Table of Contents

Chapter 1 – Introduction

A Ponzi scheme is an investment fraud that attracts individuals by promising above market risk-free returns and that repays existing investors with funds contributed by new ones (SEC, n.d.). The underlying sector that supposedly will generate those returns can involve gold mines, synthetic rubies, tropical islands and stock market investments (Mohammed, 2021). Throughout history, Ponzi schemes have evolved and have embezzled sizable amounts of money, reaching the peak with Madoff's scheme that stole 64.8 billion dollars (Carvajal et al., 2009). These frauds are documented to continuously take place throughout the world, affecting both developed and developing economies, causing dire consequences. Specifically, in the U.S. Ponzi schemes worth 5.3 billion USD have been detected in 2022 (Ponzitracker, 2023). Interestingly, over 25% of the uncovered Ponzi frauds were related to cryptocurrencies. As per what concerns developing countries, Mohammed (2021) observed that Ponzi schemes taking place in Africa distress mostly the underprivileged and breadwinners, negatively affecting the family's education, health and, in some cases, causing deaths. Those effects however are common across different continents and regions, as documented in Albania (Thanasi and Riotto, 2017) and in the Caribbeans (Carvajal et al., 2009).

Throughout history, Ponzi frauds have evolved and are now transitioning towards digital versions, exploiting cryptocurrencies. In this regard, Springer (2020) reports the advantages for perpetrators as cryptos remove geographical barriers, ensure anonymity, and deny the possibility of having the money sized. This is worsened by the continuous growth that digital currencies are experiencing. More precisely, Hicks (2023) identified 23,000 cryptocurrencies active in 2023 totaling a 1.1 trillion USD market capitalization. The aforementioned advantages for fraudsters and the hype surrounding cryptocurrency have had dramatic effects on cybercrime. In fact, Rob Wright, the head of Europol, reported, in 2018, that 5 billion USD have

been laundered in the European continent (The Economist, 2018). Further, CipherTrace identified that 4.52 billion USD have been stolen on cryptocurrency exchanges (Badawi and Jourdan, 2020). Finally, FTC discovered, in 2021, that the value of frauds and scams involving digital currencies reached 14 billion USD, causing median losses to users of around 1,900 USD (Kutera, 2022). Specifically, cybercrime related to cryptocurrencies takes place in four different ways: Ransomware, responsible for 456.8 million USD[1] in 2022, Money Laundering embezzling 23.8 billion USD[2], High Yield Investment Programs (i.e., Ponzi Schemes) stealing 7.8 billion USD[3], and Pump and Dump schemes defrauding individuals of 4.6 billion USD[4].

Regarding Ponzi schemes, the development of Ethereum paved the way for the implementation of automatic Ponzi frauds via smart contracts. The latter provide perpetrators a wide set of benefits, including: anonymity and protection against incrimination, apparent reliability and credibility provided by the open source and automatic nature of the smart contract and, immutability of the contract once deployed. The combination of those factors and the hype surrounding cryptocurrencies allows those frauds to scale.

Further, individuals are recruited to those schemes oftentimes via bots that can mimic human behavior. Specifically, Nizzoli et al. (2020) identify that by following generic cryptocurrency discussion threads on Twitter, 56.6% of the links provided to seemingly discuss deeper the topic at hand were involved in deception and fraud. Similar results are also identified when analyzing Telegram, in which the majority of crypto related talks are associated with Ponzi schemes and Pump and Dump actions (Nizzoli et al., 2020). To make those results even more worrying, Ponzi frauds are pitched also on websites

[1] Source: https://www.chainalysis.com/blog/crypto-ransomware-revenue-down-as-victims-refuse-to-pay/

[2] Source: https://www.idnow.io/blog/how-criminals-leverage-crypto-money-laundering/

[3] Source: https://cointelegraph.com/news/billions-lost-in-crypto-ponzi-schemes-in-2022

[4] Source: https://cryptopotato.com/24-of-new-tokens-in-2022-were-likely-pump-and-dump-schemes-report/

such as bitcointalk in threads destined to beginner users (Vasek and Moore, 2019).

Currently, three main types of attempts in detecting Ponzi smart contracts on Ethereum have been exploited, with each using different data sources, namely: the source code, the bytecode and the transaction records.

The first approach seeks to identify Ponzi schemes by analyzing the reward distribution entailed in the source code of the contract. The most important contribution made in detection with this data is by Chen et al. (2021a) who, through a Multi-channel Text Convolutional Neural Network and Transformer are capable of extracting structural and semantic features, reaching a 0.89 F1 score. Nonetheless, the method is not generalizable as only 23% of the smart contracts have released their source code (Zheng et al., 2020).

The second type of data has received the most attention from the literature and seeks to identify Ponzi schemes by analyzing the series of instructions that are passed to the Ethereum Virtual Machine (Chen et al., 2018). Generally, since bytecodes are not human readable, specific disassemblers are employed that can extract opcodes and operands. An important advantage of exploiting this type of data is that it is always publicly available as, without it, miners would not be able to implement the smart contract. In this regard, major contributions have been made by Shen et al. (2021), Fan et al. (2021) and Chen et al. (2021b), who, respectively, achieved 0.88, 0.96 and 0.96 F1 scores. Importantly, bytecodes-based algorithms can detect Ponzi schemes as soon as the smart contract is deployed, thus limiting the harm caused to individuals. Yet, as Chen et al. (2021b) highlight current methods will struggle in identifying Ponzi smart contracts combining multiple schemes, new Ponzi frauds creating different opcodes patterns and schemes exploiting encryption. Thus, to limit those drawbacks, transaction-based methods can be implemented, acting as a second stage detection system.

Current research focusing on transactions mostly extracts few features and proceeds to classify smart contracts based on them. In particular, Chen et al. (2018) exploit 7 features, Chen et al. (2019) employ 13, Galletta and Pinelli (2023) use 27 and Yu et al. (2021) 14, respectively attaining F1 scores of 0.44, 0.30, 0.62 and 0.79. Importantly, those classification attempts all rely on the full transaction history to detect Ponzi smart contracts, ultimately rendering vain those attempts as they cannot reduce the harm caused to investors. Additionally, the features used tend to be limited, meaning that they cannot accurately describe the behavior of Ponzi schemes.

Therefore, the goal of this thesis is to first document the differences and how Smart Ponzi schemes behave on Ethereum. Next, 87 total features, including the smart contract name, will be extracted and used for classification. This approach aims at providing the different classifiers with a vast set of characteristics that can help in detecting smart contracts. In fact, the features engineered seek to measure the full behavior of Ponzi schemes, benefiting the detection capabilities. Importantly, contrary to the bulk of the literature, the classifiers will be exposed to a limited number of transactions for classification, allowing for temporal detection of Ponzi frauds. Finally, once the best overall classifier and the best temporal one have been identified, an in-depth investigation of the most relevant characteristics used for classification will follow, identifying also possible differences across the two considered models. To summarize, the research questions will be the following:

- ➔ RQ1: How effective is the proposed framework in detecting in a timely manner Ponzi smart contracts in Ethereum?
- ➔ RQ2: What is the ideal transaction threshold required for optimal detection?
- ➔ RQ3: What are the key characteristics that are essential for the classification of Ponzi Smart Contracts?

To answer the research questions, I will analyze the transactions of 222 Ponzi smart contracts and 1814 non-Ponzi schemes gathered from Etherscan.io. Specifically, to allow models to learn with few transactions, the feature extraction procedure will take place on a limited number of transactions. In fact, the data will be randomly split into training, validation, and test, devising 10 transaction steps, each characterized by 10 transactions. Further, at each step, only the smart contracts having at least transaction step * 10 transactions will be considered. Due to the severe class imbalance and the limited available smart contracts, two different augmentations will be tested, namely: minority oversampling and Temporal Evolution Augmentation of Transaction Graph (TEAUG). Minority oversampling is applied by testing Random, SMOTE, Borderline SMOTE and Adasyn oversampling. TEAUG, instead, is a data augmentation procedure first suggested by Jin et al. (2022) which entails carrying over the smart contracts of the previous steps to increase the available observations employed by the models.
Further, to include possible topological information embedded in the network structure surrounding the smart contracts, Graph2Vec and GNNs are applied.

Overall, the different models tested showcase satisfactory detection capabilities across the various transaction steps considered. Specifically, the best overall model observed refers to a Gradient Boosting model featuring both Borderline SMOTE and TEAUG, that reached an F1 score of 0.857, an Auroc of 0.908, a Precision of 0.889 and a Recall of 0.828, using only the first 80 transactions available. Further, the best early warning model refers to a Gradient Boosting one with both TEAUG and SMOTE, that, by using only a total of 30 transactions, achieved an F1 score of 0.837, an Auroc of 0.884, a Precision score of 0.857 and a Recall of 0.818.
Finally, by investigating the most relevant characteristics to distinguish between the two types of smart contracts, 3 main families of features are

common across both the best overall model and the early warning one, namely: the contract value in in-degrees/out-degrees, the outflows/inflows from users and the contract name. Moreover, as the number of transactions used for the feature extraction procedure increases, the model shifts its main focus from the outflows/inflows from users to the contract value in in-degrees/out-degrees.

The remaining part of the thesis is structured as follows: Chapter 2 reviews the literature on Ponzi schemes, blockchain technology and current attempts at detecting those frauds; Chapter 3 details the data gathering procedure, the cleaning and feature engineering steps and performs an exploratory analysis of the features extracted across the two types of smart contracts; Chapter 4 reviews the modelling framework and explains in details the features and augmentations used, as well as the fitting metrics considered; Chapter 5 presents the results and relates them to the research questions; Finally, Chapter 6 concludes.

Chapter 2 – Literature Review

2.1 Ponzi schemes' general definition, overview, and evolution over time

2.1.1 Definition and Overview

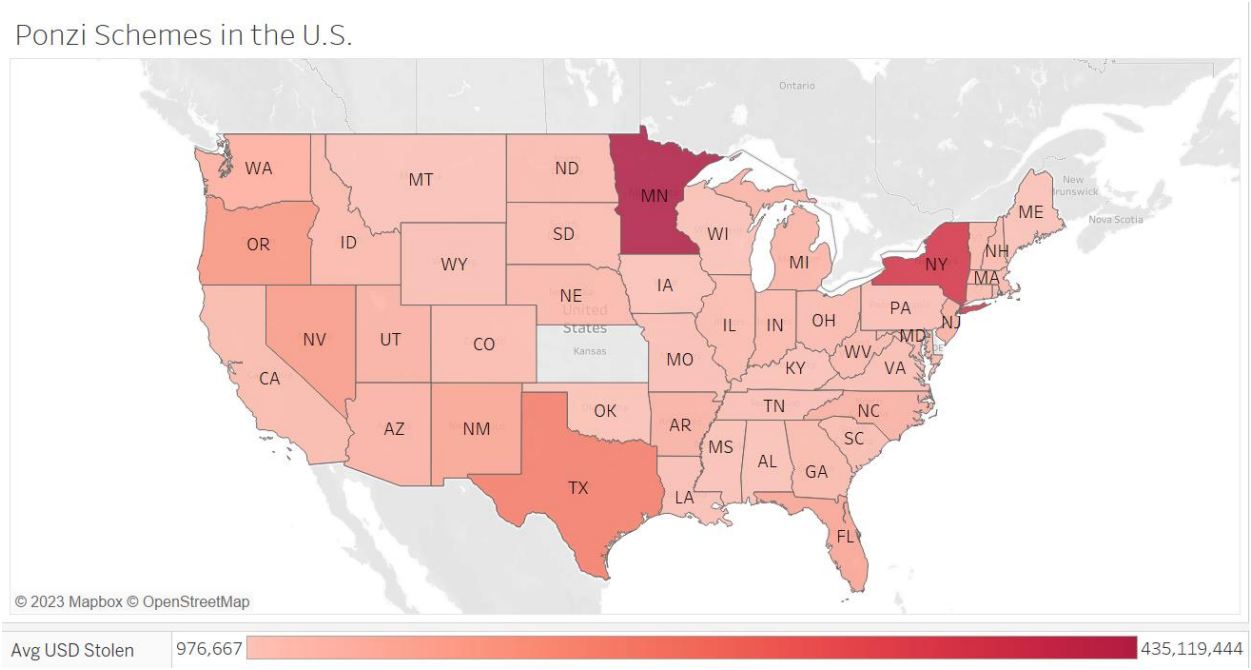
The U.S. Securities and Exchange Commission (SEC) defines Ponzi schemes in the following way:

"A Ponzi scheme is an investment fraud that involves the payment of purported returns to existing investors from funds contributed by new investors. Ponzi scheme organizers often solicit new investors by promising to invest funds in opportunities claimed to generate high returns with little or no risk. With little or no legitimate earnings, Ponzi schemes require a constant flow of money from new investors to continue. Ponzi schemes inevitably collapse, most often when it becomes difficult to recruit new investors or when a large number of investors ask for their funds to be returned" SEC (n.d.).

Generally, the Ponzi perpetrator will lure individuals by proposing investment strategies that, through the special abilities of the schemer, will generate above-market returns without being exposed to any risk (Thanasi and Riotto, 2017). The advertised opportunity can involve a plethora of sectors, including gold mines, synthetic rubies, hydroponic farming, tropical islands, and, most commonly, stock market investments (Mohammed, 2021). Importantly, the perpetrator will not be transparent about the profit-generating investment, justifying the opacity as a means to protect the business (Thanasi and Riotto, 2017). The victims are typically targeted based on a set of common characteristics, namely affinity in ethnicity, religion, or profession. Through pervasive and persistent tactics, Ponzi schemes have been able to flourish even in developed countries with a strong regulatory framework (Carvajal et al.,

2009). In this regard, the figure below summarizes the average sums of money swallowed by Ponzi schemes by US State.

Figure 1: *Average USD attracted by Ponzi Schemes in the U.S*.



Source: Own elaboration of Ponzi Tracker data[5]

The SEC established that the commonalities across Ponzi schemes relate to high risk-free and consistent returns, unregistered investments and sellers, secretive and complex business strategies, and difficulty in receiving payments (Mohammed, 2021). Relating to the latter characteristic, it has been observed that investors attempting to cash out will face difficulties. More precisely, the Ponzi schemer will try two different tactics: incentivize the investor to roll over their investments by promising even higher returns or allow only a gradual withdrawal (Mohammed, 2021). Furthermore, once rumors about fraud start to spread, Ponzi perpetrators will encourage investors to stay in the game as it is the only mean to reach financial independence. In reality, those approaches are performed to attract new funds and hide the fraud for as long

[5] Source: https://www.ponzitracker.com/ponzi-database

as possible (Thanasi and Riotto, 2017). Overall, the scheme will fail under three possible scenarios: the perpetrator takes the money and runs, recruiting slows down, and a high number of investors demand their initial capital and the interest earned (Thanasi and Riotto, 2017).

Moffatt (2018), by analyzing different Ponzi schemes, detected distinct phases of those frauds. It all starts with "The Benefit", which is the advertised and often guaranteed rate of return that will be generated. Then, "The Setup", a general overview of how the returns will be generated, is provided. The two subsequent steps are crucial for a successful Ponzi scheme: "Initial Credibility", relating to the schemer's credibility, trustworthiness, and ability, and the "Pay-off to initial investors" essential for the popularity of the fraud. Finally, after those steps have been performed, the schemer will need to "Communicate the success" and pay off the initial investors, allowing the number of investors to grow exponentially (Moffatt, 2018).

Jory and Perry (2011) complement those steps by focusing on the strategies employed to draw in investors. More precisely, perpetrators will overestimate the funds under management and send regular fictitious profit statements. Further, to win the trust of faith-based or specific ethnicities Bible-speak is commonly used (Jory and Perry, 2011). In this regard, Springer (2020) documents that perpetrators tend to be intelligent, friendly, and charismatic individuals.

The growth and death of Ponzi schemes is typically correlated with the phase of the economy. Specifically, in the first decade of the 2000s, a period characterized by a bullish stock market, the number of Ponzi schemes grew significantly (Jory and Perry, 2011). During the 2008-2009 financial crisis, most of them failed.

On average, Ponzi schemes attract 100 investors but, in certain cases, the number of victims and, also, the registered losses can be much higher. For

instance, Madoff's Ponzi scheme, the largest in history, affected around 5,000 investors for a total fraud of 64.8 billion dollars (Carvajal et al., 2009).

Importantly, a major hunting ground for perpetrators is countries with weak regulatory frameworks. This is illustrated by the case of Albania, and by more recent cases in the Caribbeans and Africa (Carvajal et al., 2009).

The major Ponzi schemes in history will be further investigated in the following section.

2.1.2 Major Ponzi Schemes in History and their consequences

Ponzi schemes are named after Charles Ponzi, a con artist that, in 1920, was capable of defrauding investors of 15 million USD in eight months (Darby, 2021). In particular, the Italian fraudster observed that there was a fixed conversion rate between postage stamps between different countries. Therefore, one Spanish postage stamp could always be converted for a US one, despite the difference in price between the two. With this observation, Charles Ponzi created the Securities Exchange Company, claiming that through a (non-existent) network of agents in different countries, arbitrage opportunities could lead to up to 50% returns in 90 days (Darby, 2021). The scheme attracted 40,000 individuals in the Boston area by exploiting two factors: the high promised returns and the remuneration scheme for the sales agents. The latter, in fact, received a 10% commission for each investment they brought in and 5% of the investments brought by the recruited subagents.

Following an investigation by the postal and legal authorities, rumors started to spread about the potential fraudulent nature of Ponzi's company, ultimately leading to an investors' run. He was later convicted to 31 years in prison and deportation to Italy (Darby, 2021).

Charles Ponzi, however, was not the first person to create such schemes. The origins, in fact, date back to the XVIII century, when John Law, a Scottish

Economist, created a scheme that caused high levels of speculation and led to the Mississippi bubble (Thanasi and Riotto, 2017).

The success of Ponzi in attracting investors' money prompted, in later years, the naissance of a plethora of those schemes. Recently, the majority of frauds originated in developing and third countries. In particular, in 2010 in India, a married couple promised returns of 20% per month for 6 months by allegedly investing in stocks. It was later revealed that the couple did not possess a stock trading license and that the business idea was in fact a Ponzi scheme (Mohammed, 2021). In China, Ezubao, a lending company harmed 900,000 investors through an online peer-to-peer lending scheme for a total of 7.6 billion USD (Mohammed, 2021). In the African context, important Ponzi schemes relate to: Barry Tannenbaum and BTC Global. The former attracted investors through a guaranteed 200% annual return which was allegedly financed by pharmaceutical imports. Overall, investors lost 1.2 billion USD (Mohammed, 2021). The latter refers to a scheme located in South Africa that involved 28,000 individuals, 80 million USD, and pivoted around Bitcoin. More precisely, investors were attracted by the promise of 2% daily returns, 14% weekly, and 50% monthly returns (Bonorchis, 2018).

Other noteworthy schemes that have eroded Africa are Pyram, Resource 5, Unique Sheperd, Savanna Gold Ltd, US Tilapia, Jastar Motors, DG Capital, and Care for Humanity (Mohammed, 2021). Those frauds had dramatic effects on the countries involved in three dimensions: loss of savings and death, leakage to the financial system, and burdens for the taxpayers. The first relates to the vast losses that affect all the investors involved who, in most cases, are underprivileged and breadwinners of their families looking for quick ways to raise money. Those losses have negative spillover effects on the family's education, health, and general livelihood (Mohammed, 2021). The second refers to the transfers of money from real banks to the Ponzi schemes and to loans taken out to invest in those fraudulent opportunities. Due to the DKM

and Menzgold Ponzi schemes which took place in Ghana, the share of non-performing loans to all bank loans observed a 90% increase, moving from 11.27% in 2014 to 21.50% in 2017 (Mohammed, 2021). Finally, once Ponzi schemes affect a vast number of individuals, bailouts from central banks are typically put in place, leading to reductions in the government budget for development projects (Mohammed, 2021).

In the European Area, major Ponzi schemes were the kings club and the Albania case. The kings club operated between 1991 and 1994 in Germany, Austria, and Switzerland. Essentially, the founders sold letters that promised a 70% yearly return to their investors. The scheme was discovered to be a Ponzi with investors losing 1 billion USD (Mohammed, 2021). The Albania case refers to the biggest Ponzi scheme in terms of reach as, overall, 57% of the Albanian population was involved (Thanasi and Riotto, 2017). In particular, the country in the 90s transitioned from a communist to a capitalist-based economy. This change spread the desire amongst the population to become rich quickly with fraudsters exploiting this climate. In the country, there were only 5 state-owned banks that could not satisfy the private sector demand, leading the way to informal lending companies. The three main newly established money collectors were Xhaferri, Populli, and Sude which had no real investments and initially offered interest rates of 5% per month (Thanasi and Riotto, 2017). The competition between those firms led to a spike in the interest rates guaranteed to their investors, reaching their maximum at 30% per month (Thanasi and Riotto, 2017). Thanks to payouts to the initial investors and significant endorsements both from the media and politicians, the attraction power of those firms grew and with it the number of individuals involved. Further, the low levels of financial knowledge and the lack of experience with financial markets led the population to not perceive the potential risks of their investments (Thanasi and Riotto, 2017). Greediness and misconception of risk caused people to sell their houses, and farmers to sell

their livestock and invest all their proceeds in Ponzi schemes. The consequences of this frenzy were disastrous. During the turbulences of 1997, 3500 people were killed, 5000 were injured and protests took place all over the country (Thanasi and Riotto, 2017). Overall, the schemes swallowed 1.2 billion USD of savings, while the average wage was only 80 USD, impacting 57% of the population and the liabilities of those firms accounted for 51% of the country's GDP (Thanasi and Riotto, 2017). Moreover, following the burst of the Ponzi bubble, inflation increased to 42%, output decreased by 7% and the Albanian Lek depreciated by almost 100% (Thanasi and Riotto, 2017).

Carvajal et al. (2009) have studied and collected information about the major Ponzi schemes that took place in the Caribbeans, focusing on their dire consequences and the key regulatory conditions required for the detection and stoppage of the frauds. In particular, adding to the aforementioned effects of Ponzi frauds in Ghana and Albania, other aftermaths of those schemes relate to a loss of confidence in the financial system and regulatory bodies, savings being diverted from productive to unproductive uses and socio-economic protests (Carvajal et al., 2009). The authors also noted that once schemes grow in size, they tend to cover up their deceptive nature via charitable and political contributions.

Regulatory actions to limit the reach and effects of Ponzi schemes are affected by a set of prerequisites, namely: independence of financial regulators, gaps in the legal and regulatory framework, international cooperation, law enforcement, and efficient court proceedings (Carvajal et al., 2009).
Concerning the first point, as observed in both the Ghana case and the Albania one, politicians, by being involved, had no incentives to support regulatory actions to stop the fraud. Further, the direct contributions made by Ponzi schemes to support governmental and social causes make it challenging to accurately investigate those companies. Thus, financial regulators and

prosecutors must have adequate independence and protection against possible lawsuits arising from their activities (Carvajal et al., 2009).

The second incorporates four elements for prompt action against those frauds, including clear provisions to prosecute the schemes, broad investigative authority, civil/administrative/criminal sanctions, and emergency relief. Those four aspects should guarantee the possibility of "following the money" a precondition that is essential for the detection of the pyramid structure of most of those schemes (Carvajal et al., 2009). International cooperation is strongly correlated to this last point as, by allowing banks to cooperate more across borders, it would be easier to trace the flow of money and, thus, detect the schemes (Carvajal et al., 2009). Finally, law enforcement and efficient court proceedings pivot around the need for experience and expertise in financial fraud detection. In this regard, international cooperation could have positive spillover effects.

The commonalities across the presented Ponzi schemes relate to the breadth of reach that perpetrators could achieve. The next section will further investigate the traits that characterize fraudsters.

2.1.3 Psychological traits of perpetrators and their victims

A study by Jory and Perry (2011) identified that founders of Ponzi schemes tend to be males with a finance background that are between 36 and 55 years old. Generally, they are charismatic salesmen, with exceptional persuasive abilities, that exploit psychological approaches to draw individuals into their frauds. Jacob and Schain (2011) identified that perpetrators tend to be individuals experiencing the so-called "Illusion of control" leading them to develop a false and exaggerated sense of their ability. Neutralization theory can help explain how those individuals can rationalize their illicit behavior as necessary and unavoidable. More precisely, such theory is characterized by 5 components: denial of responsibility, denial of injury, denial of victim,

condemnation of condemners, and appeal to higher loyalties (Jacob and Schain, 2011). The first relates to the tendency of blaming deceptive behavior on external forces, and the second and third comprise the absence of immediate effects on the victims and seeing them as mere enemies (Jacob and Schain, 2011). The latter two refer to the neutralization of blame by questioning authority and demanding the loyalty of friends (Jacob and Schain, 2011).

Bhattacharya (2003) highlights that the three essential components for the development of a Ponzi scheme are: convincing individuals about an investment opportunity, guaranteeing a high return, and building credibility through initial payouts. Relating to the first point, different factors and strategies can be leveraged, such as situation, cognition, personality, emotion, reciprocation, commitment, social proof, authority, and liking (Jacob and Schain, 2011). More precisely, perpetrators will exploit socio-economic pressures affecting breadwinners and individuals with unstable financial situations looking for an easy way out (Jacob and Schain, 2011). Typically, family and friends are the first victims as they are easily influenced by the need of returning favors, trust, and liking towards the schemer (Jacob and Schain, 2011).

Other strategies frequently used to attract individuals rely on affinity in terms of ethnic or religious groups. In fact, the common belief is that someone from the same background would never deceive you (Jacob and Schain, 2011). Springer (2020) exemplifies Ponzi schemes leveraging affinity to expand their reach, those are: eAdGear, DFRF Enterprises, Tropikadget, and Universo Foneclub. The former fraud operated between 2010 and 2014, with an alleged business relating to optimizing search engines to increase the visibility of potential clients. The main targeted group where Chinese individuals living in the United States (Springer, 2020). DFRF Enterprises attracted investors by supposedly investing in gold mines located in Brazil and Africa. The scheme operated between 2014 and 2015, targeting mainly Spanish and Portuguese

US citizens, and required an initial 1,000 USD fee (Springer, 2020). Similarly, Tropikadget was another fraud targeting Spaning and Portuguese US citizens. It operated between 2013 and 2014, with investors being charged a 94$ membership fee. Importantly, there was no underlying business and investors would receive payouts based uniquely on the number of individuals they brought into the pyramid scheme (Springer, 2020). Finally, Universo Foneclub was active in 2006 in the prepaid phonecards sector. Those items were sold at a loss, meaning that no profits were earned on the sales, and, payouts, were distributed solely through recruitment fees. The targeted group was related to Brazilian and evangelical Christian communities (Springer, 2020).

The inner dynamics of Ponzi schemes are studied in a laboratory setting by Sadiraj and Schram (2001). The authors, in fact, develop a rather complex fraud with the following characteristics: there is an Investment Fund (IF) with an initial sum of money X which will affect real returns. Individuals are either informed (i.e., they know the realization X) or uninformed (i.e., they do not know the realization X) and must decide whether to invest a fixed sum of money into the IF. Investors are paid a return financed entirely by the initial sum of money X available in the fund. At each time step, individuals can decide whether to withdraw their entire sum of money or not. In every round bankruptcy can take place when the withdrawals are greater than the money in the IF (Sadiraj and Schram, 2001).

Through survival analysis, the authors study investors' behaviors. Overall, in all the experiments, the IF bankrupt with investors keeping their money invested despite knowing that the interest paid was financed by the original investments made in the fund (Sadiraj and Schram, 2001). Importantly, the number of informed investors significantly impacts withdrawal decisions. Investors, in fact, keep their money in the fund for longer periods when only 1 informed investor is present (Sadiraj and Schram, 2001). In all experiments, herding is observed, implying that uninformed investors are significantly more

likely to withdraw if some informed did so. The phenomenon is identified to be weaker when high-interest rates are offered (Sadiraj and Schram, 2001). Relating to interest rates, the authors detect that they are crucial in determining participation rates. When interest rates are higher, there is both higher and longer participation in the IF (Sadiraj and Schram, 2001).

Recently, with the development of cryptocurrencies, Ponzi schemes are shifting to digital versions. Springer (2020) documents the advantages for perpetrators of this trend. In particular, digital currencies allow investors to partake in fraud regardless of their geographical location, making it more difficult to investigate crime and enforce one nation's laws (Springer, 2020). Furthermore, cryptocurrencies lead to complete anonymity for both participants and perpetrators, worsening the ability to impose fines and legal repercussions. Finally, the funds cannot be seized and the possibility of transferring money around the world electronically leads digital coins to be the perfect breeding ground for Ponzi schemes (Springer, 2020).

Before introducing blockchain-based Ponzi schemes, it is essential to understand the key components of this technology.

The next chapter will delve into the key characteristics of digital currencies.

2.2 Blockchain and Smart Contracts

2.2.1 Blockchain definition, key components and inner workings

According to the Financial Action Task Force (FAFT, 2014), a virtual currency is a digital representation of value that can be digitally traded and serves three distinct functions: it acts as a medium of exchange, a unit of account and a store of value. Yet, those currencies do not have legal tender status and are not issued or guaranteed by a jurisdiction (Kethineni and Cao, 2020). Through the development of blockchain technology, and the related guarantees achieved via cryptography, virtual currencies have gained in popularity. Specifically, in 2023, around 23,000 cryptocurrencies were around, leading to a total market capitalization of 1.1 trillion USD (Hicks, 2023). In order for blockchain-based digital currencies to act as a medium of exchange, different key components must be in place. More precisely, cryptocurrencies can properly function thanks to: Distributed Ledger Technology, Hash functions, Cryptographic Nonces, Asymmetric-Key Encryption, Blocks and Consensus mechanisms.

Distributed Ledger Technology (DLT) refers to the system that enables storing and updating a distributed ledger in a decentralized manner (Belotti et al., 2019). Three distinct components characterize DLTs, namely: a data model recording the current ledger state, a mechanism to alter and update the ledger via transactions and a protocol to guarantee consensus on the transactions among the different participants (Belotti et al., 2019). Thanks to those DLTs characteristics, it is then possible to introduce blockchain technology as a Peer-to-Peer (P2P) DLT, structured as a chain of blocks of transactions characterized by consensus among participants (Belotti et al., 2019).

Yaga et al. (2019) identify several advantages of P2P distributed DLTs over centrally owned ledgers:

1) Centrally owned ledgers may be lost or destroyed. A blockchain network on the other hand, thanks to its distributed design guarantees the presence of different backups distributed across the participating nodes.
2) Centrally owned ledgers may be on a homogeneous network, leading to a low system resiliency to cyber-attacks. A blockchain network instead is heterogeneous, making it much less likely to be a victim of those attacks.
3) Centrally owned ledgers are generally located in specific geographical locations, exposing them to power outage risks. Blockchains nodes are instead distributed across the globe, making them resilient to those risks.
4) Centrally owned ledgers are not transparent, whereas by design, through broadcasting mechanisms required for the validation of transactions, blockchain systems are.
5) Users must trust that centrally owned ledgers include all valid transactions. Blockchains, through distributed ledgers, do not require this trust.
6) Centrally owned ledgers may be insecure, and users must trust that critical security patches are continuously installed. Blockchain networks, by being distributed, provide no central point of attack.

Regarding this last point, by exploiting cryptography, secure data transmission mechanisms are enabled and allow immutability across the chain. There are two main applications of cryptographic technology in blockchains: Hash functions and Asymmetric-Key encryption.

The first refers to a function, which when fed data, calculates a relatively unique output of fixed size that distinctively represents the data fed to the function. Therefore, even the slightest change in the input will lead to a

completely different hash (Yaga et al., 2019). The same authors identify that hash functions are able to offer three important security properties:

- They are preimage resistant, meaning that it is computationally impossible to derive the input from the hash.
- They are collision resistant, so that it will be impossible to find a second input producing the same hash.
- Provided an input, the hash generated will always be the same, meaning that the hashing function is deterministic.

In most blockchain technologies, the hashing function used is the Secure Hash Algorithm (SHA), which can generate 2^{256} possible outputs (Yaga et al., 2019). The SHA algorithm is also applied in the validation of blocks in Proof of Work based blockchains.

Asymmetric-Key encryption refers to the system that allows to trust transactions coming from other peers in the blockchain. Specifically, asymmetric-key encryption relates to the usage of two pairs of keys: a public one and a private one (Yaga et al., 2019). In particular, the private key is used to encrypt a transaction whereas the public key is the only mechanism to decrypt it. Since the public key is openly distributed, encrypting the transaction with the private key allows to prove that the signer of the transaction has access to the private key and to guarantee the safety of the transaction. Signing transactions is the fundamental feature that guarantees integrity, authenticity, and non-repudiation of blockchains (Belotti et al., 2019).

For a transaction in the blockchain to take place, Belotti et al. (2019) document the four critical steps required:

1) Creation: The sender needs to define the origin and destination of the digital asset that is being transferred. The sender will need to sign the transaction via private key.

2) Propagation: The transaction is broadcasted to all the peers of the blockchain.
3) Validation: Validating nodes must verify and approve the transaction based on the specific consensus mechanism available. Verification of the transaction is performed by exploiting the public key of the sender and decrypting the transfer of the digital asset.
4) Propagation: The validated transaction is propagated to the entire network, updating all the DLTs.

Overall, a transaction will involve different parties. Precisely, the data-sender, that is the node sending a transaction and digitally signing it, the data-receiver which includes all the nodes receiving the transaction and that can verify its authenticity, validating nodes, the peers responsible for the application of the consensus algorithm (Yaga et al., 2019).
Concerning the consensus algorithm, different implementations are currently available across blockchains.

Poof of work (PoW) was the first consensus algorithm introduced and led to the creation of Bitcoin. It requires solving a computationally intensive puzzle and the solution is the proof that the block has been validated. More precisely, the computational work consists in identifying the cryptographic nonce that will generate, combined with the block of transactions, an hash with a specified number of leading zeros (Yaga et al., 2019). In the case of Bitcoin, the difficulty of the computational puzzle is adjusted so that new blocks are published every 10 minutes. The miner that is capable of solving this computational puzzle is rewarded with Bitcoins (Belotti et al., 2019). Yet, Bitcoin mining has important negative implications in terms of CO2 emissions. Salam (2023) documents that in 2022 such activity produced 65.4 megatons of CO2, which is more than the total emissions of Greece (56.6 megatons).

More ecofriendly solutions have been later proposed, among which the most important one is the Proof of Stake model (PoS). Such a system is based on stakes, meaning that the higher is the stake of an individual, the more likely they will want the system to succeed. Yaga et al. (2019) document that staked users can be selected according to three different selection criteria:

1) Random choice: the blockchain randomly selects users based on their stake in the system. For instance, a user with a 40% stake of the entire blockchain will have a 40% probability of being selected for validation.
2) Multi-round voting system: the blockchain will select several staked individuals to create proposed blocks. Then, different voting rounds may take place in order to validate the block.
3) Coin age: the staked crypto has an age property, meaning that only after a specified length of time the owner can use it as a stake to publish the next block. Once this operation takes place, the age of the staked crypto is reset.

Once staked individuals are selected, they are then charged with the task of validating a block of transactions. If the block of validated transactions is not adequate and contains fraudulent transactions, the staked individuals will lose their stake; otherwise, they will be rewarded with cryptocurrency (Yaga et al., 2019). Variations of the PoS model exist to ensure that the voting power is not centralized to the rich (Belotti et al., 2019).

Blockchains can ensure participation of different peers according to two modalities. Permissionless blockchains are public and open access, entailing that any individual can join the network and participate in the consensus process. Each node has a specific pseudonym, thus possibly allowing malicious nodes to enter the chain and act within the network (Belotti et al., 2019). Contrariwise, in permissioned chains, participants have restrictions on

validating rights, or also on access rights. Those two characteristics imply greater system security and privacy of transactions (Belotti et al., 2019).
Permissioned blockchains have two additional consensus systems at their disposal: Proof of Authority and Proof of Elapsed Time.
The first entails that publishing nodes have connected their real word identities to their blockchain address. Overall, this implies that publishing nodes are putting at stake their identity and reputation when validating and publishing new blocks (Yaga et al., 2019).
The second model revolves around the presence of a random time hindering nodes to validate blocks. Specifically, each publishing node must request a wait time from a trusted hardware after which block validation is enabled. Once a node publishes a new block, all the other nodes are alerted, their idle time will reset to zero and the entire process restarts (Yaga et al., 2019).

When performing block validation, it can happen that multiple blocks will be published at the same time or that malicious users may not broadcast a fraudulent transaction to the entire chain. In those instances, the official blockchain will be the longest one available as it has the largest computational work to guarantee its accuracy. Further, whenever a transaction is performed, it is usually accepted only after several blocks later (Yaga et al., 2019).

Currently, there is a debate as to whether cryptocurrencies should be government backed. At its conception Bitcoin had the goal of avoiding all possible intermediaries and offering a direct P2P system based on PoW. Proponents of government backed cryptocurrencies argue that it would be a new way to introduce financial control, to break-away from the USD as fiat currency and to maintain central banks monopoly (Kethineni and Cao, 2020). Yet, others argue that any form of state-backed crypto is pointless and against the key DLT characteristics of decentralization, transparency, and immutability (Kethineni and Cao, 2020).

Radical differences can be found across jurisdictions in different states as certain countries are currently transitioning towards cryptocurrencies whilst others have bans on them.

Countries favoring crypto are Venezuela, the Marshall Islands and Sweden. More precisely, Venezuela introduced, in 2018, the petro a state-backed cryptocurrency alternative to the almost worthless bolivar. The value of this digital currency is linked to the price of one barrel of Venezuelan oil, meaning that it is not affected by the supply and demand of currency (Kethineni and Cao, 2020). Similarly, the Marshall Islands have introduced the Sovereign as its legal tender and official country cryptocurrency and Sweden is looking to introduce the e-krone (Kethineni and Cao, 2020). For Venezuela and the Marshall Islands introducing state-backed cryptocurrencies is a way to become more independent from the USD and thus regain sovereignty (Kethineni and Cao, 2020). The importance of cryptocurrencies has also been addressed in 2018 during the G20 meeting where different European countries recognized the need to develop adequate regulation. The main challenge facing state-backed digital currencies is whether events such as tax fraud, Ponzi schemes, money laundering and other criminal activities can be prevented, thus improving cryptocurrencies' security (Kethineni and Cao, 2020).

On the other hand, countries with bans on cryptocurrencies are: South Korea, China, Thailand and Russia. Precisely, in South Korea regulators planned to prohibit cryptocurrencies that provide anonymity to its users. Similarly, in China all virtual currency transactions have been stopped and Initial Coin Offerings have been banned. The Bank of Thailand ruled Bitcoin illegal and, likewise, Russia is looking to ban Bitcoin as well (Kethineni and Cao, 2020).
The development of cryptocurrencies has led to major improvements in terms of services available on the blockchain. More precisely, with the development

of Ethereum, smart contracts have been created. The following section will provide an in-depth focus on those contracts.

2.2.2 Ethereum and Smart Contracts

Ethereum is a blockchain specifically designed for building decentralized applications running smart contracts (Wood, 2014). The programming language used in this service is Solidity, which is Turing-complete. The execution of smart contracts takes place on the Ethereum virtual machine and is performed by miners, who are rewarded Ether for their services. Transactions are validated based on the PoS model and, unlike Bitcoin, Ethereum is capable of achieving around 20 transactions per second (Wood, 2014).

The major improvement over Bitcoin is the usage of Solidity as a programming language. In fact, by being Turing complete, complex smart contracts can be implemented. Smart contracts were first conceptualized in 1994 by Nick Szabo, who defined them as computerized protocols that execute the terms of a contract whilst minimizing the need for trusted enforcers (Yaga et al., 2019). In practice, smart contracts are a collection of code and data deployed on the blockchain. To guarantee that smart contracts will not run indefinitely, due to the Turing complete nature of Solidity, the concept of gas is introduced. More precisely, each line of code in smart contracts consumes gas and an overall gas limit is provided for each smart contract. The set of operations performable by those instruments is quite varied as they can: perform calculations, store information, and automatically send funds (Zheng et al., 2020). Smart contracts additionally provide several advantages over traditional contracts. More precisely, they are immutable, meaning that they cannot be modified once they are issued and all the related transactions are public, traceable, and auditable (Zheng et al., 2020). Furthermore, no centralization is required for their execution, ultimately minimizing administrative costs, and leading to greater efficiency levels (Zheng et al., 2020).

Yet, smart contracts are plagued by challenges in their creation, deployment, and execution.

Relating to the contract creation process, two areas of difficulties relate to smart contracts readability and functional issues. The first connects to the fact that in order to deploy a smart contract, the contract developer needs to first draft in Solidity the code and then compile it so that it can be read by the EVM. Nonetheless, compiling the source code into bytecode leads to human-unreadable code, thus reducing the transparency of smart contracts (Zheng et al., 2020). This problem is exacerbated by the fact that only around 23% of the smart contracts have released their source code (Zheng et al., 2020). Recent advancements to mitigate this problem have been made by Huang et al. (2023) who developed BCGen, a bytecode comment generation model based on the transformer architecture. Specifically, the algorithm first starts by generating the control flow graph from the code and, subsequently, features are extracted that can help in the generation of natural language comments (Huang et al., 2023).

The second concerns problems in the smart contract logic and code implementation. Specifically, many contracts suffer from reentrancy problems which can be exploited by malicious users to steal digital currency (Zheng et al., 2020). Others suffer from code under-optimization. In this regard, Chen et al. (2017) documented that more than 90% of smart contracts suffer from costly gas patterns.

The main challenges that can arise in the contract deployment phase concern the contract correctness and the dynamic control flow.

The former problem instantiates when, due to the complexity of modelling smart contracts, bugs that hinder the correct contract execution are present (Zheng et al., 2020). Two main solutions have been proposed to mitigate those scenarios. Luu et al. (2016) proposed Oyente, a tool capable of identifying

security bugs relating to exceptions and timestamp-dependent problems; whereas Grech et al. (2019) developed Gigahorse, an algorithm that can decompile bytecodes and identify the implicit data and control flow dependencies relating to the contract execution.

The latter difficulty refers to the mutability of the control flow of smart contracts arising from interactions with other contracts, which can make it difficult to predict the contract behavior (Zheng et al., 2020). Charlier et al. (2017) and Nikolić et al. (2018) proposed two novel solutions. The first combined stochastic processes and graphs representing the current transactions to identify and predict future contract behaviors and interactions. The second, instead, focus on contracts locking/leaking funds and that can be terminated by anyone. They provide Maian, a system that can identify those contract vulnerabilities.

Finally, challenges can arise during the smart contract execution. Specifically, often times due to the forking of blockchain branches, the order of the transactions can be misaligned. Mavridou and Laszka (2018) developed a system based on transaction numbers that guarantees the correct ordering of events.

Overall, despite the vast improvements that blockchain technology can provide, the anonymity offered makes it a target for cybercrime activities. The next section will further investigate general characteristics of recurring crypto-related frauds.

2.2.3 Cryptocurrencies cybercrime

Cryptocurrency cybercrime has already reached dramatically high levels in recent years. In particular, in 2018, according to Rob Wright, the head of Europol, around 5 billion USD have been crypto laundered just in the European continent (The Economist, 2018). Similarly, CipherTrace identified that the

value of hacks, thefts and scams doubled from 2018 to 2019 and reached 230 times the 2017 value (Badawi and Jourdan, 2020). Specifically, more than 4.52 billion USD has been stolen by cybercriminals on cryptocurrency exchanges. Likewise, FTC discovered that in 2021, the value of frauds and scams in cryptocurrencies reached 14 billion USD, with median losses for crypto users of around 1,900 USD (Kutera, 2022).

The major forms of cyberattacks identified are: High Yield Investment Programs, which are Ponzi schemes promising exceptionally high returns, Money Laundering, enabling criminals to disguise the source of their illegal profits, Ransomware, freezing accounts and demanding a ransom to unblock them, and Pump and Dump in which groups of individuals create hype around low priced cryptocurrencies, dumping them as soon as they rise in price (Badawi and Jourdan, 2020).

Nizzoli et al. (2020) and Vasek and Moore (2019) characterize how crypto frauds operate and how individuals are drawn to them.
The first conducted a study on the manipulation effects of social media by analyzing tweets, discord and telegram messages discussing cryptocurrencies. Specifically, the authors apply topic modelling to the messages extracted from the social networks and discover that in Telegram, important recurring topics relate to Ponzi schemes and Pump and Dump actions. Similarly, tweets mentioning cryptocurrencies often times provide invite links to Ponzi frauds. Concerning Twitter, Nizzoli et al. (2020) identify the emergence of new bots that can closely mimic human behavior and that are key actors in spreading Ponzi schemes invite links. The last result identified is that, starting from generic cryptocurrency messages and following the provided links to join further discussions, 56.5% of those are involved in deception and fraud.
The second analyzed scammer’s behavior on bitcointalk to determine the key characteristics of long-lasting Ponzi schemes. More precisely, the website

provides different subforums where crypto-enthusiasts can discuss crypto-related events. Vasek and Moore (2019) analyze messages taking place in Scam accusations, Gambling games and Gambling investment games. Overall, the authors detect that the greater the involvement of the scammer in terms of messages sent, the longer the life of the fraud. In fact, when scammers post half of the total posts in the thread, the average fraud duration extends from one week to three weeks. Similarly, each additional daily comment from a victim leads to longer lifespans of the frauds (Vasek and Moore, 2019).

The authors, by exploiting the activity history of different users, are capable of understanding the characteristics of crypto frauds preys. Particularly, Ponzi victims are overrepresented in the following threads: beginners, economy, politics and society, and meta (Vasek and Moore, 2019).

The rapid advent of crypto and the related belief that digital currencies can quickly generate profits seems to be the most common narrative exploited by cryptocurrency-based high yield investment programs (Vasek and Moore, 2019).

Finally, Ponzi schemes operating on virtual currencies are gaining the attention of law enforcement authorities (Kethineni and Cao, 2020). In fact, in 2014 the U.S. District Court in Texas prosecuted BTC Savings and Trust, a Ponzi scheme promising weekly returns of around 7% and exploiting Bitcoin as the alleged source of the profits (Kethineni and Cao, 2020). Likewise, in India, police investigation discovered Bitcoin based Ponzi schemes worth 300 billion USD being conducted by the following three companies: GB Minors, Gain BTC and GB2 (Kethineni and Cao, 2020).

The commonality across those Ponzi schemes operating on Bitcoin is that real companies were linked to the frauds, making it thus possible to identify and prosecute the associated fraudsters. With the advent and rapid expansion of Ethereum, the possibility of tracking down the individuals behind Ponzi schemes is being drastically reduced. In fact, by exploiting smart contracts,

perpetrators can lure investors and make profits whilst keeping their anonymity. These features, as documented by Vasek and Moore (2019), are likely to be related to a massive growth in the magnitude of those frauds.

The next chapter will provide a focus on Ethereum based-Ponzi schemes that are exploiting smart contracts, highlight the current state of the art in their detection and motivate the empirical work analysis performed.

2.3 Smart-Ponzi schemes overview and machine learning modelling

2.3.1 Definition and general characteristics of smart Ponzi schemes

Bartoletti et al. (2020) through an in-depth analysis of the Ponzi scheme definition provided by the SEC, specify four characteristics jointly required for the identification of smart Ponzi schemes. Specifically, the contract must distribute money to its investors (R1). This condition alone is inadequate for detecting Ponzi schemes as also gambling games, bonds, insurances and other types of contracts satisfy it (Bartoletti et al., 2020). Next, the money distributed uniquely derives from investors, thus excluding contracts where Ethereum is distributed from external sources (R2) (Bartoletti et al., 2020). Thirdly, investors make a profit if and only if new investors continue funding the smart contract (R3). Importantly, gambling games and lotteries are not excluded by those three requirements as its users make money through the investments of others (Bartoletti et al., 2020). Finally, the risk of being exposed to losses grows with the time one joins the scheme (R4). Combining all those requirements Ponzi schemes can be uniquely identified (Bartoletti et al., 2020).

Linking those requirements with the original SEC definition yields that: R1 and R2 capture that Ponzi schemes "involve the payment of returns to existing investors from funds contributed by new investors", R3 relates to the requirement of Ponzi schemes of a constant flow of money from new investors and R4 corresponds to the inevitable collapse of those frauds when no new joiners are present (SEC, n.d.; Bartoletti et al., 2020). Importantly, the latter requirement excludes from the set of smart Ponzis contracts that are often accused of being Ponzis, such as: CryptoKitties and Fomo3D. The first is a game, implemented through the ERC-721 token, in which users can breed and trade virtual cats. However, since a new joiner can breed a rare feline and profit from its sale, R4 is violated (Bartoletti et al., 2020). Next, Fomo3D is a timed lottery game in which the last buyer collects the jackpot. Each time a new buying command is sent to the contract, the lottery expiry time gets

extended, the jackpot increases, and the early joiners receive a share of the buy-in amount, a Ponzi-like feature. Nonetheless, since a late joiner can win the jackpot, R4 is inevitably not satisfied (Bartoletti et al., 2020).

Generally, smart Ponzi schemes have a set of distinctive features that can help in distinguishing them from other smart contracts. Firstly, the contracts send Ethereum to accounts that have previously joined the scheme (Chen et al., 2018). This is typically accomplished through the storage of participants' information and the investments they have made (Chen et al., 2021). Specifically, this is achieved through "investing" functions, which enable transfers of Ethereum to the smart contract whilst recording the investor's information (Chen et al., 2021b). Secondly, some accounts receive more payments than others (Chen et al., 2018). Those operations are performed through "reward" actions in which via specific CALL functions, participants receive a bonus proportional to the funds available, the time of joining and the individual investment made (Chen et al., 2021b). Reward actions are activated only when the available funds meet a specified threshold or when a set time is elapsed (Chen et al., 2021b). Such pattern grants that the contract balance will be typically low, so as to maintain an image of fast and high returns (Chen et al., 2018).

The life cycle of those frauds is characterized by three distinct phases: Bootstrap, Hyper-operation and Collapse. The former relates to the naissance of the scheme, during which a limited number of investors join and wait for the returns (Song and Kong, 2022). The second is associated with a growth in popularity of the scheme and, hence, of more investors joining believing that they could achieve high returns (Song and Kong, 2022). Finally, the latter materializes once investors observe that they cannot withdraw, losing confidence in the scheme and with it their investments. Aware participants, to reduce their losses, will desperately attempt to invite investors, making it more

difficult to identify the fraudster. Before this stage, the latter generally leaves the contract (Song and Kong, 2022).

Exploiting Ethereum to introduce Ponzi smart contracts has benefits for the perpetrators. Precisely, they can keep anonymity, entailing protection against incrimination (Song and Kong, 2022). Further, the public availability of the source code, the related transactions and accounts, and the automatic nature of its execution provide users with reliability and credibility. This is worsened by the apparent indistinguishability of the fraudulent contracts operating code compared to that of safe ones (Song and Kong, 2022). Finally, once smart contracts are deployed, they become immutable, making it impossible to modify and stop their execution. Through this feature, participants will acquire confidence that the returns will not halt (Song and Kong, 2022).

Immutability itself can oftentimes cause harm to investors. In fact, the source code can contain intentional bugs aimed at increasing the founders' profits. Luu et al. (2016) identified that the most common vulnerability is associated with the send function. Specifically, contracts should check for errors in send transactions. Bartoletti et al. (2020) note that in case of errors during the investment process, if there is not an explicit error checking function, the money invested will remain in the contract while the user information will not be recorded, thus excluding her from the scheme and inappropriately stealing her money. The majority of smart Ponzis do not explicitly check for this source of errors (Bartoletti et al., 2020). Further, different schemes have a minimum amount of investment required in order to join. Yet, a plethora of them do not have return functions if the investor does not meet the minimum investment requirement, entailing that the user would lose the money and not be part of the scheme (Bartoletti et al., 2020). Additionally, some contracts entail a fee for each transaction made towards the scheme which will directly benefit the founder. Nevertheless, wrong implementations of the fee calculating function are often detected (Bartoletti et al., 2020). To exemplify, Bartoletti et al.

(2020) detected that PiggyBank computes fees in the following way: fee += investment * fee(%) instead of fee = investment*fee(%), leading to fees growing for each deposit. Other observed bugs relate to incorrect implementations of the reward functions, which will reward multiple times the contract founder and only once the other participants (Bartoletti et al., 2020). Finally, certain smart contracts will have a set of functions empowering the founder to increase the fees, change the interest rate and terminate the contract (Bartoletti et al., 2020).

To attract investors, Ponzi smart contracts exploit preposterous return promises. An instance of this is given by Rubixi, a Ponzi scheme that was advertised in the following way:

"Hello! My name is Rubixi! I’m new & verified pyramid smart contract on the Ethereum blockchain. When you send me 1 ether, I will multiply the amount and send it back to your address when the balance is sufficient. My multiplier factor is dynamic (min. x1.2 max. x3), thus my payouts are accelerated and guaranteed for months to come" (Chen et al., 2019).

An in-depth analysis of the contract transactions revealed that 22 out of 112 participants made a profit, with the creator and another participant taking more than 40% of the total profits (Chen et al., 2019). Moreover, many participants did not receive a single payback and the founder had more receiving transactions than investment ones (Chen et al., 2019).

Currently, a plethora of implementations of smart Ponzi schemes are affecting Ethereum, thus the next section will investigate and taxonomize those frauds based on their redistribution logic and dynamics.

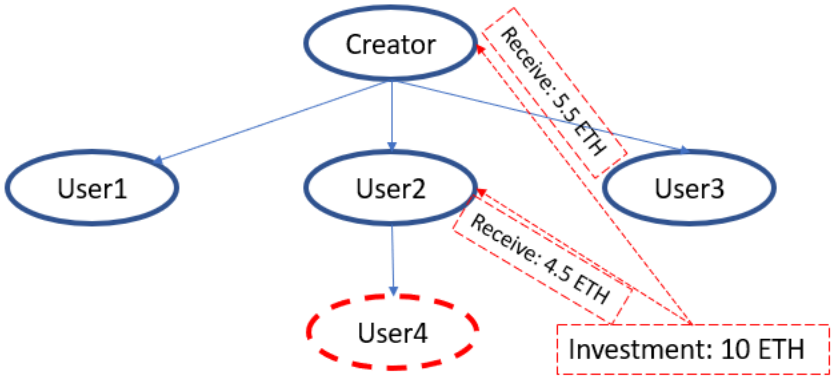
2.3.2 Smart Ponzi schemes taxonomy and redistribution patterns

Overall, four distinct implementations of Ponzi schemes have been discovered on the Ethereum blockchain; those are: Tree-shaped, Chain-shaped, Waterfall and Handover schemes (Chen et al., 2021b).

The first exploits trees as data structure by requiring each new joiner to specify the inviter, who will become her parent node. Specifically, if a user does not specify an inviter, the contract creator will be the parent node (Bartoletti et al., 2020). Whenever a new individual joins the scheme, the money will be redistributed following the tree structure, halving the sums of money at each root, thus entailing that: the contract creator will make a profit whenever a new joiner enters the scheme as he is the father node, and the profit of a specific node will depend on the number of investors attracted (Bartoletti et al., 2020). Furthermore, investors can be excluded from the scheme if: they do not meet the minimum entry toll, they specify an inviter that does not exist or they are already part of the fraud (Chen et al., 2021b). In this scheme format, there is no promised return as the profits will depend uniquely on the ability of each investor to attract new joiners.

The figure below summarizes the redistribution nature of Tree-shaped schemes. Importantly, as User4 joins the contract and indicates User2 as its inviter, User4 investment is immediately redistributed with the following logic. First, the 10% fee is subtracted from the investment and it is sent to the contract creator. Next, half of the remaining sum of money is received by User2 and the remaining sum gets sent to the contract creator.

Figure 2: *Tree-shaped scheme with a 10% transaction fee*

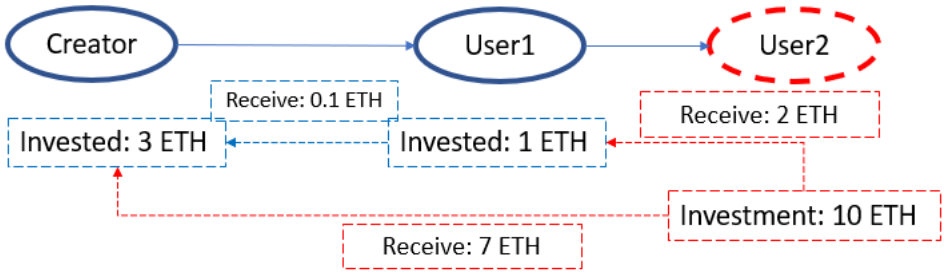


Chain-shaped schemes are a special type of tree-shaped ones as, in this case, each node will have exactly one child node. Generally, those contracts have a promised multiplier that will be applied to each investor (Chen et al., 2021b). Particularly, the earlier an investor joins the scheme, the greater will be the likelihood of receiving a payout. This is the case as the scheme starts paying back users one at a time in order of arrival and liquidates them in full. Once there is enough liquidity in the scheme and all the previous participants have been paid, the user will receive a payment from the contract and will be excluded from the scheme (Bartoletti et al., 2020). Those contracts are commonly characterized by a minimum entry toll and by fees that will be directly paid to the contract creator once a new investment is made in the scheme. Each user, provided that the scheme flourishes, will know exactly the payouts he will receive as those will be proportional to the amount invested and will depend on the stated multiplier (Chen et al., 2021b).

The figure below summarizes the redistribution nature of Chain-shaped schemes. Particularly, provided the 2X multiplier, for the contract creator to be liquidated in full, at least 6 ETH, after the fees have been collected, must be deposited in the scheme. Once User1 joins, the creator receives a 10% fee, and the remaining ETH is left in the contract as it is not enough to repay him in full. As User2 deposits 10 ETH, the contract creator receives the transaction fee and is liquidated in full, meaning that later he will receive uniquely the

transaction fees. Similarly, since User2 has deposited enough ETH, User1 gets liquidated in full and is excluded from the contract. In order for User2 to be fully liquidated, new individuals must join and the contract balance must be of at least 20 ETH.

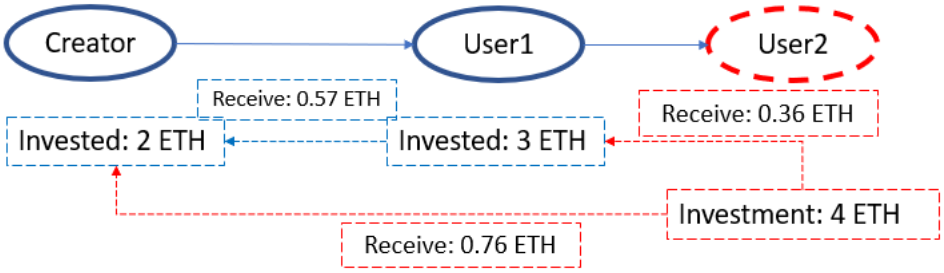
Figure 3: *Chain-shaped scheme with 2x multiplier and 10% transaction fee*



Waterfall schemes exploit a similar user ordering to chain-shaped schemes but exploit a different money distribution pattern. Each new investment is spread to the previous joiners on a first come first serve basis and always starting from the beginning of the chain. More precisely, a stated percentage of the investment will be paid to each of the earlier nodes and, thus, late joiners will typically not receive any payouts from the scheme (Bartoletti et al., 2020). To ensure that all users will receive a payout, the investments of new users will grow proportionally to the number of users (Bartoletti et al., 2020).

The figure below exemplifies a Waterfall scheme. In particular, every time a new investment is brought to the contract, the early participants receive a fee which is determined by multiplying the investment by a stated percentage. In fact, the contract creator receives the 10% transaction fee and the 10% payout from both User1 and User2. Similarly, User1 receives 10% of the investment made by User2. This procedure will continue until users receive 2x of their original investment.

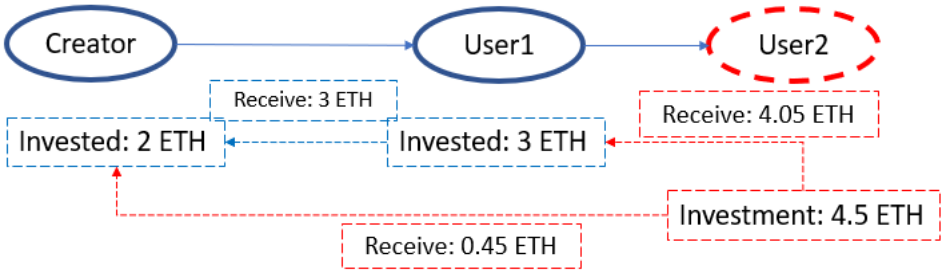
Figure 4: *Waterfall scheme with 10% transaction fee, 10% payouts and 2X multiplier*



Finally, Handover schemes are a special instance of chain-shaped contracts, characterized by an increasing entry toll whenever a new joiner enters the scheme (Bartoletti et al., 2020). In particular, whenever a new individual enters the scheme, the previous investor is immediately liquidated in full, thanks to the increasing toll at each step (Bartoletti et al., 2020). This implies that at each step there will be only one investor receiving the money and that as the scheme grows, it will become more difficult to be paid due to the high entry toll entailed (Bartoletti et al., 2020).

The figure below illustrates a Handover scheme. Whenever a new investment is made, 10% is sent to the contract creator and the remaining amount is distributed to the previous node. In order to guarantee a 1.35x multiplier, the entry toll must continuously rise to guarantee a full liquidation of the previous node.

Figure 5: *Handover scheme with 10% transaction fee and 1.35x multiplier*



Overall, given the diverse nature of those schemes, it is necessary to develop solutions for the identification of fraudulent contracts.

The next section will delve deeper into the motivations and already developed approaches in the field.

2.3.3 Detecting Smart Ponzi schemes

Detecting smart Ponzi schemes is a key requirement for the global improvement of blockchain technology. In particular, smart contracts are core components of blockchain-based systems and improving detection mechanisms can provide greater protection to investors (Hu et al., 2021). To this end, Yu et al. (2021) document that at least 725 million USD has been lost due to Ponzi schemes and fraud ICOs. Further, around 10% of the whole Ethereum smart contracts are Ponzi frauds, implying that the financial security of crypto-based technologies needs important improvements (Yu et al., 2021). Nonetheless, detecting Ponzi schemes is still an open issue due to lack of investors knowledge and regulation. More precisely, because of the great hype surrounding blockchain technology, many investors are still finding it complicated to understand the effects of smart contracts, making them targeted prey for scams (Chen et al., 2018). Similarly, national authorities and regulators are also inexperienced about smart contracts, leaving this field of the financial system in a gray area (Chen et al., 2018).
Therefore, developing approaches to detect Ponzi smart contracts is both urgent and critical to make blockchain markets safer for investors (Chen et al., 2018).

The state-of-the-art detection systems operate mainly on three different types of data: the source code, the bytecode and the transaction records.

Source code-based approaches are capable of detecting Ponzi schemes by analyzing the reward distributions entailed by the contract source code. In this regard, a key contribution is made by Chen et al. (2021a), who devised MTCformer (Multi-channel Text Convolutional Neural Networks and

Transformer) an algorithm capable of detecting Ponzi schemes from their source code by exploiting a multi-channel TextCNN and a Transformer to extract structural and semantic features. Overall, through this procedure, they are capable of beating CodeBert F1 score by 12 percentage points, reaching an F1 score of 0.89. Yet, although the classification through the source code showcases excellent performance, the method is not generalizable to new contracts as source codes are not immediately available and sometimes are hidden.

Bytecode based methods involve the bulk of the literature of Ponzi scheme detection. Bytecodes refer to the series of instructions that need to be passed to the Ethereum Virtual Machine in order to implement the logic of the smart contract (Chen et al., 2018). Bytecodes, however, are not human readable and specific disassemblers are available to transform the bytes into operation codes, characterized by *opcodes*, representing function names, and *operands*. Contrary to the source code, the bytecode, by definition, must always be publicly available as it is necessary to deploy the contract on Ethereum (Chen et al., 2018). In this regard, important contributions have been made by: Fan et al. (2020), Shen et al. (2021), Chen et al. (2021b) and . The first developed a boosting algorithm called PonziTech to achieve classification. Specifically, after extracting the opcode, they generate a feature vector summarizing each smart contract and perform data augmentation via the SMOTE algorithm. Finally, the authors achieve a 0.98 F1-Score and identify key operation codes most commonly found in Ponzi smart contracts (i.e., SLOAD and SSTORE, used to create and store associative maps) (Fan et al., 2020). Such result, however, needs to be taken with care as data augmentation is performed prior to splitting the data between train and test, possibly causing overfitting and low generalization power. Shen et al. (2021) implement an anomaly detection algorithm, IForest to classify smart contracts. Specifically, the authors extract the opcodes and reduce the noise via PCA prior to fitting the model, achieving

an F1-score of 0.88. Fan et al. (2021), have proposed AI-SPSD, a boosting based algorithm that takes in input the opcode segmented via n-grams, ultimately achieving a 0.96 F1 score.
Finally, Chen et al. (2021b) have proposed SADPonzi, an algorithm capable of achieving extremely high detection power through an analysis of the calling relationships between the opcode functions, reaching an F1 score of 0.96.

Importantly, opcodes-based detection algorithms can detect Ponzi schemes as soon as the smart contract is deployed. Yet, as highlighted by Chen et al. (2021b) the current methods will face difficulties in detecting: Ponzi smart contracts that combine multiple schemes, new Ponzi frauds exploiting the current available opcodes algorithms to create opcodes undetectable frauds, schemes exploiting encryption for each performed operation.
Therefore, although opcodes-based algorithms can achieve high detection power, they should be interpreted as an early warning system that can be strengthened by a second stage detection via transaction-based systems. Specifically, since the transaction history is always available and will always reward the contract founders, the latter algorithms should provide robustness to evolving smart Ponzi schemes.

Advances in transaction-based detection algorithms have been mostly made by: Chen et al. (2018), Chen et al. (2019), Galletta and Pinelli (2023), Yu et al. (2021) and Jin et al. (2022). The first developed a boosting classifier, exploiting 7 different features, namely: the proportion of receivers having invested before payment, the smart contract balance, the number of investments and payments, the proportion of investors having received one payment, the number of payments from the smart contract and the difference between number of investments and payments (Chen et al., 2018). Overall, the authors achieve an F1 score of 0.44, indicating a lack of discriminatory power of the developed model. The researchers tried to improve on their

previous model by exploiting a random forest classifier and increasing the number of features to 13. More precisely, the added features refer to: the mean, standard deviation and skewness of the distribution of the difference between the number of investments and payments and of their value. Yet, the results obtained are worse than the previous, reaching an overall F1 score of 0.30 (Chen et al., 2019). Greater detection is achieved by Galletta and Pinelli (2023), who, by constructing 27 features and utilizing a gradient boosting model, reach an F1 score of 0.62.

Yu et al. (2021) exploit a different suite of algorithms for the detection of smart Ponzi schemes. Specifically, they analyze blockchain data through the lenses of network analysis, treating smart contracts and individuals as nodes and the transactions between them as edges. In particular, the authors extract the 1^{st}-order neighbors of each smart contract and analyze the topological structure formed by the transactions, performing node classification for the detection of Ponzi schemes. Overall, an F1 score of around 0.79 is reached via a random forest classifier and 0.89 via graph convolutional neural networks (Yu et al., 2021).

Nevertheless, the presented papers exploit a hindsight perspective to classify smart contracts, meaning that all the Ponzi schemes transactions are used. This leads to a lack of early detection capabilities and means that Ponzi schemes are identifiable only after they have harmed a plethora of investors. Therefore, detection systems that are capable of accurately identifying Ponzi schemes with few transactions are essential to increase the blockchain's safety (Jin et al., 2022).

In this regard, Jin et al. (2022) developed an early warning system reaching an F1 score of 0.91. More precisely, exploiting a network perspective, the micro transaction graphs for each contract are extracted and the data is divided into different steps. Specifically, the authors create 10 different steps, each made of 10 transactions, leading to the following: at step 2, the classification

algorithm will exploit the first 20 transactions of the smart contracts to distinguish between Ponzis and non-Ponzis. To enhance the data available at each step, a temporal evolution augmentation strategy is applied. The latter refers to feeding, as data augmentation, all the data of previous steps, meaning that: at step 2, the classifier will be fed with the first 20 transactions of the smart contracts as well as, separately, the first 10 transactions. Overall, through this approach, early detection is possible as, already at step 2, the authors reach an F1 score of 0.88 (Jin et al., 2022).

In the following sections I will apply the concept of early detection for the classification of smart contracts and possibly expand on the results obtained by the latter authors.

Chapter 3 – Data Gathering and Exploratory Analysis

3.1 Data Gathering

The data collection procedure began with the gathering of both Ponzi and Non-Ponzi smart contracts' addresses. Such data was collected following the papers by Bartoletti et al. (2020), Chen et al. (2018) and Zheng et al. (2022) in which the authors manually labeled smart contracts as being Ponzi or Non-Ponzi. The classification is based on a manual inspection of the Solidity code and on the verification of 4 distinct requirements described in the previous section, namely:

- R1 = The smart contract distributes money to investors.
- R2 = The money gathered by the contract comes from investors only.
- R3 = Each investor makes a profit conditionally on new investors sending money to the contract.
- R4 = The risk of losing the investment grows with the time one joins the scheme.

Once those addresses were obtained, transactions' data was acquired through the Etherscan.io API. Such an API offers the possibility of gathering both internal and external historical transactions.

External transactions refer to transactions made from an Externally Owned Address or wallet towards the smart contract to activate a specific contract function (Amir, 2021).

For instance, a user wanting to activate a specific function available in a Ponzi smart contract, would need to send the required amount of ETH to activate it. Such a transaction would then be listed under the external transactions data. The features that are available for those movements are the following:

- blockNumber: the block number in which the transaction was confirmed
- timeStamp: the Unix in seconds of when the transaction was confirmed
- hash: the transaction hash of the external transaction
- from: the Ethereum address that initiated the transaction

- to: the Ethereum address that received the transaction
- value: the amount of Ether transferred in the transaction
- gas: the amount of gas used in the transaction
- gasPrice: the gas price at which the transaction was executed
- isError: a binary indicator of whether the transaction took place or resulted in an error
- txreceipt_status: the status of the transaction receipt, if available (e.g. 0 for fail, 1 for success)
- input: the input data of the transaction (if any)
- contractAddress: the address of the smart contract (if any)
- cumulativeGasUsed: the overall gas used by the contract
- gasUsed: the amount of gas used in the transaction
- confirmations: the number of confirmations that the transaction has received on the blockchain
- methodId: the function signature of the smart contract function that was called by the transaction.
- functionName: the name of the called function.

Internal transactions instead refer to functions that are activated based on the transaction made towards the smart contract. Importantly, there is not necessarily a 1-to-1 relationship between external transactions and internal ones as a single external movement can generate multiple internals. For instance, once a user makes an external transaction towards a Ponzi smart contract, a set of functions would be activated to redistribute the newly acquired ETH to the existing nodes that have already joined the scheme (Amir, 2021).

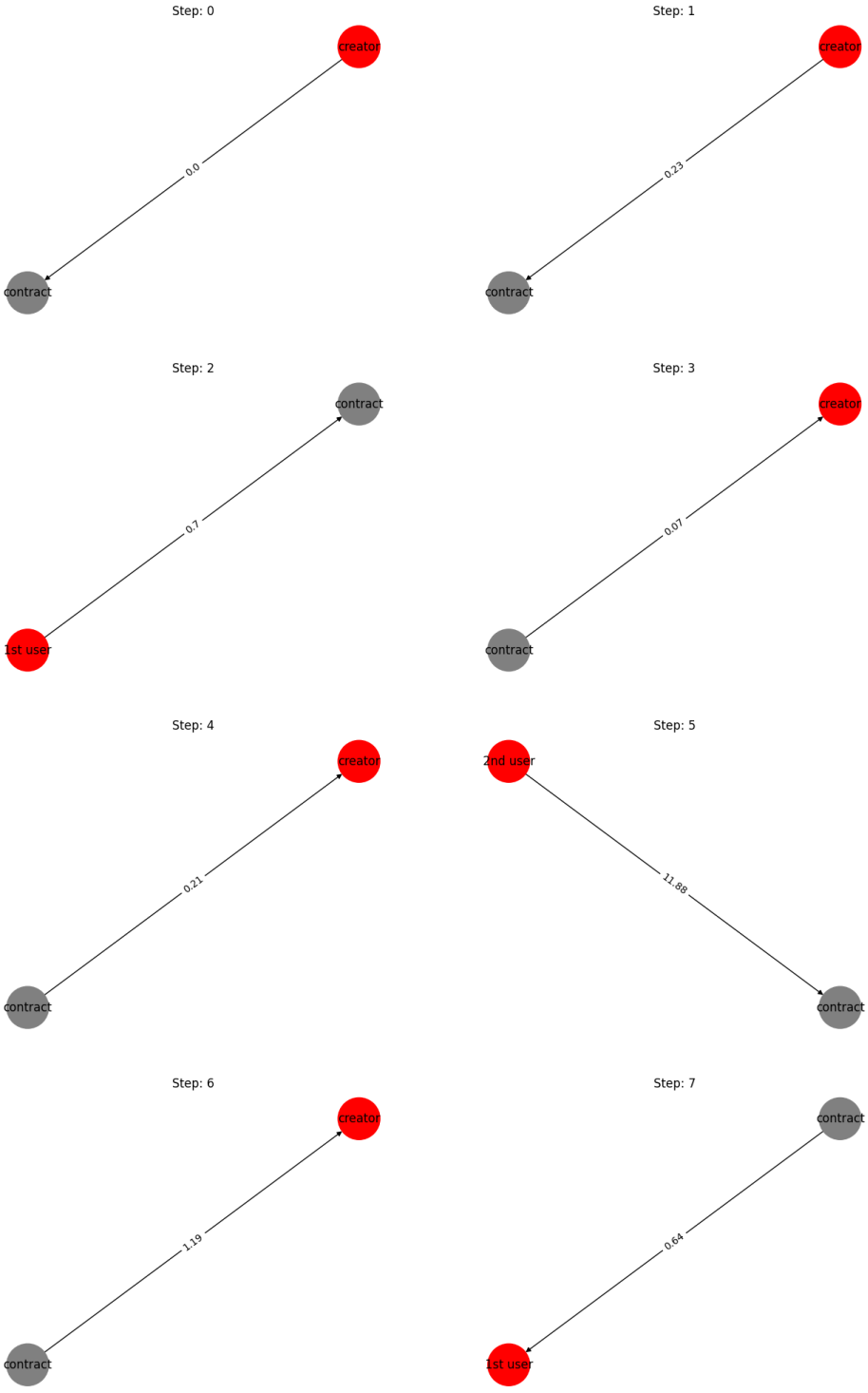
In this case, the available features are the following:

- blockNumber: the block number in which the transaction was confirmed
- timeStamp: the timestamp in seconds of when the transaction was confirmed
- hash: the transaction hash of the internal transaction

- from: the Ethereum address that initiated the internal transaction
- to: the Ethereum address that received the internal transaction
- value: the amount of Ether transferred in the internal transaction
- contractAddress: the address of the smart contract (if any) involved in the internal transaction
- input: the input data of the internal transaction (if any)
- type: the type of the internal transaction (e.g. CALL or CREATE)
- gas: the amount of gas used in the internal transaction
- gasUsed: the amount of gas used by the transaction (if any)
- traceId: the unique ID of the transaction trace (if any)
- isError: a binary indicator of whether the transaction took place or resulted in an error
- errCode: the reason for the error (if any)

The dynamics of a real Ponzi smart contract are reported below for illustrative purposes. At each time step only the nodes being related to a transaction are reported. From the graphs it is possible to observe the typical creation and network dynamic pattern of a Ponzi smart contract. In particular, time steps 0 and 1 report the network creation phase originated from the contract creator. At step 2, the first new user joins the scheme and immediately, at step 3 and 4 the contract creator receives a share of the pay-in. Next, at step 5 a new user joins, and at step 6 and 7 the existing nodes are paid a share. Importantly, the share of the contract creator is larger as, typically, the longer a user is in the scheme, the greater the payouts she will receive. This pattern of creation and redistribution of newly acquired ETH is common across many different schemes.

Figure 6: *Network Dynamics FastRealisticPyramid*



Overall, data about 495 Ponzi smart contracts was obtained for a total of 145,363 transactions, partitioned into 53,572 internal transactions and 91,791

external ones. On the other hand, 7,046 non-Ponzi smart contracts were identified, totaling 5,290,536 transactions, divided into 610,250 internal and 4,680,106 externals.

3.2 Data Cleaning and Feature engineering

Following the data gathering there were multiple data cleaning operations performed. In particular, there were instances of the same smart contracts being reported multiple times due to the contract address being formatted with upper and lower-case characters. All duplicates were thus dropped. Next, contract creation transactions were characterized by the "to" field missing and had to be fixed. Moreover, there were instances for which the "from" field in internal transactions had a different address compared to the contract one. Those instances, already observed by Bartoletti et al. (2020), were then treated as external transactions and originate in settings where the user does not directly send money to the smart contract but goes through another contract (typically a wallet one). This movement of money is an inflow with respect to the contract and must thus be treated as an external transaction. Following these procedures, the total number of transactions for Ponzi smart contracts is 121,810, with 37,795 internal and 84,015 externals, whereas for non-Ponzi smart contracts the total is 3,733,867, with 531,734 internal and 3,202,133 externals. Additionally, some contracts had to be removed due to an extremely low number of distinct nodes interacting with them. In fact, all smart contracts with fewer than 3 nodes were dropped. The rationale for this decision is related to two aspects:

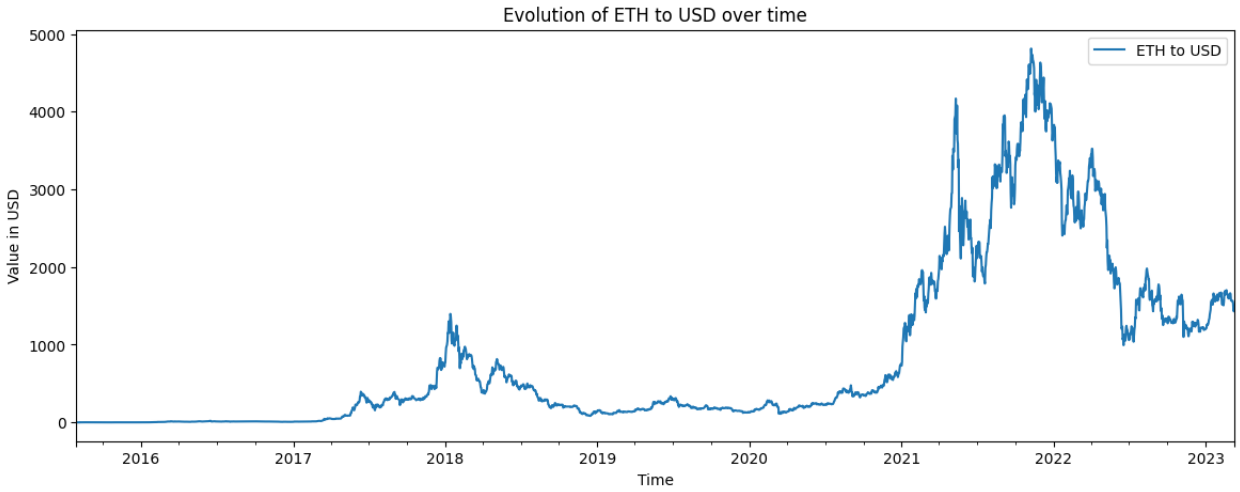
1) In order for a Ponzi scheme to take place there must be at least 2 nodes, one being the contract creator and another being a user joining a contract. However, with 2 nodes, the Ponzi scheme dynamics are not present as the recruiting of other nodes is missing. Moreover, with just two nodes, the transactions may be indistinguishable from those of a financial derivative contract.

2) Maximizing the number of smart contracts reported in the dataset can potentially improve the ability of the algorithm to detect frauds early on in their development and thus ameliorate the fraud detection capability of the model.

Finally, the data cleaning procedure yielded the following results: 239 Ponzi smart contracts, featuring 109,519 total transactions divided into: 29,969 internal transactions and 79,550 external ones; 2,151 non-Ponzi smart contracts, featuring 3,653,975 total transactions with 465,524 internal and 3,188,451 external respectively.

In addition to the data obtained from Etherscan, additional features had to be constructed. Firstly, the value reported under the column "value" was in ETH and, provided the great variability in the price of ETH, depicted in figure 7, such value was converted to USD.

Figure 7: *ETH to USD overtime*



Source: Own elaboration based on Yahoo Finance data[6]

Moreover, the "timestamp" variable also had to be converted to a human readable time format. Next, two new features were constructed. On one hand, for external transactions, the transaction cost was determined by taking the

[6] Source: https://finance.yahoo.com/quote/ETH-USD/history

product between the gas used and the gas price, in line with the Etherscan guidelines. Through this computation, the value obtained represented transaction costs in ETH which had to be converted to USD. The rationale behind this variable is that, potentially, Ponzi smart contracts will have lower transaction fees since, by nature, they thrive on the ability of attracting new users. The second innovative feature is instead given by the name of the smart contract. Said variable was extracted through web scraping with the BeautifulSoup library. Further, such a gathering proved to be challenging as the Etherscan.io website is not easily scrapable and, thus, a special algorithm had to be devised to accomplish this task. According to Etherscan, the public name tag of a contract is determined by the person who verifies the contract source code. In general, such practice is performed by the contract developer. The intuition for the usefulness of this feature is that the contract name tends to be composite and, therefore, with an appropriate embedder, an overall embedding representing the contract name can be determined. Further details are reported in the Exploratory analysis section.

3.3 Exploratory analysis

Once the gathered data was cleaned, and the new features were engineered, I proceeded to document the current situation of both Ponzi and Non-Ponzi smart contracts. Overall, 16 descriptive metrics were computed, of which 14 are directly comparable between the two types of smart contracts, whereas 2 of them focus on the redistribution patterns applicable only to Ponzi contracts. Table 1 summarizes the results by providing, for all the metrics the total, mean, standard deviation, median, minimum, 1st quartile, 3rd quartile and maximum. The following sections provide further analyses of those metrics.

Table 1: *Descriptive statistics Ponzi and Non-Ponzi Smart Contracts*

Type of smart contract	Feature	Total	Mean	Std	Median	Min	1st quartile	3rd quartile	Max
Non-Ponzi	All-degrees	3,653,975.00	1,698.73	3,589.67	86.00	3.00	16.00	894.50	20,021.00
Ponzi	All-degrees	109,519.00	458.24	1,557.74	72.00	3.00	20.50	291.50	14,608.00
Non-Ponzi	Errors external transactions	272,025.00	131.60	519.74	5.00	0.00	1.00	44.50	9,907.00
Ponzi	Errors external transactions	4,925.00	20.61	90.34	1.00	0.00	0.00	6.00	959.00
Non-Ponzi	Errors internal transactions	9,841.00	9.81	117.41	0.00	0.00	0.00	0.00	3,279.00
Ponzi	Errors internal transactions	3,315.00	13.87	170.38	0.00	0.00	0.00	0.00	2,627.00
Ponzi	Gini payins	-	0.61	0.19	0.63	0.00	0.48	0.76	0.98
Ponzi	Gini payouts	-	0.68	0.22	0.72	0.00	0.59	0.83	1.00
Non-Ponzi	In-degrees	3,188,451.00	1,508.97	3,083.70	77.00	1.00	14.00	790.00	11,388.00
Ponzi	In-degrees	79,565.00	332.91	1,213.90	48.00	3.00	14.50	209.00	10,033.00
Non-Ponzi	In-degrees distinct users	988,215.00	467.68	1,217.66	18.00	1.00	4.00	199.00	8,706.00
Ponzi	In-degrees distinct users	19,447.00	81.37	351.20	10.00	2.00	4.00	40.50	3,731.00
Non-Ponzi	Inflow transactions	$2,930,827,144.09	$1,387,045.50	$8,755,848.10	$332.78	$0.00	$1.43	$37,417.53	$182,597,243.68
Ponzi	Inflow transactions	$21,706,902.13	$90,823.86	$592,928.91	$1,110.09	$0.35	$188.29	$7,219.34	$7,290,782.93
Non-Ponzi	Lifetime (days)	-	428.17	618.60	105.00	0.00	16.50	587.00	2,742.00
Ponzi	Lifetime (days)	-	498.36	676.71	186.00	0.00	3.00	729.00	2,671.00
Non-Ponzi	Out-degrees	465,524.00	502.18	1,724.29	8.00	1.00	2.00	60.50	10,000.00
Ponzi	Out-degrees	29,954.00	128.56	448.57	23.00	1.00	5.00	74.00	4,602.00
Non-Ponzi	Out-degrees distinct users	44,055.00	47.52	307.46	2.00	1.00	1.00	4.00	4,530.00
Ponzi	Out-degrees distinct users	7,951.00	33.27	142.64	6.00	0.00	3.00	26.00	1,723.00
Non-Ponzi	Outflow transactions	$3,395,118,550.60	$2,522,376.34	$30,795,932.60	$455.89	$0.00	$4.38	$47,303.04	$1,078,281,192.51
Ponzi	Outflow transactions	$20,282,393.93	$84,863.57	$538,267.10	$1,068.21	$0.00	$146.85	$7,276.16	$5,929,645.73
Non-Ponzi	Payins by users	$2,930,827,144.09	$4,518.40	$223,541.00	$0.00	$0.00	$0.00	$651.47	$132,467,397.08
Ponzi	Payins by users	$21,706,902.13	$1,467.97	$14,880.90	$184.79	$0.00	$35.47	$634.80	$1,047,459.18
Non-Ponzi	Payouts by contracts	$3,053,158,260.62	$74,674.91	$3,406,969.16	$10.79	$0.00	$0.39	$715.86	$560,395,095.13
Ponzi	Payouts by contracts	$20,282,393.93	$1,371.64	$52,408.74	$0.00	$0.00	$0.00	$11.67	$5,633,125.29
Ponzi	Profit by user	-	-$96.34	$54,043.85	-$104.47	-$1,047,459.18	-$428.92	-$11.72	$5,633,125.29
Non-Ponzi	Transaction costs - contract	$3,864,374.50	$1,869.56	$11,811.76	$57.97	$0.00	$9.72	$551.94	$452,894.06
Ponzi	Transaction costs - contract	$44,147.92	$119.97	$568.52	$3.96	$0.00	$0.49	$26.34	$5,236.18
Non-Ponzi	Transaction costs - user	$3,864,374.50	$6.01	$176.67	$0.80	$0.00	$0.20	$2.75	$57,543.14
Ponzi	Transaction costs - user	$44,147.92	$2.30	$19.67	$0.64	$0.00	$0.10	$1.38	$2,279.62

3.3.1 Network metrics

The comparison of the network characteristics between the two types of contracts is based on 5 different measures: in-degrees, distinct users in in-degrees, out-degrees, distinct users in out-degrees, and all-degrees.

In-degrees refer to all external transactions that characterize a contract, i.e., all money flowing into the smart contract. The results reported in the above table clearly indicate a greater number of inflow transactions for non-Ponzi smart contracts as both the median and the reported quartiles are vastly greater than those of Ponzi smart contracts. Importantly, both smart contracts are characterized by a large variability in their distribution, as depicted by a direct comparison of the mean and standard deviation with the median. These results are expected as when Ponzi schemes grow in popularity, they can be more easily identified as being frauds and thus being shut down. Another possible explanation would be related to the nature of those contracts as they are characterized by the promise of large returns which are uniquely financed by the newly acquired pay-ins. However, the contracts' lifetime, computed as the difference in days between the contract creation and the last recorded transaction, demonstrates that the median lifetime for a Ponzi smart contract is actually greater than that of the non-Ponzi counterparts, falsifying such argument. Importantly, surrounding such result is a high variability, observed both in the standard deviation and in the interquartile range.

Similarly, when analyzing the in-degrees distinct users, non-Ponzi smart contracts tend to be larger and attract a greater number of individuals. In fact, all the computed measures support this result. A possible explanation for it could be related to the greater soundness of non-Ponzi smart contracts.

On the other hand, out-degrees refer to all the internal transactions that characterize a contract, i.e., all money flowing out of the contract and towards the crypto investors. In those metric, Ponzi smart contracts demonstrate their redistributive nature. In particular, it is possible to notice that the median

number of out-degrees for Ponzi schemes is much larger than that of non-Ponzi ones, with also greater quartiles characterizing the distribution. Similar results are also displayed in the distinct users in the out-degrees, for which Ponzi smart contracts have even more differences to the other smart contracts. Those results are in-line with the nature of those schemes as they are redistributive, thus allowing many different users to receive pay-outs.

Overall, as depicted by the all-degrees measure, which groups together in and out degrees, the size of the two categories of smart contracts are mostly the same in terms of number of transactions. In fact, there are comparable measures in terms of Median, minimum, and 1st quartile. Non-Ponzi smart contracts however demonstrate a much greater variability in terms of size, observed by the large IQR and the standard deviation surrounding the mean.

3.3.2 Contract pay-ins and payouts measures

To inspect differences between the two categories of contracts relating to monetary flows, 4 distinct metrics are computed: value in inflows, value in outflows, pay-ins by users, and payouts by contracts. An analysis of those metrics clearly highlights the difference between Ponzi smart contracts and non-Ponzi ones. In particular, when looking at the inflows, Ponzi schemes are characterized by a much larger median value compared to non-Ponzi ones with a definitely smaller variability in money attracted, as reported by the IQR. Non-Ponzi schemes instead showcase a much more skewed inflow distribution, having a lot of density at low values with fewer contracts attracting sizable USD. Overall, it is also worth observing that Ponzi schemes have been able to attract 21.7 million USD. The same pattern is also detected when looking at the pay-ins by users. In fact, for Ponzi smart contracts such distribution is highly concentrated around the median value, thus highlighting commonalities across those frauds. Those aspects are not detected when looking at non-Ponzi schemes, which feature an ample number of pay-ins of zero and a large

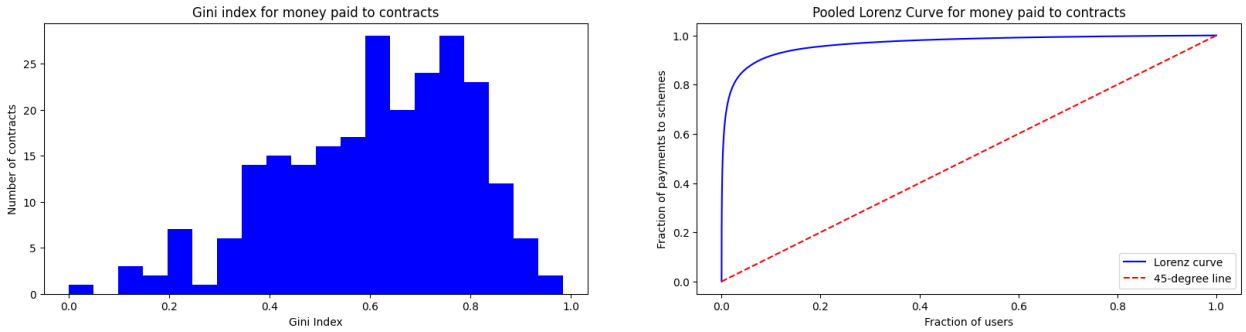
variability. It is worth pointing out that many of those smart contracts are token sales attracting large sums of money, thus explaining this result. Analogous results are detected in outflow transactions, featuring Ponzi contracts having much larger payouts with moderate variability and non-Ponzi contracts being characterized by small outflows with an extremely high variability. More interesting results are detected when analyzing payouts received by users. In this instance, non-Ponzi contracts feature larger payouts despite having smaller outflows indicating already the unfair redistribution policy of Ponzi schemes. This last explanation is highlighted in the distribution of payouts of those schemes since the 3rd quartile is particularly small. A further inspection of the redistribution inequality of Ponzi schemes will be performed in the following section.

3.3.3 Redistribution metrics

To analyze how Ponzi smart contracts redistribute money to their users, 3 different measures are determined: the pay-in Gini index, the payout Gini and the distribution of profits by users. The Gini index measures the extent to which a distribution among individuals deviates from perfect equality. Its range is between 0 and 1 with 0 representing perfect equality and 1 being perfect inequality (The World Bank, n.d.). When analyzing such metric for pay-ins it is clear that there are a lot of discrepancies between the sums of money that must be invested by different users. In fact, the Index has a median of 0.63, suggesting that users must make different contributions in order to join the scheme. This is in line with the classification made by Bartoletti et al. (2020) which shows that as Ponzi schemes evolve, the contribution that must be made by new joiners keeps on increasing, thus exacerbating the differences. Moreover, contract creators typically invest particularly limited sums of money or, even, zero USD in the contract. Figure 8 reports the pay-in Gini index distribution as well as its associated Lorenz curve. The latter plots the cumulative percentage of money invested against the cumulative number of

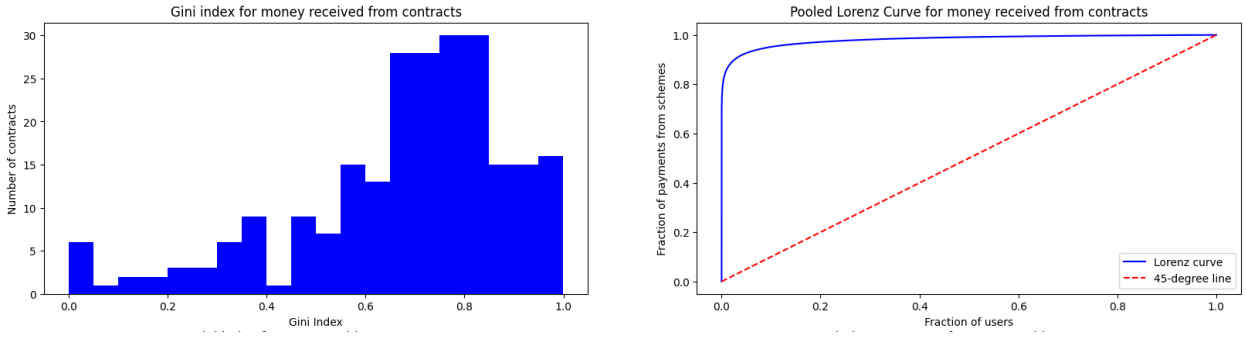
investors. In settings of perfect equality, the Lorenz curve should be a flat line, matching the bisector line (The World Bank, n.d.).

Figure 8: *Gini index and Lorenz curve of pay-ins*



An even worst result is detected when analyzing the Gini index of the payouts made by the contracts. In fact, in this case, an even higher Index is observed, highlighting a much more asymmetric redistribution of money to users. This result is in line with previous literature documenting Ponzi schemes. Figure 9 reports the payout Gini index distribution and its associated Lorenz curve. It is worth highlighting the high density observed at the two extremes. Density at zero refers to Ponzi schemes that have not yet made payouts to their users, whilst density at the other extreme indicates contracts redistributing money uniquely to the contract creators.

Figure 9: *Gini index and Lorenz curve of payouts*



Finally, the profit by user metric seeks to identify a monetary value (in USD) to the redistribution applied by those contracts. In line with the above findings, the distribution of profits is almost entirely negative, featuring negative profits

up to the 3rd quartile. Further, there seem to be users registering extremely low and high profits, in the millions, and those are all related to the scheme "Sociall: token sale". Finally, the fraction of users registering a negative profit is 86.29% and the fraction of those breaking even or making a profit is 13.71%.

3.3.4 Contract execution metrics

The last set of metrics that was computed refer to contract execution. More precisely, four different aspects are considered: the errors in external transactions, the errors in internal transactions, the transaction costs grouped by contracts and the transaction costs grouped by users. Overall, it appears that Ponzi external transactions are characterized by less errors compared to non-Ponzi, as seen by both the median, the quartiles, and the mean. A possible explanation for this has to do with the discrepancy in the number of transactions being present between the two categories of smart contracts. It is indeed more likely to encounter errors in Ponzi contracts as reported by internal transactions. In fact, although there is a significant difference between the total number of transactions, it appears that Ponzi schemes have a higher mean and standard deviation in errors compared to the non-Ponzi counterparts. It is important to note that in both instances, the third quartile indicates the absence of errors.

The last feature that was considered relates to the transaction costs. In particular, previous studies have not considered this metric although it indicates significant differences between the two categories of contracts considered. In fact, both groupings indicate that Ponzi smart contracts have less transaction costs (i.e., lower gas price and gas required for the transaction to be completed) than non-Ponzi ones. It is important to highlight that the difference between the two when grouping by contract could be related to the number of transactions that characterize the smart contracts considered. This argument, however, is not applicable when grouping by users as, in the

gathered data, individuals tend to make a comparable number of transactions to the contracts. In this instance, it becomes clear that Ponzi smart contracts are cheaper to execute than non-Ponzi ones, as seen by every metric computed. The rationale for this phenomenon is related to the nature of Ponzi schemes, which, by construction thrive on the continuous expansion of the user base. Low transaction costs, in fact, make it cheaper to join the scheme.

3.3.5 Name analysis

Upon analyzing the Ponzi schemes identified, one feature that the bulk of the literature is not considering is the name of the smart contract. Such names are mostly composite (i.e., WealthRedistributionProject) and therefore, by breaking the smart contract name into its different word components embeddings representing the contracts can be generated. To achieve this, I initially cleaned the names from numbers and special characters and then I exploited the presence of uppercase letters to individuate words. As seen in the contract name reported above, performing this task would lead to the following words: Wealth, Redistribution, Project. Following this procedure, contract names were adequately cleaned and ready to be parsed by an embedder. I decided to use Bert as the embedding algorithm given that it can produce particularly accurate word embeddings thanks to the bulk of corpora used in its unsupervised training scheme. The Bert tokenizer also exploits Wordpiece to identify possible sub-words in the provided string. Therefore, I constructed a function to look for all possible combinations of sub-words which are part of the English vocabulary. Next, the tokens were passed to the embedder and the different word representations were averaged to obtain the contract name embedding. After having computed those vectors, to identify commonalities across the smart contracts name, clustering was performed. More precisely, three distinct clustering algorithms, namely, K-Means, Hierarchical Clustering and Gaussian Mixture Modelling Clustering were applied to Ponzi smart contracts, non-Ponzi smart contracts and the combination of

the two. The choice of using multiple clustering algorithms is based on assessing the robustness of results whilst reducing the limitations characterizing each clustering method.

To determine the appropriate number of clusters, three different metrics were computed: the Silhouette score, the Calinski Harabasz score and the Davies Bouldin metric. The silhouette score was first introduced by Rousseuw (1987) and it is computed by measuring the mean intra cluster distance with the mean nearest-cluster distance for each sample. In particular, such technique penalizes scenarios in which the specified number of clusters is either too high or too low. The appropriate number of clusters indicated by this technique is the one associated with a local maximum in the score as it enables to reach good separation between clusters whilst avoiding over segmenting the data (Rousseeuw, 1987). The Calinski Harabask (CH) metric, on the other hand, is determined by taking the ratio between the inter-cluster and the intra-cluster sum of squared distances (Caliński and Harabasz, 1974). The appropriate number of clusters indicated by the technique is the maximum CH score obtained. In the determination of the number of clusters I have considered local maxima in line with the Silhouette score. Finally, the Davies Bouldin metric is determined as the average similarity of a cluster with the one most similar to it. More precisely, the score is determined by summing the within cluster dispersion of two clusters and then by dividing such difference with the Euclidean distance between the centroids (Davies and Bouldin, 1979). Therefore, the lower the score, the better the separation between the clusters.

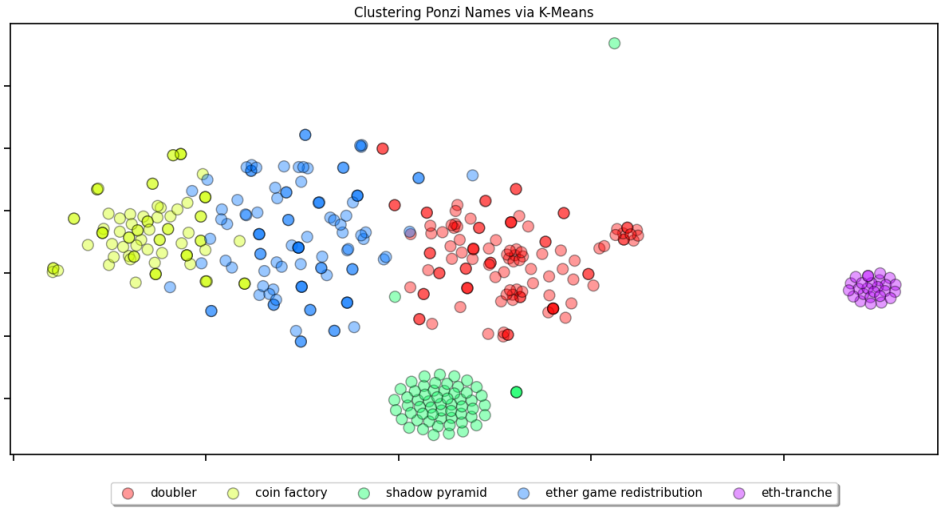
3.3.6 Ponzi schemes names clustering results

Overall, when analyzing Ponzi schemes' names, it becomes clear that five different types of contracts exist, namely: doubling contracts, coin factories, shadow pyramids, redistribution games and eth-tranche schemes. Doubling contracts are characterized by the promise of doubling the ether you deposit. An instance of this type of contract is given by Crystal Doubler which promises

doubling ether in a short a period of time with no fees attached[7]. Coin factories tempt users by promising returns whilst trying to guarantee liquidity for withdrawals. It is a highly speculative type of scheme which repays investors based on the fees of new joiners (Hoenicke, n.d.). Shadow pyramids include EthPyramid a scheme promising the following: "A no-bullshit, transparent, self-sustaining pyramid scheme"[8]. Redistribution games are games designed to allegedly distribute money to their players. An example is Protect the castle which has the following dynamics. There is a castle which is being attacked, all users who finance the defense of the castle will be repaid, by the king, 2 times the amount invested when funds allow him to. If no one contributes to the castle for 6 hours, the last 3 citizens get the King's Piggy bank. The king collects 3% of all the transactions and users must pay a 3% fee for each investment made[9]. Finally, eth-tranche refer to EthTranchePricing contracts which perform crowd sales and distribute the earnings mainly to the contract founders[10].

The resulting clusters are reported in figure 10.

Figure 10: *Ponzi smart contracts name clustering*



[7] Source: https://etherscan.io/address/0x51170b18bca7896b49c52dcc18e66e5c921e100f#code
[8] Source: https://etherscan.io/address/0xc6b5756b2ac3c4c3176ca4b768ae2689ff8b9cee#code
[9] Source: https://etherscan.io/address/0xa9fa83d31ff1cfd14b7f9d17f02e48dcfd9cb0cb#code
[10] Source: https://etherscan.io/address/0x7bbc3ad5c296ae6eb50228d2a6a37234d2db3ff1#code

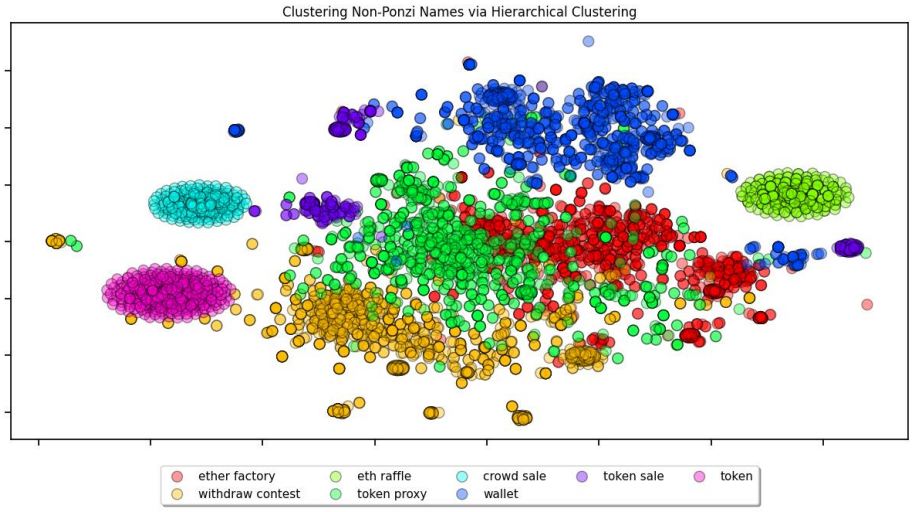
As can be seen from the above figure, clusters appear to show a high degree of separation between them.

3.3.7 Non-Ponzi smart contracts name clustering

Non-Ponzi smart contracts are, overall, characterized by the presence of 8 distinct clusters, specifically: token sales, withdraw contests, ether factories, wallets, crowd sales, tokens, eth raffles, and token proxies. The names of the identified macro-classes are rather self-explanatory. The more peculiar categories are given by: withdraw contests, ether factories and eth raffles. More precisely, withdraw contests refer to smart contract games (i.e., Poker, Dice, etc.), ether factories refer to smart contracts designed to generate new tokens[11], and eth raffles refer to smart contracts that allow individuals to purchase tickets and one winner will receive the prize pool.

The figure below reports the clustering results.

Figure 11: *Non-Ponzi smart contracts name clustering*

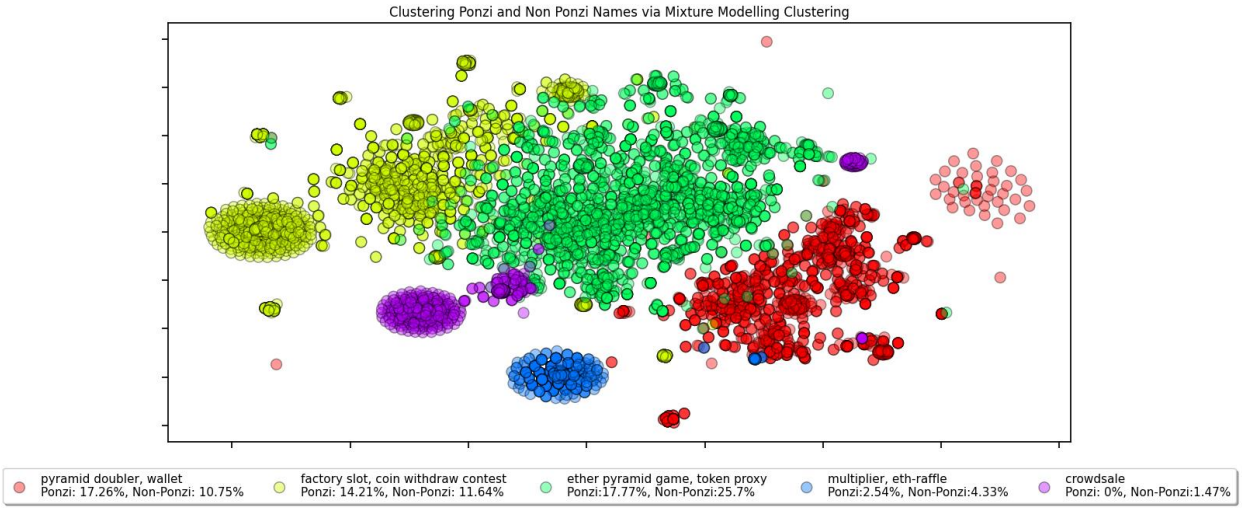


3.3.8 Combining Ponzi and Non-Ponzi word representations

To further comprehend whether the contract name could be a potential new feature to include in the modelling phase, I decided to compare how Ponzi smart contracts names would cluster with non-Ponzi ones. Figure 12 reports the clustering results.

[11] Source: https://etherscan.io/address/0x71fc91bd2a3c75ffd7464cbd26edcd5fc50425dd#code

Figure 12: *Ponzi and Non-Ponzi smart contracts name clustering*



Overall, 5 distinct clusters are formed and for each of them I have reported the categories of contracts present as well as the fraction of Ponzi and non-Ponzi contracts computed on the total number of contracts of that type available in the dataset. In all of them there appears to be one instance of smart contracts dominating the cluster, suggesting that the name has discriminatory power. The only clusters in which this is not observed are the yellow and the blue one where high similarity between the contract names is reported. In fact, the yellow cluster reports game type contracts for which the name is not indicative of the presence of Ponzi schemes. Similarly, raffles and multiplier-based contracts tend to showcase similar names across the two types of smart contracts.

Chapter 4 – Modelling Framework

4.1 Data Preparation and Feature Engineering

As highlighted in chapter 2.3.3 of the literature review, the bulk of the literature focusing on detecting Ponzi schemes by analyzing the transactions' history does not identify frauds in a timely manner. Specifically, the entire transaction history is used to train models, thus implying that if new Ponzi smart contracts are established, they can be detected only after they have grown in popularity and have hurt a plethora of individuals. Hence, detecting Ponzi schemes early on is a key requirement to limit the dire financial consequences of those frauds. To ensure that this condition is met, I decided to follow the approach introduced by Jin et al. (2022) whereby only a limited set of transactions is used to create the training and testing set. Effectively such an approach guarantees that models will be able to identify Ponzi schemes early as only small sets of transactions are available for classification. More precisely, I divide the available transactions for each smart contract in blocks of 10, and I construct 10 overall identification phases. This means that for step 1, I will consider smart contracts having at least 10 transactions and only the first 10 transactions will be used to train the different classifiers. Similarly, at step 2 only the smart contracts having at least 20 transactions will be evaluated and the classifiers will be trained on the first 20 transactions available and so on. Contrary to the approach by Jin et al. (2022), who first filter the smart contracts so as to keep only those having at least 100 transactions, by dynamically identifying the set of contracts that will be considered at each step, the number of observations and, thus the statistical power of the different proposed models, can be maximized.

Once the set of smart contracts and their related transactions have been identified, different features are constructed to summarize the movements' history.

Precisely, a total of 87 features have been engineered focusing on: the number of in-degrees and out-degrees (2 features), the number of transactions for in-degrees and out-degrees (14 features), the errors in the in-degrees and out-degrees (12 features), the number of distinct users in in-degrees and out-degrees (2 features), the transaction value in in-degrees and out-degrees (16 features), the age in minutes of the transactions (6 features), the inflows and outflows from users (14 features), the transaction costs for the contract and users (15 features) and the dummies obtained from the name embedding clustering (6 features).

Overall, the number of in-degrees and out-degrees can introduce prediction power as I would expect that a Ponzi smart contract, given a set of transactions, would typically have a lot of internals compared to a non-Ponzi, as depicted in the taxonomy provided in chapter 2.3.2 of the literature review. Next, the number of transactions for in-degrees and out-degrees seeks to describe the behavior pattern of users in terms of transactions sent and received to and from the smart contract. The 14 features extracted relate to the average, standard deviation, median, minimum, 1st quartile, 3rd quartile and maximum of the number of transactions sent/received from the smart contract. Similarly, the inflows and outflows from users perform the same task focusing on the value in USD of the different transactions rather than on the numerosity of them. I decided to extract 7 features for each inflow and outflow considered so as to properly account for the distribution of the data across users. Related to the behavior of pattern redistribution available, the number of distinct users participating and receiving money from the smart contracts is also introduced as a feature.

The errors in in-degrees and out-degrees describe the presence of errors in each block of transactions. The rationale for the set of features extracted (total, mean, standard deviation, median, 1st and 3rd quartile) is that since

Ponzi schemes tend to have specific rules for the value that must be sent to activate specific contract functions, the error distribution might differ compared to that of a non-Ponzi smart contract.

Additionally, contract level features that have been extracted relate to the transaction value for out and in degrees which tries to capture the distribution of the inflows and outflows for the considered block of transactions. Analogously, the transaction costs for the contract incorporate the contract-level total transaction costs that have been sustained. Since Ponzi schemes thrive on the ability to continuously attract new individuals, the transaction costs might introduce additional predictive power. On the same note, the transaction costs have also been computed by grouping by the user, thus identifying how they are dispersed across the smart contract's participants. Finally, the age in minutes of each transaction with respect to the time of the contract creation looks to incorporate a time dimension in the analysis, whereas the clustering dummies integrate the results of the clustering analysis performed in the previous chapter. It is important to notice that I have included 6 dummies rather than 5 as there are smart contracts for which the name is unavailable. Thus, all nameless smart contracts have been assigned to the sixth cluster.

Lastly, the data was randomly split in training (60%), validation (15%), and test (25%) stratifying on the indicator of whether the smart contract was fraudulent, so as to guarantee a constant fraction of Ponzi schemes in all the datasets.

The table below reports the total number of Ponzi and Non-Ponzi smart contracts available for each transaction step considered.

Table 2: *Ponzi and Non-Ponzi smart contracts available at each step*

Step	1	2	3	4	5	6	7	8	9	10
Ponzi	222	182	171	159	140	126	120	115	112	106
Non-Ponzi	1814	1536	1395	1298	1233	1188	1145	1099	1064	1037
Total	2036	1718	1566	1457	1373	1314	1265	1214	1176	1143

As can be seen, there is an important problem of class imbalance in all the different steps considered. In fact, Ponzi smart contracts represent about 10% of the observations available at each step and, thus, two important considerations must be made.

Firstly, the class imbalance problem can be reduced via oversampling in the training and validation step, so as to guarantee that the model will observe enough Ponzi schemes observations and tune its parameters accordingly. I decided to explore oversampling the minority class rather than undersampling the majority class to ensure the maximization of the number of distinct observations available for the classification phase. Four different oversampling approaches have been performed, namely: random, SMOTE, Borderline-SMOTE, and ADASYN oversampling.

The former entails that the observations from the minority class will be randomly selected and will populate the dataset. Although this leads to an increase in the number of observations available, having exact replicas ultimately causes overfitting (Ali et al., 2019). An improvement over random sampling is proposed by Synthetic Minority Oversampling Technique (SMOTE) which generates synthetic samples by applying k-Nearest Neighbors (KNN) on the inferior class (Ali et al., 2019). Despite the advance over random sampling, a limitation is that new samples are generated without considering the majority class observations which may cause overlapping between the classes, leading to over-generalization and harming the fit metrics (Ali et al., 2019). Borderline-SMOTE and ADASYN try to limit this phenomenon by explicitly considering the majority class during the synthetic generation of new samples. The former

accomplishes this goal by identifying the borderline between the two classes and generating minority samples on it (Ali et al., 2019). The latter achieves the desired result by creating minority samples in areas where more majority samples are identified as, in those cases, there is a high risk that the classifier will ignore those samples (Ali et al., 2019).

Secondly, the evaluation of the models' performance on test data cannot be based on accuracy as, otherwise, a dummy classifier with 0 discrimination power would achieve high performance scores. Solutions to this problem are provided by selecting the models with the highest F1 and Auroc score.
The first is a way of harmonically combining precision and recall thus identifying the models with the highest detection power. Specifically, precision refers to the fraction of real Ponzi smart contracts among those classified by the model as Ponzis (Wood, 2020). Recall instead is the fraction of predicted Ponzi smart contracts among the total number of positive examples (Wood, 2020).
The formulas for the three metrics are here reported:

$$Precision = \frac{True\ Ponzi}{True\ Ponzi + False\ Ponzis}$$

$$Recall = \frac{True\ Ponzi}{True\ Ponzi + False\ Non-Ponzis}$$

$$F1 = 2\,x\,\frac{Precision\,x\,Recall}{Precision + Recall}$$

The second, instead is the curve computed by combining specificity and sensitivity and measuring the area underneath the ROC curve. Essentially such curve plots the True Positive Rate and False Positive Rate at different thresholds, allowing to finally obtain a classification power metric. The metric is constrained between 0.5 and 1, with the latter entailing perfect detection capabilities.

Another important phenomenon depicted in table 2 is how the number of available observations, especially for Ponzi schemes, continuously reduces as the steps progress. Jin et al. (2022) propose Temporal Evolution Augmentation of Transaction Graph (TEAUG), an approach that encompasses a continuous expansion of the dataset with each transaction step considered.

In fact, TEAUG rests on the idea that at each step, the training and validation set is expanded with the observations available at the previous step. This means that at step 2, the data that will be used for the model fitting phase relates to all smart contracts having at least 20 transactions, focusing on their first 20 transactions and also all smart contracts having at least 10 transactions focusing on their first 10 transactions. Importantly, to avoid potential overlaps between the data used in training and testing, which would result in bias and not generalizable results, the smart contracts have been firstly randomly split accordingly. Hence, a smart contract that is part of training in step 1 will also be part of the training set even in later steps and even if it is considered for additional transactions.

With such an approach, the resulting available observations at each step are considerably higher, as illustrated in table 3.

Table 3: *Ponzi and Non-Ponzi smart contracts available at each step TEAUG*

Step	1	2	3	4	5	6	7	8	9	10
Ponzi	222	404	575	734	874	1000	1120	1235	1347	1453
Non-Ponzi	1814	3350	4745	6043	7276	8464	9608	10707	11770	12807
Total	2036	3754	5320	6777	8150	9464	10728	11942	13117	14260

Also in this case, the severe class imbalance has been addressed by oversampling. Specifically, at each step, the new observations are oversampled so as to avoid oversampling too many observations constructed with more transactions.

4.2 Traditional and Alternative Modelling

The training phase involves optimizing 7 different models via grid search.
More precisely, the different models tested relate to: a logistic regression, which will act as a benchmark, an elastic net with optimization on the penalty parameter and L1 ratio, a gradient boosting model optimized on the number of estimators and tree depth, a random forest classifier optimized on the number of estimators and tree depth, a multilayered perceptron tested with different hidden layer configurations, activation functions and regularization parameters, and, finally, a stacking of the previously mentioned models with a logistic regression featuring different L1 levels. Each model was run 5 times and the results were then averaged so as to obtain more robust metrics. This, however, was not applied to TEAUG due to the extremely high computational costs required for each optimization routine.

Yet, provided the specific nature of the data, alternative modellings have also been tested. In fact, the transaction history can be organized so as to produce graph networks, featuring as nodes the smart contract and the different participants and as edges the transactions between them.
Two alternative techniques have been applied: Graph2Vec embeddings and Graph Neural Networks (GNNs).
The first algorithm, developed by Narayanan et al. (2017), seeks to identify a new approach to compute an embedding representing an entire graph. This is accomplished by applying the word2vec Skipgram algorithm to rooted subgraphs. In particular, a graph is seen as a document, with its rooted subgraphs representing the words composing it. Essentially the algorithm is characterized by two steps: generating rooted subgraphs for each node in the graph and learning embeddings based on them. Overall, the technique is capable of learning graph embeddings which are task agnostic and data-driven in a completely unsupervised way.

In my application, the graph2vec embeddings are added as an extra feature to the previously defined ones in the classification phase.

The second relates to the application of neural networks taking as input different graphs. In particular, those algorithms work by obtaining node representations via message passing to ultimately generalize and allow for graph level predictions. Specifically, given a graph G = (V,E,X,W), with V indicating the set of nodes, E the set of edges, X the node attributes and W the edge attributes, the node representation for node v will be the result of aggregating the node features of its neighbors (Maheshkar, 2023). Therefore, a node representation in the graph will be updated in the following way:

$$h_v^t = f_{Update}(h_v^t, f_{Agg}(h_v^{t-1}, \{h_u^{t-1} : u \in N(v)\}))$$

Importantly, E, the adjacency matrix will act as a filter useful for the identification of the neighbors of the specific node considered. As reported in the equation above, the node representation through the first layer of the network is being updated based on its direct neighbors. As the number of layers applied increases, the representation of node v will be affected by the direct neighbors of each node in its neighbors. Therefore, if too many layers are added in the network, there is an over-smoothing problem as the representation of each node becomes an aggregation of the entire network (Elinas and Bonilla, 2022). Provided this problem and the type of networks that will be analyzed, I have opted for applying only 1 layer to obtain a graph embedding. The idea of message passing is exploited and tested with two different architectures: Graph Convolutional Neural Networks (GCNs) and Graph Attention Networks (GATs).

GCNs, by taking in input a feature matrix and an adjacency matrix, produce node-level representations via convolutions. The update of the node v representation in a GCN is provided in the equation below:

$$h_{N(v)} = \sum_{u \in N(v)} w_{u,v} h_u = \sqrt{\frac{1}{d_v}} \sum_{u \in N(v)} \sqrt{\frac{1}{d_u}} h_u$$

Effectively, the updated representation is gathered by exploiting message passing information from the neighbors of the considered node v, aggregate it and pass it to a neural network (Zhang et al., 2019). Importantly, the message passed by a specific neighbor u is weighted by the inverse of its degree. Finally, to obtain a graph representation, a pooling operation is performed. In the analysis I have applied average pooling, meaning that the graph representation will be the mean representation of the participating nodes. After the graph embedding is obtained, I then added graph-level characteristics (i.e., the features presented before) and introduced a MLRP layer to aggregate them and finally a softmax layer to predict a label for each graph. During the training phase, different hidden layer dimensions (16,32,64,128) and multiple dropout rates (0.1,0.2,0.3,0.4) were tested. Due to the extremely high computational costs (26 hours of runtime) the model was not applied 5 times.

GATs are different compared to GCNs as the weights assigned to the neighbors will not depend on the connections of the node but rather a function is introduced which will consider the embedding of the target and source node. In this way, the weights will be able to include node-level features and the local structure when updating the representation (Veličković et al., 2019). The updating equation for node v is reported below:

$$h_{N(v)} = \sum_{u \in N(v)} softmax_u(a(h_u, h_v)) h_u$$

With the attention weight being determined in the following way:

$$a(h_u, h_v) = LeakyReLU(a^T \cdot [Wh_u || Wh_v])$$

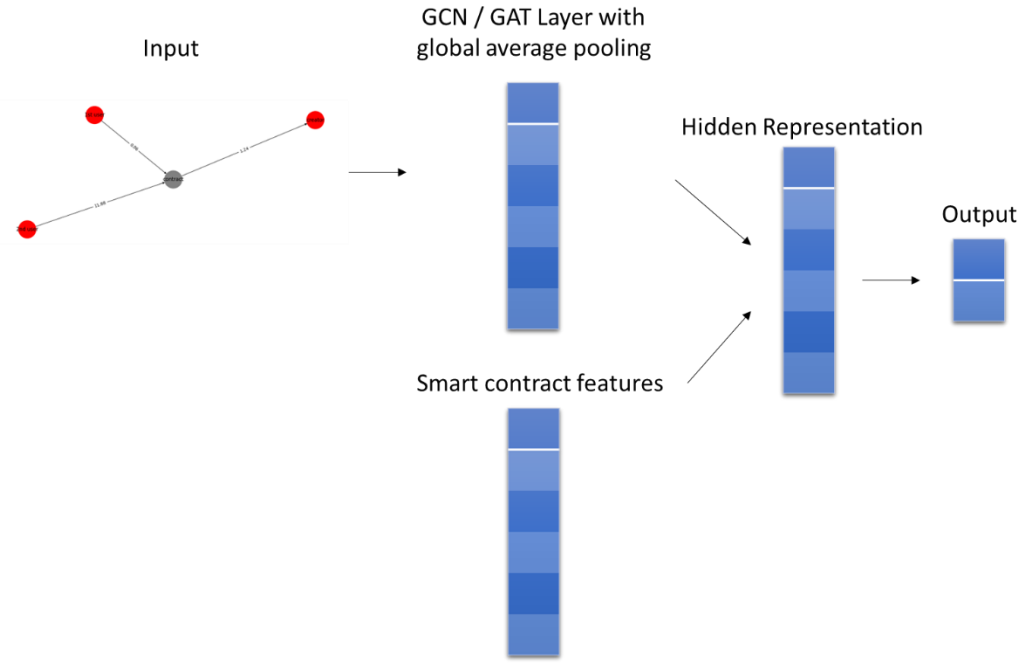
As with GCNs, after the GAT layer, global pooling is introduced to obtain a graph level representation. Similarly, graph level attributes are also added so

as to incorporate characteristics of the considered smart contract. Overall, due to the particularly high computational costs (29 hours of runtime), only one attention head was introduced, and the optimized parameters are the same as for GCNs.

For both GNNs applications, directed graphs were used to ensure that the directionality of the movements to and from the smart contract was considered. Further, different attributes relating to the transaction value sent/received, the account balance, the errors sent/received, the lifetime and transaction costs were introduced as node features. Due to the presence of multiple transactions between nodes and the smart contract, the edges were condensed into one with: directionality depending on whether the node had made a profit or a loss and weight depending on how much money the user had lost/profited.

The pipeline for the application of GNNs is reported below.

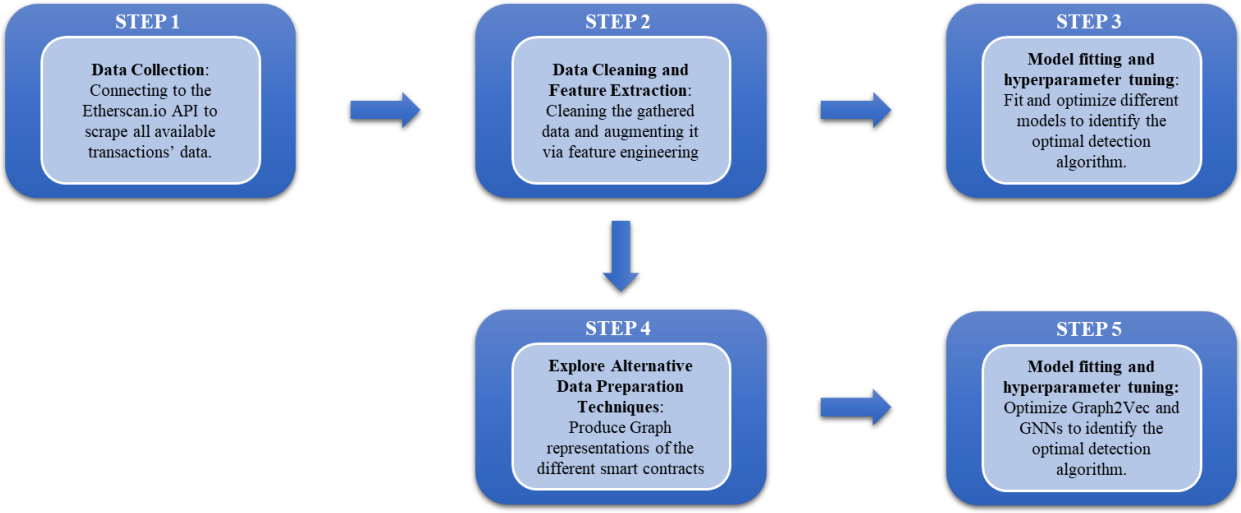
Figure 13: *Pipeline GNNs*



In a nutshell, the entire procedure that led to the development of the Ponzi scheme detection algorithm is reported below. The key steps relate to the Data

Collection via the Etherscan.io API, the Data Cleaning and Feature Extraction procedure, the Exploration of Alternative Data Preparation Techniques and the Model Fitting and Hyperparameter Tuning phase.

Figure 14: *Data and Modelling Steps*



Chapter 5 – Results and Discussion

In this section, the results of the different fitting procedures are presented, discussed and linked with the thesis research questions. Specifically, answers to the following questions will be provided:

- ➔ RQ1: How effective is the proposed framework in detecting in a timely manner Ponzi smart contracts in Ethereum?
- ➔ RQ2: What is the ideal transaction threshold required for optimal detection?
- ➔ RQ3: What are the key characteristics that are essential for the classification of Ponzi Smart Contracts?

5.1 Ponzi Smart Contracts' Detection (RQ1)

To answer the first research question, the results of the different models tested are reported in table 4, 5, 6 and 7. As highlighted in the modelling framework chapter, special emphasis should be placed on the F1 and Auroc tables as they can adequately identify the classifiers with the highest detection capabilities. Importantly, adequate discrimination power is identified starting from step 2. In fact, contrary to step 1 in which the highest recorded F1 score is 0.677, sizable improvements are detected as the number of transactions used for the feature extraction procedure increases. Overall, the different models tested highlight two main results. Firstly, introducing data augmentation in the form of both oversampling and TEAUG improves the detection power. In fact, both the F1 and Auroc scores are typically higher when considering as a benchmark the traditional machine learning approach with no augmentations. The motivation for this result is strongly related to the few observations that are available for training and validating the models, which was highlighted in the modelling framework chapter. In fact, oversampling and TEAUG greatly improve the number of available observations, especially for Ponzi smart contracts, benefiting the discrimination power of the models. Secondly, it

appears that the graph structure is not providing any additional detection power to the models, as depicted by the Graph2Vec results and the GNNs ones. Overall, the performance of the best classifier recorded is strongly satisfactory and is associated with a gradient boosting model with both Borderline SMOTE augmentation and TEAUG augmentation at step 8 of the transaction history. More precisely, the model reached an F1 score of 0.857, an Auroc of 0.908, a Precision of 0.889 and a Recall of 0.828. Those metrics are vastly greater than the results gathered by Chen et al. (2018), Chen et al. (2019), Galletta and Pinelli (2023) and Yu et al., (2021), who recorded F1 scores of 0.44, 0.30, 0.62 and 0.79 respectively. The results strongly suggest that the modelling approach used, and the features extracted, can greatly improve the detection power of traditional machine learning classifiers. In this regard, the first researchers used only 7 features, the second group 13, the third 27 and the fourth 14, leading to the conclusion that increasing the set of features available for the classifiers can better capture the fraudulent behavior of Ponzi schemes, thus improving the detection capabilities. Specifically, in my case, introducing 87 features enables the models to properly account for the distribution of a large set of behaviors and patterns characterizing Ponzi smart contracts. Yet, the results gathered are not at the discrimination power frontier as, Yu et al. (2021) and Jin et al. (2022), by exploiting GNNs achieved F1 scores of 0.89 and 0.91 respectively.

In this respect, further experimentations with different GNN structures should be tested and applications of temporal graphs are required, so as to capture the Ponzi nature of certain smart contracts. Specifically, when considering the networks generated by the smart contract and its first order neighbors, star-like graphs are constructed which by themselves do not hold structural characteristics that are relevant in the classification of Ponzi smart contracts. Instead, if a time dimension is incorporated, the redistribution mechanism could be more easily unearthed, possibly positively affecting the classification results.

Importantly, contrary to the bulk of the literature focusing on detecting smart Ponzi schemes, those results are obtained in a timely manner, exploiting only 80 total transactions to reach classification.

5.2 Optimal Transaction Threshold (RQ2)

Considering the second research question, the goal is to identify the model capable of achieving high detection metrics with the lowest number of transactions available. In this regard, the metrics recorded by the optimal gradient boosting model at step 3 with TEAUG and SMOTE oversampling are particularly important. In fact, the model achieved an F1 score of 0.837, an Auroc of 0.884, a Precision score of 0.857 and a Recall of 0.818. Despite those results being slightly worse than those recorded by the best model, special emphasis should be placed on the fact that the latter exploited 50 extra transactions to get to classification. The implications associated with those metrics are that it is possible to detect in a timely manner Ponzi schemes by focusing on the transaction history. Moreover, with only a total of 30 transactions, also including those that resulted in errors, the model could further protect users from being exposed to those frauds, limiting Ponzi schemes growth possibilities and, ultimately, their associated dire financial consequences.

Table 4: *F1 Score across the different tested models*

F1 Score		1		2		3		4		5		6		7		8		9		10	
	Models	No TEAUG	TEAUG	No TEAUG	TEAUG	No TEAUG	TEAUG	No TEAUG	TEAUG	No TEAUG	TEAUG	No TEAUG	TEAUG	No TEAUG	TEAUG	No TEAUG	TEAUG	No TEAUG	TEAUG	No TEAUG	TEAUG
traditional	Logistic	0.256	0.182	0.194	0.100	0.208	0.097	0.189	0.182	0.190	0.146	0.140	0.225	0.189	0.292	0.208	0.253	0.197	0.217	0.184	0.182
	ElasticNet	0.048	0.082	0.046	0.047	0.053	0.023	0.180	0.082	0.063	0.283	0.161	0.073	0.164	0.120	0.225	0.079	0.153	0.205	0.195	0.082
	SVM	0.201	0.228	0.218	0.234	0.221	0.218	0.226	0.228	0.205	0.229	0.233	0.218	0.217	0.207	0.229	0.214	0.208	0.220	0.220	0.228
	RandomForest	0.597	0.795	0.709	0.681	0.727	0.747	0.704	0.795	0.696	0.824	0.684	0.787	0.664	0.807	0.771	0.815	0.702	0.792	0.713	0.795
	GradientBoosting	0.611	0.784	0.687	0.683	0.766	0.765	0.723	0.784	0.682	0.806	0.782	0.800	0.656	0.836	0.745	0.852	0.723	0.792	0.747	0.784
	MLRP	0.319	0.130	0.443	0.510	0.562	0.602	0.667	0.130	0.489	0.714	0.638	0.697	0.562	0.000	0.610	0.676	0.721	0.778	0.615	0.130
	Stacking	0.586	0.750	0.526	0.389	0.763	0.750	0.719	0.750	0.674	0.794	0.719	0.787	0.530	0.836	0.741	0.800	0.517	0.468	0.594	0.750
Random Oversampling	Logistic	0.242	0.262	0.090	0.039	0.229	0.073	0.191	0.152	0.217	0.141	0.222	0.216	0.197	0.276	0.205	0.231	0.131	0.300	0.180	0.262
	ElasticNet	0.057	0.139	0.133	0.146	0.173	0.064	0.117	0.187	0.086	0.202	0.183	0.189	0.170	0.205	0.187	0.245	0.200	0.242	0.195	0.139
	SVM	0.205	0.205	0.216	0.218	0.217	0.222	0.226	0.240	0.239	0.240	0.201	0.218	0.206	0.219	0.198	0.224	0.202	0.205	0.185	0.205
	RandomForest	0.595	0.764	0.706	0.660	0.617	0.710	0.704	0.767	0.793	0.805	0.740	0.789	0.724	0.750	0.782	0.787	0.694	0.786	0.547	0.764
	GradientBoosting	0.622	0.760	0.750	0.707	0.636	0.779	0.797	0.784	0.800	0.844	0.755	0.787	0.687	0.800	0.846	0.807	0.720	0.830	0.597	0.760
	MLRP	0.475	0.688	0.489	0.430	0.521	0.680	0.577	0.438	0.570	0.612	0.633	0.643	0.621	0.735	0.639	0.597	0.539	0.597	0.404	0.688
	Stacking	0.577	0.808	0.725	0.658	0.591	0.821	0.707	0.789	0.737	0.818	0.660	0.787	0.611	0.814	0.821	0.800	0.547	0.769	0.601	0.808
Smote Oversampling	Logistic	0.266	0.269	0.129	0.092	0.180	0.125	0.216	0.194	0.246	0.113	0.199	0.222	0.156	0.283	0.185	0.320	0.228	0.302	0.149	0.269
	ElasticNet	0.096	0.202	0.159	0.000	0.162	0.100	0.142	0.223	0.147	0.170	0.149	0.143	0.133	0.210	0.143	0.182	0.193	0.211	0.132	0.202
	SVM	0.205	0.209	0.219	0.230	0.225	0.238	0.233	0.219	0.203	0.227	0.204	0.215	0.193	0.209	0.211	0.224	0.213	0.229	0.192	0.209
	RandomForest	0.619	0.750	0.680	0.667	0.724	0.737	0.671	0.725	0.672	0.775	0.766	0.771	0.706	0.716	0.700	0.730	0.679	0.759	0.636	0.750
	GradientBoosting	0.677	0.815	0.722	0.787	0.757	0.837	0.701	0.831	0.656	0.841	0.771	0.806	0.723	0.814	0.748	0.842	0.710	0.821	0.625	0.815
	MLRP	0.471	0.698	0.557	0.496	0.505	0.642	0.534	0.653	0.515	0.651	0.582	0.667	0.580	0.667	0.567	0.688	0.493	0.759	0.548	0.698
	Stacking	0.609	0.830	0.707	0.737	0.690	0.814	0.695	0.790	0.602	0.824	0.722	0.781	0.708	0.847	0.721	0.780	0.696	0.780	0.603	0.830
Borderline Smote Oversampling	Logistic	0.227	0.295	0.237	0.118	0.240	0.061	0.165	0.112	0.181	0.146	0.235	0.242	0.228	0.304	0.200	0.357	0.153	0.271	0.160	0.295
	ElasticNet	0.115	0.289	0.144	0.110	0.116	0.053	0.135	0.238	0.119	0.087	0.093	0.196	0.116	0.141	0.149	0.110	0.168	0.220	0.136	0.289
	SVM	0.204	0.209	0.212	0.216	0.206	0.234	0.233	0.245	0.216	0.227	0.195	0.223	0.216	0.213	0.216	0.215	0.180	0.226	0.190	0.209
	RandomForest	0.578	0.750	0.642	0.653	0.728	0.729	0.624	0.719	0.745	0.753	0.649	0.765	0.718	0.727	0.745	0.742	0.745	0.759	0.667	0.750
	GradientBoosting	0.677	0.846	0.713	0.769	0.762	0.814	0.652	0.816	0.752	0.800	0.700	0.794	0.856	0.842	0.831	0.857	0.756	0.836	0.667	0.846
	MLRP	0.372	0.576	0.394	0.391	0.535	0.667	0.512	0.586	0.606	0.699	0.591	0.735	0.679	0.687	0.628	0.677	0.562	0.647	0.515	0.576
	Stacking	0.609	0.815	0.708	0.745	0.717	0.814	0.642	0.482	0.588	0.778	0.622	0.769	0.827	0.828	0.810	0.857	0.748	0.793	0.633	0.815
Adasyn Oversampling	Logistic	0.228	0.190	0.139	0.145	0.194	0.072	0.247	0.191	0.206	0.130	0.130	0.235	0.164	0.278	0.171	0.238	0.138	0.266	0.110	0.190
	ElasticNet	0.164	0.199	0.099	0.131	0.112	0.097	0.107	0.197	0.185	0.169	0.104	0.198	0.174	0.179	0.200	0.186	0.117	0.195	0.142	0.199
	SVM	0.204	0.200	0.220	0.233	0.225	0.233	0.224	0.239	0.222	0.240	0.190	0.217	0.184	0.220	0.197	0.197	0.184	0.199	0.185	0.200
	RandomForest	0.597	0.750	0.682	0.660	0.717	0.714	0.668	0.727	0.672	0.756	0.686	0.761	0.665	0.696	0.695	0.719	0.640	0.733	0.633	0.750
	GradientBoosting	0.676	0.830	0.727	0.753	0.722	0.837	0.705	0.810	0.716	0.829	0.715	0.806	0.709	0.828	0.715	0.842	0.676	0.836	0.607	0.830
	MLRP	0.421	0.447	0.469	0.455	0.540	0.484	0.597	0.559	0.561	0.659	0.609	0.614	0.532	0.686	0.567	0.622	0.579	0.648	0.477	0.447
	Stacking	0.664	0.830	0.710	0.742	0.695	0.837	0.643	0.762	0.719	0.806	0.595	0.806	0.701	0.800	0.722	0.857	0.657	0.836	0.595	0.830
Graph2Vec	Logistic	0.240	0.253	0.078	0.036	0.211	0.040	0.222	0.159	0.188	0.116	0.213	0.243	0.176	0.306	0.188	0.322	0.189	0.264	0.200	0.253
	ElasticNet	0.263	0.223	0.119	0.310	0.087	0.184	0.063	0.164	0.110	0.242	0.141	0.161	0.152	0.200	0.131	0.221	0.051	0.221	0.132	0.223
	SVM	0.201	0.207	0.204	0.234	0.243	0.233	0.230	0.220	0.217	0.227	0.212	0.218	0.206	0.207	0.224	0.201	0.217	0.220	0.230	0.207
	RandomForest	0.563	0.784	0.617	0.645	0.709	0.736	0.787	0.750	0.708	0.800	0.697	0.793	0.759	0.778	0.739	0.792	0.681	0.784	0.797	0.784
	GradientBoosting	0.573	0.833	0.597	0.683	0.738	0.765	0.717	0.784	0.707	0.800	0.642	0.814	0.784	0.836	0.717	0.846	0.707	0.784	0.810	0.833
	MLRP	0.418	0.353	0.574	0.562	0.515	0.621	0.488	0.179	0.478	0.684	0.580	0.638	0.652	0.767	0.635	0.786	0.583	0.800	0.567	0.353
	Stacking	0.564	0.329	0.633	0.707	0.709	0.767	0.567	0.785	0.623	0.800	0.614	0.787	0.622	0.807	0.738	0.325	0.574	0.830	0.724	0.329
Graph Neural Networks	GCN Conv Mean	0.602	0.784	0.612	0.605	0.725	0.673	0.554	0.652	0.571	0.667	0.676	0.648	0.566	0.679	0.642	0.784	0.727	0.769	0.593	0.784
	GAT Conv Mean	0.547	0.741	0.614	0.593	0.681	0.667	0.610	0.643	0.551	0.659	0.667	0.632	0.657	0.667	0.654	0.667	0.714	0.760	0.640	0.741

Table 5: *Auroc Score across the different tested models*

AUROC Score		1		2		3		4		5		6		7		8		9		10	
	Models	No TEAUG	TEAUG	No TEAUG	TEAUG	No TEAUG	TEAUG	No TEAUG	TEAUG	No TEAUG	TEAUG	No TEAUG	TEAUG	No TEAUG	TEAUG	No TEAUG	TEAUG	No TEAUG	TEAUG	No TEAUG	TEAUG
traditional	Logistic	0.577	0.651	0.548	0.519	0.553	0.513	0.530	0.541	0.539	0.519	0.472	0.577	0.551	0.643	0.577	0.600	0.567	0.574	0.543	0.651
	ElasticNet	0.345	0.425	0.340	0.344	0.273	0.337	0.512	0.374	0.274	0.664	0.500	0.382	0.506	0.477	0.610	0.393	0.485	0.570	0.565	0.425
	SVM	0.519	0.590	0.571	0.600	0.564	0.552	0.576	0.570	0.560	0.596	0.643	0.604	0.617	0.589	0.638	0.600	0.591	0.614	0.628	0.590
	RandomForest	0.753	0.863	0.828	0.821	0.841	0.814	0.826	0.849	0.822	0.882	0.821	0.867	0.763	0.876	0.851	0.874	0.835	0.867	0.793	0.863
	GradientBoosting	0.740	0.865	0.769	0.789	0.850	0.844	0.796	0.848	0.784	0.869	0.848	0.868	0.764	0.880	0.825	0.893	0.814	0.867	0.809	0.865
	MLRP	0.609	0.803	0.687	0.732	0.726	0.812	0.777	0.534	0.694	0.833	0.785	0.841	0.716	0.500	0.736	0.881	0.794	0.866	0.736	0.803
	Stacking	0.723	0.865	0.710	0.714	0.846	0.832	0.813	0.852	0.779	0.867	0.812	0.867	0.763	0.880	0.847	0.872	0.702	0.810	0.752	0.865
Random Oversampling	Logistic	0.571	0.632	0.490	0.507	0.568	0.510	0.534	0.530	0.570	0.521	0.589	0.568	0.562	0.622	0.581	0.582	0.441	0.661	0.535	0.632
	ElasticNet	0.351	0.472	0.414	0.426	0.458	0.340	0.373	0.488	0.352	0.543	0.523	0.543	0.496	0.594	0.540	0.675	0.576	0.671	0.558	0.472
	SVM	0.531	0.591	0.564	0.561	0.557	0.559	0.576	0.595	0.634	0.622	0.569	0.614	0.590	0.618	0.571	0.624	0.582	0.587	0.542	0.591
	RandomForest	0.774	0.875	0.853	0.825	0.784	0.852	0.852	0.884	0.859	0.914	0.845	0.919	0.851	0.883	0.910	0.899	0.842	0.882	0.732	0.875
	GradientBoosting	0.742	0.844	0.826	0.801	0.752	0.837	0.861	0.848	0.841	0.873	0.831	0.867	0.798	0.890	0.892	0.887	0.799	0.887	0.720	0.844
	MLRP	0.742	0.878	0.771	0.779	0.752	0.855	0.793	0.717	0.800	0.865	0.810	0.880	0.805	0.894	0.847	0.851	0.756	0.862	0.706	0.878
	Stacking	0.712	0.881	0.795	0.761	0.716	0.861	0.794	0.858	0.799	0.870	0.762	0.867	0.730	0.891	0.863	0.901	0.695	0.850	0.715	0.881
Smote Oversampling	Logistic	0.581	0.638	0.500	0.512	0.530	0.513	0.558	0.548	0.604	0.505	0.559	0.576	0.508	0.638	0.543	0.676	0.611	0.669	0.488	0.638
	ElasticNet	0.404	0.586	0.442	0.338	0.478	0.374	0.417	0.560	0.469	0.476	0.465	0.444	0.434	0.595	0.473	0.525	0.554	0.605	0.415	0.586
	SVM	0.532	0.599	0.570	0.590	0.573	0.595	0.592	0.553	0.556	0.594	0.574	0.598	0.560	0.600	0.601	0.624	0.604	0.632	0.560	0.599
	RandomForest	0.797	0.873	0.847	0.834	0.875	0.875	0.838	0.876	0.858	0.910	0.913	0.903	0.862	0.877	0.830	0.877	0.862	0.878	0.800	0.873
	GradientBoosting	0.795	0.898	0.829	0.863	0.858	0.901	0.836	0.884	0.818	0.896	0.885	0.882	0.821	0.891	0.831	0.907	0.859	0.901	0.780	0.898
	MLRP	0.751	0.880	0.799	0.789	0.811	0.855	0.785	0.852	0.782	0.853	0.827	0.848	0.803	0.882	0.803	0.856	0.807	0.878	0.798	0.880
	Stacking	0.770	0.900	0.826	0.855	0.840	0.888	0.832	0.878	0.799	0.882	0.847	0.879	0.827	0.910	0.827	0.884	0.848	0.896	0.768	0.900
Borderline Smote Oversampling	Logistic	0.565	0.655	0.578	0.510	0.582	0.494	0.510	0.495	0.534	0.519	0.602	0.591	0.605	0.650	0.568	0.730	0.485	0.629	0.505	0.655
	ElasticNet	0.415	0.743	0.444	0.374	0.388	0.314	0.423	0.595	0.375	0.278	0.364	0.559	0.420	0.433	0.474	0.399	0.498	0.622	0.447	0.743
	SVM	0.528	0.594	0.556	0.557	0.529	0.587	0.591	0.607	0.584	0.594	0.555	0.617	0.612	0.607	0.610	0.608	0.526	0.628	0.558	0.594
	RandomForest	0.778	0.873	0.803	0.824	0.856	0.873	0.784	0.866	0.885	0.883	0.830	0.889	0.905	0.879	0.879	0.878	0.861	0.878	0.793	0.873
	GradientBoosting	0.799	0.902	0.793	0.861	0.844	0.888	0.776	0.872	0.863	0.879	0.816	0.881	0.911	0.895	0.890	0.908	0.851	0.903	0.763	0.902
	MLRP	0.676	0.813	0.699	0.735	0.781	0.852	0.739	0.809	0.828	0.874	0.791	0.872	0.889	0.859	0.849	0.854	0.820	0.859	0.741	0.813
	Stacking	0.755	0.898	0.791	0.857	0.834	0.888	0.772	0.762	0.804	0.876	0.770	0.877	0.899	0.893	0.885	0.908	0.850	0.898	0.773	0.898
Adasyn Oversampling	Logistic	0.567	0.554	0.528	0.514	0.530	0.483	0.590	0.537	0.560	0.502	0.462	0.590	0.505	0.640	0.521	0.591	0.473	0.648	0.507	0.554
	ElasticNet	0.480	0.580	0.377	0.413	0.395	0.313	0.417	0.504	0.514	0.471	0.350	0.565	0.516	0.529	0.577	0.539	0.415	0.565	0.460	0.580
	SVM	0.528	0.581	0.576	0.597	0.574	0.586	0.572	0.594	0.601	0.622	0.542	0.610	0.536	0.620	0.570	0.569	0.534	0.575	0.544	0.581
	RandomForest	0.789	0.873	0.815	0.833	0.881	0.871	0.863	0.867	0.845	0.906	0.874	0.902	0.868	0.874	0.910	0.875	0.844	0.874	0.805	0.873
	GradientBoosting	0.800	0.900	0.821	0.858	0.860	0.901	0.848	0.881	0.843	0.895	0.826	0.882	0.825	0.893	0.857	0.907	0.808	0.903	0.755	0.900
	MLRP	0.719	0.777	0.749	0.779	0.824	0.777	0.850	0.835	0.792	0.876	0.837	0.873	0.796	0.872	0.863	0.857	0.841	0.873	0.759	0.777
	Stacking	0.800	0.900	0.820	0.839	0.846	0.901	0.806	0.873	0.851	0.891	0.779	0.882	0.824	0.890	0.867	0.908	0.802	0.903	0.750	0.900
Graph2Vec	Logistic	0.570	0.609	0.476	0.500	0.555	0.504	0.565	0.535	0.540	0.509	0.581	0.585	0.533	0.643	0.548	0.661	0.549	0.618	0.576	0.609
	ElasticNet	0.637	0.628	0.414	0.678	0.361	0.506	0.315	0.472	0.436	0.610	0.469	0.482	0.497	0.580	0.465	0.621	0.338	0.621	0.474	0.628
	SVM	0.519	0.590	0.534	0.600	0.614	0.583	0.587	0.555	0.588	0.594	0.594	0.604	0.589	0.589	0.629	0.573	0.614	0.614	0.648	0.590
	RandomForest	0.747	0.863	0.778	0.798	0.841	0.848	0.884	0.852	0.838	0.856	0.841	0.854	0.856	0.845	0.858	0.857	0.817	0.852	0.939	0.863
	GradientBoosting	0.727	0.868	0.738	0.789	0.828	0.844	0.792	0.848	0.804	0.856	0.759	0.870	0.826	0.880	0.805	0.877	0.789	0.852	0.853	0.868
	MLRP	0.663	0.609	0.729	0.744	0.725	0.825	0.722	0.546	0.686	0.838	0.729	0.819	0.764	0.871	0.785	0.870	0.749	0.883	0.728	0.609
	Stacking	0.724	0.752	0.783	0.801	0.816	0.862	0.738	0.867	0.770	0.856	0.788	0.867	0.746	0.876	0.847	0.749	0.763	0.887	0.849	0.752
Graph Neural Networks	GCN Conv Mean	0.811	0.861	0.814	0.845	0.865	0.878	0.768	0.880	0.848	0.869	0.831	0.867	0.801	0.858	0.729	0.869	0.899	0.862	0.763	0.861
	GAT Conv Mean	0.816	0.902	0.794	0.862	0.863	0.848	0.727	0.845	0.857	0.878	0.784	0.871	0.804	0.864	0.705	0.859	0.893	0.861	0.742	0.902

Table 6: *Precision Score across the different tested models*

Precision Score		1		2		3		4		5		6		7		8		9		10	
	Models	No TEAUG	No TEAUG	TEAUG	No TEAUG	TEAUG	No TEAUG	TEAUG	No TEAUG	TEAUG	No TEAUG	TEAUG	No TEAUG	TEAUG	No TEAUG	TEAUG	No TEAUG	TEAUG	No TEAUG	TEAUG	No TEAUG	TEAUG
traditional	Logistic	0.577	0.548	0.519	0.553	0.513	0.530	0.541	0.539	0.519	0.472	0.577	0.551	0.643	0.577	0.600	0.567	0.574	0.543	0.651
	ElasticNet	0.345	0.340	0.344	0.273	0.337	0.512	0.374	0.274	0.664	0.500	0.382	0.506	0.477	0.610	0.393	0.485	0.570	0.565	0.425
	SVM	0.519	0.571	0.600	0.564	0.552	0.576	0.570	0.560	0.596	0.643	0.604	0.617	0.589	0.638	0.600	0.591	0.614	0.628	0.590
	RandomForest	0.753	0.828	0.821	0.841	0.814	0.826	0.849	0.822	0.882	0.821	0.867	0.763	0.876	0.851	0.874	0.835	0.867	0.793	0.863
	GradientBoosting	0.740	0.769	0.789	0.850	0.844	0.796	0.848	0.784	0.869	0.848	0.868	0.764	0.880	0.825	0.893	0.814	0.867	0.809	0.865
	MLRP	0.609	0.687	0.732	0.726	0.812	0.777	0.534	0.694	0.833	0.785	0.841	0.716	0.500	0.736	0.881	0.794	0.866	0.736	0.803
	Stacking	0.723	0.710	0.714	0.846	0.832	0.813	0.852	0.779	0.867	0.812	0.867	0.763	0.880	0.847	0.872	0.702	0.810	0.752	0.865
Random Oversampling	Logistic	0.445	0.080	0.250	0.220	0.182	0.134	0.200	0.149	0.143	0.142	0.190	0.122	0.211	0.118	0.184	0.077	0.208	0.105	0.175
	ElasticNet	0.035	0.079	0.086	0.097	0.040	0.069	0.109	0.049	0.115	0.104	0.109	0.094	0.114	0.103	0.139	0.112	0.138	0.110	0.084
	SVM	0.115	0.122	0.124	0.124	0.128	0.129	0.139	0.136	0.139	0.114	0.123	0.116	0.126	0.110	0.129	0.113	0.116	0.103	0.116
	RandomForest	0.595	0.666	0.623	0.620	0.673	0.664	0.733	0.866	0.756	0.770	0.718	0.716	0.706	0.721	0.750	0.669	0.786	0.609	0.750
	GradientBoosting	0.841	0.860	0.829	0.829	0.909	0.870	0.879	0.960	0.964	0.857	0.828	0.782	0.800	0.906	0.821	0.886	0.880	0.910	0.826
	MLRP	0.402	0.389	0.296	0.475	0.607	0.511	0.359	0.494	0.484	0.604	0.519	0.591	0.658	0.555	0.479	0.523	0.469	0.332	0.595
	Stacking	0.870	0.939	0.862	0.915	0.941	0.870	0.857	0.964	0.900	0.876	0.828	0.890	0.828	0.941	0.774	0.915	0.833	1.000	0.840
Smote Oversampling	Logistic	0.421	0.108	0.167	0.136	0.139	0.153	0.226	0.164	0.114	0.132	0.172	0.099	0.203	0.109	0.225	0.140	0.205	0.090	0.182
	ElasticNet	0.055	0.092	0.000	0.092	0.061	0.084	0.126	0.083	0.098	0.085	0.082	0.076	0.121	0.081	0.100	0.109	0.118	0.075	0.113
	SVM	0.115	0.123	0.131	0.128	0.137	0.134	0.128	0.117	0.132	0.117	0.124	0.110	0.118	0.120	0.129	0.120	0.132	0.109	0.118
	RandomForest	0.594	0.624	0.618	0.662	0.686	0.622	0.660	0.592	0.705	0.685	0.711	0.654	0.649	0.711	0.676	0.603	0.733	0.638	0.724
	GradientBoosting	0.755	0.766	0.833	0.770	0.857	0.693	0.889	0.633	0.879	0.744	0.833	0.800	0.828	0.838	0.857	0.669	0.821	0.662	0.815
	MLRP	0.380	0.473	0.378	0.374	0.538	0.443	0.561	0.421	0.560	0.482	0.600	0.510	0.556	0.487	0.629	0.369	0.733	0.463	0.611
	Stacking	0.654	0.734	0.729	0.675	0.833	0.685	0.800	0.562	0.875	0.721	0.781	0.744	0.862	0.773	0.767	0.661	0.742	0.645	0.846
Borderline Smote Oversampling	Logistic	0.391	0.183	0.132	0.168	0.091	0.118	0.104	0.124	0.130	0.153	0.186	0.143	0.226	0.122	0.241	0.091	0.191	0.097	0.206
	ElasticNet	0.065	0.084	0.066	0.066	0.033	0.081	0.136	0.068	0.050	0.054	0.112	0.068	0.079	0.086	0.064	0.094	0.125	0.078	0.169
	SVM	0.114	0.119	0.123	0.117	0.133	0.134	0.142	0.124	0.132	0.111	0.128	0.122	0.121	0.123	0.122	0.101	0.130	0.107	0.121
	RandomForest	0.543	0.633	0.611	0.710	0.673	0.639	0.667	0.689	0.707	0.597	0.722	0.614	0.667	0.702	0.697	0.740	0.733	0.742	0.724
	GradientBoosting	0.738	0.890	0.795	0.826	0.833	0.752	0.886	0.750	0.824	0.750	0.806	0.881	0.889	0.872	0.889	0.795	0.852	0.891	0.880
	MLRP	0.301	0.306	0.270	0.451	0.586	0.481	0.500	0.520	0.617	0.553	0.694	0.568	0.622	0.533	0.611	0.459	0.550	0.493	0.487
	Stacking	0.752	0.884	0.745	0.748	0.833	0.737	0.380	0.533	0.778	0.718	0.758	0.843	0.857	0.841	0.889	0.778	0.767	0.717	0.815
Adasyn Oversampling	Logistic	0.232	0.167	0.127	0.125	0.077	0.169	0.149	0.131	0.107	0.081	0.171	0.097	0.192	0.103	0.182	0.084	0.170	0.102	0.121
	ElasticNet	0.092	0.058	0.079	0.065	0.057	0.062	0.113	0.104	0.097	0.059	0.110	0.099	0.102	0.111	0.103	0.066	0.108	0.080	0.111
	SVM	0.114	0.124	0.133	0.127	0.132	0.127	0.138	0.126	0.139	0.107	0.122	0.102	0.126	0.109	0.110	0.103	0.112	0.104	0.112
	RandomForest	0.559	0.699	0.607	0.638	0.648	0.574	0.681	0.614	0.674	0.597	0.692	0.570	0.615	0.571	0.657	0.559	0.688	0.616	0.724
	GradientBoosting	0.730	0.809	0.761	0.686	0.857	0.673	0.842	0.712	0.853	0.761	0.833	0.750	0.857	0.681	0.857	0.715	0.852	0.706	0.846
	MLRP	0.324	0.378	0.327	0.413	0.369	0.473	0.429	0.487	0.545	0.513	0.482	0.435	0.600	0.427	0.511	0.462	0.535	0.387	0.328
	Stacking	0.700	0.762	0.786	0.655	0.857	0.630	0.744	0.702	0.806	0.629	0.833	0.732	0.800	0.679	0.889	0.680	0.852	0.686	0.846
Graph2Vec	Logistic	0.450	0.071	0.111	0.169	0.167	0.148	0.227	0.126	0.121	0.131	0.214	0.110	0.236	0.110	0.241	0.109	0.190	0.115	0.183
	ElasticNet	0.159	0.073	0.199	0.053	0.114	0.038	0.101	0.070	0.148	0.086	0.092	0.094	0.112	0.081	0.126	0.031	0.126	0.083	0.128
	SVM	0.112	0.114	0.134	0.139	0.135	0.131	0.129	0.123	0.132	0.121	0.126	0.117	0.120	0.127	0.117	0.124	0.127	0.131	0.119
	RandomForest	0.600	0.644	0.652	0.698	0.744	0.781	0.769	0.710	0.897	0.679	0.885	0.786	0.875	0.776	0.875	0.700	0.870	0.705	0.833
	GradientBoosting	0.721	0.763	0.800	0.815	0.838	0.915	0.879	0.825	0.897	0.810	0.889	0.981	0.920	0.849	0.957	0.894	0.870	0.942	0.952
	MLRP	0.505	0.712	0.595	0.568	0.542	0.499	0.333	0.645	0.650	0.755	0.595	0.836	0.767	0.714	0.815	0.660	0.815	0.767	0.857
	Stacking	0.705	0.703	0.829	0.776	0.786	0.600	0.816	0.736	0.897	0.665	0.828	0.884	0.852	0.766	0.200	0.607	0.880	0.723	0.204
Graph Neural Networks	GCN Conv Mean	0.718	0.722	0.667	0.717	0.596	0.667	0.604	0.667	0.667	0.639	0.590	0.682	0.783	0.773	0.909	0.645	0.833	0.571	0.833
	GAT Conv Mean	0.634	0.692	0.614	0.653	0.674	0.857	0.628	0.463	0.587	0.818	0.545	0.611	0.667	0.810	0.773	0.833	0.864	0.667	0.741

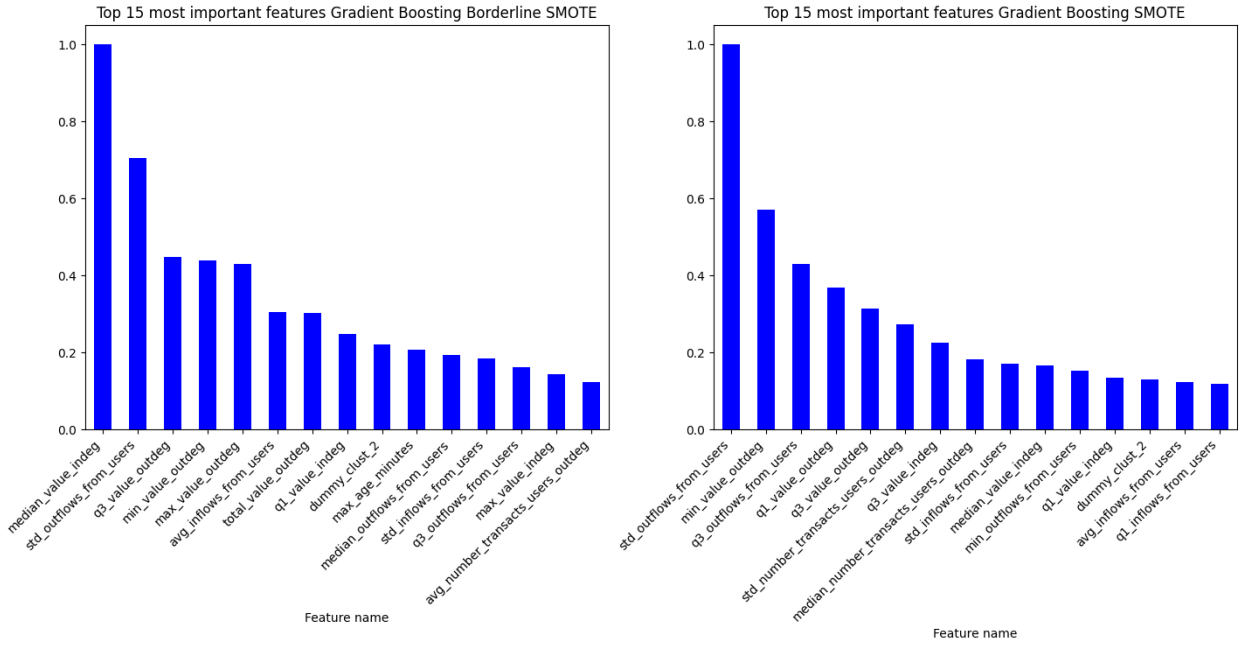
Table 7: *Recall Score across the different tested models*

Recall Score		1		2		3		4		5		6		7		8		9		10	
	Models	No TEAUG	No TEAUG	TEAUG	No TEAUG	TEAUG	No TEAUG	TEAUG	No TEAUG	TEAUG	No TEAUG	TEAUG	No TEAUG	TEAUG	No TEAUG	TEAUG	No TEAUG	TEAUG	No TEAUG	TEAUG	No TEAUG	TEAUG
traditional	Logistic	0.577	0.548	0.519	0.553	0.513	0.530	0.541	0.539	0.519	0.472	0.577	0.551	0.643	0.577	0.600	0.567	0.574	0.543	0.651
	ElasticNet	0.345	0.340	0.344	0.273	0.337	0.512	0.374	0.274	0.664	0.500	0.382	0.506	0.477	0.610	0.393	0.485	0.570	0.565	0.425
	SVM	0.519	0.571	0.600	0.564	0.552	0.576	0.570	0.560	0.596	0.643	0.604	0.617	0.589	0.638	0.600	0.591	0.614	0.628	0.590
	RandomForest	0.753	0.828	0.821	0.841	0.814	0.826	0.849	0.822	0.882	0.821	0.867	0.763	0.876	0.851	0.874	0.835	0.867	0.793	0.863
	GradientBoosting	0.740	0.769	0.789	0.850	0.844	0.796	0.848	0.784	0.869	0.848	0.868	0.764	0.880	0.825	0.893	0.814	0.867	0.809	0.865
	MLRP	0.609	0.687	0.732	0.726	0.812	0.777	0.534	0.694	0.833	0.785	0.841	0.716	0.500	0.736	0.881	0.794	0.866	0.736	0.803
	Stacking	0.723	0.710	0.714	0.846	0.832	0.813	0.852	0.779	0.867	0.812	0.867	0.763	0.880	0.847	0.872	0.702	0.810	0.752	0.865
Random Oversampling	Logistic	0.167	0.104	0.021	0.242	0.045	0.335	0.122	0.406	0.139	0.519	0.250	0.500	0.400	0.752	0.310	0.443	0.536	0.644	0.519
	ElasticNet	0.149	0.435	0.489	0.777	0.159	0.390	0.683	0.349	0.833	0.813	0.688	0.867	1.000	0.986	1.000	0.957	1.000	0.896	0.407
	SVM	0.967	0.939	0.872	0.874	0.818	0.900	0.854	0.966	0.889	0.844	0.969	0.880	0.867	0.986	0.862	0.993	0.893	0.926	0.889
	RandomForest	0.596	0.752	0.702	0.614	0.750	0.750	0.805	0.731	0.861	0.713	0.875	0.733	0.800	0.855	0.828	0.721	0.786	0.496	0.778
	GradientBoosting	0.495	0.665	0.617	0.516	0.682	0.735	0.707	0.686	0.750	0.675	0.750	0.613	0.800	0.793	0.793	0.607	0.786	0.444	0.704
	MLRP	0.596	0.674	0.787	0.586	0.773	0.665	0.561	0.680	0.833	0.669	0.844	0.660	0.833	0.759	0.793	0.571	0.821	0.533	0.815
	Stacking	0.433	0.596	0.532	0.437	0.727	0.600	0.732	0.600	0.750	0.531	0.750	0.467	0.800	0.731	0.828	0.393	0.714	0.430	0.778
Smote Oversampling	Logistic	0.196	0.161	0.064	0.270	0.114	0.365	0.171	0.491	0.111	0.406	0.313	0.367	0.467	0.621	0.552	0.636	0.571	0.430	0.519
	ElasticNet	0.433	0.587	0.000	0.716	0.273	0.480	0.976	0.663	0.639	0.650	0.563	0.547	0.800	0.607	1.000	0.857	1.000	0.570	0.963
	SVM	0.978	0.957	0.936	0.921	0.909	0.910	0.756	0.783	0.806	0.794	0.813	0.780	0.900	0.897	0.862	0.971	0.857	0.807	0.889
	RandomForest	0.647	0.748	0.723	0.800	0.795	0.730	0.805	0.777	0.861	0.869	0.844	0.767	0.800	0.690	0.793	0.779	0.786	0.637	0.778
	GradientBoosting	0.615	0.683	0.745	0.744	0.818	0.710	0.780	0.680	0.806	0.800	0.781	0.660	0.800	0.676	0.828	0.757	0.821	0.593	0.815
	MLRP	0.625	0.696	0.723	0.786	0.795	0.675	0.780	0.669	0.778	0.744	0.750	0.673	0.833	0.683	0.759	0.750	0.786	0.681	0.815
	Stacking	0.589	0.683	0.745	0.744	0.795	0.705	0.780	0.663	0.778	0.725	0.781	0.680	0.833	0.676	0.793	0.736	0.821	0.570	0.815
Borderline Smote Oversampling	Logistic	0.160	0.339	0.106	0.423	0.045	0.280	0.122	0.337	0.167	0.506	0.344	0.573	0.467	0.559	0.690	0.500	0.464	0.467	0.519
	ElasticNet	0.502	0.526	0.340	0.460	0.136	0.415	0.976	0.463	0.361	0.313	0.813	0.427	0.633	0.572	0.379	0.793	0.929	0.556	1.000
	SVM	1.000	0.978	0.851	0.879	0.977	0.900	0.902	0.834	0.806	0.788	0.875	0.940	0.900	0.897	0.897	0.857	0.857	0.852	0.778
	RandomForest	0.618	0.652	0.702	0.749	0.795	0.610	0.780	0.811	0.806	0.713	0.813	0.867	0.800	0.793	0.793	0.750	0.786	0.607	0.778
	GradientBoosting	0.625	0.596	0.745	0.707	0.795	0.575	0.756	0.754	0.778	0.656	0.781	0.833	0.800	0.793	0.828	0.721	0.821	0.533	0.815
	MLRP	0.495	0.609	0.702	0.665	0.773	0.555	0.707	0.737	0.806	0.638	0.781	0.847	0.767	0.772	0.759	0.736	0.786	0.541	0.704
	Stacking	0.531	0.591	0.745	0.702	0.795	0.570	0.659	0.703	0.778	0.569	0.781	0.813	0.800	0.786	0.828	0.721	0.821	0.570	0.815
Adasyn Oversampling	Logistic	0.229	0.139	0.170	0.428	0.068	0.465	0.268	0.491	0.167	0.331	0.375	0.527	0.500	0.497	0.345	0.386	0.607	0.156	0.444
	ElasticNet	0.764	0.348	0.404	0.409	0.341	0.390	0.756	0.817	0.667	0.438	1.000	0.713	0.733	0.966	0.966	0.500	0.964	0.637	0.963
	SVM	0.989	1.000	0.957	0.963	0.977	0.925	0.878	0.937	0.889	0.813	0.969	0.967	0.867	1.000	0.931	0.893	0.929	0.852	0.926
	RandomForest	0.640	0.665	0.723	0.819	0.795	0.800	0.780	0.743	0.861	0.806	0.844	0.800	0.800	0.890	0.793	0.750	0.786	0.652	0.778
	GradientBoosting	0.629	0.661	0.745	0.763	0.818	0.740	0.780	0.720	0.806	0.675	0.781	0.673	0.800	0.752	0.828	0.643	0.821	0.533	0.815
	MLRP	0.647	0.622	0.745	0.791	0.705	0.815	0.805	0.663	0.833	0.750	0.844	0.687	0.800	0.848	0.793	0.779	0.821	0.622	0.704
	Stacking	0.633	0.665	0.702	0.740	0.818	0.660	0.780	0.737	0.806	0.594	0.781	0.673	0.800	0.772	0.828	0.636	0.821	0.526	0.815
Graph2Vec	Logistic	0.164	0.087	0.021	0.279	0.023	0.450	0.122	0.371	0.111	0.563	0.281	0.433	0.433	0.655	0.483	0.714	0.429	0.778	0.407
	ElasticNet	0.764	0.326	0.702	0.233	0.477	0.175	0.439	0.257	0.667	0.406	0.625	0.400	0.933	0.345	0.897	0.143	0.893	0.333	0.889
	SVM	1.000	1.000	0.957	0.977	0.864	0.950	0.732	0.914	0.806	0.875	0.813	0.867	0.767	0.931	0.724	0.893	0.821	0.963	0.778
	RandomForest	0.538	0.596	0.638	0.721	0.727	0.795	0.732	0.709	0.722	0.719	0.719	0.733	0.700	0.745	0.724	0.664	0.714	0.919	0.741
	GradientBoosting	0.476	0.496	0.596	0.674	0.705	0.590	0.707	0.623	0.722	0.531	0.750	0.653	0.767	0.621	0.759	0.586	0.714	0.711	0.741
	MLRP	0.382	0.483	0.532	0.474	0.727	0.480	0.122	0.383	0.722	0.475	0.688	0.540	0.767	0.600	0.759	0.529	0.786	0.474	0.222
	Stacking	0.473	0.604	0.617	0.656	0.750	0.540	0.756	0.566	0.722	0.613	0.750	0.500	0.767	0.717	0.862	0.571	0.786	0.733	0.852
Graph Neural Networks	GCN Conv Mean	0.519	0.531	0.553	0.733	0.773	0.474	0.707	0.500	0.667	0.719	0.719	0.484	0.600	0.548	0.690	0.833	0.714	0.615	0.741
	GAT Conv Mean	0.481	0.551	0.574	0.711	0.659	0.474	0.659	0.679	0.750	0.563	0.750	0.710	0.667	0.548	0.586	0.625	0.679	0.615	0.741

5.3 Most important classification features (RQ3)

Considering the third research question, the goal is to identify the most important features that are useful to distinguish between Ponzi smart contracts and non-Ponzi ones. In this regard, the figure below reports the top 15 features that are used by the best overall model and the best early warning one.

Figure 15: *Top 15 most important features identified*



Starting with the best overall model, 4 different families of features are identified, relating to: the contract value in in-degrees/out-degrees, the outflows/inflows from users, the contract name, and the time dimension. To better understand how Ponzi and non-Ponzi smart contracts differ on those characteristics, table 8 reports, for each feature, the mean, standard deviation, 1st quartile, median and 3rd quartile across the two groups.

The majority of the features identified (8/15) pertain to the first family, involving the contract value observed in in-degrees and out-degrees. In this regard, by focusing on the in-degree features, it is possible to determine that Ponzi smart contracts tend to have median, q1 and max values greater than non-Ponzi ones, with an associated low variability.

Overall, this result is in line with the continuous need for increasing investments by new users, a general trait common to all Ponzi schemes highlighted in the literature review section. Similarly, the features in the out-degrees all highlight the redistributive nature that characterizes Ponzi smart contracts. More precisely, all the available features showcase a higher Q1, median and Q3 compared to non-Ponzi smart contracts with lower standard deviation.

Next, other relevant characteristics to distinguish between Ponzi and non-Ponzi smart contracts are identified when, instead of grouping the transactions by contract, emphasis on the users is placed. In fact, Ponzi participants in the first 80 transactions tend to receive much larger payouts and more receiving transactions compared to non-Ponzi ones, albeit showcasing a higher standard deviation. This result is expected as Ponzi schemes thrive on distributing new users' investments to existing participants. Similarly, the higher standard deviation is a result of the payout structure which typically rewards better early joiners. Nonetheless, although Ponzi schemes showcase higher payouts, by focusing on the pay-ins by users, the nature of the frauds is identifiable. Specifically, by comparing Q1 and the median of the median_outflows_from_users, representing the payouts received by users, and the average_inflows_from_users, representing pay-ins made by investors, it is possible to notice that the payouts are significantly lower than the pay-ins. This result holds true for the majority of Ponzi participants but not all, as Q3 shows higher payouts than pay-ins, indicating that a fraction of the investors, likely the first joiners, can attain positive and significant profits.
Comparing the pay-ins between Ponzi and Non-Ponzi participants, it is possible to notice that the former category requires larger investments than the latter. Yet, a higher standard deviation is detected for Ponzi frauds, in line with the requirement of continuous and growing investments required to sustain it.

The remaining important features employed by the classifier relate to the name cluster and the maximum time in between transactions. The former indicates that cluster 2, which includes ether games and token proxies, is more populated by non-Ponzi smart contracts compared to fraudulent ones. The latter showcases that Ponzi schemes tend to be more active as indicated by the lower maximum time elapsed between the last transaction available and the contract creation timestamp. Specifically, by comparing the median, the last Ponzi transaction, ceteris paribus, took place 8 hours before the comparable non-Ponzi one.

Table 8: *Descriptive Statistics Optimal Model Features*

Contract	Feature	Mean	STD	Q1	Median	Q3
Ponzi	median_value_indeg	40.75	92.44	0.70	8.31	35.07
Non-Ponzi	median_value_indeg	931.07	19,063.90	0.00	0.00	35.58
Ponzi	std_outflows_from_users	172.03	480.78	0.00	21.09	135.51
Non-Ponzi	std_outflows_from_users	10,191.78	203,355.22	0.00	0.00	0.00
Ponzi	q3_value_outdeg	242.72	1,137.49	1.39	14.18	103.40
Non-Ponzi	q3_value_outdeg	6,694.26	106,953.30	0.00	0.00	1.02
Ponzi	min_value_outdeg	113.77	989.06	0.00	0.23	2.70
Non-Ponzi	min_value_outdeg	1,079.59	42,986.67	0.00	0.00	0.00
Ponzi	max_value_outdeg	739.38	4,099.96	3.95	64.94	319.21
Non-Ponzi	max_value_outdeg	34,594.40	391,327.73	0.00	0.00	2.21
Ponzi	avg_inflows_from_users	-369.92	1,241.05	-267.36	-92.45	-27.84
Non-Ponzi	avg_inflows_from_users	-32,948.55	486,296.77	-907.73	-33.68	0.00
Ponzi	total_value_outdeg	1,546.54	7,535.72	12.14	161.16	971.10
Non-Ponzi	total_value_outdeg	66,839.69	696,303.10	0.00	0.00	4.55
Ponzi	q1_value_indeg	16.75	54.59	0.00	0.55	10.17
Non-Ponzi	q1_value_indeg	334.43	6,144.18	0.00	0.00	2.87
Ponzi	dummy_clust_2	0.24	0.43	0.00	0.00	0.00
Non-Ponzi	dummy_clust_2	0.43	0.49	0.00	0.00	1.00
Ponzi	max_age_minutes	747.16	580.99	110.53	820.47	1,368.79
Non-Ponzi	max_age_minutes	993.62	508.97	540.30	1,282.37	1,409.03
Ponzi	median_outflows_from_users	863.86	7,415.03	2.05	18.70	98.18
Non-Ponzi	median_outflows_from_users	43,085.52	502,509.85	0.00	0.00	1.47
Ponzi	std_inflows_from_users	613.68	2,393.60	23.62	92.70	337.88
Non-Ponzi	std_inflows_from_users	43,495.93	641,811.58	0.00	6.97	1,243.60
Ponzi	q3_outflows_from_users	950.02	7,416.57	5.32	39.01	209.63
Non-Ponzi	q3_outflows_from_users	49,300.60	530,483.37	0.00	0.00	2.66
Ponzi	max_value_indeg	2,206.62	10,378.49	48.62	201.60	915.78
Non-Ponzi	max_value_indeg	63,352.99	630,717.15	0.00	44.74	3,629.36
Ponzi	avg_number_transacts_users_outdeg	2.62	3.78	1.00	1.44	3.33
Non-Ponzi	avg_number_transacts_users_outdeg	2.11	5.83	0.00	0.00	1.00

Further, when comparing the two distributions reported in figure 15, it is worth noting the similarities in the features used for classification. In fact, out of the top 15 features reported, 9 of them, pertaining to the contract value in in-degrees/out-degrees, the outflows/inflows from users and the contract name, are common between the two. This result indicates that even when fewer transactions are considered for classification, common traits are displayed by Ponzi smart contracts.

To further comprehend how the two types of smart contracts differ, table 9 reports, for each feature employed by the best early warning model, the mean, standard deviation, 1st quartile, median and 3rd quartile across the two smart contracts categories. Similar to the other model considered, three main families of features are employed, relating to: the contract value in in-degrees/out-degrees, the outflows/inflows from users and the contract name. By focusing on the contract value in in-degrees and out-degrees, it is possible to observe the higher contributions that have been made towards fraudulent contracts, as noted by all the indeg metrics, and their redistributive nature shown by the outdeg ones. Importantly, significant behavioral differences in terms of how the contract acts are already distinguishable and are adequately valued by the classifier even when a small set of transactions is considered. Moreover, associated to the higher outflows from the contract, it is also possible to observe a greater number of individuals receiving paybacks, underlying the importance of repaying users early on to allow the fraud to grow.

Unlike the top available model, when only 30 transactions are considered, the classifier places greater emphasis on user level characteristics. More precisely, a total of 8 features involving investors are exploited, contrary to the 6 used by the other model. Focusing on the payouts received by individuals, when compared to non-Ponzi smart contracts, Ponzi participants tend to receive both greater payouts, as shown by min_outflows_from_users, q1_outflows_from_users and q3_outflows_from_users but are exposed to significantly higher variability. The first result can be explained by the fact that early individuals must receive payouts in order to guarantee the longevity of the scheme, whereas the second indicates that even when only 30 transactions are exploited, there is already significant unfairness in the redistribution of the funds attracted. The pay-ins figures instead indicate that there is greater

variability in the contributions to Ponzi smart contracts with also greater investments made by individuals.

Finally, the remaining feature considered relates to the contract name, indicating, as before, that cluster 2, which includes ether games and token proxies, is more populated by non-Ponzi smart contracts.

Table 9: *Descriptive Statistics Early Model Features*

Contract	Feature	Mean	STD	Q1	Median	Q3
Ponzi	std_outflows_from_users	84.93	256.37	0.00	3.13	47.60
Non-Ponzi	std_outflows_from_users	4,902.83	129,732.77	0.00	0.00	0.00
Ponzi	min_value_outdeg	99.77	861.23	0.00	0.45	5.67
Non-Ponzi	min_value_outdeg	622.55	19,128.92	0.00	0.00	0.00
Ponzi	q3_outflows_from_users	371.53	2,535.34	0.02	17.55	113.33
Non-Ponzi	q3_outflows_from_users	27,476.71	370,816.65	0.00	0.00	0.93
Ponzi	q1_value_outdeg	112.54	866.11	0.00	1.71	10.90
Non-Ponzi	q1_value_outdeg	1,703.95	36,087.71	0.00	0.00	0.05
Ponzi	q3_value_outdeg	181.66	909.87	0.00	8.56	64.74
Non-Ponzi	q3_value_outdeg	5,604.74	91,411.46	0.00	0.00	0.31
Ponzi	std_number_transacts_users_outdeg	0.82	1.44	0.00	0.00	1.08
Non-Ponzi	std_number_transacts_users_outdeg	0.23	1.07	0.00	0.00	0.00
Ponzi	q3_value_indeg	110.52	212.18	6.35	25.42	98.19
Non-Ponzi	q3_value_indeg	3,364.38	71,643.88	0.00	0.00	127.10
Ponzi	median_number_transacts_users_outdeg	1.59	1.88	1.00	1.00	2.00
Non-Ponzi	median_number_transacts_users_outdeg	1.05	2.62	0.00	0.00	1.00
Ponzi	std_inflows_from_users	394.56	1,734.60	8.55	66.46	220.44
Non-Ponzi	std_inflows_from_users	29,539.10	504,804.77	0.00	0.42	513.49
Ponzi	median_value_indeg	38.86	85.67	1.07	8.92	34.22
Non-Ponzi	median_value_indeg	1,158.87	27,004.71	0.00	0.00	19.67
Ponzi	min_outflows_from_users	277.29	2,523.21	0.00	2.73	21.81
Non-Ponzi	min_outflows_from_users	22,812.95	348,809.37	0.00	0.00	0.09
Ponzi	q1_value_indeg	17.83	57.65	0.00	0.99	10.24
Non-Ponzi	q1_value_indeg	388.81	8,582.59	0.00	0.00	1.04
Ponzi	dummy_clust_2	0.23	0.42	0.00	0.00	0.00
Non-Ponzi	dummy_clust_2	0.41	0.49	0.00	0.00	1.00
Ponzi	avg_inflows_from_users	-301.22	1,204.27	-224.10	-75.86	-19.09
Non-Ponzi	avg_inflows_from_users	-26,843.80	494,002.07	-569.60	-20.28	0.00
Ponzi	q1_inflows_from_users	-360.14	1,749.11	-242.78	-89.02	-18.95
Non-Ponzi	q1_inflows_from_users	-34,459.54	615,822.76	-556.39	-19.90	0.00

To conclude, the results clearly indicate that it is possible to detect Ponzi schemes early on by focusing on the transaction history. In fact, satisfactory detection power is observable even when only 30 transactions are fed to the classifiers. Further, by focusing on the characteristics used by the gradient boosting model to discern between the two categories of smart contracts, important insights into the behavioral patterns of Ponzi smart contracts are provided.

The next chapter will summarize the thesis and conclude.

Chapter 6 – Conclusion

The goal of the thesis is to document the current situation of Ponzi schemes on Ethereum and propose a framework capable of identifying those frauds early on, so as to protect investors. This is achieved by training the different tested models on features extracted from a subset of transactions. Specifically, 10 transaction steps, each made by 10 transactions are defined, and 87 different features, relating to the number of transactions, their value, errors, age, transaction costs and smart contract name, are computed at each step. To limit the class imbalance problem detected and to avoid having too few smart contracts on which to train the classifiers, two different data augmentation schemes have been applied, namely: minority class oversampling and TEAUG. The first involves testing different oversampling strategies applied to Ponzi smart contracts, that is: Random, SMOTE, Borderline SMOTE and Adasyn oversampling. The second, instead, is a data augmentation procedure first suggested by Jin et al. (2022) which entails carrying over the smart contracts of the previous transaction steps to increase the available observations employed by the models.

Overall, the findings indicate that the proposed early detection framework provides satisfactory detection capabilities, as seen by the performance of the best overall model available. The latter, which is a Gradient Boosting model combining both Borderline SMOTE and TEAUG, in fact, by exploiting 80 transactions to perform the feature extraction procedure, attained an F1 score of 0.857, an Auroc of 0.908, a Precision of 0.889 and a Recall of 0.828. Furthermore, the results suggest that it is possible to signal the presence of Ponzi schemes even with fewer transactions, as highlighted by the best early detection model. More precisely, by exploiting only 30 total transactions, a Gradient Boosting classifier, with both SMOTE and TEAUG, can achieve an F1 score of 0.837, an Auroc of 0.884, a Precision score of 0.857 and a Recall of

0.818. Finally, by investigating the importance attributed to the different covariates by both the best overall model and the best early warning one, important implications can be drawn. Specifically, 3 main families of features are common across the two models, namely: the contract value in in-degrees/out-degrees, the outflows/inflows from users and the contract name. Moreover, as the number of transactions used for the feature extraction procedure increases, the model shifts its main focus from the outflows/inflows from users to the contract value in in-degrees/out-degrees. Furthermore, two main patterns can be easily distinguished in Ponzi smart contracts, namely: the continuous need for increasing investments and the redistributive nature. The former can be identified by considering the in-degree features, which showcase higher values compared to non-Ponzi smart contracts with also lower variability. Additionally, when grouping by users, Ponzi participants invest more compared to non-Ponzi ones.

The latter, instead, is detected when analyzing the out-degree features. In particular, when grouping by smart contract, the out-degree transactions are characterized by higher values compared to non-fraudulent contracts. Similarly, when grouping by individuals, the payouts received by Ponzi participants are higher, albeit with a higher standard deviation. This result is expected as Ponzi schemes thrive on distributing new users' investments to existing participants, better rewarding the early joiners. Yet, by comparing the inflows and outflows of users, Ponzi participants register significant losses.

The remaining important feature employed by the classifier relates to the name cluster. In fact, cluster 2, which includes ether games and token proxies, is more populated by non-Ponzi smart contracts compared to fraudulent ones.

Overall, these findings complement the current literature focusing on detecting Ponzi smart contracts on the Ethereum blockchain by exploiting the transaction history. Precisely, the proposed framework and features extracted significantly improve the discrimination capabilities currently attained. In fact, similar

works by Chen et al. (2018), Chen et al. (2019), Galletta and Pinelli (2023) and Yu et al., (2021), achieved, respectively, F1 scores of 0.44, 0.30, 0.62 and 0.79. Importantly, those results are achieved by exploiting the full transaction history, whereas the ones presented in this research employ only a limited set of them for classification. This difference is especially important given that Ponzi schemes should be identified as early as possible to limit their growth possibilities and negative financial consequences. The detection power obtained, however, is not at the Ponzi smart contract detection frontier as Yu et al. (2021) and Jin et al. (2022), achieved, respectively, F1 scores of 0.89 and 0.91. Nonetheless, those authors use a more black-box approach as they implement GNNs. The current research, instead, by exploiting tree-based methods, provides important insights into the characteristics that are mostly relevant for distinguishing between Ponzi and non-Ponzi smart contracts at different stages of their development.

Yet, the presented research has limitations which can serve as recommendations and directions for future research.

Firstly, due to the particularly high computational costs, the metrics achieved with TEAUG and with GNNs are the result of a single run, exposing them to some variability. To strengthen those results, metrics of at least 5 runs should be recorded and averaged. Further, when focusing on implementing GNNs, only two types of graph layers have been tested, namely Graph Convolutions and Graph Attention. Given the rapid advancement and the development of highly sophisticated layers, future research should consider testing different GNN structures. In this regard, due to the high number of transactions available between the participants and the smart contracts, Multigraphs worsened the detection capabilities, and the different edges had to be merged into one before applying the suite of algorithms. Those edges contained only information regarding the value being exchanged between the individual and the smart contracts, potentially limiting the information available to the model.

Future research should consider avoiding edges from being merged and should focus on implementing Temporal GNNs. In particular, the latter, by considering the entire evolution of the smart contract, could improve current detection capabilities. In fact, Temporal GNNs could unearth the entire redistribution mechanism and detect early smart contracts showcasing Ponzi traits.

The last limitation of this study is related to the lack of manual classification of new smart contracts to increase the dataset size. In effect, I exploited publicly available datasets which are populated by a restricted number of Ponzi frauds. Future research could benefit by expanding the smart contracts available on which to train the models.

References

Ali, H., Salleh, M. N. M., Hussain, K., Ahmad, A., Ullah, A., Muhammad, A. & Khan, M. (2019). A review on data preprocessing methods for class imbalance problem. *International Journal of Engineering & Technology*, *8*, 390-397.

Amir, R. (2021). Understanding an Ethereum Transaction. *Etherscan Information Center*. https://info.etherscan.com/understanding-an-ethereum-transaction/

Badawi, E., & Jourdan, G. V. (2020). Cryptocurrencies emerging threats and defensive mechanisms: A systematic literature review. *IEEE Access*, *8*, 200021-200037

Bartoletti, M., Carta, S., Cimoli, T., & Saia, R. (2020). Dissecting Ponzi schemes on Ethereum: identification, analysis, and impact. *Future Generation Computer Systems*, *102*, 259-277.

Belotti, M., Božić, N., Pujolle, G., & Secci, S. (2019). A vademecum on blockchain technologies: When, which, and how. *IEEE Communications Surveys & Tutorials*, *21*(4), 3796-3838.

Bhandari, A. (2023). Guide to AUC ROC Curve in Machine Learning : What Is Specificity? *Analytics Vidhya*. https://www.analyticsvidhya.com/blog/2020/06/auc-roc-curve-machine-learning/

Bhattacharya, U. (2003). The optimal design of Ponzi schemes in finite economies. *Journal of Financial Intermediation*, *12*(1), 2-24.

Bonorchis, R. (2018). Cryptocurrency 'Scam' Has South African Police Chasing Billions. *Bloomberg.com.* Retrieved May 1, 2023, from https://www.bloomberg.com/news/articles/2018-05-25/south-african-police-probing-1-billion-rand-investment-scam

Caliński, T., & Harabasz, J. (1974). A dendrite method for cluster analysis. *Communications in Statistics-theory and Methods*, *3*(1), 1-27.

Carvajal, A., Monroe, H. K., Pattillo, C. A., & Wynter, B. (2009). Ponzi Schemes in the Caribbean, IMF Working Papers, 2009(095), A001.

Charlier, J., Lagraa, S., & Francois, J. (2017, September). Profiling smart contracts interactions with tensor decomposition and graph mining. In *European Conference on Machine Learning and Principles and Practice of Knowledge Discovery (ECML-PKDD)-Workshop on MIning DAta for financial applicationS (MIDAS)*.

Chen, T., Li, X., Luo, X., & Zhang, X. (2017). Under-optimized smart contracts devour your money. In *2017 IEEE 24th international conference on software analysis, evolution and reengineering (SANER)* (pp. 442-446). IEEE.

Chen, W., Li, X., Sui, Y., He, N., Wang, H., Wu, L., & Luo, X. (2021b). Sadponzi: Detecting and characterizing ponzi schemes in ethereum smart contracts. *Proceedings of the ACM on Measurement and Analysis of Computing Systems*, *5*(2), 1-30.

Chen, W., Zheng, Z., Cui, J., Ngai, E., Zheng, P., & Zhou, Y. (2018). Detecting ponzi schemes on ethereum: Towards healthier blockchain technology. In *Proceedings of the 2018 world wide web conference* (pp. 1409-1418).

Chen, W., Zheng, Z., Ngai, E. C. H., Zheng, P., & Zhou, Y. (2019). Exploiting blockchain data to detect smart ponzi schemes on ethereum. *IEEE Access*, *7*, 37575-37586.

Chen, Y., Dai, H., Yu, X., Hu, W., Xie, Z., & Tan, C. (2021a). Improving Ponzi scheme contract detection using multi-channel TextCNN and transformer. *Sensors*, *21*(19), 6417.

Darby, M. (2021, April 14). In Ponzi We Trust. *Smithsonian Magazine*. https://www.smithsonianmag.com/history/in-ponzi-we-trust-64016168/

Davies, D. L., & Bouldin, D. W. (1979). A cluster separation measure. *IEEE transactions on pattern analysis and machine intelligence*, (2), 224-227.

Elinas, P., & Bonilla, E. V. (2022). Revisiting Over-smoothing in Graph Neural Networks.

Fan, S., Fu, S., Xu, H., & Cheng, X. (2021). Al-SPSD: Anti-leakage smart Ponzi schemes detection in blockchain. *Information Processing & Management*, *58*(4), 102587.

Fan, S., Fu, S., Xu, H., & Zhu, C. (2020). Expose your mask: smart Ponzi schemes detection on blockchain. In *2020 International Joint Conference on Neural Networks (IJCNN)* (pp. 1-7). IEEE.

Financial Action Task Force. (2014). FAFT report: Virtual currencies key definitions and potential AML/CFT risk. Retrieved June 23, 2023, from http://www.fatf-gafi.org/media/fatf/documents/reports/Virtual-currency-key-definitions-and-potential-aml-cft-risks.pdf

Galletta, L., & Pinelli, F. (2023). Sharpening Ponzi Schemes Detection on Ethereum with Machine Learning. *arXiv preprint arXiv:2301.04872*.

Grech, N., Brent, L., Scholz, B., & Smaragdakis, Y. (2019, May). Gigahorse: thorough, declarative decompilation of smart contracts. In *2019 IEEE/ACM 41st International Conference on Software Engineering (ICSE)* (pp. 1176-1186). IEEE.

Hicks, C. (2023, March 16). Different Types of Cryptocurrencies. *Forbes Advisor*. https://www.forbes.com/advisor/investing/cryptocurrency/different-types-of-cryptocurrencies/

Hoenicke, J. (n.d.). *Ponzi Token – A self-trading token on the Ethereum network*. https://test.jochen-hoenicke.de/crypto/ponzitoken/

Hu, T., Liu, X., Chen, T., Zhang, X., Huang, X., Niu, W., & Liu, Y. (2021). Transaction-based classification and detection approach for Ethereum smart contract. *Information Processing & Management*, *58*(2), 102462.

Huang, Y., Huang, J., Chen, X., He, K., & Zhou, X. (2023). BCGen: a comment generation method for bytecode. *Automated Software Engineering*, *30*(1), 5.

Jin, J., Zhou, J., Jin, C., Yu, S., Zheng, Z., & Xuan, Q. (2022, December). Dual-Channel Early Warning Framework for Ethereum Ponzi Schemes. In *Big Data and Social Computing: 7th China National Conference, BDSC 2022, Hangzhou, China, August 11-13, 2022, Revised Selected Papers* (pp. 260-274). Singapore: Springer Nature Singapore.

Jory, S. R., & Perry, M. J. (2011). Ponzi schemes: A critical analysis.

Kethineni, S., & Cao, Y. (2020). The rise in popularity of cryptocurrency and associated criminal activity. *International Criminal Justice Review*, *30*(3), 325-344.

Kutera, M. (2022). Cryptocurrencies as a subject of financial fraud. *Journal of Entrepreneurship, Management and Innovation*, *18*(4), 45-77.

Luu, L., Chu, D. H., Olickel, H., Saxena, P., & Hobor, A. (2016). Making smart contracts smarter. In *Proceedings of the 2016 ACM SIGSAC conference on computer and communications security* (pp. 254-269).

Luu, L., Chu, D. H., Olickel, H., Saxena, P., & Hobor, A. (2016, October). Making smart contracts smarter. In *Proceedings of the 2016 ACM SIGSAC conference on computer and communications security* (pp. 254-269).

Maheshkar, S. (2023). An Introduction to Message Passing Graph Neural Networks (GNNs). *W&B*. https://wandb.ai/graph-neural-networks/spatial/reports/An-Introduction-to-Message-Passing-Graph-Neural-Networks-GNNs---VmlldzoyMDI2NTg2

Mavridou, A., & Laszka, A. (2018). Designing secure ethereum smart contracts: A finite state machine based approach. In *Financial Cryptography and Data Security: 22nd International Conference, FC 2018, Nieuwpoort, Curaçao, February 26–March 2, 2018, Revised Selected Papers 22* (pp. 523-540). Springer Berlin Heidelberg.

Moffatt, M. (2018). The 5 Elements of a Ponzi Scheme. *ThoughtCo*. https://www.thoughtco.com/ponzi-scheme-definition-and-overview-1147436

Mohammed, U. (2021). Effect of Ponzi schemes on a country: the case of Ghana. *Journal of Financial Crime*, *28*(3), 926-939.

Narayanan, A., Chandramohan, M., Venkatesan, R., Chen, L., Liu, Y., & Jaiswal, S. (2017). graph2vec: Learning distributed representations of graphs. *arXiv preprint arXiv:1707.05005*.

Nikolić, I., Kolluri, A., Sergey, I., Saxena, P., & Hobor, A. (2018, December). Finding the greedy, prodigal, and suicidal contracts at scale. In *Proceedings of the 34th annual computer security applications conference* (pp. 653-663).

Nizzoli, L., Tardelli, S., Avvenuti, M., Cresci, S., Tesconi, M., & Ferrara, E. (2020). Charting the landscape of online cryptocurrency manipulation. *IEEE Access*, *8*, 113230-113245.

2022 ponzi schemes — Ponzitracker. (n.d.). Ponzitracker. https://www.ponzitracker.com/2022-ponzi-schemes

Rousseeuw, P. J. (1987). Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. *Journal of computational and applied mathematics*, *20*, 53-65.

Sadiraj, K., & Schram, A. (2001). *Informed and uninformed investors in an experimental Ponzi scheme*. ACE Programme Management.

Salam, E. (2023). Bitcoin is terrible for the environment – can it ever go green? *The Guardian*.

https://www.theguardian.com/technology/2023/apr/26/bitcoin-mining-climate-crisis-environmental-impact

SEC. (n.d.). *SEC Enforcement Actions Against Ponzi Schemes*. Retrieved May 1, 2023, from https://www.sec.gov/spotlight/enf-actions-ponzi.shtml

Shen, X., Jiang, S., & Zhang, L. (2021). Mining bytecode features of smart contracts to detect Ponzi scheme on blockchain. *Computer Modeling in Engineering & Sciences*, *127*(3), 1069-1085.

Song, L., & Kong, X. (2022). A Study on Characteristics and Identification of Smart Ponzi Schemes. *IEEE Access*, *10*, 57299-57308.

Springer, M. (2020). *The Politics of Ponzi Schemes: History, Theory and Policy*. Routledge.

Thanasi, E., & Riotto, J. (2017). The spectacular rise and disastrous collapse of a financial scheme: the case of Albania. *Open Journal of Business and Management*, *5*(01), 194.

The Economist. (2018, April 30). Crypto money-laundering. *The Economist*. https://www.economist.com/finance-and-economics/2018/04/26/crypto-money-laundering

The World Bank. (n.d.). *Glossary | DataBank*. https://databank.worldbank.org/metadataglossary/world-development-indicators/series/SI.POV.GINI

Vasek, M., & Moore, T. (2019). Analyzing the Bitcoin Ponzi scheme ecosystem. In *Financial Cryptography and Data Security: FC 2018 International Workshops, BITCOIN, VOTING, and WTSC, Nieuwpoort, Curaçao, March 2, 2018, Revised Selected Papers 22* (pp. 101-112). Springer Berlin Heidelberg.

Veličković, P., Cucurull, G., Casanova, A., Romero, A., Lio, P., & Bengio, Y. (2017). Graph attention networks. *arXiv preprint arXiv:1710.10903*.

Wood, G. (2014). Ethereum: A secure decentralised generalised transaction ledger. *Ethereum project yellow paper*, *151*(2014), 1-32.

Wood, T. (2020). F-Score. *DeepAI*. https://deepai.org/machine-learning-glossary-and-terms/f-score

Yaga, D., Mell, P., Roby, N., & Scarfone, K. (2019). Blockchain technology overview. *arXiv preprint arXiv:1906.11078*.

Yu, S., Jin, J., Xie, Y., Shen, J., & Xuan, Q. (2021). Ponzi scheme detection in ethereum transaction network. In *Blockchain and Trustworthy Systems: Third International Conference, BlockSys 2021, Guangzhou, China, August 5–6, 2021, Revised Selected Papers 3* (pp. 175-186). Springer Singapore.

Zhang, S., Tong, H., Xu, J., & Maciejewski, R. (2019). Graph convolutional networks: a comprehensive review. *Computational Social Networks*, *6*(1), 1-23.

Zheng, Z., Chen, W., Zhong, Z., Chen, Z., & Lu, Y. (2022). Securing the Ethereum from Smart Ponzi Schemes: Identification Using Static Features. ACM Transactions on Software Engineering and Methodology.

Zheng, Z., Xie, S., Dai, H. N., Chen, W., Chen, X., Weng, J., & Imran, M. (2020). An overview on smart contracts: Challenges, advances and platforms. *Future Generation Computer Systems*, *105*, 475-491.