coins, capped at 21 million. This scarcity was intended to prevent inflation and maintain the value of the currency over time. Bitcoin was designed to be accessible to anyone with an internet connection. This accessibility aimed to empower individuals who lacked access to traditional banking services. Bitcoin facilitated peer-to-peer transactions, allowing individuals to send funds directly to one another without the need for banks or other intermediaries. This feature aimed to reduce transaction costs and delays. The transparency of the blockchain is intended to deter fraudulent activity and provide an auditable record of all transactions. It is a global currency that could be used for international transactions without the need for currency conversion or high exchange fees.

2.2 An alternative means of payment

As mentioned, Bitcoin was created as an attempt to solve problems existing with other online means —e.g., bank transactions, payment cards, prepaid

cards, e-Wallets. These means of payment may not meet the expectations of some people involved in online monetary transactions, for instance due to the following limits or properties:

- trust-based model: all transactions are carried out by a third-party financial institution, which must be trusted by the sender of the transaction as well as the recipient;
- non-irreversible transactions: the institution mediating the transaction can cancel or reverse the operation;
- transaction costs: intermediaries fees can be somewhat high and thus limit the minimum value of a transaction;
- availability: some payment methods may not be available in some regions or countries;
- lack of anonymity: due to the trust required in some institutions, in some cases users are forced to provide more information than they want.

2.3 Transactions at the heart of the blockchain

Figure 2.1 is a schematic representation of a Bitcoin transaction. Each transaction has at least one sender and one or more recipient addresses. The sender must digitally sign the transaction, in order to prove he owns the bitcoins that he is about to spend. These bitcoins must have been received by the current owner in a previous transaction. To complete the transaction, the sender must have the receiver's public key (also known as Bitcoin address), the hash (identifier) of the previous transaction (where these bitcoins where received by the sender), and his own private key. Public and private keys exist in a ratio of 1:1. Public keys must be used to receive payments and private keys are used to sign payments. In this way only someone in possession of both, public and private keys of the emitting bitcoin address, can make a payment.
There is a special type of transaction, called *Coinbase* transactions [Bashir and Prusty, 2019], which appear once in each block. These transactions correspond to the generation of bitcoins resulting from the mining operation. They have no inputs and can have an unlimited number of outputs. They

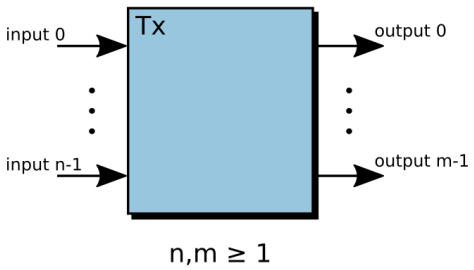


Figure 2.1: A Bitcoin transaction.

are the transactions that reward miners who were able to calculate a nonce ("Number Only used oNCE") value (see Section 2.6). The amount created in a Coinbase transaction varies over time depending on rules defined in the Bitcoin protocol, the *halving system*.

Bitcoin protocol uses a transaction model called UTXO (Unspent Transaction Output), as opposed to the usual approach of money transactions, based on *user accounts*. In a user account principle, the multiple payments received by a single user are merged in their account, as we are used to with traditional bank accounts. Instead, in the UTXO mode, each transaction is composed of one or more entries and exits, each entry being the exit of another transaction, except for the very specific case of Coinbase transactions. A user receiving two payments thus control two UTXO —-output of transactions that are not yet spent. The sum of all UTXOs controlled by a user corresponds to the available balance of that usre. Each UTXO is associated with an address and an amount of cryptocurrency that can be spent in transactions. Managing UTXOs is important for the security and efficiency of the blockchain. Blockchain nodes verify each transaction to ensure that the UTXOs used to fund the transaction are valid and have not already been spent in the past. UTXOs are also important in ensuring the transparency and integrity of the blockchain by allowing users to verify the full transaction history: by tracking money from UTXO to UTXO, we can trace back the entire history of Bitcoins present in a particular UTXO back to their original creation.

2.4 Security based on asymmetric cryptography

Bitcoin keys can be generated freely, by anyone, at any time. What makes the system secure and unlikely to be broken, is the low probability of the same keys being created twice. The hash algorithm used for creating Bitcoin addresses is RIPEMD-160. For this reason, there are 2^{160} possible pairs of private/public Bitcoin keys.

2.5 Organization in *chain of blocks*

One of the challenges in this payment system is that the recipient of a transaction cannot verify whether the sender has not already spent the same coins at the exact same time with another user, essentially attempting a double-spending. To address this issue, all transactions must be publicly known, and only the *first* spending of an UTXO is considered valid.

To achieve consensus among all network nodes regarding the *order* of transactions, the blockchain is organized in a sequential manner. Periodically, a *block* of transactions is created, containing a set of transactions. This block is timestamped, and contains a reference to the previous block in the chain. Each new block added at the end of the chain strengthen the inalterability of the older ones: indeed, each block is summarized by a hash, containing information about all the transactions it incorporates, along with the hash of the previous block, creating a chronological link between blocks and giving rise to the "chain" structure of the blockchain as depicted on Figure 2.2. Altering a transaction in an old block would thus require altering all the hashes of all the transactions added afterwards.

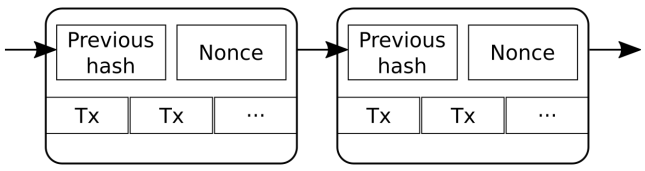


Figure 2.2: Blockchain representation.

2.6 Proof-of-work mechanism

To ensure the functionality of the blockchain, a proof-of-work system is implemented [Ghimire and Selvaraj, 2018]. The fundamental concept here is to search for a specific *nonce* value that, when used in the hashing process, results in a hash starting with a predetermined number of zero bits. While the verification of this hash can be done quite easily, the computational effort required to find the suitable value is exponentially challenging.

Nodes participating in this process, often referred to as miners, strive to discover this nonce value that produces the desired hash. The miner who successfully finds this nonce is rewarded with the transaction fees paid by users for their transactions. Additionally, when new bitcoins are created, they also go to the miner responsible for solving the proof-of-work puzzle. Bitcoins are created through Coinbase transactions, which occur once per block. When the Bitcoin network was initially launched, each block rewarded miners with 50 BTC. However, the Bitcoin protocol has a built-in mechanism called "halving" (see Table 2.1), which occurs approximately every 210,000 blocks. After a halving event, the block reward is divided by 2. As of the time this manuscript is being written, the block reward for miners is 6.25 BTC. This means that miners receive 6.25 Bitcoins for successfully mining a block. However, it is important to note that the value of Bitcoin can fluctuate significantly over time when converted to other currencies, such as the U.S. dollar. For instance, at the current exchange rate of approximately 1 BTC to 30,000 USD, the reward paid to miners for each block is more than USD 180,000. The dynamic nature of Bitcoin's value thus impacts the rewards earned by miners for their efforts in securing the network and processing transactions. Bitcoin's halving mechanism ensures that the rate of new Bitcoin creation decreases over time, ultimately capping the total supply of Bitcoin at 21 million coins. This scarcity is one of the factors that can influence its value in the global market.

Transaction fees in the Bitcoin network are contingent on the amount of data within the transaction. Unlike traditional financial institutions, which charge a percentage of the transaction amount, Bitcoin fees are fixed by a market mechanism: each block can contain only a maximal amount of bits. Each transaction requires a certain amount of bits, depending for instance on the number of inputs/outputs, but not —or very loosely— related to

	block	BTC
Bitcoin launch	0	50
1st halving	210000	25
2nd halving	420000	12.5
3rd halving	630000	6.25
4th halving	840000	3.125
5th halving	1050000	1.5625
6th halving	1260000	0.78125
7th halving	1470000	0.390625
8th halving	1680000	0.1953125
9th halving	1890000	0.09765625

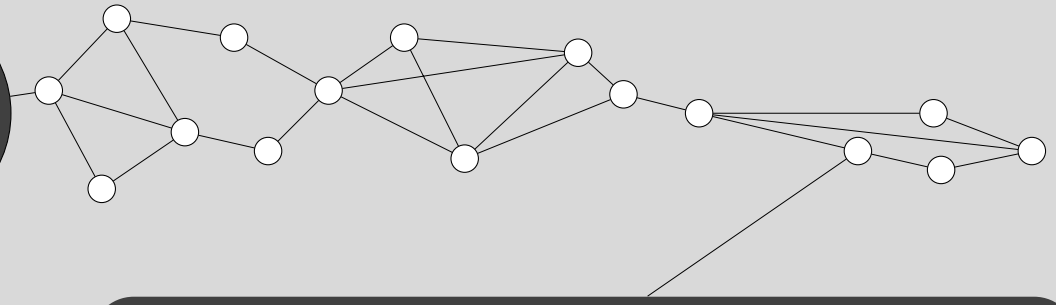
Table 2.1: Values paid after some blocks (non-exhaustive list).

the amounts transfered. Thus a transaction $t1$ requiring twice the amount of bits compared with a transaction $t2$ will also roughly require twice its transaction fees. Thus, besides the block reward, miners also earn transaction fees as part of their compensation for adding a new block to the Bitcoin blockchain. Unlike the fixed Coinbase reward, transaction fees are not set by the network but are determined by the users who initiate the transactions. Miners prioritize transactions based on the fees attached to them, aiming to maximize their overall profit. Miners consider several factors when selecting transactions to include in a block. They are incentivized to include transactions with higher fees because these contribute more to their earnings. Users who want their transactions to be processed quickly often attach higher fees to incentivize miners. Miners also consider the size of a transaction in bytes. Some miners calculate the fee rate, which is the fee amount divided by the transaction size (in bytes), and prefer transactions with higher fee rates as they offer better compensation relative to the data size. This fee market within Bitcoin serves as a mechanism to allocate limited block space efficiently. Users can choose their transaction fees based on their urgency and willingness to pay, and miners are motivated to include transactions with higher fees to optimize their revenue. It's a dynamic system that adjusts to network conditions and user preferences.

Another critical aspect of the Bitcoin system is determining when a block is accepted into the blockchain. This requires validation from a majority of nodes within the network. Each CPU in the network holds one vote, not one IP address. When multiple chains co-exist in this distributed system,

the majority consensus consists considers that the longest chain of blocks, indicative of the most extensive proof-of-work effort, is considered the valid chain. This mechanism ensures the integrity and security of the blockchain by requiring the consensus of the majority of participants.

3



State-of-the-art methods for analyzing Bitcoin transaction data

3.1 Introduction

Bitcoin transaction analysis involves first grouping transactions originating from the same user and then either identifying the users or analyzing their activities within the Bitcoin network. This grouping process is essential because Bitcoin transactions are pseudonymous by design, as they are linked to addresses rather than real-world identities. Clustering transactions helps deanonymize users by linking multiple addresses to a single entity. Researchers and authorities use clustering to trace the movement of cryptocurrency, especially in the context of illicit activities like money laundering or fraud. Grouping transactions can reveal patterns in a user's behavior.

Another preliminary task needed for Bitcoin transaction analysis consists in de-anonymizing users: even if we know the multiple transactions of a single user, we still ignore their identity. Using external databases or public information, the analyst can gather information about pseudonymous users, thus uncovering either their identity or hints about this identity, such as a category of user.

The analysis of user activity within the Bitcoin network provides insights into their behavior and intentions. This generally covers the study of transaction Frequency. How often a user engages in transactions can reveal his level of activity and involvement in the cryptocurrency ecosystem. The size and value of transactions can indicate the user's financial activities, whether they are involved in microtransactions, large investments, or other activities. Understanding with whom a user frequently interacts can provide clues about their network of counterparties, be it individuals, businesses, or exchanges.

Bitcoin transaction analysis is not limited to academic research. Governments and regulatory bodies employ these techniques for various purposes, such as anti-money laundering or fraud detection.

In the following, we review the most important research contributions in those directions.

3.2 Address Clustering

Bitcoin addresses are freely and easily generated. Creating many addresses may enhance users' privacy: users can receive payments on multiple ad-

dresses, without explicit relation between them; they can also move funds among their own addresses with the objective of complexifying the tracking of their coins. For this reason, it is common for a user to have multiple Bitcoin addresses. This makes the analysis of the transaction graph more difficult. It is therefore necessary to group the addresses of the same user, with the aim of simplifying the analysis of transactions and reducing computational costs. This task is not straightforward as addresses are not directly related to each other nor to the identity of the owner. There is currently no method considered fully reliable for clustering user addresses, and all methods result in approximations, errors, and incompleteness; address clusters may in particular contain the addresses of several users.

The final objective of address clustering is usually to produce en *entity graph*. From the raw data in the blockchain, one can construct an *address graph*, in which nodes are addresses and edges represent a transfer from one address to one or multiple other addresses. This graph is particularly complex, due to the multiple inputs/outputs of a single transaction, and is better represented as a multigraph, with bitcoin transactions as multi-edges. By grouping addresses, one instead naturally creates (see Figure 3.1) a simple directed graph — all the addresses in input belonging naturally to the same entity as explained later.

Different techniques have been designed, either directly derived from the Bitcoin protocol or reflecting common practices, to produce what is called an entity graph. The main approaches are detailed below.

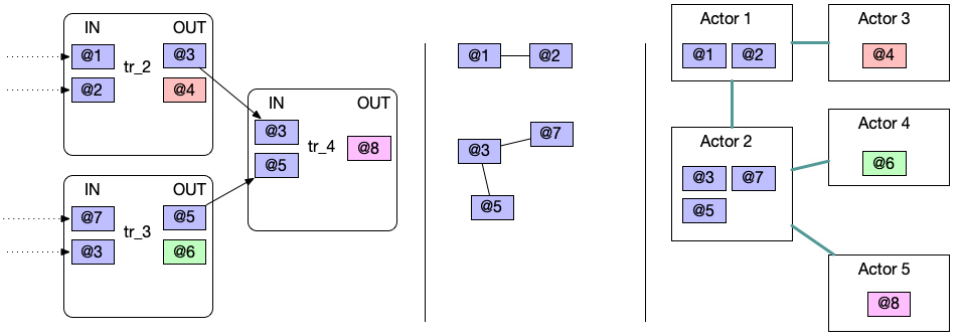


Figure 3.1: Multi-input heuristic (H1) 3.2.1. Left: Three Bitcoin transactions, involving multiple inputs and multiple outputs. Center: Network of co-spending built from the data on the left. Right: Network of transactions between entities resulting from combining left and center.

3.2.1 Using heuristics

Many heuristics have been proposed in the literature with the goal of clustering Bitcoin's entity addresses; however, two of them stand out as the most commonly used.

Multi-input heuristic (H1). First mentioned by [Nakamoto, 2008] and widely used, this heuristic is based on the Bitcoin transaction protocol. It indicates that all addresses that appear as input in a transaction belong to the same entity. It is known that CoinJoin transactions – a method of obfuscation in which different users combine their inputs into a single transaction– allow two separate entities to make a single payment, but [Harrigan and Fretter, 2016] conducted a graph analysis of labeled data from known addresses and showed some reasons for which this heuristic is effective. [Nick, 2015] used Bloom filters to evaluate the validity of this heuristic and concluded that it can identify more than 69% of the addresses in an entity's wallet.

One-time change output heuristic (H2). Due to the UTXO nature of Bitcoin transactions, which requires the sender to spend the full amount received in a previous transaction, the difference between the amount to send and the amount in input of the transaction is sent back to an address belonging to the same user making the payment. In some cases, it is easy to recognize such a change output because the change address is the same as one of the addresses used as input to the transaction. One can also recognize the change output *a posteriori* based on the heuristic H1: the output address is used as input to another transaction along with one of the input addresses of the current transaction. For all other cases, the identification is not direct and this is the subject of this second heuristic. On top of this, users looking for improved anonymity use Bitcoin addresses only once, meaning that they send the surplus to an address that will never receive any more payments, thus participating in only one more transaction — as the sender. Heuristic H2 analyzes the following characteristics of the transactions:

- The transaction is not a CoinBase transaction. Indeed, CoinBase transactions are generated by the Bitcoin system and not by a user. There is thus no surplus, so no amount to be sent back using a change

output;

- None of the output addresses is also an input address of the transaction. If one address is also an input address, it will be considered the change output;
- There are only two outputs. Transactions having two outputs are more likely to have exactly one payment output and one change output, thus facilitating the task of identifying the change output;
- Only one of the output addresses appears for the first time in the blockchain. As mentioned, a one-time change output will send the surplus to an address that will be only used once;
- The address that appears for the first time no longer appears as an output in the blockchain. For the same reason as the previous item.

Many studies use a variation of this heuristic such as [Androulaki et al., 2013, Chang and Svetinovic, 2018, Ermilov et al., 2017, Meiklejohn et al., 2013, Kappos et al., 2022, Cazabet et al., 2018, Zhang et al., 2020, Wang et al., 2020, Neudecker and Hartenstein, 2017]. All these variations attempt to identify the one-time output of the change based on some information using one or more of the listed characteristics.

Address reuse-based change address detection. This heuristic presented in [Zhang et al., 2020] is inspired in H2. The main difference is that one output may be considered a one-time change output if the other outputs are found as an output of another transaction in any moment in time.

Two other heuristics can be found in the literature, but as they are related only to the mining process, they concern a more restricted number of transactions.

Coinbase transaction heuristic. Having no input and any number of outputs, Coinbase transactions [Zheng et al., 2020] are used to reward miners. As only the first one able to compute the nonce value (see Section 2.6) is rewarded, one may consider all the output addresses as belonging to the miner; they can thus be aggregated in the same cluster.

Mining pool heuristic. For transactions having at least one known mining address as an output, and n_o number of outputs ([Zheng et al., 2020] used $n_o = 100$), one may consider it as a mining pool associated transaction, and cluster all the output addresses together. Although somewhat bold, this reasoning was validated by [Lewenberg et al., 2015].

3.2.2 Using patterns

Bitcoin transactions follow certain patterns and structures that can be analyzed to understand how they work and to detect various types of activities. Many transaction patterns have been listed by [Ferrin, 2015]. The authors of [Chang and Svetinovic, 2018] use these patterns to create address clusters.

Relay pattern. It is a single transaction pattern whose transaction has a single input and a single output. This is a surprisingly popular type of transaction. As it is unlikely that the amount associated with an address will be exactly the same as that needed for a payment, this type of transaction is commonly seen as being used to transfer coins from one address to another belonging to the same user. Thus, the two addresses of the transaction are considered to belong to the same cluster of entities.

Sweep pattern. A sweep transaction has multiple inputs and a single output. This type of transactions is used to consolidate money associated with multiple addresses into a single address. It is frequently used to facilitate coin management. In this case, all addresses associated with the transaction are merged if the number of entries is greater than that observed on average.

Distribution pattern. It refers to transactions having any number of inputs and three or more outputs. These transactions are commonly used when a user makes a group payment. It can have multiple inputs because the user collects coins from multiple previous payments, and have more than two outputs because, besides the change output, many payments are made. If more than 75% of the output addresses already belong to the same cluster, and the majority of the input addresses already belong to the same cluster, the authors merge the clusters.

3.2.3 Using physical network data

Physical network data may be used in order to generate Bitcoin address aggregates. Regardless of the fact that IP addresses are not stored in transaction data, a Bitcoin node may have access to this information and link Bitcoin and IP addresses. This knowledge may be used in conjunction with one or more of the presented heuristics (3.2.1) as shown in previous works [Kang et al., 2020, Neudecker and Hartenstein, 2017]. This however requires a heavy setup, since the IP address information is not in the blockchain, but must be collected by continuously listening to the peer-to-peer Bitcoin blockchain network through nodes. This solution is also inefficient when users hide their origin through mechanisms such as Tor.

3.2.4 Using machine learning techniques

Unsupervised approaches. From the network of cryptocurrency transactions, researchers have developed methods to extract user behavior preferences, such as frequent transaction patterns, average transaction amounts, and transaction frequencies. Several approaches have been proposed to group addresses based on similarities in transaction behaviors.

In an early work, [Reid and Harrigan, 2011] discuss the possibility of clustering addresses of the same user. They discuss the H1 heuristic and mention several ways to improve over it, using off-chain or on-chain information. In a case study, they show how some addresses can be unidentified as belonging to the same user, considering the topological structure of transaction graphs. They show that such networks have a non-trivial topological structure, that has implications for anonymity.

In [Androulaki et al., 2013], the authors focus on the privacy provided by Bitcoin when it is used to support the daily transactions of individuals. It is a theoretical work, not using on-chain data, but simulated transactions following the UTXO principle. On top of H1 and H2, they introduce the concept of *Behavior-based* analysis, according to which one could attack the privacy of Bitcoin users, i.e., perform address clustering, based on the transaction history. They consider various transaction behavior features, including transaction times, sender-recipient indexes, and transaction amounts.

The work presented in [Monaco, 2015] proposed using multiple temporal features to capture the dynamics of Bitcoin users' transaction behavior and

observed that these patterns over time can reveal users' identities. The work is theoretical and does not propose a practical way to do large-scale address clustering. It can work only on users having a large number of transactions (100+ per month), since it is based on the comparison of time series of activity. More precisely, They use the phase space reconstruction technique [Kantz and Schreiber, 2004] which is useful for reconstructing all the dynamic variables of the system while preserving its properties. Their early work demonstrates the feasibility of some user identification from blockchain data.

In [Cazabet et al., 2018], the authors proposed a method based on network clustering, i.e., community detection, for address clustering. It consists of constructing a graph where the vertices of the graph are the clusters from H1 which are connected if they contain addresses involved in the same transaction. A community detection algorithm is then applied to this graph to search for dense subgraphs. The authors compared the performance with three existing heuristics: H1 and two different versions of H2. The results showed that the proposed heuristic had the best recall score but low precision, and was only partially capable of improving over H1. Performances obtained with H2 were much lower.

[Shao et al., 2018] propose to use deep neural network and clustering methods to identify change addresses. In input of their framework, they describe each address by some statistical features describing the characteristics of the transaction history in which the address appears. The feature vector associated with an address is based on transaction history and is represented as a variable length sequence where each element is a concatenation of features related to a transaction. The length of the sequence corresponds to the size of the transaction history. From these sequences, they build a representative vector in Euclidean space to ease the comparisons between sequences. To that end, they use Skipgram (Word2Vec) algorithm [Mikolov et al., 2013a, Mikolov et al., 2013b] to generate transaction embeddings that are combined with manually extracted features resulting in vectors of 120 dimensions. Note that this method does not require a labeled dataset, but instead takes only into account the similarity of the feature description of addresses. The resulting vectors are used to recognize an unknown address from a given test set using the k-nearest Neighbors (k-NN) method. They also cluster the addresses using K-Means on vectors pre-processed by PCA.

Due to the computational complexity of their method, they worked only with less than 9000 curated addresses.

[Zhang et al., 2018] introduced a multi-resolution clustering system for de-anonymizing Bitcoin addresses. They first use a classification algorithm to detect the category of some users, and then use a clustering algorithm to group addresses based on some computed features. However, the paper does not provide details on the features used or the details of the method. We only know that the results are evaluated by using H1 clustering as a ground truth, and only 30,000 addresses are considered.

The authors in [Tovanich and Cazabet, 2022, Tovanich and Cazabet, 2023] proposed a method to associate multiple H1 clusters of the same user based on the way coins flow through the Bitcoin user network. Namely, they first identify some *tag entities*, chosen among the most active entities. Then, they use tainted flow tracking to see which of these entities are encountered with what frequency by coins flowing from each entity (i.e., H1 cluster) of interest. Using Graph Neural Networks, they embed the transaction flow of each actor in a vector that they call a *fingerprint*. Then, they use clustering to rediscover groups of H1 clusters probably belonging to the same user, with the hypothesis that two clusters of the same user should have a similar fingerprint

Supervised approaches. At the time of our first publication on the topic, no method existed for address-clustering using supervised learning. The bottleneck to overcome was the lack of a reliable training set, a problem to which we proposed an original solution, as presented in Section 5.3.3. However, approximately at the same time as ours, another work([Möser and Narayanan, 2022]) was published, using a comparable approach. Both works were developed independently in parallel, at about the same time, although the first version of our work was published first.

[Möser and Narayanan, 2022] propose to use machine learning models to detect change outputs within Bitcoin transactions. This is made possible through the creation of a new ground truth dataset obtained from the Bitcoin blockchain. The evaluation of this dataset reveals that, in the majority of cases, it is possible to accurately identify change addresses with a high degree of precision. Furthermore, the paper discusses the application of these machine learning models to cluster change addresses. By imposing

constraints based on the predictions generated by their model, similar to those presented by [Ermilov et al., 2017], they demonstrate that such constraints can effectively prevent cluster collapse. In essence, these constraints help maintain the integrity of change address clusters during the clustering process.

Compared with that work, our contribution uses a different set of output features, and uses different methods to solve the problem of *catastrophic merges*(See Section 5.4). Furthermore, we added a new contribution in which we show that targetting the training on individual users greatly improved the classification performances (Chapter 6).

3.3 De-anonymizing entity identities

Another classic problem of Bitcoin data analysis consists of associating an address, or a group of addresses, to the real-world identity of their owner. Some addresses, such as those associated with well-known entities like WikiLeaks or Silk Road, are publicly disclosed and can be directly linked to specific identities or organizations. Several online services, including online stores and cryptocurrency exchanges, require users to provide identification information before using their services. This information can link cryptocurrency addresses to real-world identities. Web crawlers can also be used to scan social networks and online forums like bitcointalk.org to identify Bitcoin addresses in user signatures or posts. This information can be used to link addresses to specific users [Reid and Harrigan, 2011]. Software tools like BitIodine [Spagnuolo et al., 2014] offer automated analysis frameworks for parsing the blockchain, constructing transaction graphs, applying clustering heuristics, and adding external information to link addresses to users or entities. IP addresses associated with Bitcoin transactions can be used to link transactions to specific geographical locations or network users[Koshy et al., 2014, Biryukov et al., 2014]. Publicly available data from Bitcoin faucets, which record and publish recipient IP addresses to prevent abuse, can also be used for this purpose [Reid and Harrigan, 2011].

3.4 Labelling entities categories

When identifying the real-world identity of an actor is not possible, one can instead target the identification of the category it belongs to, among a predefined number of typical categories. Entities can notably be grouped according to their field of activity. After presenting the main categories of entities, we discuss the main machine learning methods to predict these categories. We end this section with a focus on cybercriminal activity, which is a particular case of actor and transaction category identification, which has been extensively studied in the literature.

3.4.1 Main categories of entities

Exchange. Exchange platforms allow people to convert state currencies into bitcoins and vice versa. Usually, they offer their own marketplaces, where users can trade and exchange different coins. They also allow users to make and receive payments on the blockchain, providing a service similar to a retail bank. These companies are the most common entrance door to Bitcoin, since users can quickly and simply purchase coins on these platforms with dollars or euros, using bank transfer, or even credit card.

Marketplaces. Entities in this category include entities —mostly companies— that sell goods, as well as entities who sell services. While most of this activity concern online shopping, there are also physical stores that accept Bitcoin payments.

Gambling activity. Gambling-related transactions are widespread on the platform. The first games started with simple probability/multiplier bets, occurring entirely on the blockchain. Users send some coins to an address, and receive in return their gains, if any, to the same address they sent from. Nowadays, many alternatives exist, including classic casino games, accessible via full websites, on which users can spend money from a virtual account, accepting payments and payouts on the Blockchain [Gainsbury and Blaszczynski, 2017]. Among the best-known platforms, we can mention SatoshiDice[1]. As there is no guarantee of being refunded, and users must trust the *casino*, some websites share a list of betting Bitcoin transactions,

[1] `http://www.satoshidice.com/`. Last accessed September 30, 2023

followed by reward payment transactions, so that one can check that the platform indeed pay the expected payouts.

Mining Pool. As proof-of-work problems require a lot of processing power, users have started to form pools where they can share their resources in order to have a greater possibility of solving these problems [Tovanich et al., 2021b]. If successful, each participant is rewarded in proportion to the amount of work provided.

Mixer. Mixers are entities offering a service to their customers, which is to make their activities harder to track, and more generally to increase the anonymity of their activities [Moser, 2013, Möser et al., 2013]. The most common approach consists of pulling several amounts received from multiple users in a single account, and then spending from this account as requested by the users. The consequence is that it becomes impossible to associate senders and recipients. This implies that mixers must be trustable agents, as nothing in the Bitcoin blockchain guarantees that they will make payment in return(Unlike in more expressive blockchains such as Ethereum, in which a smart contract could guarantee this behavior). The non-reversible nature of Bitcoin transactions also means that it will not be possible to be refunded, unless this be the intent of the mixer. Such transactions may be linked to money laundering. It requires the use of a reliable third-party institution that mediates between transactions, contradicting one of the primary motivations of Bitcoin.

Faucet. Faucets are web services that reward users in exchange for performing certain actions. These tasks might involve completing a captcha or playing a game. Users often have to install an app to be rewarded.

Criminal Activity. The anonymity of Bitcoin attracts many users looking for an anonymous means of payment for their illegal activities. This may include black market transactions, where illegal items are traded, but also scammers, Ponzi schemes, ransomware or any form of criminal activity (see Section 3.4.3).

3.4.2 Supervised learning for activity identification

Various authors focused on different entity types, but a common setting consists of starting from a set of entities whose type is known and then training a model to recognize these types, based on the local properties of these actors.

[Lin et al., 2019] used different supervised learning models to predict entity categories. They compared results from Logistic Regression, Perceptron, SVM, AdaBoost, Random Forest, XGBoost, LightGBM, and Neural Network. They used 28 features to train the models. However, the training data set was relatively small, comprising 26308 addresses divided into 1353 entities. The results show good performance for LightGBM and Neural Network. LightGBM was able to identify 98% of the mixing entities, but only identified 73% of the faucets. [Toyoda et al., 2018] did a similar study, and were able to identify 94% of mixing entities using Random Forests, but achieved a lower 41% accuracy for exchanges.

[Yin and Vatrapu, 2017] focused on the recognition of criminal entities. They considered 12 different categories, including 5 related to cybercrime. Their data has been pre-processed by a data provider. After testing thirteen supervised learning models, they found that the four best performing models were Bagging, Gradient Boosting, Extremely Random Forests, and Random Forests. The best results was obtained using bagging models, achieving an accuracy of 29.81%.

[Harlev et al., 2018] employed transaction features in a supervised machine learning framework to de-anonymize Bitcoin addresses, particularly focusing on very active entities. Using 434 chosen entities representing a total of 200 Million transactions, they trained a classifier to label their activity, among an extended list of categories such as Exchange, Gambling, RansomWare and Others. The method used to obtain the address clustering for the chosen users is not detailed, because it is provided by a private partner.

Inspired by the work of [Ranshous et al., 2017], which uses transactions network motifs and supervised models, [Jourdan et al., 2018] proposed their own motifs, representing different ways bitcoins flow from one address to another. A LightGBM ([Ke et al., 2017]) implementation of a decision tree was used as a supervised model. The authors reached for instance accuracy, precision and F1 scores of 0.95, 1.0 and 0.97 respectively for gambling

entities.

3.4.3 Cybercrime analysis

Criminal activity has been widely studied in the bitcoin network, which justifies detailing here the most important studies that have been carried out. These studies aim to fight cybercrime taking place in the Bitcoin blockchain.

Dark web marketplaces. An online marketplace called *Silk Road* was established in February 2011, enabling the trading of illegal items. Operating within the TOR[2] network and trading only in bitcoins, Silk Road ensured anonymity for its users. The items traded were mainly illegal drugs, but also prescription drugs, firearms (allowed until March 2012) and fake documents. Silk Road seller's guide defined some unauthorized items, such as stolen credit cards, assassinations and counterfeit currency [Christin, 2013]. Silk Road was the best known anonymous marketplace, but not the only one. Black Market Reloaded, Sheep Marketplace, Atlantis are some other examples. All of the marketplaces mentioned are no longer operational. Silk Road was taken down by the FBI[3] while the others went closed by their owners. The work of [Christin, 2013] has made it possible to better understand the operating of those websites. For this, the author created a Silk Road account and crawled the website for a few months with the aim of collecting data. The study shows the prevalence of drugs offered as products. It also appears that the United States was the source of almost half of the products announced and around 35% of the buyers. This marketplace alone represented 1.2 million USD/month during the period studied. The study carried out by [Soska and Christin, 2015] made it possible to scrape data from 35 marketplaces. The authors identified more than 100 million USD in illegal sales per year during the period analyzed.
The work of [ElBahrawy et al., 2020] shows that the dark web market ecosystem is resilient, as users migrate from one market to another. They used a Chainalysis preprocessed dataset [4] produced using heuristics and machine learning. The work shows that among the nearly 40,000 entities in the dataset, around 8,300 interacted with dark web marketplaces. They also

[2] `https://www.torproject.org/`. Last accessed September 30, 2023
[3] Federal Bureau of Investigation in USA
[4] `https://www.chainalysis.com/`. Last accessed September 30, 2023

discovered that almost 2 billion USD was received via Bitcoin addresses on the Dark Web market between June 18, 2011 and July 24, 2019.

Ransomware

Ransomware is a type of malware that typically blocks the victim's access to their own data and threatens permanent blocking if a ransom is not paid. Once infected, a computer sees its data encrypted and the malware spreads via the local network, but also via the Internet. This type of cybercrime is not exclusive to Bitcoin, but the network's anonymity as well as global access to this type of payment comes in handy for ransomware operators. Some ransomware attacks have been widely covered in the media, such as the WannaCry attack in 2017. It is estimated that more than 300,000 computers were infected worldwide, including thousands of computers from UK hospitals.
The work conducted by [Huang et al., 2018] analyzed the transaction graph and was able to track transactions, from purchase of Bitcoins by the victims, to the conversion of received Bitcoins into fiat currency by the ransomware operators. They also showed how different ransomware families manage their bitcoins using different addresses. A ransomware family called CryptoLocker was investigated by [Liao et al., 2016] from September 5, 2013 to January 31, 2014. In this work, the authors used heuristics and transaction graph analysis to identify more than 310,000 USD in ransom payments during this period.

Money Laundry

Mixers were already introduced in Section 3.4.1 A study by [Möser et al., 2013] uses taint analysis to link payments to and from mixing services. Three mixing services were studied in this work: Bitcoin Fog, BitLaundry and the Send Shared functionality of Blockchain.info. The authors were able to connect addresses sending and receiving payments to BitLaudry, while Bitcoin Fog and Send Shared from Blockchain.info managed to hide the connection between input and output payments.
In another work, [Balthasar and Hernandez-Castro, 2017] perform an analysis of numerous mixers in order to evaluate the quality of service. They completely discourage users seeking anonymity from using DarkLaunder,

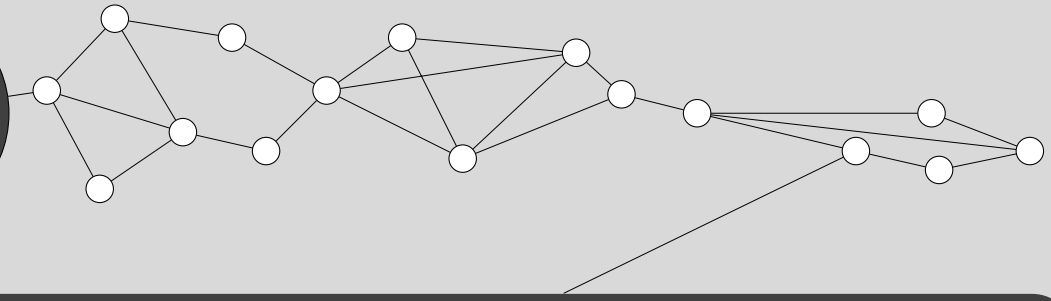
Bitlaunder, and CoinMixer because they have many vulnerabilities in their systems. The study also indicates that even Alphabay and Helix, considered major players, have weaknesses in their anonymizing process. Even worst, [Van Wegberg et al., 2018] tested the services of five different services, but only got refunds from three of them, showing that this type of service can also simply be scams.

3.5 Summary and motivations of the thesis contributions

In conclusion, Bitcoin transaction analysis serves a multifaceted purpose, encompassing user clustering, identification, and activity analysis. It is employed by various stakeholders, including researchers, law enforcement, and regulatory bodies, to enhance transparency, security, and compliance within the Bitcoin network.

In this thesis, we first address the problem of finding a way to determine whether multiple addresses are associated with the same user or entity, i.e., address clustering. This step is essential when it comes to studying Bitcoin transactions; and none of the methods reviewed above manage to accomplish this task adequately. Our second main contribution aims to analyze user activity based on an original definition of user roles. Compared with most other works presented here, it does not require labeled datasets —that are often known to be incomplete and imprecise. It does not focus on one particular type of users of interest, but rather offer a new way to understand the real Bitcoin economy, i.e., involving exchange of money between different entities, a problem that was not addressed in previous works.

4



Construction of a supervised dataset for change output prediction

As seen in previous chapters, the change address of a Bitcoin transaction is an important element to exploit for address clustering. The approach we propose is to learn supervised predictive models to recognize that an output address of a transaction corresponds to a change address. As in state-of-the-art methods, we use descriptors calculated from the Blockchain as features. But unlike previous works, we do not really on unsupervised methods or heuristics, but instead rely on a supervised method. The novelty of our work lies in the proposal of an original approach to generate a large quantity of training and testing examples based on a ground truth constructed from an *a posteriori* interpretation scheme. In this chapter, we first describe the set of transactions studied, as well as the external data considered for the analysis. Then, we explain how example couples (X, y) are constructed for the machine learning process. It involves describing examples by their labels and features. Finally, we present machine learning models and the methods used for their evaluation. The hardware infrastructure and the Python libraries used for implementing our methods are described in Annex A.

4.1 Blockchain Data collection and preprocessing

Since Blockchain data is available to everyone, we collected all data by setting up a Bitcoin node. Bitcoin nodes are computers, running the latest version of the Bitcoin software, whose role is to verify and relay transactions and blocks, maintain a copy of the full blockchain, and enforce the consensus rules of the Bitcoin protocol. Nodes must maintain a full copy of the Bitcoin blockchain, and thus start by downloading the content of the blockchain. In most of the work presented here, we considered data from block 0 (January 3, 2009) to block 667542 (January 25, 2021), representing a total of approximately 600 million transactions. Bitcoin's decentralized nature imposes an efficient coding of the data, which is originally in a binary format. In order to transform the data into a directly exploitable format, we used the Bitcoin-etl (Appendix A) python library, which conveniently takes care of difficulties such as ill-formed Bitcoin addresses, changes in the protocol over time, etc. It outputs a set of JSON files, each transaction being represented as a JSON object. For each transaction, we consider the following information:

- The unique hash of the transaction

- For each output, a unique ID (e.g., 1, 2, 3, etc.), the output address, and the amount associated to that output
- The transaction fees
- The ID of the block containing the transaction
- The timestamp of that block
- For each input, a reference to the associated transaction output, i.e., the transaction hash and output ID.

Note that in the original data, we do not have direct access to the information about input addresses and amounts. Indeed, as explained in the introduction, each transaction input is the output of a previous transaction. It is thus necessary and sufficient to encode, for each input, the information of its origin.

The first preprocessing step consists in retrieving the input addresses. The problem is apparently simple: for transaction t_1, we have access to a reference of the form $(t_2, output_1)$, so we need to access the information of transaction t_2 to retrieve the amount and address of $output_1$. However, due to the size of the data —hundreds of millions of transactions representing hundreds of GB— the problem is not trivial. We realized that usual databases were not adapted to solve the problem in an acceptable time, due to their reliance on hard drive access, while it was not possible to store the data as a dictionary in-memory due to the limits of our hardware, even with 128GB of RAM. We thus had to develop specific optimized codes, and efficient large-data processing tools such as Spark (Appendix A).
The other costly pre-processing step consists of applying the H1 heuristic for clustering addresses belonging to the same user. The problem cannot be solved locally, transaction by transaction, but requires taking all transactions into consideration simultaneously. For instance, one transaction in 2010 might contain in input addresses a_1 and a_2, while a transaction in 2020 might contain in input addresses a_2 and a_3. All three transactions thus belong to the same user, but one needs to consider all transactions simultaneously to discover this information. We solved the problem following [Cazabet et al., 2018]: we build a graph in which Bitcoin addresses are nodes, and an edge connects two addresses if they appear in the input of the same

transaction. More precisely, if a transaction has n addresses in inputs, i.e., $[a_1, a_2, ..., a_n]$, we add to the graph the links $(a_1, a_2), (a_2, a_3), ..., (a_{n-1}, a_n)$. Solving H1 then consists in discovering the connected components of this graph. This task was performed using an efficient network analysis library, SNAP (Appendix A). We thus associated a unique user ID to each address. We then enrich our dataset using this information: each transaction has now a unique user ID in input, corresponding to the sender, and to each output also has an associated user ID.

4.2 Data augmentation

We added two types of external information to our dataset. First, the blockchain information contains only information about the amounts transferred in BTC, not in Dollars. We retrieved from an external source, aggregating historical information from various exchange platforms, the average daily conversion rate from BTC to Dollars. We used this conversion rate to associate to each BTC amount its value in Dollars at the conversation rate of the day it took place.

Another type of information we added concerns user identities. In principle, after applying H1, one needs to collect only one address of a particular entity to identify all its other addresses. Several sources thus provide such databases in which they collect addresses for well-known companies. This information is relatively simple to obtain since one needs to exchange only once with such a company to know one of its addresses. We relied mainly on the `walletexplorer.com` website, a widely used source of such information, which relies on the H1 heuristic to provide all the addresses of a large number of well-known actors. We used a web crawler to retrieve at least one address for each entity in their database. An important element for the rest of our analysis is that, for some entities, the platform is aware of multiple H1 clusters belonging to the same entity. In that case, we collected at least one address for each of those multiple clusters, and retained this information.

We then enriched our dataset by associating the corresponding identity to each of our user IDs, i.e., ID corresponding to a set of addresses as identified by H1. First, this allows us to know who particular actors are, helping us to identify some interesting actors such as Exchange platforms. Second, it makes it possible to identify some known errors of the H1 heuristic (see

Section 3.2.1), especially when the same actor is known to possess multiple H1 aggregates. It thus constitutes a ground truth for our experiments, allowing us to overcome the limit of using H1 as a ground truth, for a method whose goal is to improve over H1.

4.3 Label assignation to transaction output links

The methods we proposed were the first to use supervised machine learning to detect change outputs. The challenge of using this supervised approach is that there are no labels in the original Bitcoin data to know if an output is or not a change output. To overcome this limit, we propose to add such labels to the dataset, by using a two-step approach. In the first step, we apply heuristic H1 to discover the first level of address aggregates.

Definition 1 (H1 aggregates) *Considering a set of transactions $\mathcal{T}$, the set of H1 address aggregates constructed from $\mathcal{T}$ and denoted by $\mathcal{A} = \{A_1, \ldots, A_p\}$ is such that for all i from 1 to p, and for all two distinct addresses $a, b \in A_i$, there exists a sequence of pairs of addresses $P = (a, c_1), (c_1, c_2), \ldots, (c_n, b)$ so that for all $(p, q) \in P$, there exists $t \in \mathcal{T}$ with $p, q \in t.input$.*

In the second step we use this information to attribute partial labeling to the address outputs. In practice, for each transaction output, if we observe that one or more output addresses belong to the same aggregate as the transaction input addresses, then we label them as change addresses, while the other outputs are labeled as payment. We call this approach **H1-labeling**.

Definition 2 (H1-labeling) *Considering H1 aggregates $\mathcal{A} = \{A_1, \ldots, A_p\}$, if there exists a transaction $t \in \mathcal{T}$ with $t.input \subseteq A_i$ and $o \in t.output$ then*

$$\begin{cases} \text{label(o)} = \text{change } \textit{if } \text{o} \in \text{A}_\text{i} \\ \text{label(o)} = \text{payment } \textit{otherwise} \end{cases}$$

H1-labeling has two limits: 1) When an output is labeled as payment, we do not know if it is in fact a genuine payment, or just a limit of the H1 heuristic, which failed to associate the address in output with the aggregate in input. 2) The method works only *a posteriori*, i.e., H1 heuristic is based on the principle that users eventually tend to combine their addresses in

input of the same transaction. But such a combination might occur days, months or years after the address was used for the first time.
To mitigate the first problem, we limit our training to examples having exactly one change address and one payment address. These examples are most likely to be correctly labeled. To mitigate the second problem, we split our dataset so that there are several years after the end of the training dataset to observe the combination of addresses in input of the same transaction, potentially years later.

4.4 Features used to describe output transaction links

Supervised machine learning methods learn a function to predict a target based on item features. In our problem, the target is a binary value, change/payment. An essential part of the process is thus to describe a transaction output with a set of useful features, that provide relevant information to judge if an output is susceptible or not to be a change output. We computed features based on the current transactions as well as previous ones. Computing some of these features already represent a challenge in itself, for instance the **nb_apps** features requiring to compute for each address, for each transaction, the number of time this address appeared before that transaction.
These features are used as input for the supervised machine learning model and are described as follows:

- **value_out**: the value in Satoshi[1] associated to the transaction link;
- **usd_out**: the value in USD associated to the transaction link;
- **nb_apps**: the number of **prior** usages of the transaction output address as an output of another transaction. This feature is related to the H2 heuristic, in which one of the elements used to identify a change address is the fact that it is used for the first time. We use instead the count of the number of times, which gives more flexibility to the machine learning model;

[1] A Satoshi is the smallest unit of a Bitcoin, worth one hundred millionth (10^{-8}) of a Bitcoin

- **total_in**: the sum of values (in Satoshi) associated to the input links of the transaction;
- **total_out**: the sum of values (in Satoshi) associated to the output links of the transaction;
- **dec_bc**: the number of decimals of the Bitcoin value (in Satoshi) associated to the output link;
- **perc_out**: the output value divided by the sum of all output values of the transaction;
- **id_out**: the index of the output in the transaction (if there are 3 outputs to a transaction, they are labeled with indices {1,2,3}). It has been shown [Meiklejohn et al., 2013] that some wallet managers are biased, and tend to systematically put the change address in a fixed location, i.e., the output of ID 1;
- **nb_inputs**: the number of inputs in the transaction;
- **nb_outputs**: the number of outputs in the transaction;
- **fee**: the fees paid for this transaction;
- **diff_zero**: number of non-zero digits among the 8 last digits of the value (in Satoshi) associated to the output link. It captures the property of being or not a *round number*. We used multiple features around this idea. The underlying assumption is that payments tend to have an amount fixed by one of the parties involved, and thus to be a *simple* number. On the contrary, the change is the difference between the input and the change, and thus would retain the same number of digits as the sum of the inputs;
- **wz_one**: a Boolean indicating if the remaining of the output value, after removing all zeros equals '1'. It could allow to differentiate typical values such as payments, 1BTC, 10BTC, etc. ;
- **wz_five**: a Boolean indicating if the remaining of the output value, after removing all zeros equals '5'. Same logic as above;

- **wz_others**: a Boolean indicating if the remaining of the output value, after removing all zeros equals '2', '3', '4', '6', '7', '8', '9', '15', '25', '55' or '99'. We consider that those could indicate particular behaviors, such as splitting a sum in equal parts or typical payment sums;
- **sup_ins**: a Boolean being True if this output is smaller than the smallest input minus transaction fee, False otherwise. In principle, the change value should be smaller than the smallest input, otherwise the input value was not needed to do the transaction;
- **entropy_val_ins, entropy_val_outs**: Shannon entropy of values distribution of input and output values. This diversity index makes it possible to differentiate the cases where most of the values come from a single input from the cases where they are distributed in a more homogeneous manner;
- **entropy_ads_ins, entropy_ads_outs**: Shannon entropy of addresses repetition distribution of input and output values. When transactions have many inputs, in some cases they have the same public key repeated multiple times. This feature is designed to differentiate this case from that where all the addresses are different, even when the number of inputs is the same.
- **timestamp**: the timestamp the transaction took place;
- **weekday**: the day of the week the transaction took place;
- **year**: the year the transaction took place;
- **month**: the month the transaction took place;
- **day**: the day in the month the transaction took place;

4.5 Machine learning models and their interpretation

Different machine learning models can be used to associate a label to a transaction output described by the features enumerated in Section 4.4. We chose not to use deep learning methods in this research, since they are more costly than traditional methods, and seem to offer little gain on *tabular* data,

i.e., data in which there is no structure to exploit, like in images, text, graph or sequences. On the contrary, it has been shown[Grinsztajn et al., 2022] that tree-based methods are the best performing in contexts of a moderate number of features, and with no structure, which correspond to our context. We introduce here briefly the methods used in our research:

- Decision Tree builds a tree in which each internal node represents a "test" on an attribute, each branch represents the test result, and each leaf node represents a class label. Paths from root to leaf represent classification rules. Branching is learned[Quinlan, 1996] on training data to optimize an objective such as Gini impurity of the classes in the leaf nodes. It has a great ability to learn nonlinear relationships while being simple and interpretable.

- CatBoost [Prokhorenkova et al., 2018] is a fast gradient boosting on Decision Trees. Boosting is used to build ensemble models in an iterative way. On the first iteration, the algorithm learns a shallow tree, fitting poorly the data, but with the objective of avoiding overfitting. In the second iteration, the algorithm learns another shallow tree to reduce the error made by the first tree. The algorithm repeats this procedure until it builds a decent-quality model. Gradient Boosting can be used for any continuous objective function, generally the Logloss for classification, and the root mean square error for regression. The method computes the gradients of the loss function to optimize for each input object, and then learns the decision tree which predicts the gradients of the loss function.

- XGboost [Chen and Guestrin, 2016] is another gradient-boosted decision tree (GBDT) machine learning library that provides parallel tree boosting. It is one of the leading machine learning library for regression, classification, and ranking problems.

A *Grid Search* approach has been used to optimize the hyperparameters.
Since XGboost and CatBoost are *black box* models, we propose to identify the most important features for the prediction using explainable Machine Learning methods by conducting a feature analysis based on SHAP values [Lundberg and Lee, 2017, Lundberg et al., 2020]. The SHAP value approach is a feature attribution method that assigns each feature a value that reflects

its importance in the prediction process. The features together contribute to the prediction process and it is difficult to measure the importance of each of them independently. To measure the influence of a feature i, it considers the variation in the prediction values using a subset of features S with and without the feature i. Shapley values consist in training a model $f_{S\cup\{i\}}$ with that feature and another model f_S without that feature. Then, predictions from the two models are compared and this for all possible subsets $S \subseteq F \setminus \{i\}$ (F is the set of features of the dataset). The Shapley values are a weighted average of all possible differences:

$$\phi_i = \sum_{S\subseteq F\setminus\{i\}} \frac{|S|!(|F| - |S| - 1)!}{|F|!} \left(f_{S\cup\{i\}} - f_S\right)$$

The method SHAP (SHapley Additive exPlanations) uses the Shapley values to compute an additive explanatory model g that is a linear combination of Shapley values:

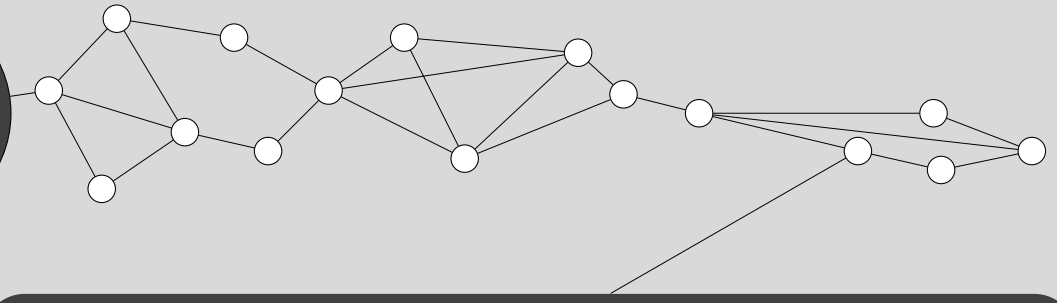
$$g(x') = \phi_0 + \sum_{i=1}^{n_c} \phi_i x'_i$$

with ϕ_0 the average output of the model, ϕ_i the explained effect of feature i and x' a binary encoding of instance x. This explanatory model is constrained to be roughly equal to f in the vicinity of x. However, computing Shapley values for each possible subset of features is too expensive. [Lundberg and Lee, 2017] introduces the concept of Shapley kernel to approximate Shapley values and makes it possible the use of this approach on real-world datasets. We use python's SHAP[2] package to compute the SHAP values of our models.

This way of building a supervised dataset for change address prediction is used in chapter 5 for the automatic discovery of clusters of addresses using the change address, and in chapter 6 to detect the address cluster for a specific user.

[2] `https://shap.readthedocs.io/en/latest/index.html`

5



Automatic discovery of change augmented address clusters

In this chapter, we consider the problem of building a supervised machine learning model to predict whether a transaction output is a change or a payment. Then, we use this model to improve the process of discovering address clusters thanks to the detected change outputs. Finally, we evaluate the results by comparing them to a ground truth.

5.1 Building train and test sets

To constitute the training set, we consider the transactions from block 0 to block 501950, corresponding to 31 December 2017. Hereafter, we call this dataset D-2017. To label the output addresses as change or payment, we use the user information provided by H1-labeling (see Section 4.3), i.e., if an output is sent to the same user as the input, it is a change output. Among the transactions in D-2017, we retain only the ones composed of exactly two outputs, one being a change and the other not. These transactions are the most likely to be correctly labeled because Bitcoin's most common behavior is to make a payment in one output and send the rest to a change address. It also has the benefit of maintaining a balanced dataset between change and payment training examples.
The dataset on which we perform change prediction, that we call the test set, is composed of the transactions of D-2017 in which 1) there are exactly two outputs, 2) none of the output address has been labeled as a change using H1-labeling. These transactions are the most likely to include a change address that was not detected thanks to H1 alone. We ignore transactions with more than two outputs, because these may correspond to different and more complex cases.

5.2 Augmented clusters with change addresses

We presented in the last section how to build a training and test set for the prediction of change outputs. However, our final objective on which we will evaluate the quality of our method is to improve address clustering over what can be found using heuristic H1. We are therefore searching for a new type of clusters called *change-augmented clusters*.

Definition 3 (Change-augmented clusters) *We consider a set of transactions $\mathcal{T}$, and $t.C$ the set of detected change outputs of a transaction $t \in \mathcal{T}$*

using a supervised machine learning model. The set of change-augmented address clusters constructed from $\mathcal{T}$ *and denoted by* $\mathcal{A} = \{A_1, \ldots, A_p\}$ *is such that for all* i *from 1 to* p*, and for all two distinct addresses* $a, b \in A_i$*, there exists a sequence of pairs of addresses* $P = (a, c_1), (c_1, c_2), \ldots, (c_n, b)$ *so that for all* $(p, q) \in P$*, there exists* $t \in \mathcal{T}$ *with* $p, q \in t.input$*, or* $p \in t.input$ *and* $q \in t.C$*.*

Said differently, if one H1 cluster sends coins to another H1 cluster using a discovered change address, then both clusters are merged into a single one. In practice, we can do the detection exactly like what is done to discover H1 clusters, i.e., by searching the connected components in a graph of identity evidence. More precisely, for H1, nodes are addresses and links are added between addresses appearing in input of the same transaction. In change-augmented clusters, nodes are defined as being H1 clusters (found in the usual way in a preprocessing step), and an edge is added between cluster A1 and cluster A2 if an address belonging to A2 appears as a detected change output of a transaction initiated by A1. Clusters are found by searching the connected components of this graph, and considering H1 clusters in the same connected component as forming a single change-augmented cluster.

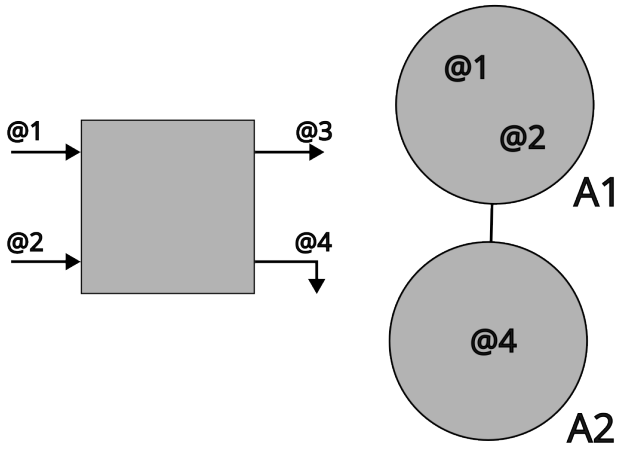


Figure 5.1: Change-augmented clusters construction. On the left is a transaction with 2 addresses in input and 2 addresses in output. On the right are two H1 clusters, A1 and A2. Address @4 is detected as a change output. Clusters containing input and change output addresses are thus linked.

5.3 Evaluation and Ground truth clusters

In order to evaluate our method, we need to compare the discovered clusters with ground-truth clusters, to take into account the problem of catastrophic merge. This section describe this problem, and the process used to build reliable ground truth clusters.

5.3.1 Catastrophic Merge

We cannot evaluate the method using directly the change-prediction score, such as an accuracy or F1 score, because some errors have a much more negative effects than others on the final result. Indeed, a major pitfall of cluster discovery is that since a single occurrence of a cluster identified as receiving the change from an address from another cluster is enough to merge the two clusters into a single one, a single false positive (a payment identified mistakenly as a change) can have a catastrophic effect on the final clustering, since two potentially large separate clusters would merge. Conversely, a false negative (a change output being wrongly assumed to be a payment) is less critical, in particular since a different change transaction between the two same clusters can be rightfully detected and will be enough to correct this missed detection.
only that the machine learning approach can do *as well as* the H1 heuristic, since all positive change examples have been labeled using H1 itself, 3) A very efficient method that would recognize all change addresses, including these missed by H1, would in fact have a *lower* score than a method only able to reproduce H1 change recognition.

5.3.2 Ground truth

Since we cannot know for sure all the addresses of an actor, we can only compare our improved clusters with clusters discovered using other methods, considering them as ground truth. However, we are confronted to a problem of endogeneity: since we used H1 to train the model, we should not evaluate our model by comparing it with the result of the same H1 clustering, although this has been done in previous works, due to the absence of better solution. There are two main drawbacks in doing so: 1)at best, it proves only that the machine learning approach can do *as well as* the H1 heuristic,

since all positive change examples have been labeled using H1 itself, 2) A very efficient method that would recognize all change addresses, including those missed by H1, would in fact have a *lower* score than a method only able to reproduce H1 change recognition. To solve this problem, we introduce two concepts for building a relevant ground truth, *A posteriori test set* and *External ground truth*.

5.3.3 A posteriori test set

To address the first problem, we define our ground truth *a posteriori*. More precisely, we split our dataset in two parts: the studied dataset **D-2017**, composed of transactions up to bloc 501950 (December 31, 2017), and the *a posteriori* ground truth dataset **D-2021**, composed of all transactions up to the end of the collected dataset. We use D-2017 for training and for evaluation, and we use D-2021 to compute the ground truth clusters, that we call **H1-AP** for *a posteriori*. The principle is that H1 applied to D-2017 will incorrectly label some changes as payment, and thus several different clusters according to H1 will in fact be merged into a single cluster in our ground truth, thanks to the additional information found in D-2021 (see Figure 5.3).

5.3.4 External ground truth

To address the second problem, we improve our ground truth using an external source. We used data extracted from the website WalletExplorer[1], commonly used in the literature to recognize known actors (e.g., [Ermilov et al., 2017, Möser and Narayanan, 2022]). The website provides for a few hundred clusters the name of the actor to which this cluster corresponds to. Although the exact details of the process are not known, this information is said to be obtained through manual collection. Since it is enough to know the identity of one address of a cluster to label the cluster, one transaction with a known actor (Exchange platform, Gambling service, etc.) usually allows to label its whole cluster. The information we are interested in from this website is that, for several known entities, the website provides several associated clusters. We leverage this information to improve our ground truth. Instead of using directly H1 clusters in our ground truth, we create

[1] `https://www.walletexplorer.com`

Ground Truth Clusters **GT-A** by merging H1 clusters found on D-2021 according to the actors defined by WalletExlorer. We chose to split the dataset in 2017 because 1) We want to have enough data after the end of the study dataset to discover change addresses, 2) WalletExplorer is known to be less reliable for actors after 2017 (missing information). The ground truth and studied datasets are summarized in Figure 5.3. The types of merges that we can observe, and the consequence on our evaluation score are summarized in Figure 5.2.

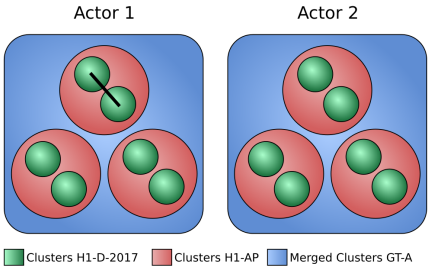
A drawback of this approach is that we had to restrict our analysis to six actors[2], identified thanks to WalletExplorer. These actors have been chosen according to their size (number of transactions), for having multiple known clusters according to Wallet Explorer, for their diversity (Mining Pool, Exchanges, Gambling services), and for their period of activity, compatible with our a posteriori approach, i.e., having activity before and after 2017). Having six actors only is of course a limit, but reflects our choice of having less data of greater quality, instead of taking the risk of evaluating our method by comparing it with a biased ground truth composed of a large number of unknown clusters. We have collected all transactions such that the sender of the transaction is among the chosen actors. At the end, we have at our disposal 520 578 addresses belonging to the six chosen actors: 354 006 are train examples that are output addresses (50% change 50% payment outputs), and 2 557 002 test examples to evaluate as being or not change addresses.

5.4 Three alternative methods to post-process the predictions of ML models

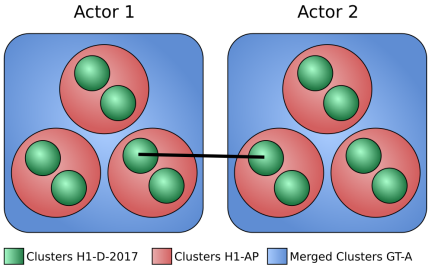
Our final objective is not to detect change outputs, but to improve address clustering. To decide whether an output should be considered a change output, we do not directly use the class provided by the machine learning model to avoid the risk of **catastrophic merge**.

We propose and evaluate three alternative methods to post-process the predictions of the machine learning models in order to minimize catastrophic merges.

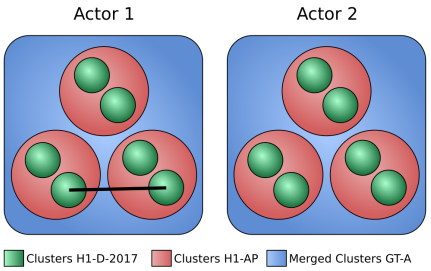
[2]Bter.com, PrimeDice.com, BitcoinVideoCasino.com, FaucetBOX.com, BTCCPool, BitZino.com.



(a) Merging of two H1-D-2017 clusters belonging to a same H1-AP cluster. Beneficial Merge.



(b) Merging of two clusters H1-D-2017 belonging to two different actors according to WalletExplorer. Detrimental Merge.



(c) Merging of two H1-D-2017 clusters belonging to two different H1-AP clusters belonging to the same ground truth. Beneficial Merge.

Figure 5.2: Different types of possible merges.

5.4.1 M1: Classification with variable confidence threshold.

The reference approach is to use a variable threshold on the confidence probability of the classification. By imposing a high confidence value, we detect fewer change transactions, but reduce the probability of a catastrophic merge. The right threshold is the best compromise between increasing false negatives and decreasing false positives.

5.4.2 M2: Limit to one change per transaction.

As mentioned when describing the train and test sets, we focus on transactions that have two outputs. It is known that these transactions tend to have a single payment and a single change output. Therefore, if two outputs are classified as a change with confidence above the threshold, and there is a significant difference ($> 1\%$) between the two, only the one with higher

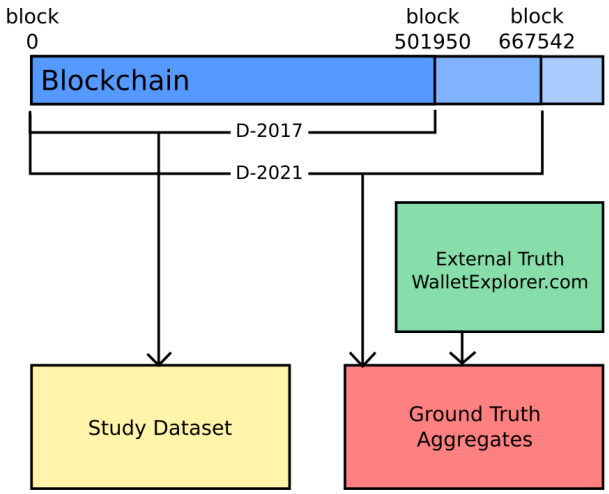


Figure 5.3: Computation of the studied dataset **D-2017** and of the Ground Truth Clusters **GT-A** based on D-2021 and WalletExplorer data.

confidence is classified as a change. If there is less than 1% difference, we classify both as payment.

5.4.3 M3: Requiring repeated change detection for cluster merges.

To avoid that a single false positive merges two clusters, we add the constraint that several change outputs must be observed between two H1 clusters to merge them. In practice, we have defined improved cluster detection as the search for connected components of the induced graph in which the nodes are H1 clusters and the edges (u_1, u_2) represent the existence of a transaction with an address of u_1 as input and an address of u_2 as change output. In this variant, we add an edge in this induced graph only if we observe $x > k$ such transactions between the two H1 clusters, with k a threshold. In the following experiments, we use $k = 2$, a sufficient limit to observe significant changes.

5.5 Experimental results

To evaluate the performance of our approach, we compare clusters obtained by the different variants with our ground truth. We use the most common cluster comparison metrics, namely *Homogeneity*, *Completeness*, *v-score/NMI*, *aNMI* and *Rand Index*. We first report the *Homogeneity* scores obtained using H1-AP (Table 5.1) or GT-A (Table 5.2) as ground truth. We observe that the M3 method obtains a value below 1 with H1-AP, while it obtains a score of 1 using a threshold of 0.8 or more using GT-A as ground truth. This clearly confirms the validity and relevance of our original validation approach: the M3 variant merges some clusters which are considered different using only the information from the Blockchain (H1 a posteriori), but which are recognized as correct when they are validated using external ground truth. It thus confirms that the process is able to recognize change addresses that could not have been detected simply by the H1 heuristic. This has never been shown before. Note that H1 alone also has a homogeneity score of 1 because it never wrongly merges clusters. Since observations are similar for other scores, we will report only the comparison with the GT-A ground truth.

Heuristic H1 **1.000**

Threshold	M1	M2	M3	XGB/M3	CatBoost/M3
0.7	0.593	0.903	0.780	0.875	0.875
0.75	0.593	0.903	0.799	0.972	0.972
0.8	0.644	0.903	0.948	0.972	0.972
0.85	0.644	0.903	0.948	0.972	0.972
0.9	0.799	0.903	0.948	0.972	0.972
0.95	0.920	**1.000**	0.978	**1.000**	0.972

Table 5.1: Homogeneity Score - Ground truth H1 on H1-AP.

The *Completeness* measures the gain obtained by rightfully merging the H1-D-2017 clusters. We observe in Table 5.3a that 1) the M1 method fails to obtain scores higher than the reference H1 score, 2) M2 only succeeds for the maximum threshold chosen, and 3) M3 makes it possible to improve relatively to the baseline.

To confirm these results, we use three commonly used scores that synthesizes both aspects captured by Completeness and Homogeneity. The first one is the v-score (or NMI), which is defined as the harmonic mean of Completeness

Heuristic H1	**1.000**				
Threshold	M1	M2	M3	XGB/M3	CatBoost/M3
0.7	0.562	0.890	0.732	0.883	0.883
0.75	0.562	0.890	0.761	**1.000**	**1.000**
0.8	0.623	0.890	**1.000**	**1.000**	**1.000**
0.85	0.623	0.890	**1.000**	**1.000**	**1.000**
0.9	0.761	0.890	**1.000**	**1.000**	**1.000**
0.95	0.908	**1.000**	**1.000**	**1.000**	**1.000**

Table 5.2: Homogeneity Score - Ground truth on GT-A.

and Homogeneity. Given the previous observations, it naturally confirms the superiority of the results of method M3 (Table 5.3b).
To make these results more robust, we use two randomly adjusted scores commonly used in clustering and community detection assessment: the aNMI (Table 5.3c) (random-adjusted version of the NMI), and the Adjusted Rand Index (ARI) in Table 5.3d. The two scores confirm the observation that the results obtained with the M3 method are superior both to the other methods and to the reference heuristic H1.
We see that setting an appropriate value for the threshold significantly improves the results. It must be large enough for the homogeneity score to reach the value one, as seen in the Table 5.2, but too high values have a negative effect on completeness (see Table 5.3a), because they are too restrictive, so there is a fair balance to get the highest scores on the combined scores (see Table 5.3b).

5.6 Discussion

Using external sources of data, we have been able to show that the aggregates we found are not only able to discover the same clusters as H1 heuristic find using posterior data, but also improves on these results. Previous works were either proposing H1 improvement, but without quantitative evaluation, or validating only using H1 or manual validation (e.g., [Cazabet et al., 2018, Shao et al., 2018]).
Of course, our work has limits. First, we validated the first approach only with six actors. Although there is no restriction on the number of actors we could detect, we restricted ourselves to these actors due to the limitations of ground truth, as we would not have had any relevant ground-truth for an

Heuristic H1	0.626				
Threshold	M1	M2	M3	XGB/M3	CatBoost/M3
0.7	0.593	0.617	0.588	0.632	0.632
0.75	0.593	0.617	0.597	0.644	0.644
0.8	0.606	0.617	**0.661**	0.644	0.644
0.85	0.606	0.617	**0.661**	0.644	0.644
0.9	0.597	0.617	**0.661**	0.644	0.644
0.95	0.618	0.627	0.641	0.626	0.644

(a) Completeness Score - Ground truth on GT-A.

Heuristic H1	0.770				
Threshold	M1	M2	M3	XGB/M3	CatBoost/M3
0.7	0.577	0.729	0.652	0.737	0.737
0.75	0.577	0.729	0.669	0.784	0.784
0.8	0.614	0.729	**0.796**	0.784	0.784
0.85	0.614	0.729	**0.796**	0.784	0.784
0.9	0.669	0.729	**0.796**	0.784	0.784
0.95	0.736	0.770	0.781	0.770	0.784

(b) V-Score/NMI - GT-A Ground truth.

Heuristic H1	0.770				
Threshold	M1	M2	M3	XGB/M3	CatBoost/M3
0.7	0.577	0.729	0.652	0.737	0.737
0.75	0.577	0.729	0.669	0.784	0.784
0.8	0.614	0.729	**0.796**	0.784	0.784
0.85	0.614	0.729	**0.796**	0.784	0.784
0.9	0.669	0.729	**0.796**	0.784	0.784
0.95	0.736	0.770	0.781	0.770	0.784

(c) aNMI - GT-A Ground truth.

Heuristic H1	0.481				
Threshold	M1	M2	M3	XGB/M3	CatBoost/M3
0.7	0.286	0.432	0.345	0.481	0.481
0.75	0.286	0.432	0.351	0.446	0.446
0.8	0.299	0.432	**0.532**	0.508	0.508
0.85	0.299	0.432	**0.532**	0.508	0.508
0.9	0.351	0.432	**0.532**	0.508	0.508
0.95	0.446	0.481	0.501	0.481	0.508

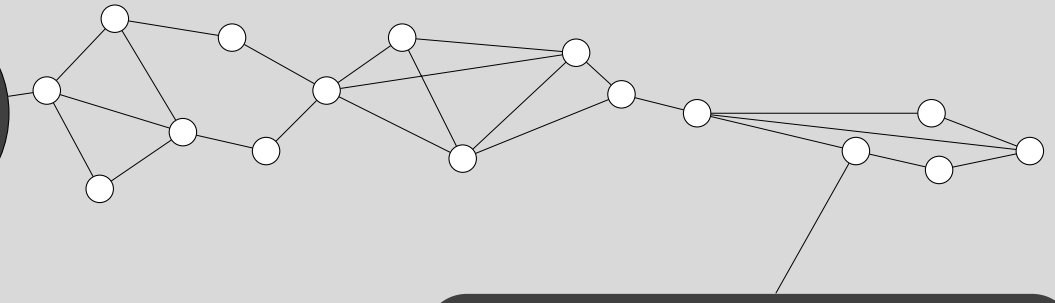
(d) Rand Index - GT-A Ground truth.

Table 5.3: Scores obtained with Completeness, V-Score, aNMI and Rand Index. We observe that the M3 method yields the best scores overall according to all metrics, both among tested variants and compared to H1 baseline, in most contexts. We also observe that we obtain higher scores using the decision tree than with competing methods.

evaluation on thousands of aggregates, which, in our opinion, is unreliable: if the ground truth is built only on H1, then showing that the output of the algorithm is similar to the ground truth is not a satisfactory approach. We also restrict our work to transactions with only two outputs. Considering transaction with more output addresses could be addressed in future work.

This work has several perspectives. First, it can be used as a first processing step in future works studying Bitcoin actor's activity. Second, the process is not specific to Bitcoin but can also be applied to any cryptocurrency using the UTXO model. Third, the effectiveness of the method can be seen as underlying weaknesses of the anonymity of Bitcoin in its current design, and can be analyzed to change either the protocol or to provide recommendation to users to increase their privacy.

6



User specialized change detection

In the previous chapter, we have proposed an original supervised machine learning based method to solve a common problem in Bitcoin: Detecting clusters of addresses. The problem considered was comparable to the usual task addressed by state-of-the-art methods, i.e. the discovery of a set of address clusters from a set of transactions. However, we observed that in the literature, the proposed methods were generally not reused after their publication. Subsequent articles on Bitcoin user analysis still rely on the H1 heuristic to identify entities' address clusters. Based on our own experience, we believe there are two main reasons for that:

1. Published results are often difficult to apply and reproduce. Given the large amount of Bitcoin data, even computing the simple H1 heuristic is a difficult task. Computing on the entire dataset a much more complex approach, requiring the computation of various features and/or computationally expensive models, such as deep neural networks, is an even greater challenge.

2. Despite the safeguards added to most methods, including ours, the risk of catastrophic merges (and in practice the little information we have about unwanted merges occurring on address clusters other than the large known ones) makes the use of these methods a risky choice when analyzing behaviors in the Bitcoin Blockchain.

This analysis does not contradict the good results obtained by these methods, as demonstrated in the previous chapter. However, we think that change detection would be more actionable for a slightly different task that we define here: the change detection of individual entities. We argue that in various applications, researchers and practitioners are mostly interested in analyzing one particular enity, or a subset of entities of interest (e.g., malicious entities [Yazdinejad et al., 2020, Dalal et al., 2021], Mining Pools [Tovanich et al., 2021a], Major exchanges [Jourdan et al., 2018], etc.). In this section, we investigate how our supervised machine learning approach, contrary to unsupervised ones, could be used for the change detection on a particular entity, with the objective to better detecting the activity of this entity in particular.

6.1 Training and testing protocols

To identify the change addresses of a target entity, we first build a dataset of all the transactions for which this entity has addresses used as input. We then compute the same features as defined in Section 4.4 in Chapter 4 for each output of these transactions. Contrary to the previous chapter, we do not restrict ourselves to transactions with two outputs, since we are less concerned about catastrophic merges, that would be more easily identified with a single entity than with the millions of clusters obtained by cluster detection on the whole Bitcoin dataset. Furthermore, we are using all the transactions with at least one known change output, instead of just one, for the training and testing steps. This means that all the transactions having multiple known change outputs that were rejected in the previous section are now present in the dataset. In addition, we use the number of outputs as an additional predictive feature. This feature was irrelevant before but applicable here, as now we are using transactions with any number of outputs. We used the same machine learning models whose hyper-parameters have been selected with grid search.
To build our training and testing sets, we can now rely on the commonly used supervised machine learning approach: we randomly split all outputs in two sets: the train and test sets. The quality evaluation process can be framed as any classification evaluation problem, using the ROC-AUC score. In this scenario, this score can be interpreted as the probability for a randomly chosen change output in the test set to have a higher confidence probability to be a change output than a randomly chosen payment output (according to the trained classifier). A score of 0.5 thus means that the classifier is doing no-better than random prediction, while a score of 1 means that all change addresses have a higher confidence score (probability of class *change*) than all payment addresses.

6.2 Experimental results

We have selected different Walletexplorer entities, from different categories and sufficiently active in terms of number of transactions to have statistically significant results. For each of them, we trained a model to recognize their change outputs based on their own transactions, using a division of

two-thirds for training and one-third for testing. We also make a baseline experiment, for which we use a random sample of 500 000 transactions for training, called hereafter *un-targeted model*. On the contrary, when the model is trained on the entity's activity, we use the name of the entity to label that model. The objective of this comparison is to see if it is more reliable to train with less data on a single actor, or to have more training data to better generalize the definition of a change address.

6.2.1 ROC-AUC score for change detection using different ML models

We computed ROC-AUC scores for change address detection using Decision Tree (DT), XGBoost (XGB), and CatBoost (CatB) machine learning models. The models were trained either specifically on the target (-t) or on generic data (-g). The results are evaluated on specific and on unknown entities for generic training, i.e. to detect changes from randomly chosen transaction outputs.

Training Dataset	DT-t	DT-g	XGB-t	XGB-g	CatB-t	CatB-g
BTCC.com	0.999	0.861	**1.00**	0.985	**1.00**	0.985
BitPay.com	0.998	0.839	**0.999**	0.979	**0.999**	0.978
CoinRoyale.com	0.998	0.938	**0.999**	0.992	**0.999**	0.994
Bter.com	0.995	0.823	**0.999**	0.950	**0.999**	0.943
Bitfinex.com	0.994	0.884	**0.995**	0.960	**0.995**	0.959
Banx.io	0.991	0.809	**0.999**	0.941	**0.999**	0.934
Cryptsy.com	0.992	0.862	**0.994**	0.965	**0.994**	0.965
PrimeDice.com	0.981	0.841	**0.987**	0.955	**0.987**	0.954
Unknown entity	-	0.855	-	**0.967**	-	0.966

Table 6.1: ROC-AUC scores for change detection using ML models. DT: Decision Tree, XGB: XGBoost, CatB: CatBoost. XXX-t means that the model has been trained specifically for the user of interest, while XXX-g means generic (non-targeted) training

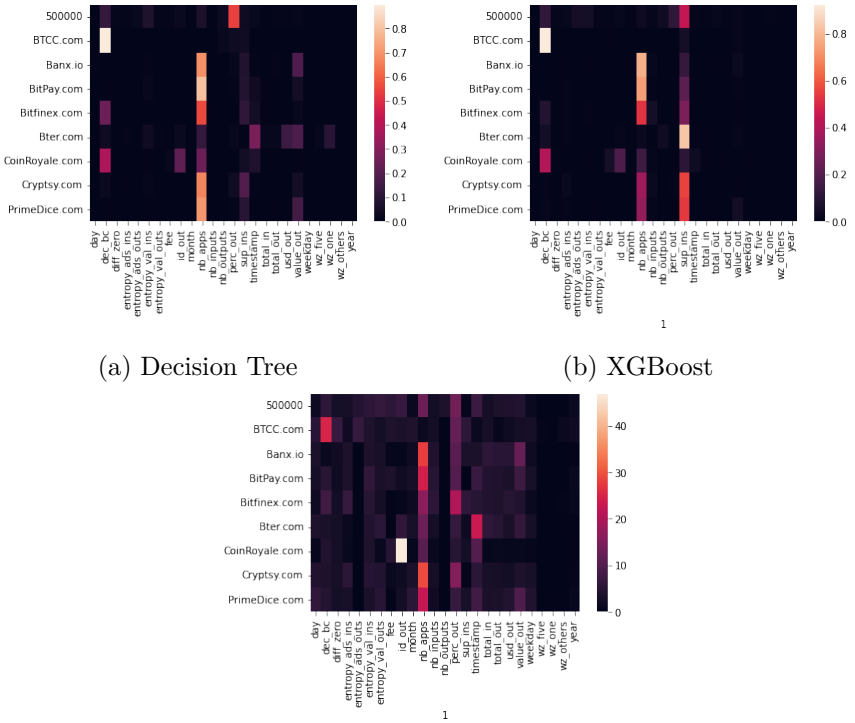
We observe that targeted learning always improves the score, with the highest results obtained with XGBoost and CatBoost.

The ROC-AUC scores for change detection for the targeted models and for the un-targeted one are shown in Table 6.1. We can observe that the method yields very convincing results in most cases, confirming that a machine learning approach can be trained to recognize change outputs. Another obvious

point is the superiority of results obtained by training on targeted data, i.e. on the previous activity of the entity itself, compared with training on non-specific data, despite the largest quantity of data available in that case. This confirms that the results obtained with previous methods are certainly sub-optimal and could be widely improved by using targeted learning.

6.2.2 Model feature importance

To better understand how entity-specific learning improves change detection, we explore feature importance for the different trained models. We first generate a heatmap based on the computation of the importance of each feature for each entity in Figure 6.1.



(a) Decision Tree (b) XGBoost

(c) CatBoost

Figure 6.1: Heatmaps of feature importance for different entities (targeted datasets) and for the transactions collected randomly (un-targeted dataset). Some features are important independently of the model, such as *nb_apps*, while others are more important in one of the models (e.g., *sup_ins*). We observe clear differences in feature importance both between one or more targeted datasets and between targeted and un-targeted datasets.

We can notice some interesting results.

- Different entities seem to manage differently their transactions: the features importance vary from one entity to another;
- Different methods tend to agree on many aspects (e.g., *id_out* importance for CoinRoyale, the importance of *nb_apps*), but differ on some other variables (e.g., *sup_ins*, *perc_out*).
- One of the features (*nb_apps*, the number of previous appearances of the output address) is very important in many of the datasets. This element was known from previous heuristics such as H2, but we can see here that it is not reliable enough to be used as a single criteria, and that its importance depends on the entity;
- Contrary to the previous observation, *BTCC.com* does not rely on the number of previous appearances of the output address at all. Another characteristic is important instead: the number of decimals of the output value in BTC;
- The timestamp feature is the most important one for the *Bter.com* entity according to two models. This can probably be explained by a change in transactions management at some time;
- *id_out*, the id-number of the outputs, that should not contain useful information in theory, is considered very relevant for CoinRoyale.

We explore in more details some of these relations using a visualization of the learned decision trees with depth limited to 3, to facilitate their interpretation (see Figures 6.2, 6.3).
In this way, we can find other important information, as well as understand some of the previous observations. In these trees, a darker color tone indicates that the node contains mostly elements of one class (orange=0=(non-change), blue=1=change). On the contrary, lighter colored nodes have more mixed examples. Some important findings are:

- When examining *Banx.io* tree (Figure 6.2), we can learn that every address used as a change output is used for the first time, although all addresses used for the first time are not change outputs;

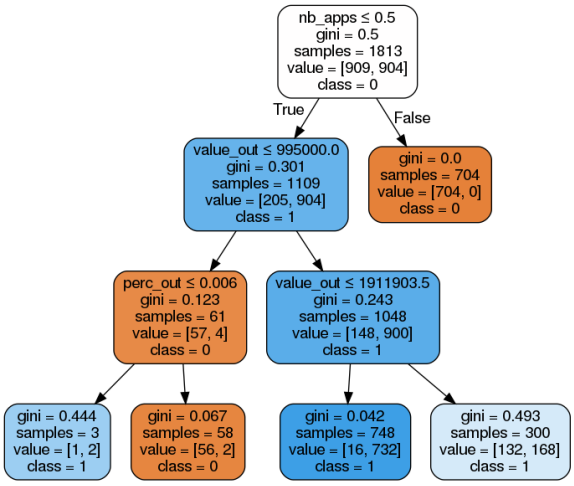


Figure 6.2: Banx.io decision tree. Orange(0) = (non-change), blue(1) = change

- The *BTCC.com* tree (Figure 6.3) shows us the importance of the decimal digits in the output value, as at the first branch it can separate many payments from change outputs. This shows us that most of the payments have more "rounded" values;

XGBoost and CatBoost beying *black box* models, whose interpretation is not as straightforward as drawing a tree, we resort to an XAI framework, SHAP [Lundberg and Lee, 2017, Lundberg et al., 2020], to explain the role of features. We show the results for XGBoost for some entities in Figure 6.4. Features are sorted by order of overall importance from top to bottom, and the plots show how a feature value (low to high) impacts the SHAP value, i.e., the classification decision (positive values indicating higher chances to be a change transaction in our case). Some interesting observations can be made:

- Despite using different indicators on a different method, the overall feature importance is coherent with what was observed with decision trees, with *perc_out* being the most relevant feature for the untargeted model, and *nb_apps* being very important for most entities
- The effect of an important feature can be reversed, such as *perc_out* between Bitfinex and un-target.
- We clearly see the effect of *nb_apps* for Banx.io observed in decision

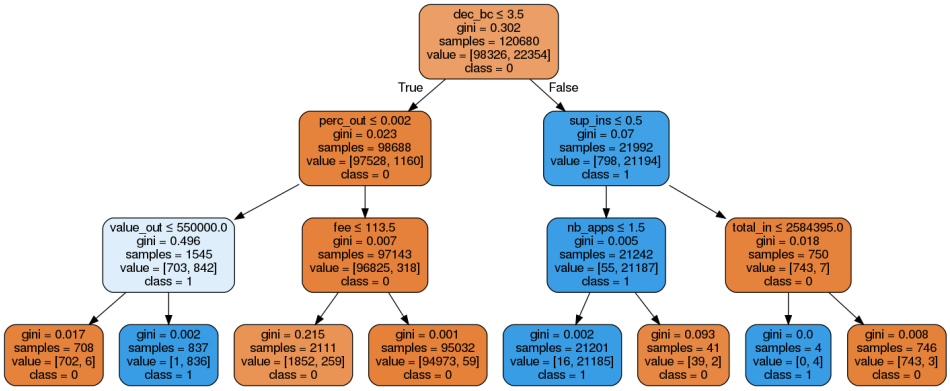


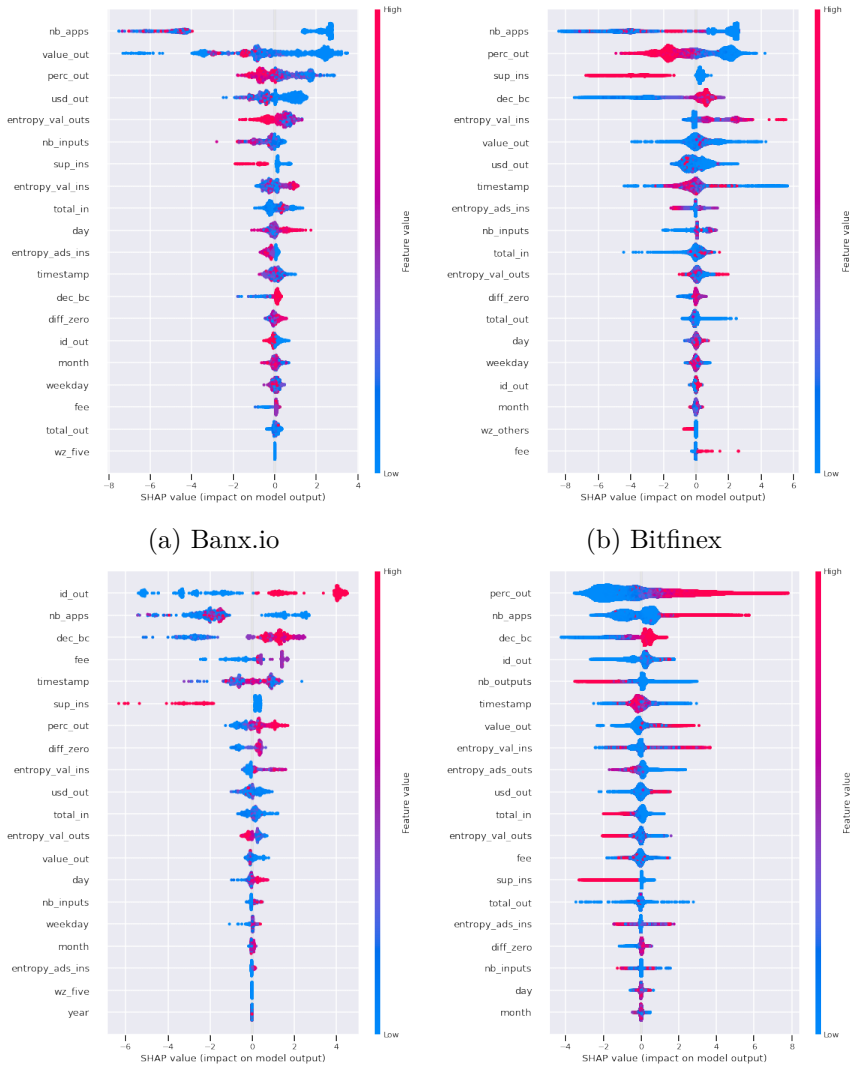
Figure 6.3: BTCC.com decision tree. Orange(0) = (non-change), blue(1) = change

trees, with the all-blue values on the right. This allows to interpret the similar but less reliable effect on Bitfinex, or even the opposite effect on un-targeted. This observation shed new lights on the H2 heuristic, which seems to be useful only for some very specific entities.

- The remarkable impact of *id_out* on CoinRoyale is clearly visible, with nearly all red values being on the right and all blue ones on the left, indicating clearly that this entity, unlike others, nearly always put its change address on the output of highest id, certainly due to a deterministic custom code, that can be considered a conception error.

All these observations confirm that the feature importance evaluation process is robust between methods, and that training an algorithm on a specific entity is clearly an advantage over ad-hoc heuristics or learning on undifferentiated entities, as previous methods do, due to the peculiarities of each entity behaviors.

We have shown that training a change detection model for a single entity is more efficient that training a model for the recognition of change addresses in general. In our opinion, such a targeted application is more actionable than the traditional *all-in-one* approach.

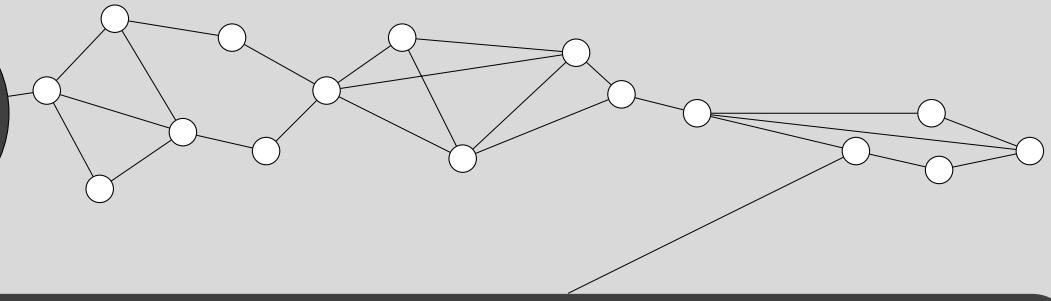


(a) Banx.io (b) Bitfinex

(c) CoinRoyale (d) un-targeted

Figure 6.4: SHAP values for XGboost for three entities and the un-targeted model. We clearly observe strong differences between the models. See in particular how *id_out* is specific of CoinRoyale, how the effect of *perc_out* is reversed between the un-targeted model and Bitfinex, or how *nb_apps*, which is among the most important feature in most cases, is mostly meaningless for BTCC.

7



Temporal study of bitcoin activity based on predefined categories

While numerous new usages of blockchains have developed in recent years, e.g., Decentralized Finance (DeFi), Non-fungible Tokens (NFTs), smart contracts, their usage as a currency, or at least as a store and exchange of value, remains one of the most important. In this chapter, we propose to study the real economic activity in the Bitcoin blockchain that involves transactions from/to retail users rather than between organizations such as marketplaces, exchanges, or other services.

7.1 Identifying real economic transactions

To analyze the real economic activity and its evolution in Bitcoin cryptocurrency, we first need to distinguish transactions corresponding to an exchange of value between two different entities (called "payments") from artificial transactions done for other reasons. Here, we describe the various processing steps and how they are relevant to filter out artificial transactions.

Address clustering. The blockchain identifies transactions between pseudonymous Bitcoin addresses, cryptographic public keys. We use the common-input heuristic H1 (see Section 3.2.1) [Harrigan and Fretter, 2016], that states that all inputs of a given transaction are owned by the same entity. This makes it possible to discover clusters of addresses, each cluster corresponding to a unique *Bitcoin entity*, the nature of which is unknown (eg companies, individuals).

UTXO outputs. Bitcoin transactions do not necessarily correspond to a single payment from one entity to another, but frequently have several *UTXO [Nakamoto, 2008] outputs*. Each of these outputs is a payment, and we first create a dataset of individual *payments*, from the source entity – unique thanks to the common-input heuristic – to each of the output entities. The number of unique payments is thus larger than the number of Bitcoin transactions as stored in the blockchain.

Change. Because of the UTXO [Nakamoto, 2008] mechanism, actors need to send back change to themselves, creating artificial coins exchange without economic signification. We remove these transactions that we are able to identify by removing *self-transactions*, between the same Bitcoin entity.

Dust & Micro Outputs. A phenomenon which has been identified and described is the use of *dust*, small-amounts sent often to many recipients, for a variety of reasons, for instance *forced address reuse* [Loporchio et al., 2023]. We think that, more generally, due to the high transaction fees and lack of technical solutions to use Bitcoin for micro-payments, all small amount payments can be considered as noise and discarded. Although the amounts are small, these transactions can bias our data as they are typically sent in very large numbers. We thus fix a minimum threshold of 0.5 USD, below which a transaction output is removed from our payment dataset.

Macro Outputs. We also get rid of transactions with very large amounts, as they may represent unconventional real payments. Those large payments may correspond to cash management between addresses of the same entity, combined payments between exchanges, or any other type of transactions that are not between customers and businesses and customers and customers. We set a conservative upper limit of 10,000 USD, assuming that transactions beyond this amount can be assimilated to professional investor profiles, even if they are carried out by individual users.

Clarification on Trading. A common belief about Bitcoin is that there is no "real" activity, most transactions being due to trading, that we would not consider in our analysis. Luckily, trading activities in Bitcoin are nearly exclusively performed by Exchange platforms (e.g., Binance or Kraken). Such transactions are conducted by internal scripture and never written to the blockchain. Many transactions that we observe are certainly linked to private trading activities, but only indirectly: customers moving their capital out of an Exchange to a privately owned Bitcoin address, and *vice versa*.

7.2 Defining entity types

We propose that entities involved in real transactions be divided into three categories. The first category contains companies and other actors offering a service to customers. This category contains Exchange platforms, Gambling services, retail outlets, etc. Despite existing works mentioned in the state of the art, our experiments make us think that it is impossible to reliably

distinguish these different types, in particular with the emergence of diversified super-players such as Binance, which offer multiple services at once. However, we think that, by definition, this type of players can be recognized by their frequent interactions with private users. We will categorize them as **Frequent receivers(FR)**. A second category is composed of the actors that are not FR, but interact with FR entities. We can consider them as individuals private Bitcoin users, since they are engaged in transactions with entities offering services. Finally, the last category contains all other entities, i.e., those that are not FR, and which never interact with FR. They represent a kind of *deep level* of the Bitcoin economy, on which we cannot say much. This is where money laundering, mixing services, and other type of transactions are supposed to happen. More formally, we define these categories as follows:

Frequent Receivers (FR). Entities who receive a steady stream of payments over a period of time, probably businesses. We define that this type of user must receive at least one genuine payment every day for 20 days in a month (which leaves two possible closing days per week). This criterion should exclude any non-professional user, who does not receive payments every day. In addition to this, a business user must have expenses related to his business activity. He must therefore participate in 10 other transactions, either as as emitter or receiver.

First Neighbors of FR (N1). This category identifies customers of FR entities. These are entities that do not meet the previous conditions, but trade (pay or receive) with entities classified as FR. We include transactions in which an actor of type N1 *receives* payments from one of FR type, because it can be a refund, a prize, in the case of gambling, or simply a transfer to himself from an Exchange or a Wallet manager.

The Others (TO). They are the entities that are not included in the two previous categories.

We have classified entities into each category on a monthly basis. Since an entity can change its behavior over time, it can be classified differently at two time periods. The reasons for a change in an entity's activity may be of

different natures: entity clusters may no longer be used to receive or send payments, changes in the entity's internal policy may occur, or this change may be related to specificities of the entity's market.

7.3 Evolution of the volume of authentic transactions

Payment volume. We first study the relationship between the total volume of all payments in Bitcoin compared to the real (filtered) payments defined in the previous section. Figure 7.1 (left) shows the evolution of the number of payments for each month in the studied period. Figure 7.1 (right) outlines the difference by plotting the ratio of the two. The ratio is stable around a mean of 0.36, with maximum and minimum values of respectively 0.52 in January 2018 and 0.15 in July 2015. While both values tend to grow with time, they also undergo important differences. For instance, in July 2015, there was a peak in the total number of payments (Fig. 7.1 (left)), but not in our filtered payments, resulting in the minimum value. After the growth and sharp decline in February 2018, the total volume of transactions grew while the real economy remained constant(Fig. 7.1 (left)).

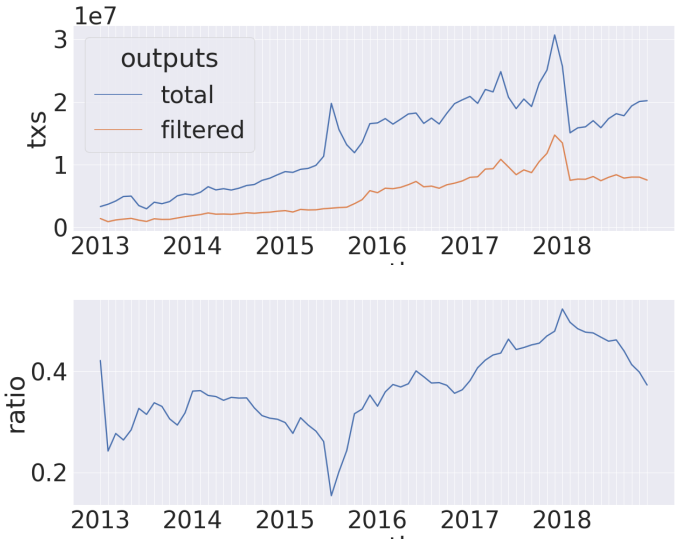
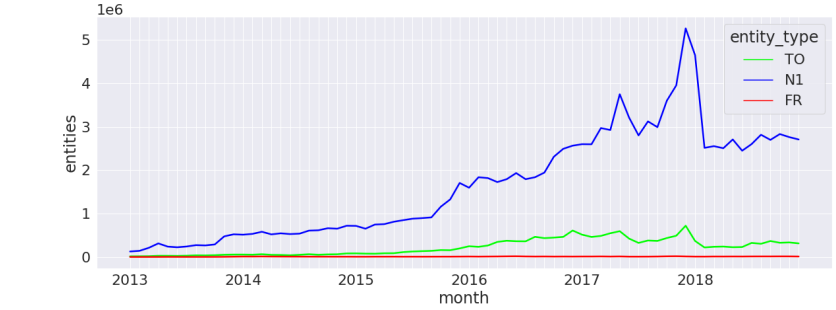
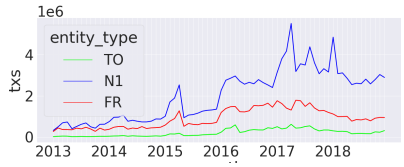


Figure 7.1: (Left) Total payment volume. (Right) Ratio of filtered to total payments.

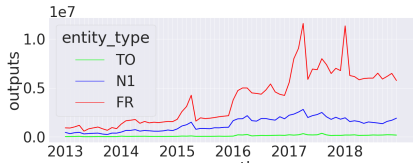
Payments entity types. We then analyze the prevalence of the three categories. Figures 7.2a, 7.2b and 7.2c show the number of entities, the volume of payment sent and received in each category, respectively. In addition, Figures 7.2d and 7.2e depict the amount of USD sent and received by category. These figures indicate that (1) FR entities are the least common but receive the most payments and the largest amounts; (2) The N1 entity type has the highest number of entities and is the category that sends the most payments with the largest amounts (although close to FR); and (3) TO (The Others) are the least present in transactions and are the ones who trade the smallest volume in USD.



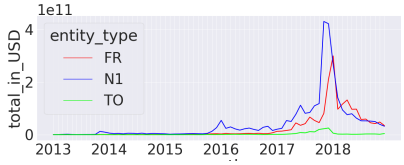
(a) Number of entities in each category.



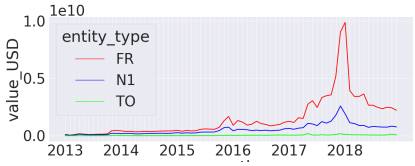
(b) Sum of payment sent (input).



(c) Sum of payment received (output).



(d) Sum of USD sent for each category.



(e) Sum USD received for each category.

Figure 7.2: Activity by category from Jan. 2013 to Dec. 2018.

Given the sulfurous reputation of cryptocurrency, it can be surprising that so few actual payments occur on the deep part of the Bitcoin economy(TO). Most of the real payments are received by big players (FR) and sent by entities that interact with those big players. Moreover, we found that the

behavior of each category is not driven by a global tendency. For example, in Figure 7.2b, we can observe that the number of payments where N1 entities appear as a sender suffers a huge rise during the first months of 2017, while we do not see the same effect for the other categories. This observation also aligns with Figure 7.2c for the number of payments from FR entities.

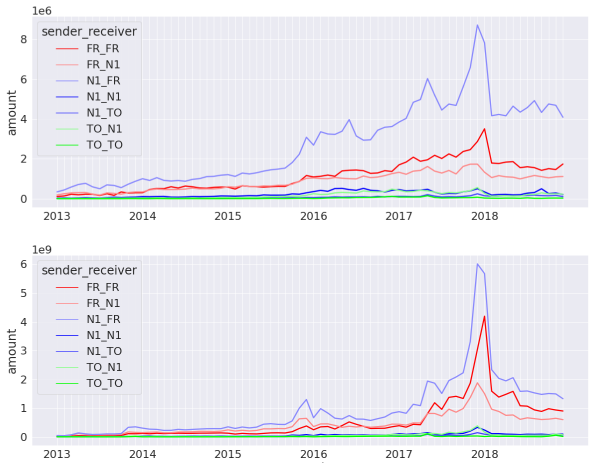


Figure 7.3: Exchanges between categories. CAT1_CAT2 in the legend identifies transactions from CAT1 to CAT2. Number of payments between categories (top). Amount of USD exchanged between categories (bottom).

Payments between categories. We turn our attention to the volumes exchanged between pairs of categories. Figure 7.3 shows the number of payments and the amount of USD exchanged between categories. Most payments and USD amounts are transferred from N1 to FR entities, while FR to N1 also has a high volume, although in third position overall. We can thus confirm that transactions from N1 to FR entities are the most common in what we consider real economic exchanges between individuals and companies. Another common type of exchange occurs among FR entities (i.e., from one FR to another FR). This result may appear surprising at first, but it can potentially be explained by transactions initiated by exchange customers. Many individuals use exchange platforms (considered as FRs) similar to retail banks, where they can request a Bitcoin equivalent of bank transfers. In other words, they ask the exchange platform to make payments on their behalf to another person or company. The majority of FR-to-FR payments probably represent this kind of payments. However, it is worth

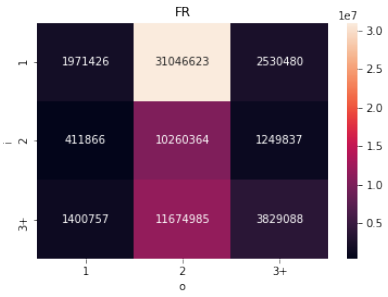
further investigation since it could also involve exchanges between businesses (B2B) or artificial transactions resulting from complex fund management.

7.4 Analyzing Transaction Types

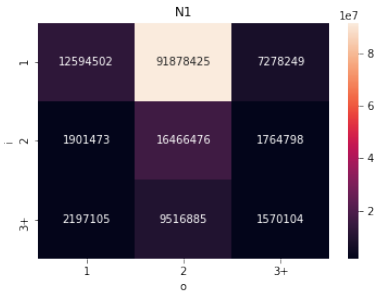
Another point of interest is to check the transaction types for each one of the entity types. For this, we considered the number of inputs and outputs of the transactions. The amounts were fixed at 1, 2 and 3 or more, for inputs and outputs. Combining these values, we have nine transaction types.
This kind of analysis is interesting so we can understand better the behavior of each entity type, and see if they behave as expected. We could assume, for instance, that customers will tend not to make multiple payments in the same transaction (i.e., transactions with more than two outputs). Companies, on the other hand, might have financial incentives to do transactions with three or more outputs, since the transaction fees are not proportional to amounts, but to the number of bits composing the transaction, and stacking two payments in a single transaction makes it less costly than issuing two different Bitcoin transactions.
Analyzing only the transactions where the entity types appear as the payer, we may see the absolute numbers for each type of transaction. Figure 7.4 shows three heatmaps for each entity type. We can observe some interesting behaviors as:

- The most common type of transaction is (2)→(2) (2 inputs, 2 outputs); it is the most used by all three categories;
- FR entities use mainly transactions with 2 outputs, but is the category that uses the most (3+)→(3+) transactions (3 or more inputs and outputs);
- N1 entities use primarily (1)→(2) transactions;
- Proportionally, TO entities are the ones who use (1)→(1) transactions the most.

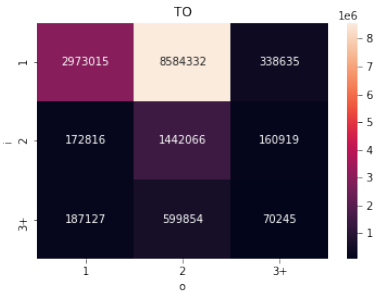
These numbers give us a first idea about the practices of each entity type. In order to have a different perspective on these numbers, we compute relative frequencies, i.e., the number of observed transactions of one type divided



(a) Total number of FR transactions by type.



(b) Total number of N1 transactions by type.



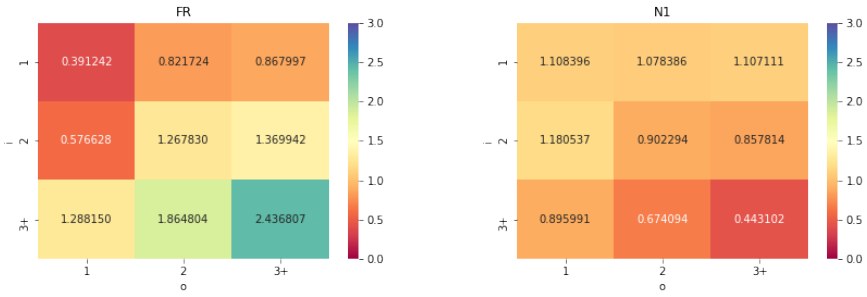
(c) Total number of TO transactions by type.

Figure 7.4: Total number of transactions by transaction type (i: number of inputs, o: number of outputs.

by the expected number of such transactions. A value of 1 thus means that the frequency is the same as on the global population; 2 means that the frequency is twice the value for the general population, and inversely 0.5 means that the frequency is half the value for the general population. The results are shown in figure 7.5.

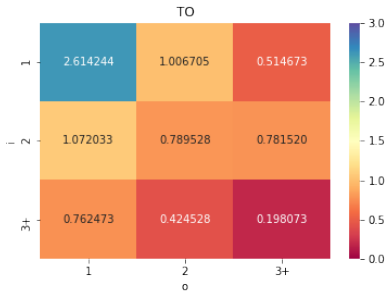
The orange color represents a relative frequency value of about 1, representing a value close to the global average, colder colors a higher value, representing cases more likely to occur than the average, while hotter colors a lower value, representing cases less likely to occur. The values present in the heatmaps provide us with some interesting information :

- FR entities are the one entity type using multiple inputs more often;
- transactions (3+)→(3+) (3 or more inputs and outputs) are mainly used by FR;



(a) Relative frequencies values for FR transactions by type.

(b) Relative frequencies values for N1 transactions by type.



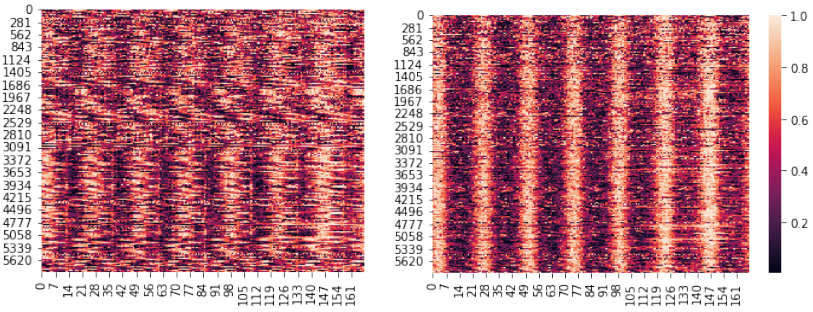
(c) Relative frequencies values for TO transactions by type.

Figure 7.5: Relative frequencies values by transaction type (i: number of inputs, o: number of outputs.

- N1 entities tend to use slightly more transactions with just one input, than the other categories;
- TO uses much more transactions (1)→(1) than any other entity type, and much less (3+)→(3+) transactions. The usage of (1)→(1) is coherent with a tendency to have transactions not corresponding to real payments, since it means that the transaction has no change, and a single source, corresponding to a Relay pattern (See section 3.2.2)

7.5 Temporal analysis

Since FR entities are supposed to have frequent interactions with clients, we can study their behavior in more detail from the rich information about payments they are involved in. This section analyzes the patterns of their weekly average behaviors.



(a) Normalized patterns, before alignment

(b) Normalized patterns, after alignment

Figure 7.6: Normalized patterns of activities for all FR and years. Each row is a FR entity, each column corresponds to an hour of an average week. Dark colors correspond to low values, light colors to high values.

7.5.1 Hourly behavioral patterns

We investigate temporal activities in an average week for entities identified as frequent receivers (FR). For each entity, we compute the payment volume for every weekly hour over a year with the following procedure.

1. We define a vector w of length $24 \times 7 = 168$, corresponding to the hours of an average week, i.e., $w[0]$ corresponds to Monday from 0 a.m. to 1 a.m., and $w[167]$ to Sunday from 11 p.m to 12 p.m.

2. For each entity, we count the total number of payments received for each hour in a week according to Anywhere on Earth (AoE) time.

3. Normalize the vector so that the sum of each entity's vector equals to 1.

We remove some noisy data for FR entities with a likely non-human pattern of activity: those with more than 80 zeros in their mean weekly activity pattern, i.e., more than 80-hour slots without a single transaction during the year. The idea is that if an entity is indeed a company, it should receive transactions at nearly any time. Note that we use received payments, whose arrival time cannot be controlled by the receiver. Figure 7.6a shows that most FR (rows in the heatmap) have a regular pattern for each day of the week, thus following the pattern of typical human behavior.

7.5.2 Temporal alignment

From Figure 7.6a, it is evident that weekly patterns among entities are not aligned. This is expected due to the global nature of Bitcoin users residing in different time zones. Therefore, we assume that daily activity follows a similar pattern everywhere on average. For example, if the peak activity occurs at 2 p.m. in Japan, it would also occur at 2 p.m. in New York — in the local time zones. This assumption is an approximation: each country has its unique characteristics, and some services might attract customers at different hours. Nevertheless, we believe it is reasonable for most entities, allowing for a deviation of up to two hours.

We propose finding an optimal alignment that minimizes the sum of absolute differences between weekly patterns. As we couldn't find an existing method in the literature, we developed a custom method (Algorithm 7.1) to solve the alignment task. We use the Mean Absolute Error (MAE) to the median of all patterns as objective, as it is less sensitive to large deviations than the Mean Square Error, to account for the presence of large values which follow a power-law distribution rather than a normal one.

Figure 7.6b displays the normalized activity patterns after the alignment process. The alignment algorithm yields two primary outcomes: (1) A vector S that assigns a time zone shift to each entity, and (2) A matrix W^S of weekly patterns where the goal is to align the patterns of all entities as closely as possible.

7.5.3 Validation of the alignment

To assess the effectiveness of our alignment process, we carefully handpicked 10 entities from the WalletExplorer collection of known users. We specifically chose these entities because we could confidently assign them a country of usage —not only domiciliation, thereby establishing a ground-truth time zone through manual research on their websites or historical data. Table 7.1 presents the list of these entities along with their expected approximate time zone and the corresponding estimated time zone shift. In most cases, we can accurately identify the region of the world with a reasonable level of precision.

Algorithm 7.1: Aligning process by minimizing the mean absolute error between normalized weekly patterns

```
   Data: W: Weekly entities activity matrix
 1 M ← medianByCol(W);
 2 W^S ← W;
 3 S ← [0] ∗ W.nbRow;
 4 previousTotalError ← +∞ ;
 5 currentTotalError ← W.length∗2;
 6 while previousTotalError > currentTotalError do
 7 |   previousTotalError ← currentTotalError ;
 8 |   currentTotalError ← 0;
 9 |   for i in W.nbRow do
10 |   |   w ← W_i minError ← +∞ ;
11 |   |   bestShift ← NULL ;
12 |   |   for shift in [0..23] do
13 |   |   |   w^S ←shifted(W_i, shift) error ← MAE(w^S, W_i) ;
14 |   |   |   if error < minError then
15 |   |   |   |   minError ← error ;
16 |   |   |   |   bestShift ← shift ;
17 |   |   |   end
18 |   |   |   W_i^S ← w^S
19 |   |   |   S[i] ← bestShift ;
20 |   |   |   currentTotalError ← currentTotalError + minError ;
21 |   |   end
22 |   end
23 end
```

7.5.4 Estimation of Bitcoin's activity geographical distribution

The alignment process allows us to estimate the time zone of FR entities. Although it provides an estimation, and does not distinguish between countries with similar time zones, it enables us to estimate the geographical distribution of Bitcoin's main players and track how this distribution evolves over time. Figures 7.7 (a and b) presents the number of entities for each time zone over the years, normalized by year(row), i.e., $c^{yearly}(y,h) = \frac{c(y,h)}{\sum_{i \in H} c(y,i)}$, where H is the 24 possible hour shifts. We compare this with a subset composed only of known entities from the widely used WalletExplorer[Janda, 2013] dataset. Both datasets display similar patterns, with a little less activity in Asia for the WalletExplorer one. WalletExplorer thus has a reason-

Entity	Expected Country	Approx. Time Zone	Avg. Estimated Shift
MeXBT.com	Mexico	GMT-6	-4.00
MercadoBitcoin.com.br	Brazil	GMT-3	-2.12
FoxBit.com.br	Brazil	GMT-3	-1.50
Paymium.com	France	GMT+2	+1.80
SimpleCoin.com.cz	Czech Republic	GMT+2	+2.00
BTC-e.com	Russia	GMT+3	+3.00
Exchanging.ir	Iran	GMT+3:30	+4.50
BX.in.th	Thailand	UTC+07	+8.75
Huobi.com	China	GMT+08	+9.67
CoinSpot.com.au	Australia	GMT+8/GMT+10	+11.40
Bitfinex.com	Unknown	Unknown	-3.50
SilkRoadMarketPlace	Unknown	Unknown	-3.00

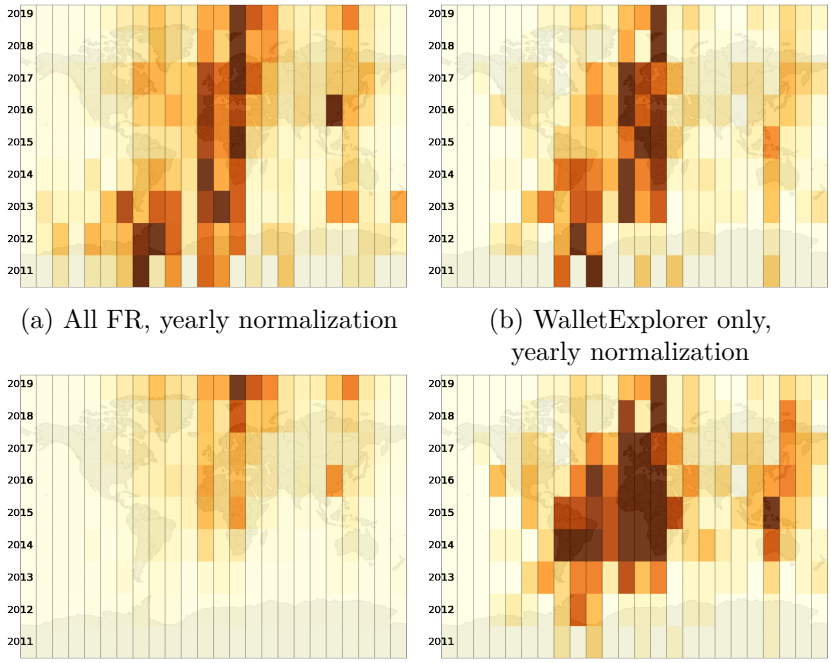
Table 7.1: Time zone of the 10 selected entities. We can observe that the estimated shift is close to the expected time zone. For the two entities at the bottom, for which we do not have *a priori* knowledge of the location, the method assigns a time zone that we can interpret as being located in the Americas.

able geographical representativeness of FR entities. We also observe a shift in time from an activity initially concentrated in the Americas to a larger concentration in the Euro-African time zones.

Figures 7.7 (c and d) display the same data, but with a global normalization, i.e., $c^{global}(y,h) = \frac{c(y,h)}{\sum_{x \in Y, i \in H} c(x,i)}$, where Y is the set of studied years. We observe a clear difference between the two. For all FR entities, we observe that the latest years concentrate most of the entities and that the historical importance of American entities nearly disappear compared with the later ones. East-European and African time zones seem to be the fastest growing and dominate the number of entities. Asia was particularly important for some years and regained importance at the end of our dataset.

On the other hand, we can observe that the WalletExplorer dataset has a completely different pattern. Most entities were active between 2014 and 2017, leading to a strongly biased view of the activity of the main players in the Bitcoin economy. This observation is coherent with the warning present on the WalletExplorer website, mentioning that *"Name database is NOT updated (except some very rare cases) since 2016"*[1]. Although this limitation is known, our work allows a quantitative estimation for the loss in representativeness, still frequently used (e.g., [Sun et al., 2022]). We can

[1] `https://www.walletexplorer.com/info`



(a) All FR, yearly normalization

(b) WalletExplorer only, yearly normalization

(c) All FR, global normalization

(d) WalletExplorer only, global normalization

Figure 7.7: Number of FR entities identified by time zone and by year. Each horizontal line corresponds to a year, and each vertical line to a time zone. The continents' contours are provided for easier interpretation of the time zone information.

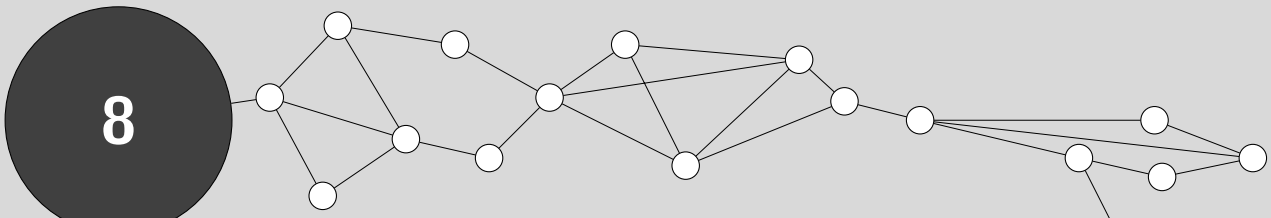
confirm this observation through the data: the fraction of all FR entities known in WalletExplorer declined from 30% in 2014 to less than 3% in 2019.

7.6 Discussion

Our work contributes to the understanding of the nature of Bitcoin real activity flow between different entity types.

We observed that a small number of commercial entities (FR) receive a massive amount of the payments, while customers (N1) make most of these payments. This is worrying for the decentralized nature of Bitcoin as cryptocurrency: most of the economic activity seems to go through central entities playing a role similar to the one of bank in the standard economic ecosystem.

In addition, we could locate entities geographically based on their activity. The time zone inference presented here should be viewed as an approximate representation rather than an unbiased reflection of reality. A surprising feature of the data is the minor importance of the American Continent in the latest years. A possible explanation can be that entities in North America in particular are global players, used all over the world. Hence, the algorithm may attribute them to an average value in the middle of the map. In future work, we could differentiate between local and global players by analyzing the amplitude of the weekly pattern. Indeed, a more international entity is likely to have a flatter temporal pattern compared to a national player with a more distinct one.



8 Conclusion

Bitcoin is the first and the most known cryptocurrency nowadays. Being a relatively new phenomenon, it instigates many studies in order to understand what it really is and how it is used. Its freely accessible data allows anyone with the skill to exploit it to do so. Who are Bitcoin users, what they are using it for, how much activity is really taking place, and how to identify cybercriminals, are just a few of the questions we may ask ourselves. These studies are not straightforward, as one is confronted with many challenges when studying blockchain data. The enormous quantity of data may be difficult to analyze and/or will take a very long time to compute. The lack of direct links between Bitcoin addresses and users' real names makes some tasks very difficult if not impossible. Because of these factors, data may be pre-processed before being analyzed. Address aggregates must be generated, and blockchain data must be enriched with data coming from external sources collected elsewhere on the internet.

8.1 Contributions

The first step presented in Chapter 4, echoing the difficulties just mentioned, was to collect and prepare the data. This was an important step, since all subsequent steps relies necessarily on the quality of the data used in input. In chapter 5 a method for improving address aggregates generation was presented. Using the data mentioned in the previous paragraph, we were able to predict the change output of transactions 7.1 using supervised learning models. Based on these predictions, we were able to achieve better results when aggregating Bitcoin addresses compared with previous works, in particular thanks to strategies for avoiding potential catastrophic errors. We then showed in chapter 6 the interest in using this method for a single-user study. Not only does the method allow identifying change outputs with higher precision, but the computational cost and time are also greatly reduced.

We presented also an analysis of the Bitcoin real activity in chapter 7. Be-

cause in part of the UTXO 7.1 nature of Bitcoin, not all transactions represent real exchanges. We presented three entity types (FR, N1, and TO 7.2) to represent different roles in the Bitcoin network. We could find a great concentration of transactions and amounts being traded, between a relatively small number of users. FR users (representing commercial actors) represent a small percentage of addresses but concentrate the most transactions and exchanged values found in the blockchain.

8.2 Difficulties

During this work, the complexity of the computations was an important point to consider. Many times we had to put a halt to the study itself, to find a less complex computational approach. On many occasions, estimated time to complete some calculations was not practical and needed a deeper understanding of the problem, in order to come up with a feasible method. Different data types containers and the problem complexity, as well as different programming languages or frameworks, had to be considered in different stages of this work.
The richness of details present in the data used also demanded a lot of attention. During data preparation, many computations took a considerable amount of time and sometimes a little misunderstanding of data details could jeopardize the quality of the results. Many points had to be carefully analyzed before launching the calculations, otherwise the reliability of the resulting data could be compromised.

8.3 Future Work

The Bitcoin blockchain data may inspire us to many other related subjects. With the current studies in mind, one may think of some other explorations that could be made aiming at a continuity of the present work.

CoinJoin detection : CoinJoin (See section 2.3) transactions may lead to errors when generating address aggregates, due to its capacity to foul the H1 heuristic, leading to erroneous merging of aggregates from different users. The results of address aggregation would greatly benefit from a study

in this matter.

Mixing : Mixing services may help cybercriminals, increasing anonymity and avoiding bitcoin flow tracing. A study in mixers' incoming and outgoing transactions may come to an aid in cybercrime prevention or recognition. This can also improve address aggregates generation, by identifying and linking addresses sending payments to mixing services and the ones receiving payments.

8.4 Conclusion

This work aimed at a better understanding of the Bitcoin transactions environment. Chapters 4, 5 and 6 presented a method for aggregating the bitcoin addresses of same users. We conducted an empirical analysis and showed that the method performed well and has promising applications. The high scores obtained were certainly due to various aspects of the method. This part of the work was the longest due to determining how to better construct the learning dataset, by selecting the right share of the collected data, and establishing and generating the right features. Because of trial and error experimentation, and long computational time, this process took some time to give the first positive results.
Chapter 7 shows a study about the real bitcoin flow. The main objective of this part is to filter transactions not representing real exchanges that may happen because of UXTO (2.3) and/or managing protocols from individual users. After that, we presented a heuristic to categorize users that may represent commercial entities, users that trade with them, and others that do not enter in either category. By performing a data analysis we were able to show that the group representing commercial users represents a small minority of addresses trading in the blockchain, but is responsible for the largest amounts traded. We could also determine temporal behavior patterns that allowed us to locate users geographically. Those results bring a new perspective when compared to previous works and may be useful in understanding the bitcoin phenomenon.

[Androulaki et al., 2013] Androulaki, E., Karame, G. O., Roeschlin, M., Scherer, T., and Capkun, S. (2013). Evaluating user privacy in bitcoin. In *International conference on financial cryptography and data security*, pages 34–51. Springer.

[Balthasar and Hernandez-Castro, 2017] Balthasar, T. d. and Hernandez-Castro, J. (2017). An analysis of bitcoin laundry services. In *Nordic Conference on Secure IT Systems*, pages 297–312. Springer.

[Bashir and Prusty, 2019] Bashir, I. and Prusty, N. (2019). *Advanced Blockchain Development: Build highly secure, decentralized applications and conduct secure transactions*. Packt Publishing Ltd.

[Biryukov et al., 2014] Biryukov, A., Khovratovich, D., and Pustogarov, I. (2014). Deanonymisation of clients in bitcoin p2p network. In *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security*, pages 15–29.

[Cazabet et al., 2018] Cazabet, R., Baccour, R., and Latapy, M. (2018). Tracking bitcoin users activity using community detection on a network of weak signals. In *Complex Networks and Applications*, pages 166–177.

[Chang and Svetinovic, 2018] Chang, T.-H. and Svetinovic, D. (2018). Improving bitcoin ownership identification using transaction patterns analysis. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 50(1):9–20.

[Chen and Guestrin, 2016] Chen, T. and Guestrin, C. (2016). Xgboost: A scalable tree boosting system. In Krishnapuram, B., Shah, M., Smola, A. J., Aggarwal, C. C., Shen, D., and Rastogi, R., editors, *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, August 13-17, 2016*, pages 785–794. ACM.

[Christin, 2013] Christin, N. (2013). Traveling the silk road: A measurement analysis of a large anonymous online marketplace. In *Proceedings of the 22nd international conference on World Wide Web*, pages 213–224.

[Dalal et al., 2021] Dalal, S., Wang, Z., and Sabharwal, S. (2021). Identifying ransomware actors in the bitcoin network. *arXiv preprint arXiv:2108.13807*.

[ElBahrawy et al., 2020] ElBahrawy, A., Alessandretti, L., Rusnac, L., Goldsmith, D., Teytelboym, A., and Baronchelli, A. (2020). Collective dynamics of dark web marketplaces. *Scientific reports*, 10(1):1–8.

[Ermilov et al., 2017] Ermilov, D., Panov, M., and Yanovich, Y. (2017). Automatic bitcoin address clustering. In *2017 16th IEEE International Conference on Machine Learning and Applications (ICMLA)*, pages 461–466. IEEE.

[Ferrin, 2015] Ferrin, D. (2015). A preliminary field guide for bitcoin transaction patterns. In *Proc. Texas Bitcoin Conf.*

[Gainsbury and Blaszczynski, 2017] Gainsbury, S. M. and Blaszczynski, A. (2017). How blockchain and cryptocurrency technology could revolutionize online gambling. *Gaming Law Review*, 21(7):482–492.

[Ghimire and Selvaraj, 2018] Ghimire, S. and Selvaraj, H. (2018). A survey on bitcoin cryptocurrency and its mining. In *2018 26th International Conference on Systems Engineering (ICSEng)*, pages 1–6. IEEE.

[Grinsztajn et al., 2022] Grinsztajn, L., Oyallon, E., and Varoquaux, G. (2022). Why do tree-based models still outperform deep learning on typical tabular data? *Advances in Neural Information Processing Systems*, 35:507–520.

[Harlev et al., 2018] Harlev, M. A., Yin, H. S., Langenheldt, K. C., Mukkamala, R. R., and Vatrapu, R. (2018). Breaking bad: De-anonymising entity types on the bitcoin blockchain using supervised machine learning. In Bui, T., editor, *51st Hawaii International Conference on System Sciences, HICSS 2018, Hilton Waikoloa Village, Hawaii, USA, January 3-6, 2018*, pages 1–10. ScholarSpace / AIS Electronic Library (AISeL).

[Harrigan and Fretter, 2016] Harrigan, M. and Fretter, C. (2016). The unreasonable effectiveness of address clustering. In *2016 Intl IEEE Conferences on Ubiquitous Intelligence & Computing, Advanced and Trusted Computing, Scalable Computing and Communications, Cloud*

and Big Data Computing, Internet of People, and Smart World Congress (UIC/ATC/ScalCom/CBDCom/IoP/SmartWorld), pages 368–373. IEEE.

[Huang et al., 2018] Huang, D. Y., Aliapoulios, M. M., Li, V. G., Invernizzi, L., Bursztein, E., McRoberts, K., Levin, J., Levchenko, K., Snoeren, A. C., and McCoy, D. (2018). Tracking ransomware end-to-end. In *2018 IEEE Symposium on Security and Privacy (SP)*, pages 618–631. IEEE.

[Janda, 2013] Janda, A. (2013). Walletexplorer.com: Smart bitcoin block explorer.

[Jourdan et al., 2018] Jourdan, M., Blandin, S., Wynter, L., and Deshpande, P. (2018). Characterizing entities in the bitcoin blockchain. In *2018 IEEE international conference on data mining workshops (ICDMW)*, pages 55–62. IEEE.

[Kang et al., 2020] Kang, C., Lee, C., Ko, K., Woo, J., and Hong, J. W.-K. (2020). De-anonymization of the bitcoin network using address clustering. In *Blockchain and Trustworthy Systems: Second International Conference, BlockSys 2020, Dali, China, August 6–7, 2020, Revised Selected Papers 2*, pages 489–501. Springer.

[Kantz and Schreiber, 2004] Kantz, H. and Schreiber, T. (2004). *Nonlinear Time Series Analysis*. Cambridge nonlinear science series. Cambridge University Press.

[Kappos et al., 2022] Kappos, G., Yousaf, H., Stütz, R., Rollet, S., Haslhofer, B., and Meiklejohn, S. (2022). How to peel a million: Validating and expanding bitcoin clusters. In *31st USENIX Security Symposium (USENIX Security 22)*, pages 2207–2223.

[Ke et al., 2017] Ke, G., Meng, Q., Finley, T., Wang, T., Chen, W., Ma, W., Ye, Q., and Liu, T.-Y. (2017). Lightgbm: A highly efficient gradient boosting decision tree. *Advances in neural information processing systems*, 30.

[Koshy et al., 2014] Koshy, P., Koshy, D., and McDaniel, P. (2014). An analysis of anonymity in bitcoin using p2p network traffic. In *International*

Conference on Financial Cryptography and Data Security, pages 469–485. Springer.

[Lewenberg et al., 2015] Lewenberg, Y., Bachrach, Y., Sompolinsky, Y., Zohar, A., and Rosenschein, J. S. (2015). Bitcoin mining pools: A cooperative game theoretic analysis. In *Proceedings of the 2015 international conference on autonomous agents and multiagent systems*, pages 919–927.

[Liao et al., 2016] Liao, K., Zhao, Z., Doupé, A., and Ahn, G.-J. (2016). Behind closed doors: measurement and analysis of cryptolocker ransoms in bitcoin. In *2016 APWG symposium on electronic crime research (eCrime)*, pages 1–13. IEEE.

[Lin et al., 2019] Lin, Y.-J., Wu, P.-W., Hsu, C.-H., Tu, I.-P., and Liao, S.-w. (2019). An evaluation of bitcoin address classification based on transaction history summarization. In *2019 IEEE International Conference on Blockchain and Cryptocurrency (ICBC)*, pages 302–310. IEEE.

[Loporchio et al., 2023] Loporchio, M., Bernasconi, A., Di Francesco Maesa, D., and Ricci, L. (2023). Is bitcoin gathering dust? an analysis of low-amount bitcoin transactions. *Applied Network Science*, 8(1):1–28.

[Lundberg and Lee, 2017] Lundberg, S. and Lee, S. (2017). A unified approach to interpreting model predictions. *CoRR*, abs/1705.07874.

[Lundberg et al., 2020] Lundberg, S. M., Erion, G. G., Chen, H., DeGrave, A. J., Prutkin, J. M., Nair, B., Katz, R., Himmelfarb, J., Bansal, N., and Lee, S. (2020). From local explanations to global understanding with explainable AI for trees. *Nat. Mach. Intell.*, 2(1):56–67.

[Meiklejohn et al., 2013] Meiklejohn, S., Pomarole, M., Jordan, G., Levchenko, K., McCoy, D., Voelker, G. M., and Savage, S. (2013). A fistful of bitcoins: characterizing payments among men with no names. In *Proceedings of the 2013 conference on Internet measurement conference*, pages 127–140.

[Mikolov et al., 2013a] Mikolov, T., Chen, K., Corrado, G., and Dean, J. (2013a). Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

[Mikolov et al., 2013b] Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., and Dean, J. (2013b). Distributed representations of words and phrases and their compositionality. *Advances in neural information processing systems*, 26.

[Monaco, 2015] Monaco, J. V. (2015). Identifying bitcoin users by transaction behavior. In *Biometric and surveillance technology for human and activity identification XII*, volume 9457, pages 25–39. SPIE.

[Moser, 2013] Moser, M. (2013). Anonymity of bitcoin transactions. *Münster Bitcoin Conference*.

[Möser et al., 2013] Möser, M., Böhme, R., and Breuker, D. (2013). An inquiry into money laundering tools in the bitcoin ecosystem. In *2013 APWG eCrime researchers summit*, pages 1–14. Ieee.

[Möser and Narayanan, 2022] Möser, M. and Narayanan, A. (2022). Resurrecting address clustering in bitcoin. In *International Conference on Financial Cryptography and Data Security*, pages 386–403. Springer.

[Nakamoto, 2008] Nakamoto, S. (2008). Bitcoin: A peer-to-peer electronic cash system. *Decentralized Business Review*, page 21260.

[Neudecker and Hartenstein, 2017] Neudecker, T. and Hartenstein, H. (2017). Could network information facilitate address clustering in bitcoin? In *Financial Cryptography and Data Security: FC 2017 International Workshops, WAHC, BITCOIN, VOTING, WTSC, and TA, Sliema, Malta, April 7, 2017, Revised Selected Papers 21*, pages 155–169. Springer.

[Nick, 2015] Nick, J. D. (2015). Data-driven de-anonymization in bitcoin. Master's thesis, ETH-Zürich.

[Prokhorenkova et al., 2018] Prokhorenkova, L. O., Gusev, G., Vorobev, A., Dorogush, A. V., and Gulin, A. (2018). Catboost: unbiased boosting with categorical features. In Bengio, S., Wallach, H. M., Larochelle, H., Grauman, K., Cesa-Bianchi, N., and Garnett, R., editors, *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada*, pages 6639–6649.

[Quinlan, 1996] Quinlan, J. R. (1996). Learning decision tree classifiers. *ACM Comput. Surv.*, 28(1):71–72.

[Ranshous et al., 2017] Ranshous, S., Joslyn, C. A., Kreyling, S., Nowak, K., Samatova, N. F., West, C. L., and Winters, S. (2017). Exchange pattern mining in the bitcoin transaction directed hypergraph. In *International conference on financial cryptography and data security*, pages 248–263. Springer.

[Reid and Harrigan, 2011] Reid, F. and Harrigan, M. (2011). An analysis of anonymity in the bitcoin system. In *SocialCom/PASSAT*, pages 1318–1326. IEEE Computer Society.

[Shao et al., 2018] Shao, W., Li, H., Chen, M., Jia, C., Liu, C., and Wang, Z. (2018). Identifying bitcoin users using deep neural network. In *International Conference on Algorithms and Architectures for Parallel Processing*, pages 178–192. Springer.

[Soska and Christin, 2015] Soska, K. and Christin, N. (2015). Measuring the longitudinal evolution of the online anonymous marketplace ecosystem. In *24th {USENIX} security symposium ({USENIX} security 15)*, pages 33–48.

[Spagnuolo et al., 2014] Spagnuolo, M., Maggi, F., and Zanero, S. (2014). BitIodine: Extracting Intelligence from the Bitcoin Network. In *Financial Cryptography and Data Security*, Lecture Notes in Computer Science (LNCS), pages 457–468, Barbados. Springer Berlin Heidelberg.

[Sun et al., 2022] Sun, Y., Xiong, H., Yiu, S. M., and Lam, K. Y. (2022). Bitanalysis: A visualization system for bitcoin wallet investigation. *IEEE Transactions on Big Data*.

[Tovanich and Cazabet, 2022] Tovanich, N. and Cazabet, R. (2022). Pattern analysis of money flows in the bitcoin blockchain. In *International Conference on Complex Networks and Their Applications*, pages 443–455. Springer.

[Tovanich and Cazabet, 2023] Tovanich, N. and Cazabet, R. (2023). Fingerprinting bitcoin entities using money flow representation learning. *Applied Network Science*, 8(1):1–22.

[Tovanich et al., 2021a] Tovanich, N., Soulié, N., Heulot, N., and Isenberg, P. (2021a). An empirical analysis of pool hopping behavior in the bitcoin blockchain. In *2021 IEEE International Conference on Blockchain and Cryptocurrency (ICBC)*, pages 1–9. IEEE.

[Tovanich et al., 2021b] Tovanich, N., Soulié, N., and Isenberg, P. (2021b). Visual analytics of bitcoin mining pool evolution: on the road toward stability? In *2021 11th IFIP International Conference on New Technologies, Mobility and Security (NTMS)*, pages 1–5. IEEE.

[Toyoda et al., 2018] Toyoda, K., Ohtsuki, T., and Mathiopoulos, P. T. (2018). Multi-class bitcoin-enabled service identification based on transaction history summarization. In *2018 IEEE international conference on internet of things (iThings) and IEEE green computing and communications (GreenCom) and IEEE cyber, physical and social computing (CPSCom) and IEEE smart data (SmartData)*, pages 1153–1160. IEEE.

[Van Wegberg et al., 2018] Van Wegberg, R., Oerlemans, J.-J., and van Deventer, O. (2018). Bitcoin money laundering: mixed results? an explorative study on money laundering of cybercrime proceeds using bitcoin. *Journal of Financial Crime*.

[Wang et al., 2020] Wang, M., Ichijo, H., and Xiao, B. (2020). Cryptocurrency address clustering and labeling. *arXiv preprint arXiv:2003.13399*.

[Yazdinejad et al., 2020] Yazdinejad, A., HaddadPajouh, H., Dehghantanha, A., Parizi, R. M., Srivastava, G., and Chen, M.-Y. (2020). Cryptocurrency malware hunting: A deep recurrent neural network approach. *Applied Soft Computing*, 96:106630.

[Yin and Vatrapu, 2017] Yin, H. S. and Vatrapu, R. (2017). A first estimation of the proportion of cybercriminal entities in the bitcoin ecosystem using supervised machine learning. In *2017 IEEE international conference on big data (Big Data)*, pages 3690–3699. IEEE.

[Zhang et al., 2020] Zhang, Y., Wang, J., and Luo, J. (2020). Heuristic-based address clustering in bitcoin. *IEEE Access*, 8:210582–210591.

[Zhang et al., 2018] Zhang, Z., Zhou, T., and Xie, Z. (2018). Bitscope: Scaling bitcoin address de-anonymization using multi-resolution clustering. In

Proceedings of the 51st Hawaii International Conference on System Sciences, pages 1–11.

[Zheng et al., 2020] Zheng, B., Zhu, L., Shen, M., Du, X., and Guizani, M. (2020). Identifying the vulnerabilities of bitcoin anonymous mechanism based on address clustering. *Science China Information Sciences*, 63(3):1–15.

A.1 Hardware

Beyond usual personal computers, a more powerful server was purchased for the project. Bought at the beginning of the project, and later updated to increase the amount of RAM and SSD capacity, its final characteristics were as follows:

- OS: Ubuntu 18.04.6 LTS x86_64
- Kernel: 5.4.0-131-generic
- CPU: Intel Xeon E5-2630 v4 (20) @ 3.100GHz
- GPU: NVIDIA GeForce GTX 1080
- Memory: 128 GB RAM
- HDD 5To
- RAID 0 4To (2x SSD 2To)

A.2 Tools and libraries

Multiple tools and libraries were used during this study, in particular to overcome the difficulty of dealing with very large datasets. Some tools were abandoned after some time since they were not fit for this kind of work, given our material constraints. We may name Neo4j as an example, a database system with a native graph storage structure. Because of the graph nature of Bitcoin transactions, this approach seemed appropriate, but we realized that the performances were not compatible with our large-scale analysis, unless the full database was charged in We list in the following libraries and tools used for this thesis:

- scikit-learn[1] : well-known and widely used machine learning Python

[1] https://scikit-learn.org/

library;

- XGBoost[2]: one of the leading machine learning models at the current time. It stands for Extreme Gradient Boosting, and it implements gradient-boosted random forests;
- CatBoost[3]: another high-performance machine learning model. As XGBoost, it also implements gradient-boosted trees;
- bitcoin-etl[4]: Python library to decode and convert data from the raw binary format found in the blockchain to a more standard and interpretable JSON format.
- Snap.py[5]: a python interface for the SNAP library. This library is used for the analysis and manipulation of large networks;
- Networkx[6]: a Python library for creating and manipulating networks. Differently from SNAP, networkx is not adapted for working with networks with a large number of nodes.
- PySpark[7]: pyspark is a python interface to the Spark engine, initially developed in Scala. Spark is a high-performance, large-scale data processing software suite.
- SHAP[8]: a Python library for interpreting predictions of machine learning models. SHAP applies a game theory approach for the explicability of the predictions obtained.

[2]https://xgboost.ai/
[3]https://catboost.ai/
[4]https://github.com/blockchain-etl/bitcoin-etl
[5]https://snap.stanford.edu/snappy
[6]https://networkx.org/
[7]https://spark.apache.org/docs/latest/api/python/
[8]https://github.com/slundberg/shap

The works presented in this thesis were published in the following papers:

- Tubino, R.R., Cazabet R. and Robardet C. (2022). Vers une meilleure identification d'acteurs de Bitcoin par apprentissage supervisé. *Conférence francophone sur l'Extraction et la Gestion des Connaissances (EGC 2022),* 28 January 2022, Blois (France), pp. 171-182. HAL : hal-04193274.

- Tubino, R.R., Cazabet R. and Robardet C. (2022). Towards a better identification of Bitcoin actors by supervised learning. *Data and Knowledge Engineering,* vol. 142, p. 102094.doi : 10.1016/j.datak.2022.102094. HAL : hal-03879416.

- Tubino, R.R., Cazabet R., Tovanich N. and Robardet C. (2023). Temporal and Geographical Analysis of Real Economic Activities in the Bitcoin Blockchain. *LIMBO@ECML/PKDD 2023: International workshop on LearnIng and Mining for BlOckchains,* 18 September 2023, Turin (Italy). HAL : hal-04188062.