Shahjalal University of Science and Technology
Department of Computer Science and Engineering

An Efficient Future-proof Approach to Detect Illicit Accounts on Ethereum blockchain Using Modified Online Bagging Nested Ensemble (NE) Method

Abdullahil Kafi

Reg. No.: 2017331072

4^{th} year, 2^{nd} Semester

Sukanto Kumar Das

Reg. No.: 2017331091

4^{th} year, 2^{nd} Semester

Department of Computer Science and Engineering

Supervisor

Mohammad Abdullah Al Mumin

Professor

Department of Computer Science and Engineering

3rd March, 2023

A Thesis submitted to the Department of Computer Science and Engineering, Shahjalal University of Science and Technology, in partial fulfillment of the requirements for the degree of Bachelor of Science in Computer Science and Engineering.

By

Abdullahil Kafi
Reg. No.: 2017331072
4^{th} year, 2^{nd} Semester

Sukanto Kumar Das
Reg. No.: 2017331091
4^{th} year, 2^{nd} Semester

Department of Computer Science and Engineering

Supervisor
Mohammad Abdullah Al Mumin
Professor
Department of Computer Science and Engineering

3rd March, 2023

Recommendation Letter from Thesis Supervisor

The thesis entitled *An Efficient Future-proof Approach to Detect Illicit Accounts on Ethereum blockchain Using Modified Online Bagging Nested Ensemble (NE) Method* submitted by the students

1. Abdullahil Kafi
2. Sukanto Kumar Das

is under my supervision. I, hereby, agree that the thesis can be submitted for examination.

Signature of the Supervisor:
Name of the Supervisor: Mohammad Abdullah Al Mumin
Date: 3rd March, 2023

Certificate of Acceptance of the Thesis

The thesis entitled *An Efficient Future-proof Approach to Detect Illicit Accounts on Ethereum blockchain Using Modified Online Bagging Nested Ensemble (NE) Method* submitted by the students

1. Abdullahil Kafi
2. Sukanto Kumar Das

on 3rd March, 2023, hereby, accepted as the partial fulfillment of the requirements for the award of their Bachelor Degrees.

Head of the Dept.	Chairman, Exam. Committee	Supervisor
Md Masum.	Mohammad Abdullah Al	Mohammad Abdullah Al
Professor and Head	Mumin	Mumin
Department of Computer	Professor	Professor
Science and Engineering	Department of Computer	Department of Computer
	Science and Engineering	Science and Engineering

Abstract

Blockchain is the underlying foundational technology that enabled the advent of many other decentralized, trustless and distributed applications of which cryptocurrency is an example. Even after one decade of its first public appearance through Bitcoin, Blockchain technology's popularity is still on the rise. Because of its immutability, decentralized manner and trustlessness, organizations and countries around the globe are embracing and adopting Blockchain technology more and more. Ethereum is one such public blockchain platform. Ethereum's support for smart contracts gave birth to numerous applications of the technology ranging from buying and selling of Non-Fungible Token to investing in and trading cryptocurrencies. Since banking and finance is indissolubly linked to it, its security and vulnerabilities are of great interest to many researchers. Scams, fraudulent and illicit activities are a part of the cryptocurrency market. Miscreants exploit the lack of cryptocurrency literacy of many people.

In this thesis, propose an efficient approach to detect illicit accounts on Ethereum blockchain using modified Online Bagging nested ensemble method. We collected transaction records of 18,057 Ethereum externally owned accounts. We trained, tested and benchmarked our approach against other popular approaches which showed significant improvement in terms of model retraining time and resource consumption, while also maintaining an effective performance measure with an average accuracy of 96 percent. We also showed the shortcomings other popular approaches suffer from when dealing with unseen data. Our contribution is three-faceted. Firstly, we compile an up-to-date dataset consisting of records of more than 18 thousands Ethereum accounts which can be used in further related research work. Secondly, we demonstrate the weaknesses of traditional approaches. Finally, we propose an effective algorithm for efficiently building classifiers for crypto-scam detection.

.

Keywords: Blockchain, Cryptocurrency, Ethereum, Machine Learning, Anomaly Detection, Scam Detection.

Acknowledgements

We would like to thank the Department of Computer Science and Engineering, Shahjalal University of Science and Technology, Sylhet 3114, Bangladesh, for supporting this research. We are very thankful to our honorable supervisor Mohammad Abdullah Al Mumin, co-supervisor Dr. Md Jabed Morshed Chowdhury for their worthy supports and directions. Without their supports and directions, this work could not be done.

Contents

Abstract I

Acknowledgements III

Table of Contents IV

List of Tables VI

List of Figures VII

1 Introduction **1**

1.1 Motivation 2

1.2 Report Organization 2

2 Crypto Scams **3**

2.1 Ponzi Schemes 5

2.2 Fake Crypto Services 5

2.3 Malware 7

2.4 Blackmail 8

2.5 Advance Scams 8

2.6 Money Laundering 9

2.7 Fake ICO 10

3 Background and Related Work **11**

3.1 Related Work 14

4 Methodology **21**

4.1 Decision Tree 21

4.2 Random Forest Classifier 22
4.3 Ensemble Learning 22
4.4 Bagging 23
4.5 Online Learning 24
4.6 Online Bagging 26
4.7 Performance metrics 27

5 Data Collection And Preprocessing **29**
5.1 Data Collection 29

6 Training and Evaluation **34**

7 Conclusion and Future Work **47**

References **48**

Appendices **54**

A Data Preprocessing **55**

B Feature Reduction **57**

C Data Normalization and Augmentation **62**

D Training **64**

E Testing and Evaluation **66**

List of Tables

3.1 Number of publications by scam type 14

5.1 1st to 23rd feature of Etherium addresses extracted from Etherscan 32
5.2 24th to 42th feature of Etherium addresses extracted from Etherscan 33

6.1 Model Training Time (Offline Batch Learning vs Online Learning) 37
6.2 Performance Matics (Accuracy, Precision, Recall, F1) 37

List of Figures

2.1 Cryptory Ponzi scheme [1] 5
2.2 An example of fake exchange service 7
2.3 Fake mixing service at Bitcoin Mixer [2] 8
2.4 Wallpaper of a device attacked by WannaCry 9
2.5 WannaCry screen requests payment 10

3.1 Blockchain ledger 12
3.2 Public Blockchain 13
3.3 Private Blockchain 14
3.4 Consortium Blockchain 15
3.5 The anomaly detection procedure by J. Kim [3] 16
3.6 This diagrame demonstrates an AutoEncoder structure with a mixed ReLU and tanh activation unit example, where the encoder compresses the input data and the decoder restores the compressed representation. 17
3.7 The three steps framework of phishing detection on Ethereum by [4] 18
3.8 Detection and analysis model of Bitcoin Generator Scam by Qi Yuan [5] 18
3.9 Country based source and destination of scam exchange funds found in [6] 19

4.1 Random Forest Classifier 23
4.2 Training Sub-sample data in bagging ensemble learning method 24
4.3 Voting for binary classification in bagging ensemble learning method 25

6.1 Decision Tree 35
6.2 Random Forest 36

6.3 K-Nearest Neighbors 38
6.4 Decision Tree (Online Learning) 39
6.5 Random Forest (Online Learning) 40
6.6 K-Nearest Neighbors (Online Learning) 41
6.7 Time Comparison for Decision Tree (Online vs. Offline) 42
6.8 Training Time Comparison for Random Forest (Online vs. Offline) 43
6.9 Training Time Comparison for K-Nearest Neighbors (Online vs. Offline) 44
6.10 XGB vs. Online Bagging.png 45
6.11 Feature Importance of Base Learners 46

Chapter 1

Introduction

According to IBM [7], Blockchain is a distributed and tamper-proof ledger that supports the transactional and asset tracking activities within a network of businesses. Decentralization, immutability and anonymity are the key properties of Blockchain technology that make it stand out among many existing and frontier technologies. Its unique nature gave rise to many innovations and inventions of different interesting applications. The World Intellectual Property Organization says, Blockchain is a leading-edge technology that is exerting a profound impact on the operation of businesses, while simultaneously transforming multiple innovation and creative ecosystems [8].

With the advent of Blockchain, different types of cryptocurrency have gained a lot of attention. Cryptocurrency refers to digital or virtual currency that is protected by cryptography. Bitcoin is the first such currency that first made an appearance and came into the scene back in 2008. The first bitcoin was mined by Satoshi Nakamoto who published the whitepaper for the digital currency in 2008 [9]. Satoshi Nakamoto is a pseudonym who is still unknown. All other cryptocurrencies evolve after bitcoin based on Blockchain. Ethereum [10] is another such name that holds the place right after Bitcoin. Ethereum is a decentralized platform that enables the operation of and houses many other decentralized applications. Its native currency is called *ether*. Ethereum alone has enabled the birth of hundreds of other cryptocurrencies like DogeCoin, DAI and Shiba Inu etc. These digital currencies have become alternatives to fiat currency which is not dependent on any central authority.

Like many other technologies, security threats and vulnerabilities have been a major concern for Blockchain too. The anonymity that this technology provides has enabled miscreants to commit crimes without revealing identity. Scams such as phishing attacks, money laundering, blackmailing are rife in Blockchain avenue, while many other illicit activities also take place.

1.1 Motivation

The introduction of cryptocurrency is making a positive impact on the current era of technology. So the security issue is very important for a stable system and trust of the user. The illegal use of cryptocurrency for different types of scams and money laundering also raise a question on the motive of inventing crypto currency. So the security of cryptocurrency and detecting the illicit use of crypto currency can increase the users trust and satisfaction, thus sprade the use of cryptocurrency. We tried to investigate the security issue of crypto currency and the illicit use of crypto currency. Our aim is to develop a system to detect the illegal use of cryptocurrency with the help of machine learning.

1.2 Report Organization

This report is organized in six chapters. Each chapter has some sections and some subsections also.

In Chapter 1, we introduce our thesis topic and the motivation behind pursuing the topic.

In Chapter 2, we present the background of crypto scams.

In Chapter 3, we discuss the background and preview previous research related to crypto scam detection.

In Chapter 4, we discuss the methodology and performance matrix.

In Chapter 5, we discuss data collection and data preprocessing.

In Chapter 6, we present our models, their evaluation and analysis.

In Chapter 7, we brief our future work plan and conclude the report.

Chapter 2

Crypto Scams

Scam is an umbrella term. It can be defined in numerous ways. In our case, we define scam as an illicit act that involves one or more individuals who deliberately deceive others with the intention of acquiring something valuable through illegal means. In a crypto scam, perpetrators exploit the market value of digital assets as well as their potential for conversion into fiat currencies, all the while capitalizing on the advantage of anonymity afforded by these transactions. Note that, we do not consider each and every illegal or illicit activity that can take place in the Cryptocurrency market. Our sole focus is crypto-scam.

Cybercriminals have exploited the Pseudo-anonymity nature of Bitcoin in their illicit activities. As per the findings of CipherTrace's 2019 report [11], the aggregate worth of different types of scams had escalated to twice the value from the previous year, with illegal activities resulting in the unlawful acquisition of over $4.4 billion in 2019 alone.

There are various types of cryptocurrency-based attacks, with "High Yield Investment Programs" (HYIP) being a well-known example of a scam perpetrated by cybercriminals [12], [13], [14], [15]. HYIP is an illegitimate investment scheme that promises exorbitant returns on investment, often in excess of 1-2% per day [12]. This type of scam is commonly referred to as a Ponzi scheme [13], after Charles Ponzi, a notorious fraudster who claimed to be running an arbitrage scheme in the early 1920s, offering investors a 100 percent profit within 90 days.

Money laundering (ML) [16], [17] and ransomware [18], [19], [20] are additional examples of fraudulent activities in the cryptocurrency realm. Perpetrators leverage the anonymity of blockchain technology to conceal the origins of illicit funds and convert them into cash for deposit into banks. This enables them to profit from illegal activities both on-chain and off-chain. To obscure the connection between the criminals and the related funds, they use cryptocurrency to launder money obtained from various sources, such as cybercrimes, digital fraud, thefts of cryptocurrencies from online exchanges, as well as conventional crimes and schemes. This is achieved by channeling the money through a complex sequence of commercial transactions or banking transfers [16].

A type of attack known as ransomware involves the use of malicious software to deny access to a victim's device data, typically by locking and encrypting it, until a ransom is paid [20]. The payment for such ransoms is frequently made using cryptocurrencies, particularly Bitcoin.

In a recent incident, officials in Riviera Beach agreed to pay a ransom of 65 bitcoins, which was valued at US $600,000 at the time, to a cybercriminal who had taken control of and disabled the city's computer systems. This resulted in a service outage that compelled the local police and fire departments to document numerous emergency calls using pen and paper [21]. Crypto scams can be classified in various ways based on distinct criteria [22], [23]. E. Badawi et . al [22] proposed a taxonomy for such scams, which encompasses seven key characteristics that identify different forms of illegal activities, and can be blended together to describe hybrid scams. These features are as follows:

1. Ponzi schemes
2. Fake crypto services
3. Malware
4. Blackmail
5. Advance–fee scams
6. Money laundering
7. Fake ICO

2.1 Ponzi Schemes

The cryptocurrency market is plagued by a prevalence of unlawful activities, with Ponzi schemes or HYIPs representing the most frequent form of fraudulent activity [20]. These schemes are characterized as scams that purport to be high yield investment programs (HYIPs) [20], [24].

Ponzi schemes tempt investors to invest funds in their program by offering early investors an abnormally high return. However, when new investments begin to dwindle or the organizers deem it appropriate, they halt the payouts and abscond with all the investments. In practice, Ponzi schemes pay initial investors with the funds deposited by new investors, which leads to the scheme collapsing once new investments stop coming in. Ultimately, in Ponzi schemes, most of the investors at the lower end of the hierarchy end up losing their entire investment. The example of a Ponzi scheme is illustrated in figure 2.1.

Figure 2.1: Cryptory Ponzi scheme [1]

2.2 Fake Crypto Services

There are various services offered in the cryptocurrency ecosystem to facilitate the use and management of cryptocurrencies for users. These services include exchange platforms, mixers, and wallets. However, some criminals take advantage of the demand for these services and create

fraudulent services that pretend to be legitimate. These fraudulent services have been categorized into five types by E. Badawi et al. [22] as follows:

Fake exchange : Fraudulent exchange services offer cryptocurrency purchase prices that are unrealistically low compared to the market rates, misleading users into thinking they are getting a good deal. These fake exchanges exploit users by promising them quick and easy access to cheap cryptocurrencies. An example of such a fraudulent service is illustrated in Figure 2.2, which depicts a fake exchange service located at paybillsbitcoin.com (DELETED).

Fake wallet : Wallet services make it easy for users to manage credentials, sign and initiate transactions, hence, send and receive cryptocurrencies with ease. Uninitiated users are susceptible to running into wallet scams. For instance, some wallet services take away all the money, while others steal a small chunk of the daily deposit. There are some other services that start to withdraw once the deposit amount reaches a certain figure.

Fake mixing : It is possible to track the flow of cryptocurrency between addresses in blockchain systems. However, mixing services can complicate the transaction process by making it difficult to connect the sender and receiver addresses. Mixing services achieve this by randomizing the number of transactions, adding delays to transactions, and using temporary addresses, among other techniques. Some scammers exploit the demand of such services. They develop fake mixers that take away the money after receiving it without sending it to the respective receiver of the money. The figure 2.3 shows a website of a fake mixer service called “BitcoinMixer”. Many users lost their money to this fake mixer, as reported on BitcoinTalk.

Fake mining pool : Cryptocurrencies that work based on the Proof Of Work mechanism require miners to put in computational efforts to validate and create blocks. As a result they receive a certain amount of cryptocurrency in the form of incentives. Some criminals have found a way to take advantage of it. They ask people who want to be miners to join a pool of existing miners and invest money to purchase hardwares required for mining. But, in reality, the invested money is not used to purchase hardwares but rather to pay to previously joined users.

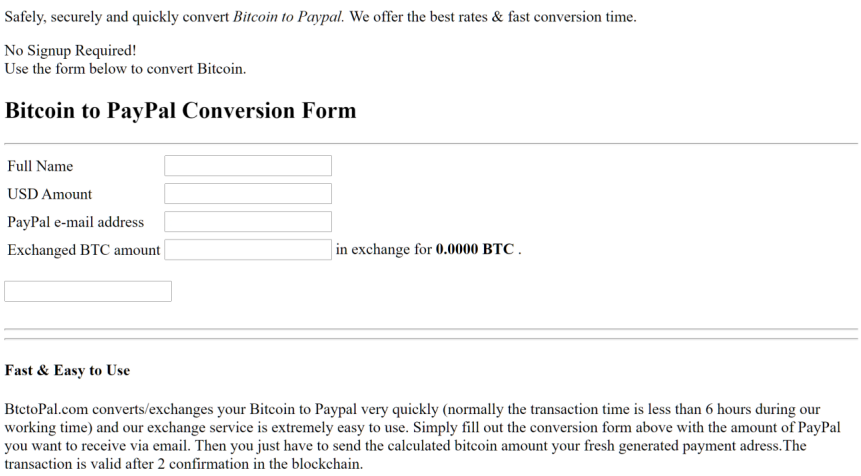

Figure 2.2: An example of fake exchange service

Fake donation : Usually, people of charitable personality trait make donations out of good will to organizations, projects or people. Scammers exploit people's good will or virtue by creating fake donation campaigns, and instead of giving the money to the respective parties, they take away the money.

2.3 Malware

Malware, short for malicious software, is a kind of intrusive software program developed by hackers that is installed on a computer without the cognizance of any of the authorized users of that computer. Once installed, it can add, manipulate or delete data stored on that computer. The untraceability nature of cryptocurrencies has been taken advantage of by malware developers. Two most relevant out of the many types of malware are Ransomware and Crypto Loggers. After being installed Ransomware encrypts the data, and asks for ransom from the victim, usually in cryptocurrency. The figures 2.4 and 2.5 show the wallpaper of a device attacked by WannaCry ransomware and a pop-up message on that device asking for ransom. Crypto Loggers attempt to steal secret keys needed to make transactions so that attacker can transfer all crypto-assets from owners account to their own accounts.

Figure 2.3: Fake mixing service at Bitcoin Mixer [2]

2.4 Blackmail

Blackmail scam is a kind of scam where the attacker claims to have obtained sensitive data or compromised user's device and then demands that ransom be paid, usually in cryptocurrency because of its anonymous nature.

2.5 Advance Scams

According to the FBI [25], an advance fee scheme involves a situation where the victim makes a payment to an individual or entity with the expectation of receiving something of higher worth, such as a contract, loan, investment, or gift. However, in the end, the victim ends up receiving very little or nothing at all. In this kind of attack, the victim usually receives a message through email or any other social media account promising a unreasonably huge amount of money in return, if the victim makes a small payment first. Once the payment is made, the scammer either disappears or asks the victim to send even more money. The website elonmuskdrop.com ran one such campaign. It is now

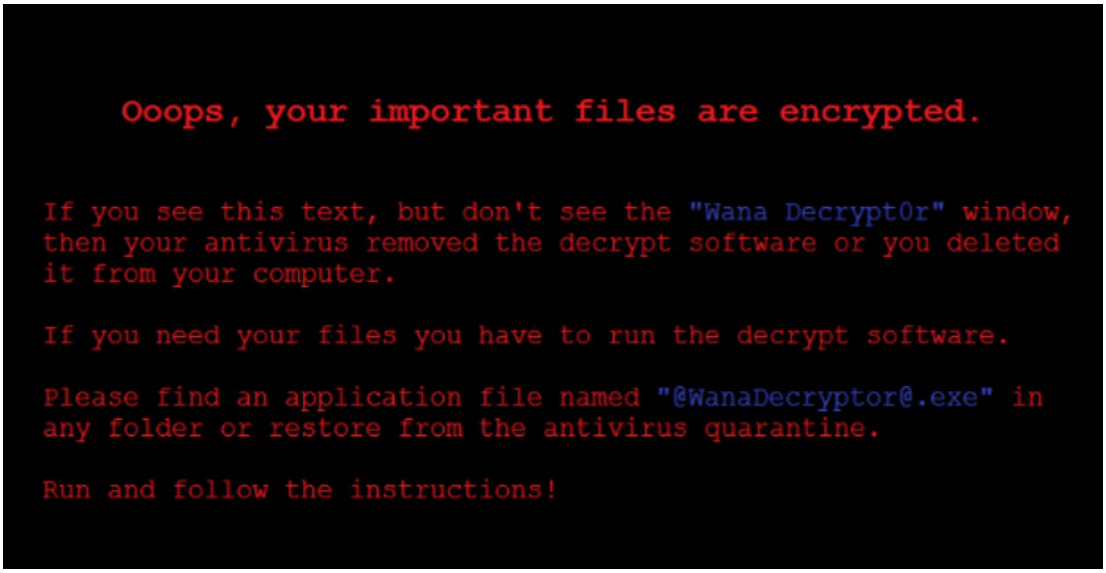

Figure 2.4: Wallpaper of a device attacked by WannaCry

flagged by Ethereum phishing detector, CryptoScamDB, CoinAbuse and many more organizations.

2.6 Money Laundering

Money Laundering makes money yielded through illicit activities look legitimate by obfuscating the actual sources and making it look as though it is coming from legitimate sources.

According to CNBC [24], Criminals in the cryptocurrency realm employ a prominent technique to launder money, which involves transferring digital assets across multiple blockchains, circumventing centralized services capable of tracing and freezing transactions. This is made possible through cross-chain bridges, and the amounts involved are becoming increasingly significant. t is claimed that crypto criminals laundered a total of $540 million by utilizing a service known as RenBridge. According to a very recent report by BBC [16], "Crypto money laundering rises 30%".

Figure 2.5: WannaCry screen requests payment

2.7 Fake ICO

Investopedia defines an initial coin offering (ICO) as the equivalent of an initial public offering (IPO) in the cryptocurrency industry [26]. It is a method of raising capital wherein companies sell investors a new digital token or cryptocurrency. Fake ICO scams follow the same strategy by running fake campaigns enticing users into acquiring fake coins with actual money that has real worth.

Chapter 3

Background and Related Work

Blockchain, the technology behind all cryptocurrencies, is based on a peer–to–peer (P2P) networking system where all the nodes are connected to each other. The database of blockchain is called a ledger that is distributed among every node. Once a transaction request is made, it is broadcasted to a peer–to–peer network and every node in that network becomes aware of it. New block is added based on a consensus protocol among POW, POS. Each block gets added to the ledger of each node and each block stores the hash of the previous block. This creates a chain–like connection between the blocks. In other words, the ledger is like a chain of blocks (figure 3.1), connected with hash. Thus to alter any transaction one has to alter every all of the previous blocks in the network which is not feasible. That is why cryptocurrency is becoming more popular due to its transparency, immutability and decentralized property.

There are three variants of blockchain [27]. They are Public blockchain, Permissioned or private blockchain and Federated or consortium blockchain.

Public Blockchain: Anyone can participate without restriction in a public or permissionless blockchain (figure 3.2). Every node in the network can see, access and inspect the transactions or data related to them. All full nodes take part in the validation process of a transaction. Most types of cryptocurrencies like Bitcoin, Ethereum, Neo, Otum, Waves are based on public blockchain.

Private Blockchain: In a private blockchain (figure 3.3) only a single entity or organization has

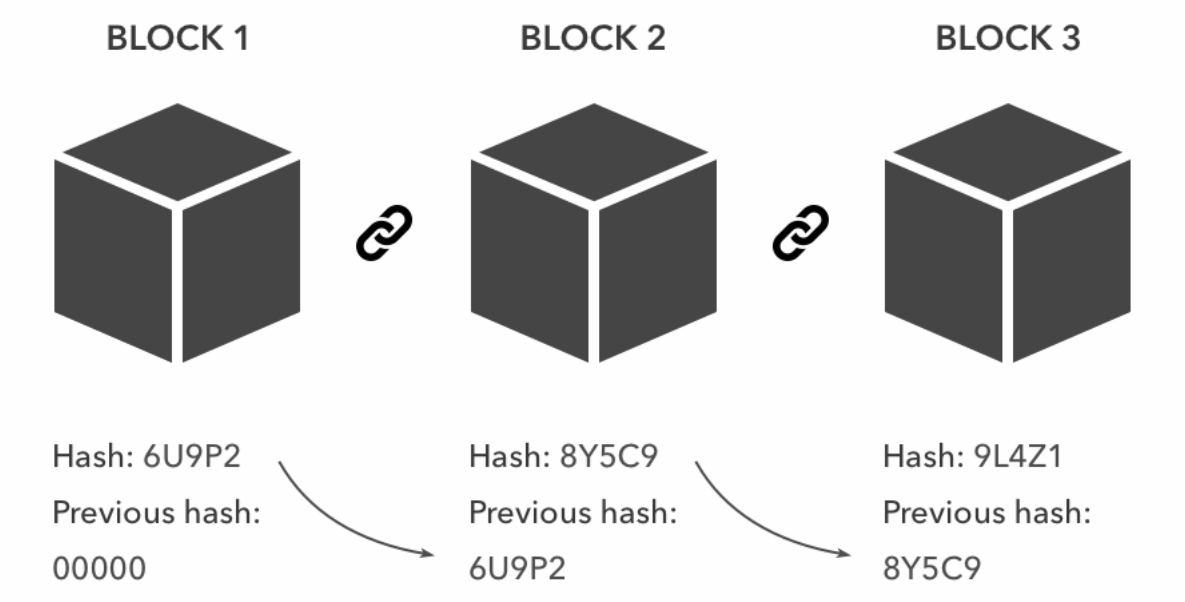

Figure 3.1: Blockchain ledger

the ultimate authority over the entire network. This organization has all kinds of write permissions and sets rules as to who can read data based on its requirements. Private blockchain usually use Proof of Authority [28] as its consensus algorithm.

Consortium Blockchain: Unlike private blockchain, in a federated or consortium blockchain (figure 3.4) there are multiple organizations. Each organization can set rules as to who can access the data stored on Blockchain related to that specific organization. Only a set of preselected equally privileged nodes who partake in the consensus process in a federated blockchain network. Hyperledger Fabric and Corda are two examples of consortium blockchain.

Machine learning is an area of study within artificial intelligence that involves creating models and algorithms that allow computers to learn from data and make predictions or decisions based on that learning. Instead of being explicitly programmed to perform a particular task, machine learning algorithms are designed to automatically improve their performance on specific tasks by learning from past examples or data. This versatility allows machine learning to be useful for a wide range of applications, including image recognition, natural language processing, predictive analytics, and recommendation systems. The three main types of machine learning are supervised

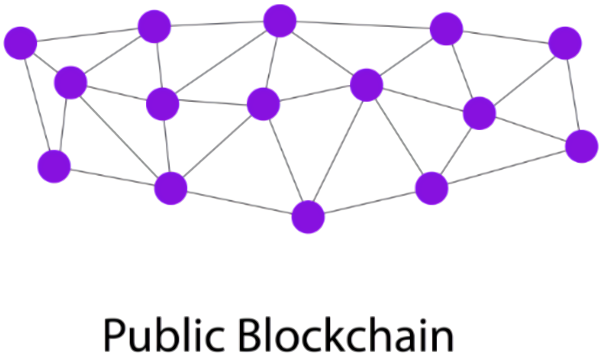

Figure 3.2: Public Blockchain

learning, unsupervised learning, and reinforcement learning. Each type has its own unique purpose, strengths, and weaknesses, and the choice of which to use depends on the problem being addressed.

Supervised learning involves training the algorithm on labeled data to predict outputs for new data, while unsupervised learning involves finding patterns in unlabeled data. Reinforcement learning involves training the algorithm to make decisions based on feedback from the environment, with the goal of finding a policy or strategy that maximizes a long-term reward signal.

According to the coinmarketcap website the total market capital of the top 10 cryptocurrencies is more than 750 billion dollars [29]. Bitcoin, the topest cryptocurrency alone, gained the market capital of 360 billion dollar. The second one, Ethereum possessed 187 billion dollar market capital. Due to the huge monetary value, cryptocurrencies gain the attention of the hackers. The anonymous feature of crypto is like a blessing for criminal activities or organizations. The use of crypto currency for illicit activity like money laundering, phishing, ransom, gambling, ponzi schemes etc are evident from LIU at el[30].Our aim of this research is to gather knowledge about different crypto currency scams, and the detection of scam using machine learning models. Machine learning models can be trained with labeled or unlabeled data to learn the hidden pattern of the data. Different types of data like block chain network traffic data, the data from blockchain ledger can be used for training models for different purposes. We have collected scam related data from different websites. We have also trained data with different models to maximize the performance

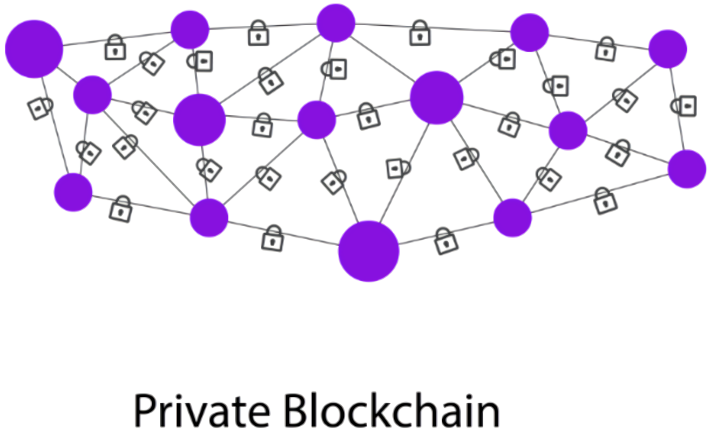

Figure 3.3: Private Blockchain

of detecting scam. We also tried to determine the significant features using different techniques.

3.1 Related Work

The table 3.1 shows the number of publications related to crypto scams from 2013 to 2020.

Scam type	**2013**	**2014**	**2015**	**2016**	**2017**	**2018**	**2019**	**2020**	**total**
Ponzi schemes	0	0	1	0	0	6	5	4	16
Fake crypto services	1	0	1	0	0	2	2	2	8
Malware	0	1	1	1	1	2	1	0	7
Blackmail	2	0	0	0	0	0	1	1	4
Advance-fee scam	0	0	0	0	0	0	0	1	1
Money laundering	1	0	1	0	0	1	2	1	6
Fake ICO	0	0	0	0	0	1	1	1	3

Table 3.1: Number of publications by scam type

Many research works have been done regarding Ponzi schemes on the Bitcoin domain [14], [31], [15], [32], [12], [13] and on Ethereum Public Blockchain Platform [32], [33]. Few others' works aim to achieve automatic detection of Ponzi schemes by analyzing transactions [31], [15],

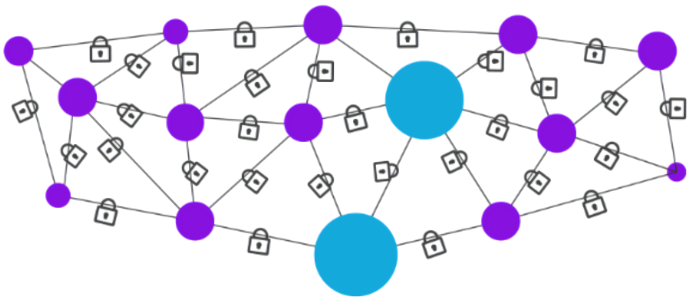

Figure 3.4: Consortium Blockchain

and [13].

Vasek, et al [14] gathered 349 candidate scams' list from a thread of bitcointalk.org [34], a suspected list of fraudulent services at badbitcoin.org [35] and cryptohyips.com that tracks Bitcoin-based HYIPs. They identified 192 scams out of these 349 candidates and categorized them into four groups: ponzi schemes, mining scams, scam wallets and fraudulent exchanges.

A. Holub, et al. [36] tracked *Coinhoarder*, a Ukrainian Bitcoin Phishing campaign, for over six months. They estimated between September 2017 to December 2017 alone, it swiped around USD 10 million and tens of millions of USD worth of cryptocurrency in total.

The works of [19], [37] and [38] involve ransomware in the cryptocurrency avenue. Authors of [19] performed a measurement analysis of CryptoLocker. Cryptolocker is a trojan horse malware that, once installed on a computer, encrypts data stored on that computer as well as all connected media and then asks for ransom [39] . They estimated close to USD 310,472.38 worth of Bitcoin payment as ransom from September, 2013, to January, 2014.

[37] proposed a lightweight framework to deanonymize bitcoin addresses managed by the same entity and to classify ransom.

Yining Hu, et al. [40] built binary classifiers to distinguish money laundering related transactions from regular ones. The authors collected transaction data over three years in the Bitcoin network and generated transaction graphs. Their findings suggest that the output values and neighborhood information carry the most importance while classifying transactions into money laundering transactions and regular ones.

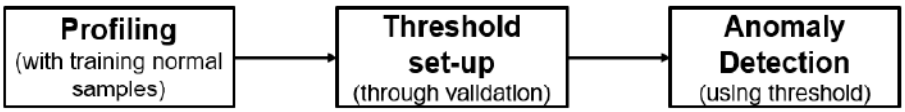

Figure 3.5: The anomaly detection procedure by J. Kim [3]

J. Kim, at el. [3] proposed a mechanism (figure 3.5) to identify anomalous activities in the Blockchain with a semi-supervised svm classifier using a data collection engine and an anomaly detection engine. Instead of inspecting data stored on the ledger, the authors took a slightly different approach. They analyzed the statistics of blockchain network traffic. Since network traffic statistics are complex in nature, the detection engine is built on top of an AutoEncoder (figure 3.6) which helps structure the dataset and reduce the dimension of the data. The learning method used for anomaly detection is a one-class semi-supervised learning which can detect new activity patterns even if they were not fed to or previously seen by the model. The attack traffic data were collected from a simulated environment where a node was set up for simulating attacks which was not connected to the Bitcoin Mainnet. Another active bitcoin node was set up for non-malicious data collection. The anomaly detection engine was hosted on that machine which also performs data collection. Attacks such as DoS and Eclipse were simulated using the node set up in an isolated environment. The data collection process went on for 7.4 days and the total number of records collected in that period were 639,360, 39.4% of which were non-anomalous. They trained the dataset for four groups of features using six different ML algorithms, OC-SVM, LR, GB, RF, DNN, AE(proposed). For the first group the f1 score was very poor due to limited features. Their

proposed AE model performs better than all other models.

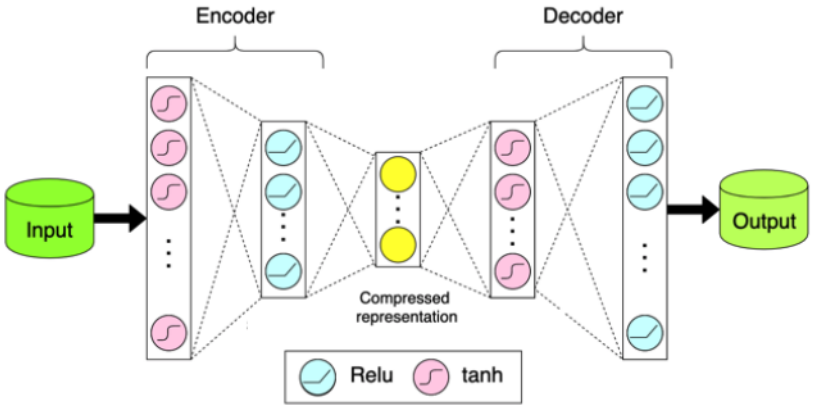

Figure 3.6: This diagrame demonstrates an AutoEncoder structure with a mixed ReLU and tanh activation unit example, where the encoder compresses the input data and the decoder restores the compressed representation.

Qi Yuan et al. [4] proposed Node2vec, a network embedding method to extract features and then used one–class Support Vector Machine to classify phishing addresses on Ethereum based on transaction data (figure 3.7). They created a network of Ethereum transactions where every account represents a node and each edge signifies a transaction between two accounts. They collected the transaction related data from Etherscan block explorer. They collected the phishing related data from EtherScamDB (https://etherscamdb.info/scams) and Etherscan (https://etherscan.io/). Both of these two authoritative websites report Phishing scams and also report the corresponding phing accounts. The labeled node with phishing in their network using this dataset. They used two networking embedding and three non-embedding methods for learning the latent features of any node. They used Deepwalk, Node2vec as network-embeding and Time Features Only, Amount Features Only, Time + Amount Features. The embedding size d=64, walks per node r=20, walk length l=5, context size k=10 are kept the same for both Node2vec and Deepwalk methods. They evaluate the performance measure for all the methods for features extraction. The Node2vec method resulted in the best performance among all the methods.

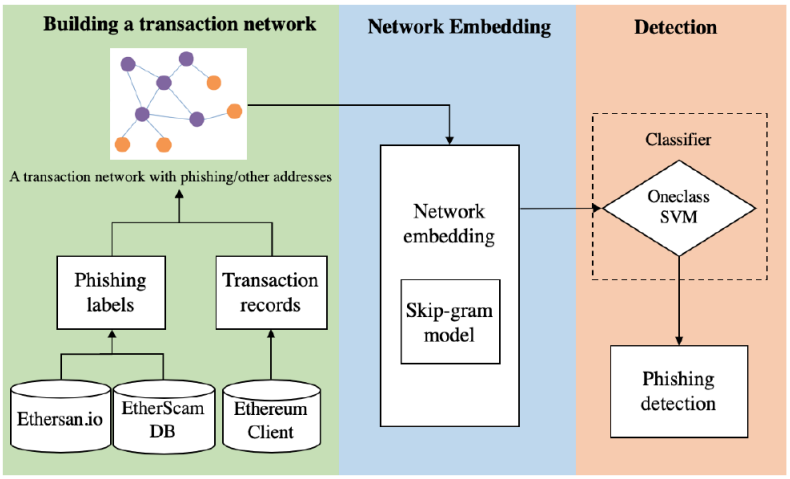

Figure 3.7: The three steps framework of phishing detection on Ethereum by [4]

Emad et al. [5] conducted an analysis (figure 3.8) of Bitcoin Generator Scams (BGS) where users are promised free bitcoins for a small fee. The authors collected a dataset of 330 BGS pages and 330 benign pages from Google search, Bitcoin.fr, Cutestat.com, and Internet archive. They developed a web crawler using predefined search queries to search for BGS pages using popular search engines such as Google.com, Bing.com, search.yahoo.com, and search.1and1.com. They used Python beautifulsoup and CSS selectors to extract and crawl URLs from the search results. A text-based classifier was developed using five classifiers including Support Vector Classifier (SVC), Naive Bayes (NB), k-nearest neighbors (KNN), Random Forest (RF), and Multi-layer Perceptron (MLP) to detect BGS pages. The authors compared the F1 scores obtained from the classifiers. The web crawling process was performed daily from November 2019 to February 2021, and the authors analyzed the online wallet addresses used by BGS, the number of BGS URLs detected per day, the number of Bitcoin addresses detected per day, the daily deposited money to BGS addresses, the daily incoming transactions to BGS addresses, and a payback analysis.

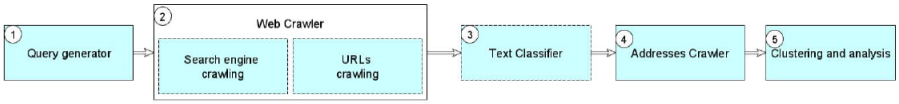

Figure 3.8: Detection and analysis model of Bitcoin Generator Scam by Qi Yuan [5]

In the exploratory analysis, Ross Phillips et al. [6] examined the scam websites listed on CryptoScamDB. They collected website snapshots from URLScan and extracted the content from

the snapshots. Additionally, they gathered historical domain registration data for each website to determine who registered the domain name. Using corpus data, they clustered the websites into various types with the DBSCAN algorithm, such as Celebrity/Exchange Giveaway (Advance-Fee) and MyEtherWallet-Unclear Format (Phishing). In total, 171 clusters were identified, with an average of 24 websites per cluster. The authors utilized Google Analytics ID, domain registrant details, domain registrar, and IP address of the websites as features for campaign clustering on 1,560 previously identified advanced fee-type websites. The campaign-based clustering technique identified 25 campaigns, with an average of 17 websites per campaign cluster. The authors analyzed the behavior of the advance-fee type clusters, such as the magnitudes of Bitcoin and Ether inflows. They also examined the blockchain address extracted from the scam websites and found that the same blockchain address was present in multiple clusters, indicating that multiple campaigns owned by the same entity were using the blockchain address. Additionally, they analyzed the source and destination categories of the funds and the country-based source and destination of exchange funds (figure **??**.

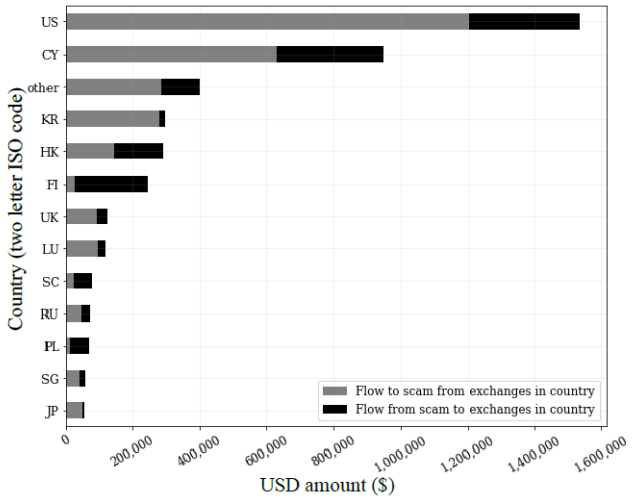

Figure 3.9: Country based source and destination of scam exchange funds found in [6]

Steven et al. [41] proposed a model based on the Extreme Gradient Boosting (XGBoost) algorithm to detect illicit accounts on the Ethereum blockchain. The authors constructed a balanced dataset comprising 2179 scam addresses and 2502 normal addresses, with 42 features. The optimal value of the model parameter was determined using grid search validation. In the experiment conducted by the authors, the features of "time difference between the first and last transaction"

and "the total available Ether balance" were identified as the most important in detecting illicit accounts using the XGBoost model.

In [42], the authors employed the Correlation Attribute Evaluator in the Weka software tool to select the features for their prediction model. They discovered that out of 42 features, only six were effective in their prediction model. The authors evaluated their model on two datasets, one consisting of 42 features and the other consisting of the selected six features. The three machine learning algorithms, j48, RF, and KNN, were applied to both datasets. The results showed a significant improvement in time measurement using the three algorithms and an increase in the f-measure for the model that utilized the Random Forest algorithm.

Aziz et al. [43] proposed the utilization of the Light Gradient Boosting Machine (LGBM) algorithm for identifying fraud in Ethereum transactions. In their investigation, they assessed the performance of eight distinct machine learning algorithms, specifically, Logistic Regression, Random Forest, MLP Classifier, K-Nearest Neighbors (KNN), Extreme Gradient Boosting (XGBoost), Support Vector Classification (SVC), ADAboost, and LGBM Classifier. The dataset utilized in the study was obtained from Kaggle [44]. The outcomes of the experiment indicate that the LGBM algorithm exhibited the highest accuracy and was also memory-efficient. Additionally, the study identified that the attributes "Time Diff between first and last (Mins)" and "Unique received from addresses" were the most significant in identifying fraudulent transactions for the best performing model.

Chapter 4

Methodology

In this section, we will give a brief overview of the machine learning models that we have used and their performance measure metrics we have used to evaluate the models.

4.1 Decision Tree

The decision tree is a prevalent and robust machine learning tool that is utilized for classification and prediction purposes. It is structured like a tree, where each internal node represents a test on an attribute, each sub-tree portrays a test outcome, and each terminal node contains a class label. To train a decision tree, we split the source set into subsets based on an attribute value test. This process is called *recursive partitioning*. The recursive partitioning terminates when the subset at a node possesses the same target variable value, or when further splitting does not improve the predictions. Information gain and Gini impurity are two widely used methods for selecting the best attributes at each node. These methods are employed as splitting criteria for decision tree models. The split that results in higher information gain or lower Gini impurity is chosen. The equations for *Entropy*, *Information Gain*, and *Gini Impurity* are given by,

$$Entropy(S) = -\sum_{x} p(x) \log_2 p(x) \tag{4.1}$$

$$InformationGain(S, a) = Entropy(S) - \sum_{v} \frac{|S_v|}{|S|} Entropy(S_v) \tag{4.2}$$

$$GiniImpurity(S) = 1 - (p_i)^2$$

4.2 Random Forest Classifier

Random Forest Classifier (RF) is an ensemble learning technique used for classification tasks. It combines multiple decision trees (in figure) that are built on randomly sampled subsets of the data and predictor variables. The predictions are made by taking the majority vote of the classes predicted by each tree in the ensemble. This improves accuracy, stability, and reduces the risk of overfitting. Random Forest Classifier has several advantages over single decision trees, such as the ability to handle a large number of predictor variables and non-linear relationships between predictors and outcomes. However, it can be computationally expensive and difficult to interpret predictions, especially when dealing with a large number of trees. It is a powerful and widely used classification technique that is useful for analyzing and classifying large and complex datasets. It provides feature importance measurement that has been leveraged in different use cases.

4.3 Ensemble Learning

Ensemble learning refers to the practice of merging multiple machine learning models in order to enhance the accuracy of predictions. The combined ensemble model can be trained and employed for making predictions. Ensemble learning helps to overcome the bias-variance tradeoff by combining diverse models, thereby increasing the variance of the parameter. The bias-variance tradeoff principle states that reducing the bias in the estimated parameters may lead to an increase in the variance of the parameter estimated across different samples. Ensemble learning methods are typically categorized into three groups: Bagging, Boosting, and Stacking.

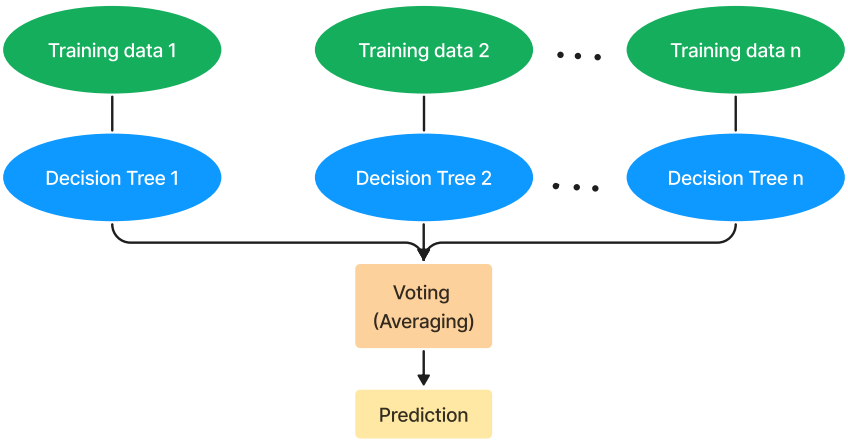

Figure 4.1: Random Forest Classifier

4.4 Bagging

Bagging is an ensemble learning technique that involves randomly sampling the training data to enhance the predictive performance of a single machine learning model. This method is particularly useful for models that tend to overfit the data. Bootstrapping is the approach used for subsampling the data, which involves random sampling with replacement, allowing the subsets of the training data to overlap. After obtaining the output from each model, the final prediction for each data point can be achieved using either regression voting, where predictions are the average of the contributing models, or classification voting, where predictions are based on the majority vote of the contributing models.

Given a data set $S = (x_1, y_1), ...(x_n, y_n)$ of size N, where $x_n \in X, y_n \in Y = 0, 1$, bagging constructs M classifiers $h_m, m = 1, ..., M$ with bootstrap replicas S_m of S, where S_m is obtained by drawing examples from the original data set S with replacement, usually having the same number of examples as S. The pseudo-code for bagging is shown in Algorithm 1, where I(W) is the indicator function of event W.

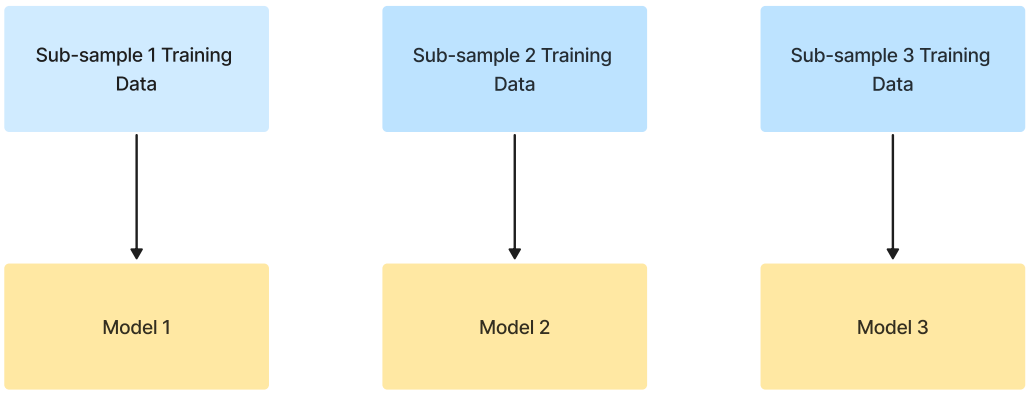

Figure 4.2: Training Sub-sample data in bagging ensemble learning method

4.5 Online Learning

Incremental learning, also known as online machine learning, is a type of machine learning in which a model learns continuously from new data, updating its parameters incrementally over time instead of being trained on a fixed dataset. This is different from traditional machine learning, where a model is typically trained on a fixed dataset and then used to make predictions on new, unseen data.

There are several advantages to incremental learning over traditional machine learning. Firstly, incremental learning allows models to adapt to changing environments and evolving data distributions. In many real-world applications, the underlying data generating process can change over time, and the ability to adapt to such changes is crucial for maintaining high predictive accuracy. Secondly, incremental learning can be more efficient in terms of computation and memory usage compared to batch learning, as it updates the model incrementally, only requiring small computations for each new data point. This makes incremental learning particularly useful for applications with large and continuously growing datasets. Lastly, incremental learning can improve the generalization ability of a model by training it on a more diverse and representative set of data over time. In traditional machine learning, the model is trained on a fixed dataset, which may not fully represent the range of data that the model will be exposed to in the future. Incremental learning,

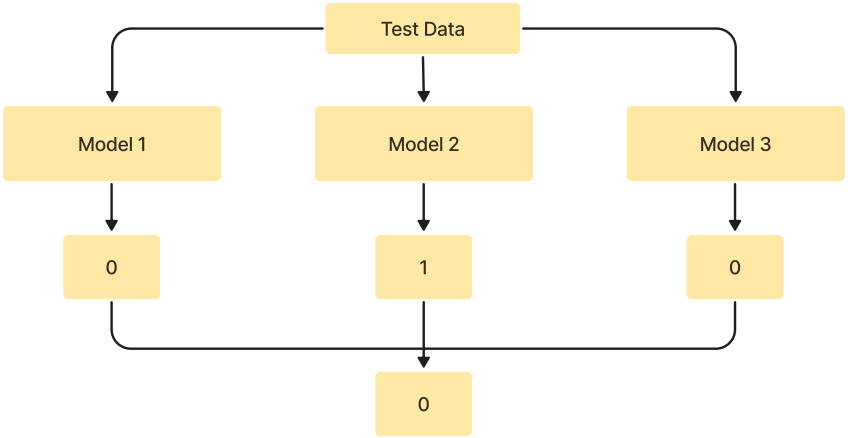

Figure 4.3: Voting for binary classification in bagging ensemble learning method

on the other hand, allows the model to gradually learn from new and more representative data over time, leading to better generalization performance.

However, incremental learning also presents some challenges, such as the need to balance exploration and exploitation of new data, handling concept drift, and avoiding catastrophic forgetting of previously learned information. Nonetheless, these challenges have been addressed through various techniques and algorithms, making incremental learning a promising approach for a wide range of applications.

In online machine learning, the mean, count, standard deviation, and variance can be updated in real-time as new data becomes available. Here's how they can be calculated: Mean and Count: The mean and count at any moment can be calculated using the following update rule.

$$mean_{new} = mean_{old} + \frac{(x - mean_{old})}{(n+1)}$$

$$new_{count} = old_{count} + 1$$

where old_{mean} and old_{count} are the old mean and count, x is the new data point, and n is the old count.

Standard Deviation and Variance : The standard deviation and variance can also be updated in real-time using the following equations:

$$new_{variance} = (old_{count} * old_{variance} + \frac{(x - old_{mean} * (x - new_{mean}))}{(old_{count} + 1)}$$

$$new_{std_deviation} = sqrt(new_{variance})$$

where $old_{variance}$ and old_{mean} are the old variance and mean, x is the new data point, and new_mean is the new mean calculated using the update rule above.

These equations allow the mean, count, standard deviation, and variance to be updated in real-time as new data becomes available, making it possible to perform online machine learning tasks that require these statistical measures. This can be particularly useful in scenarios where data is arriving continuously and the model needs to adapt and update its parameters in real-time.

4.6 Online Bagging

Bootstrap aggregating or bagging is a type of ensemble method used in machine learning. Ensemble methods involve combining multiple weaker models known as base models to improve performance rather than using a single machine learning model. Online bagging is a variant of bagging that has been shown to perform comparably to batch bagging, while also being faster. In a study by Nikunj et al. [45], both theoretical and experimental evidence demonstrated the effectiveness of online bagging. Figure X illustrates the Online Bagging algorithm.

Algorithm 1 Online Bagging Algorithm

```
1: function OnlineBagging(h, L_0, d)
2:     For each base model L_m ∈ h, m ∈ (1, 2, ...M),
3:         Set k according to Poisson (1).
4:             Do k times
5:                 L_m ∈ h, m ∈ (1, 2, ...M)
6: end function
```

4.7 Performance metrics

Performance metrics in machine learning are used to evaluate the effectiveness and efficiency of an algorithm in making predictions or classifications. These metrics help to assess the accuracy, precision, recall, and other important parameters of a model.The following are the performance metrics used to determine the usefulness of the models:

1. Accuracy: This measures the proportion of correctly classified instances over the total number of instances (Equation 4.3).

2. Precision: This measures the proportion of true positive instances (correctly predicted positives) over the total number of predicted positive instances (Equation 4.4).

3. Recall: This measures the proportion of true positive instances over the total number of actual positive instances (Equation 4.5).

4. F1 score: This is the harmonic mean of precision and recall and is used to balance the trade-off between them (Equation 4.6).

5. Confusion matrix: This is a table that summarizes the true positive, false positive, true negative, and false negative results of a classification algorithm .

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{4.3}$$

$$Precision = \frac{TP}{TP + FP} \tag{4.4}$$

$$Recall = \frac{TP}{TP + FN} \tag{4.5}$$

$$F1 = \frac{2 * Precision * Recall}{Precision + Recall} = \frac{2 * TP}{2 * TP + FP + FN} \tag{4.6}$$

Where TP = True Positive, TN = True Negative, FP = False Positive, FN = False Negative

Chapter 5

Data Collection And Preprocessing

5.1 Data Collection

As mentioned in the RELATED WORK section, different approaches have been employed to classify accounts and detect fraudulent transactions in Blockchain. In this paper, we present five different supervised learning models to distinguish fraudulent transactions from valid transactions in the Ethereum Public Blockchain platform. We used a labeled dataset from Kaggle [44]. The dataset contains 9841 records in total, out of which 7662 (77.86%) are of non–fraudulent transactions and 2179 (22.14%) are of fraudulent transactions. Each transaction record has 47 features such as : average and total time between sent and received transactions for account in minutes, minimum and maximum value received, unique received address, total ether sent, number of created contracts and so on. The transactions have been classified based on their labels as a "Fraud" or "Non-Fraud" respectively.

We also created a new dataset consisting of transaction-related addresses, which was composed of both benign and malignant addresses. The malignant addresses were collected from CryptoscamDB [46], while the benign addresses were collected from Etherscan [47], Ethplorer [48], and Etherchain [49]. Specifically, 150 Ethereum addresses were randomly collected from the daily transaction history over a period of 30 days. These addresses were then combined, and a check was performed to identify any scam addresses in the collected pool. As a result of this check, a total of 28 scam addresses were identified and removed, along with any duplicate addresses.

Ultimately, this process resulted in the creation of a final dataset of 13,223 benign addresses.

A feature extractor was developed to extract features from Ethereum addresses. To collect transaction-related data of the Ethereum dataset, four open-source APIs provided by Etherscan [50] were utilized. A concise explanation of each API and its designated use case is presented below.

Get Ether Balance for a Single Address : This api provides the amount of Ether held by a specified address. We used this api to get the total ether balance of the addresses. This API was employed to obtain the overall Ether balance of the addresses.

Get a list of 'Normal' Transactions By Address : This api provides a record of all the transactions executed by a particular address. The record conforms to JSON format and includes transaction data, with each transaction encompassing various attributes such as "timeStamp", "from", "to", "contractAddress", "isError", and "value". From these attributes, a total of 21 features were derived.

Get Contract ABI for Verified Contract Source Codes : This api provides the Application Binary Interface (ABI) of a verified smart contract. This api was used to check whether an address is a contract address or an account address.

Get a list of 'ERC20 - Token Transfer Events' by Address : This api provides a list of ERC-20 tokens that were transferred by a particular address, with an optional feature to filter the results by the token contract used. We have extracted 21 features related to ERC-20 token transactions from the information present in the token transfer transaction. This information includes the "timeStamp", "from", "to", "contractAddress", "isError", and "value".

In the subsequent section, a succinct overview of CryptoScamDB and Etherscan will be provided as they constitute our primary sources of data.

CryptoScamDB is a platform that tracks and reports cryptocurrency scams, aiming to ed-

ucate people about common cryptocurrency scams, provide tools for checking the legitimacy of cryptocurrency projects, and raise awareness about cryptocurrency security. It operates as an open-source project and offers an API for developers to integrate its scam database into their applications.

Etherscan is a blockchain explorer that provides users with a variety of tools and services to explore, analyze, and interact with the Ethereum blockchain. It allows users to view transaction details, search for addresses, monitor network activity, and track the progress of smart contracts. Etherscan also provides data and insights into the performance of the Ethereum network, including gas prices and block times. It is a popular and widely used platform for Ethereum users, developers, and businesses, and is considered a valuable resource for navigating and understanding the Ethereum blockchain.

Our dataset is composed of 18057 addresses where 4834 addresses (26.77%) are scam related and 13223 (73.23%) addresses are benign addresses. We extract 43 features for each record. Our new dataset contains 2655 new scams related addresses than the dataset from kaggle. The features in our dataset is according to the transection of the addresses till 15 december 2022. The features in our dataset are more consistent than the features in the kaggle dataset.

Feature Name		Description	Data Type
1	Avg min between sent tnx	Average time between sent transactions for account in minutes	float64
2	Avg min between received tnx	Average time between received transactions for account in minutes	float64
3	Time Diff between first and_last (Mins)	Time difference between the first and last transaction	int64
4	Sent_tnx	Total number of sent normal transactions	int64
5	Received_tnx	Total number of received normal transactions	int64
6	NumberofCreated_Contracts	Total Number of created contract transactions	int64
7	UniqueReceivedFrom_Addresses	Total Unique addresses from which account received transactions	int64
8	UniqueSentTo_Addresses20	Total Unique addresses from which account sent transactions	int64
9	MinValueReceived	Minimum value in Ether ever received	float64
10	MaxValueReceived	Maximum value in Ether ever received	float64
11	AvgValueReceived5Average	Average value in Ether ever received	float64
12	MinValSent	Minimum value of Ether ever sent	float64
13	MaxValSent	Maximum value of Ether ever sent	float64
14	AvgValSent	Average value of Ether ever sent	float64
15	MinValueSentToContract	Minimum value of Ether sent to a contract	float64
16	MaxValueSentToContract	Maximum value of Ether sent to a contract	float64
17	AvgValueSentToContract	Average value of Ether sent to contracts	float64
18	TotalTransactions (IncludingTnxtoCreate_Contract)	Total number of transactions	int64
19	TotalEtherSent	Total Ether sent for account address	float64
20	TotalEtherReceived	Total Ether received for account address	float64
21	TotalEtherSent_Contracts	Total Ether sent to Contract addresses	float64
22	TotalEtherBalance	Total Ether Balance following enacted transactions	object
23	TotalERC20Tnxs:	Total number of ERC20 token transfer transactions	int64

Table 5.1: 1st to 23rd feature of Etherium addresses extracted from Etherscan

Feature Name		Description	Data Type
24	ERC20TotalEther_Received:	Total ERC20 token received transactions in Ether	object
25	ERC20TotalEther_Sent:	Total ERC20 token sent transactions in Ether	object
26	ERC20TotalEtherSentContract:	Total ERC20 token transfer to other contracts in Ether	object
27	ERC20UniqSent_Addr:	Number of ERC20 token transactions sent to Unique account addresses	int64
28	ERC20UniqRec_Addr:	Number of ERC20 token transactions received from Unique addresses	int64
29	ERC20UniqRecContractAddr:	Number of ERC20 token transactions received from Unique contract addresses	int64
30	ERC20AvgTimeBetweenSent_Tnx	Average time between ERC20 token sent transactions in minutes	float64
31	ERC20AvgTimeBetweenRec_Tnx	Average time between ERC20 token received transactions in minutes	float64
32	ERC20AvgTimeBetweenContract_Tnx	Average time ERC20 token between sent token transactions	float64
33	ERC20MinVal_Rec	Minimum value in Ether received from ERC20 token transactions for account	float64
34	ERC20MaxVal_Rec	Maximum value in Ether received from ERC20 token transactions for account	float64
35	ERC20AvgVal_Rec	Average value in Ether received from ERC20 token transactions for account	float64
36	ERC20MinVal_Sent	Minimum value in Ether sent from ERC20 token transactions for account	float64
37	ERC20MaxVal_Sent	Maximum value in Ether sent from ERC20 token transactions for account	object
38	ERC20AvgVal_Sent	Average value in Ether sent from ERC20 token transactions for account	float64
39	ERC20UniqSentTokenName	Number of Unique ERC20 tokens transferred	int64
40	ERC20UniqRecTokenName	Number of Unique ERC20 tokens received	category
41	ERC20MostSentTokenType	Most sent token for account via ERC20 transaction	object
42	ERC20MostRecTokenType	Most received token for account via ERC20 transaction	int64

Table 5.2: 24th to 42th feature of Etherium addresses extracted from Etherscan

Chapter 6

Training and Evaluation

For the sake of benchmarking against other popular proposed models in scam detection we chose [42] and [41] and compared the results of our proposed model with them in terms of performance metrics (accuracy, precision, recall and f1 score) and model building time measurement for different size of datasets.

Ibrahim at el. [42] proposed a Fraud detection model using Decision Tree, Random Forest and K-nearest neighbors (KNN) algorithms. We trained three classification models using these three algorithms with increasing size of datasets and plotted the performance metrics vs. size graphs (Figure 6.1[1], Figure 6.2[2] and Figure 6.3[3]) to demonstrate the differences in the trend of performance between the models built using traditional Offline learning and Online learning.

We built the models again but incrementally. Albeit, the performance metrics (see Figure 6.4, Figure 6.5 and Figure 6.6) did not look great in the beginning, as the size of the dataset grew the differences faded. Online learning was proven better on two counts. Firstly on unseen data all three models trained in offline mode performed significantly poorer compared to their performances on test data and, secondly, the time taken for training models as the size of the dataset grew was linearly growing, which was not the case when they were trained online. Figure 6.7, Figure 6.8, Figure 6.9 and Table 6.1 show the Time taken for retraining the models for every 10 new records for the traditional offline learning vs. the online learning using Decision Tree and Random Forest algorithms respectively. From this table we get a clear picture of the advantage of online learning

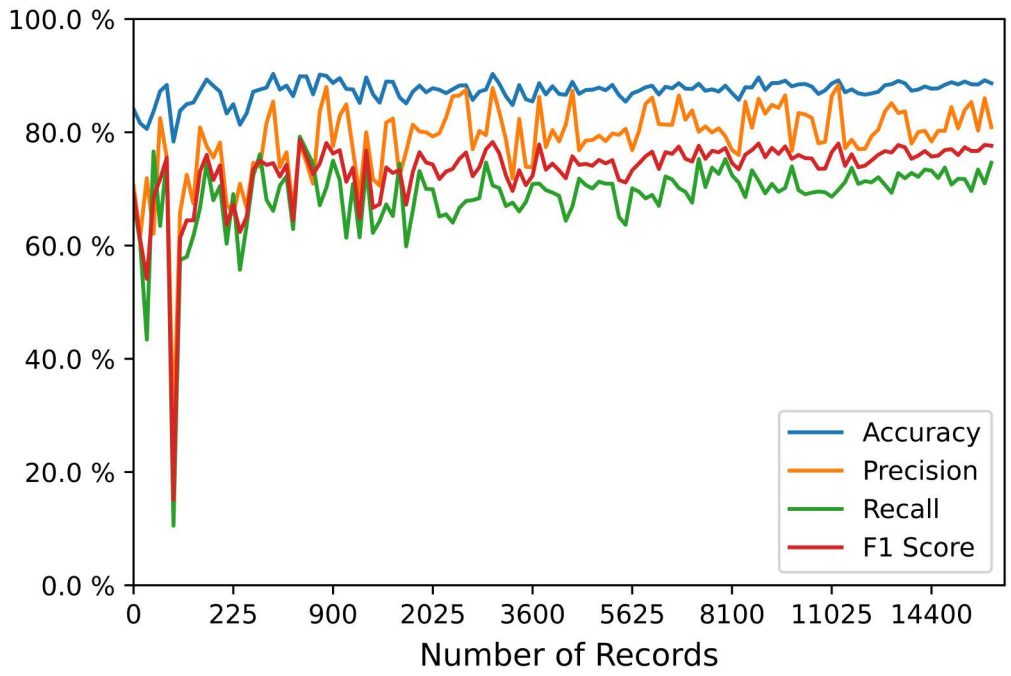

Figure 6.1: Decision Tree

over offline learning in case of streaming data.

Farrugia at el. [41] proposed XGBoost classifier to detect illicit accounts in the Ethereum public blockchain based on their transaction history. While their proposed model performed extremely well for their dataset with an average accuracy of 96.3 percent, its accuracy came down to 92 percent when tested against unseen data which is considered good for a machine learning model. But the real problem with the XGBoost classifier lies in the amount of time and resources it consumes during training. It took more than 600 times the amount taken by our proposed Online Bagging ensemble model to be trained and it grows as the size of the dataset increases (Figure:6.10).

Using the Online adaptation of the Bagging ensemble, we achieved more than 92 percent accuracy and an f1 score of 86 percent. With the introduction of drift detection retraining technique along with hyperparameter tuning and scaling of the data, the model was further improved and

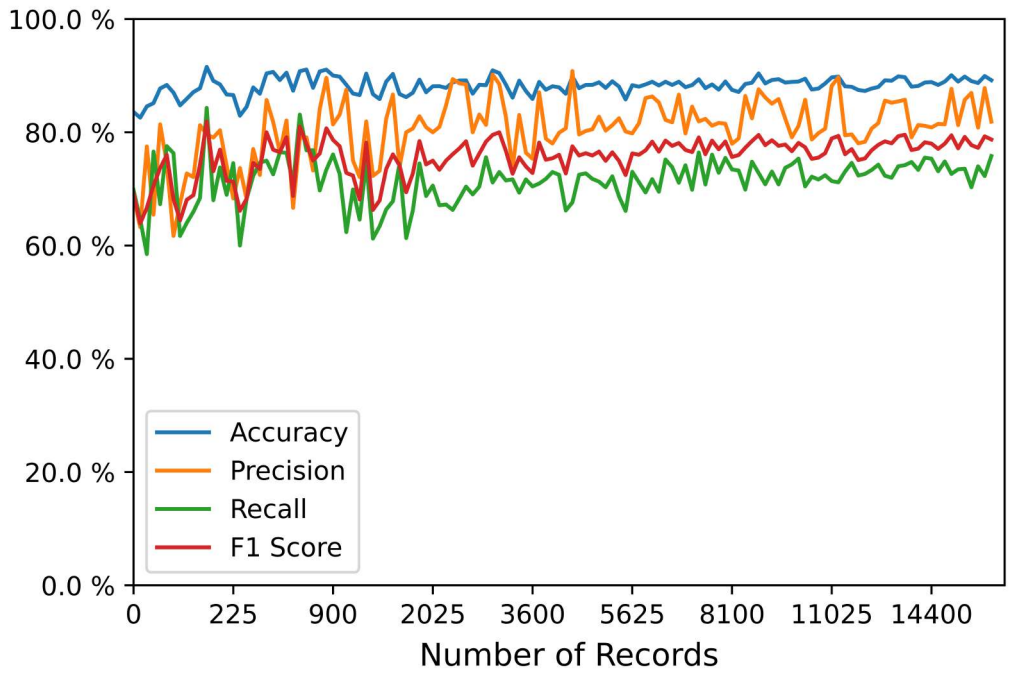

Figure 6.2: Random Forest

the resulting accuracy was 94.22 percent and the f1 score was improved to 89.19 percent. Finally, replacing Poisson's distribution with the modified log-normal probability distribution in Online Bagging algorithm improved the performance even further. With this approach, we eventually were able to achieve an average accuracy of 96 percent and F1 score above 92 percent. But the most significant positive findings was the model retraining time measurement. It required significantly less time to train with a growing dataset. In fact, the retraining time was independent of the volume of the dataset since no previous data record needed to be fed to the model. The model was only fitted to the new stream of data. Figure 6.11 shows the feature importance of each of the 7 base learners of our model. Table 6.2 shows the performance of all of the models built as a part of the study.

No. of records	Decision Tree		Random Forest	
	Offline ML	Online ML	Offline ML	Online ML
18007+10	100.7	59.6	1880.34	59.0
18017+10	89.7	71.9	1856.95	86.5
18027+10	140.5	13.7	1871.99	31.0
18037+10	154.1	37.9	1908.53	48.1
18047+10	157.1	17.6	1848.55	31.9

Table 6.1: Model Training Time (Offline Batch Learning vs Online Learning)

Algorithm	**Accuracy**	**Precision**	**Recall**	**F1 Score**
Decision Tree	0.90	0.86	0.74	0.79
Random Forest	0.90	0.87	0.75	0.80
K-Nearest Neighbors	0.87	0.79	0.72	0.75
Online Descision tree	0.84	0.65	0.91	0.76
Online Random Forest	0.92	0.84	0.89	0.86
Online K-Nearest Neighbors	0.87	0.74	0.81	0.77
XGBoost Classifier	0.92	0.86	0.94	0.89
Online Bagging with Drift Detection Retraining	**0.96**	**0.93**	**0.92**	**0.93**

Table 6.2: Performance Matics (Accuracy, Precision, Recall, F1)

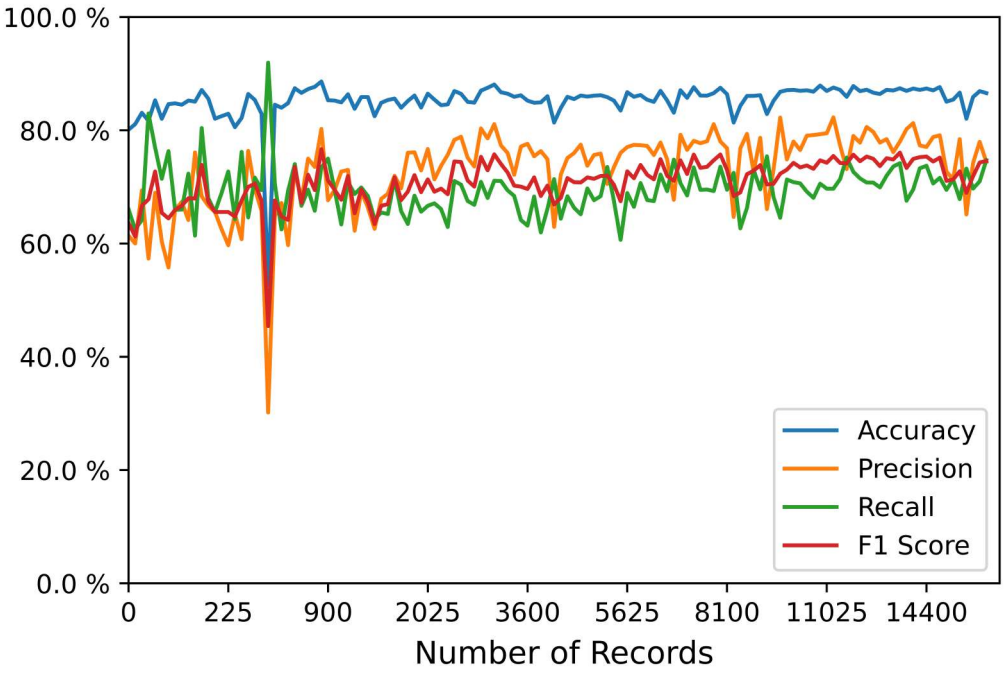

Figure 6.3: K-Nearest Neighbors

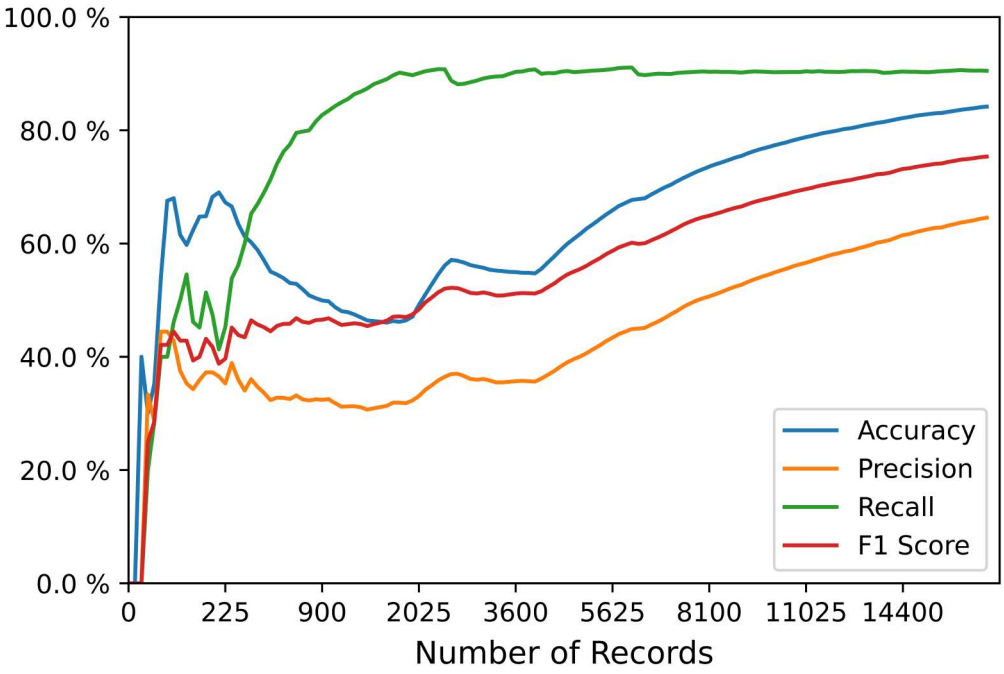

Figure 6.4: Decision Tree (Online Learning)

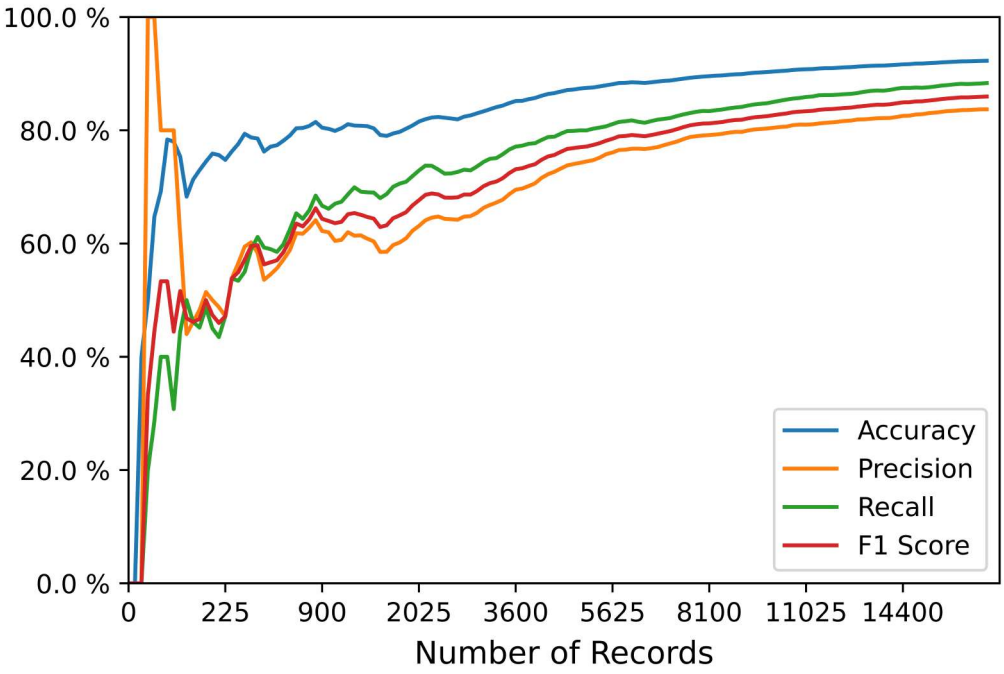

Figure 6.5: Random Forest (Online Learning)

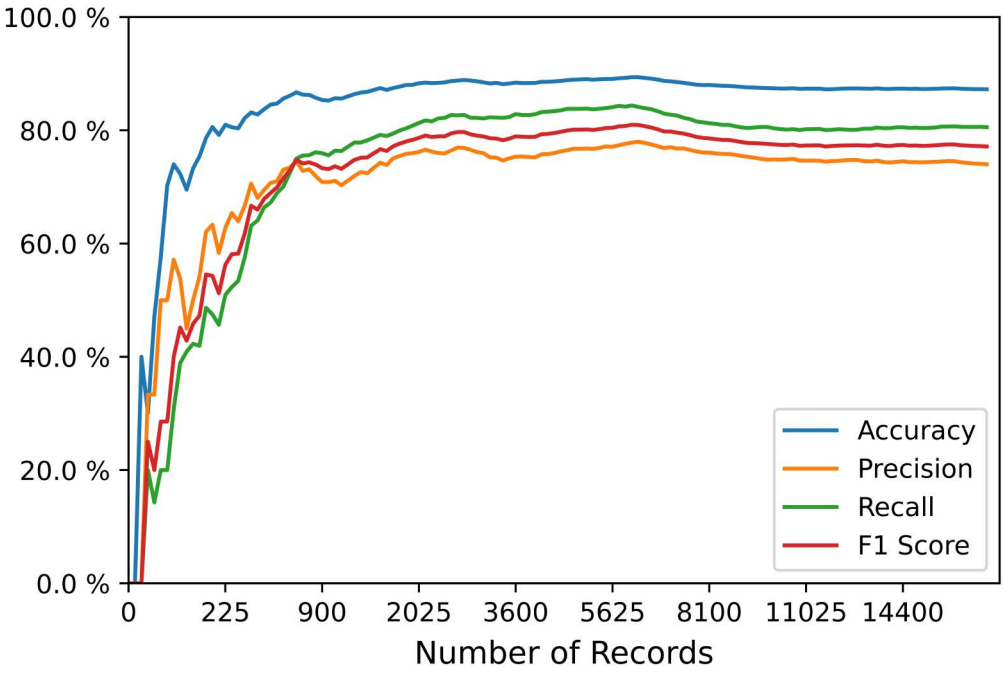

Figure 6.6: K-Nearest Neighbors (Online Learning)

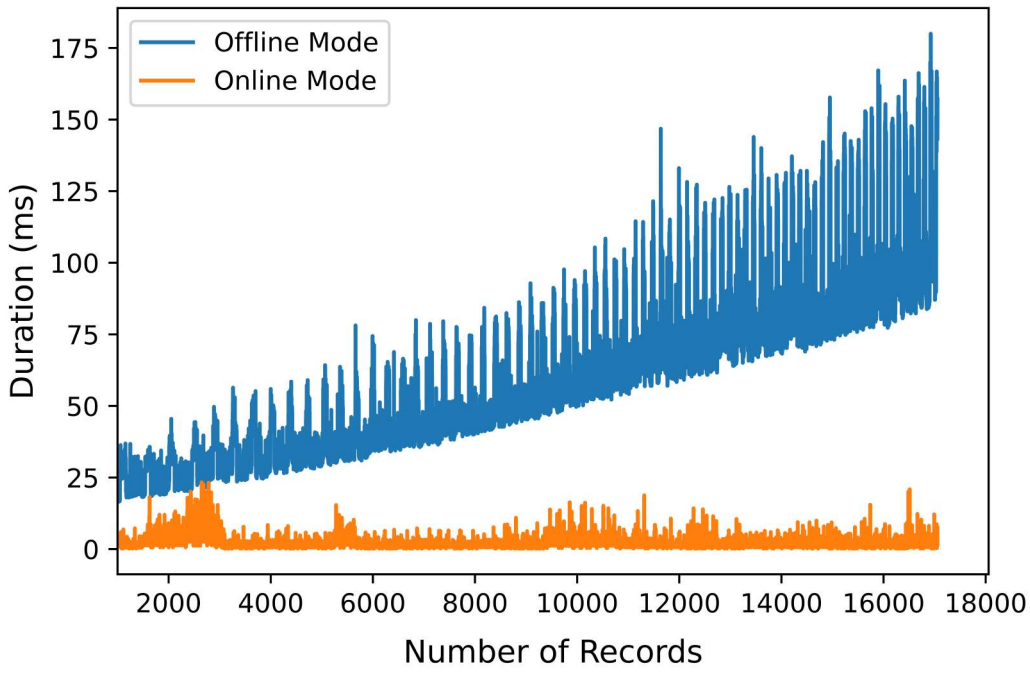

Figure 6.7: Time Comparison for Decision Tree (Online vs. Offline)

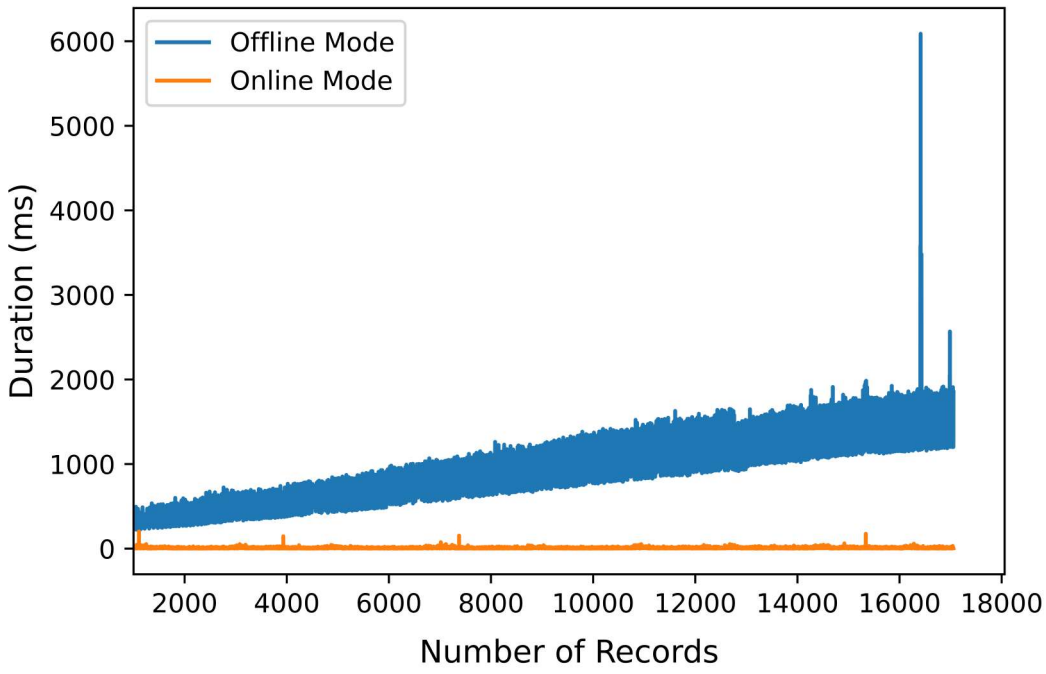

Figure 6.8: Training Time Comparison for Random Forest (Online vs. Offline)

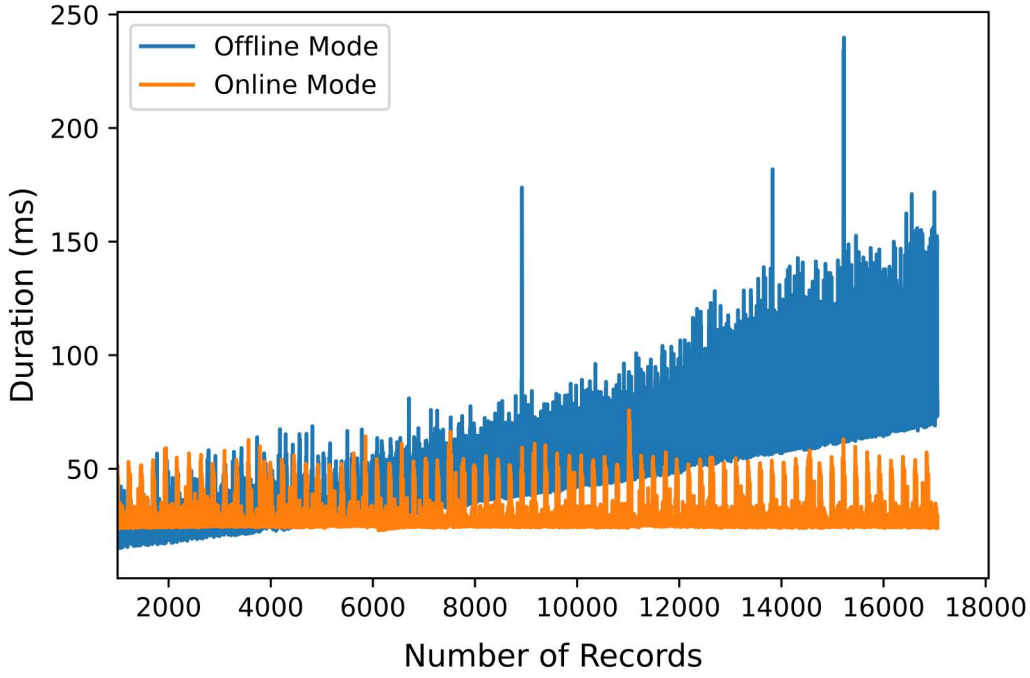

Figure 6.9: Training Time Comparison for K-Nearest Neighbors (Online vs. Offline)

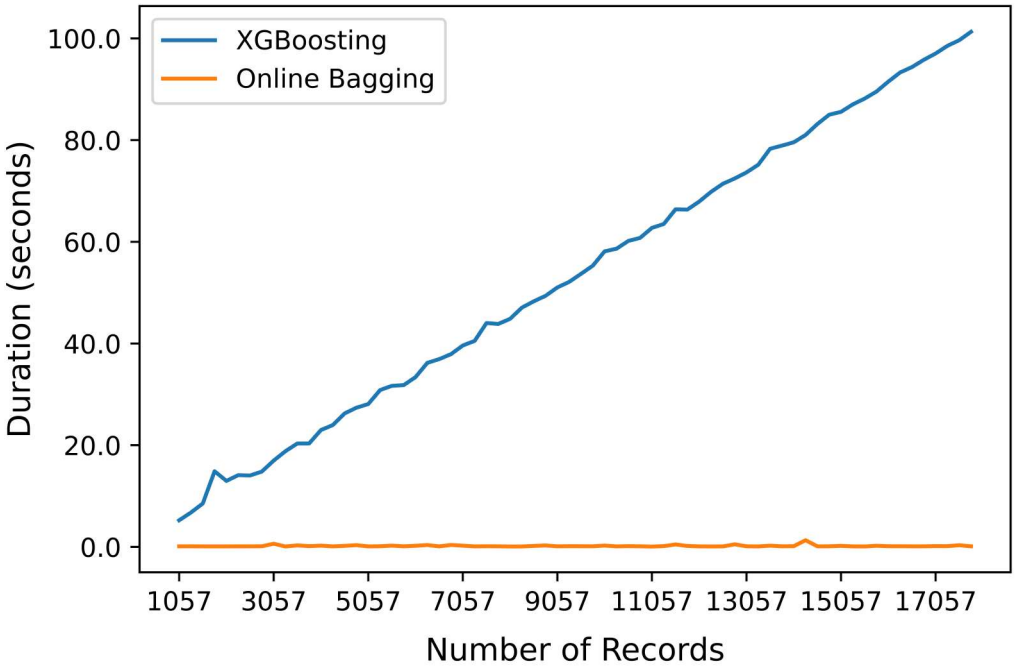

Figure 6.10: XGB vs. Online Bagging.png

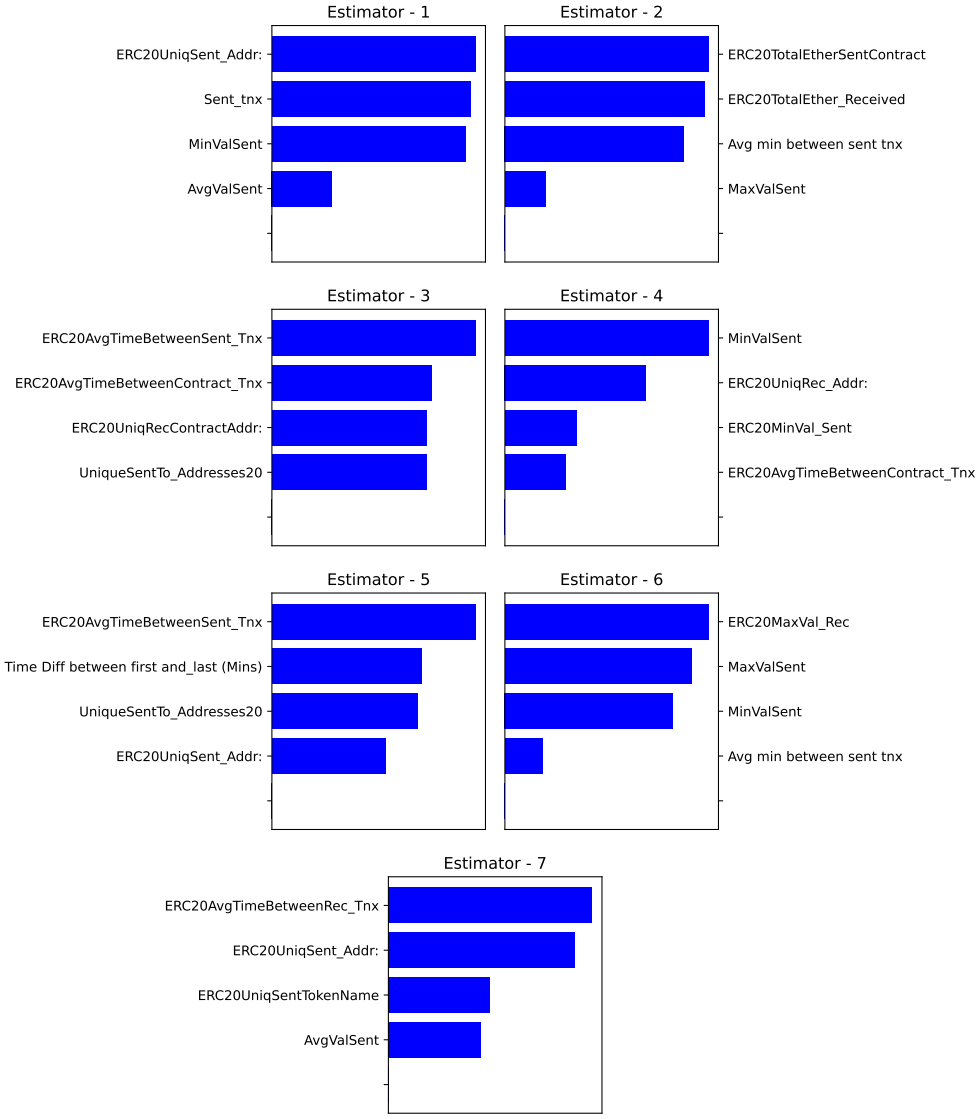

Figure 6.11: Feature Importance of Base Learners

Chapter 7

Conclusion and Future Work

This study aimed to build an efficient fraudulent actor detection classifier for the Ethereum public blockchain network at the account level. The main focus was on building a computationally efficient model that requires minimum resources and time to train and can adapt to change in the distribution and trend of data. While traditional offline machine learning models work for a given set of data, they suffer from data drift in the long run and have to be retrained again and again which is expensive. Online Bagging ensemble technique addresses all of the shortcomings of the traditional machine learning classifier in case of streaming data while also maintaining good performance. It can detect both data drift and concept drift, pick up new emerging trends in the data and forget obsolete patterns of the data. The biggest advantage of Online Bagging method over the traditional machine learning approach is the ability to adjust its parameters seeing only the new data stream without having to see any of the previous data again. While all Online learning techniques can learn from streaming data, Bagging method accounts for imbalanced data and balances the bias-variance trade-off.

In this study, we worked with a total of 18057 addresses which is a minute fraction of the total addresses transacting on Ethereum platform everyday. A larger dataset would help us further evaluate our proposed approach and build a better model. In future, we plan to aggregate a much larger dataset and fine tune our approach. It is also possible to work with a subset of features that we have worked with to further reduce the complexities and resource demands of the model training and testing.

References

[1] "Bitcointalk." [Online]. Available: https://bitcointalk.org/index.php?topic=633822.0

[2] "Bitcoin mixer." [Online]. Available: https://bitcoinmixers.cc/

[3] J. Kim, M. Nakashima, W. Fan, S. Wuthier, X. Zhou, I. Kim, and S.-Y. Chang, "Anomaly detection based on traffic monitoring for secure blockchain networking," in *2021 IEEE International Conference on Blockchain and Cryptocurrency (ICBC)*. IEEE, 2021, pp. 1–9.

[4] Q. Yuan, B. Huang, J. Zhang, J. Wu, H. Zhang, and X. Zhang, "Detecting phishing scams on ethereum based on transaction records," in *2020 IEEE International Symposium on Circuits and Systems (ISCAS)*. IEEE, 2020, pp. 1–5.

[5] E. Badawi, G.-V. Jourdan, G. Bochmann, and I.-V. Onut, "An automatic detection and analysis of the bitcoin generator scam," in *2020 IEEE European Symposium on Security and Privacy Workshops (EuroS&PW)*. IEEE, 2020, pp. 407–416.

[6] R. Phillips and H. Wilder, "Tracing cryptocurrency scams: Clustering replicated advance-fee and phishing websites," in *2020 IEEE International Conference on Blockchain and Cryptocurrency (ICBC)*. IEEE, 2020, pp. 1–8.

[7] "What is blockchain technology? - ibm blockchain." [Online]. Available: https://www.ibm.com/topics/what-is-blockchain

[8] "Blockchain and intellectual property." [Online]. Available: https://www.wipo.int/cws/en/blockchain-and-ip.html

[9] S. Nakamoto, "Bitcoin: A peer-to-peer electronic cash system," *Decentralized Business Review*, p. 21260, 2008.

[10] V. Buterin *et al.*, "A next-generation smart contract and decentralized application platform," *white paper*, vol. 3, no. 37, pp. 2–1, 2014.

[11] A. Schlabach, "Q3 2019 cryptocurrency anti-money laundering report," Feb 2020. [Online]. Available: https://ciphertrace.com/q3-2019-cryptocurrency-anti-money-laundering-report/

[12] M. Vasek and T. Moore, "Analyzing the bitcoin ponzi scheme ecosystem," in *International Conference on Financial Cryptography and Data Security*. Springer, 2018, pp. 101–112.

[13] K. Toyoda, P. T. Mathiopoulos, and T. Ohtsuki, "A novel methodology for hyip operatorsâ bitcoin addresses identification," *IEEE Access*, vol. 7, pp. 74 835–74 848, 2019.

[14] M. Vasek and T. Moore, "Thereâs no free lunch, even using bitcoin: Tracking the popularity and profits of virtual currency scams," in *International conference on financial cryptography and data security*. Springer, 2015, pp. 44–61.

[15] M. Bartoletti, B. Pes, and S. Serusi, "Data mining for detecting bitcoin ponzi schemes," in *2018 Crypto Valley Conference on Blockchain Technology (CVCBT)*. IEEE, 2018, pp. 75–84.

[16] C. Brenig, G. Müller *et al.*, "Economic analysis of cryptocurrency backed money laundering," 2015.

[17] M. Möser, R. Böhme, and D. Breuker, "An inquiry into money laundering tools in the bitcoin ecosystem," in *2013 APWG eCrime researchers summit*. Ieee, 2013, pp. 1–14.

[18] M. Spagnuolo, F. Maggi, and S. Zanero, "Bitiodine: Extracting intelligence from the bitcoin network," in *International conference on financial cryptography and data security*. Springer, 2014, pp. 457–468.

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

[20] S. Bistarelli, M. Parroccini, and F. Santini, "Visualizing bitcoin flows of ransomware: Wannacry one week later." in *ITASEC*, 2018.

[21] "Welcome to fbi.gov," Mar 2022. [Online]. Available: https://www.fbi.gov/?came_from=https%3A%2F%2Fwww.fbi.gov%2Fscams-and-safety%2Fcommon-scams-and-crimes%2Fretracted-pages%2Fadvance-fee-schemes

[22] E. Badawi and G.-V. Jourdan, "Cryptocurrencies emerging threats and defensive mechanisms: A systematic literature review," *IEEE Access*, vol. 8, pp. 200 021–200 037, 2020.

[23] A. Higbee, "The role of crypto-currency in cybercrime," *Computer Fraud & Security*, vol. 2018, no. 7, pp. 13–15, 2018.

[24] M. Sigalos, "Crypto criminals laundered $540 million by using a service called renbridge, new report shows," Aug 2022. [Online]. Available: https://www.cnbc.com/2022/08/10/crypto-criminals-laundered-540-million-using-renbridge-elliptic-says.html

[25] "Common scams and crimes," May 2016. [Online]. Available: https://www.fbi.gov/scams-and-safety/common-scams-and-crimes

[26] J. Frankenfield, "Initial coin offering (ico): Coin launch defined, with examples," Sep 2022. [Online]. Available: https://www.investopedia.com/terms/i/initial-coin-offering-ico.asp

[27] "What is blockchain?" [Online]. Available: https://www.oracle.com/middleeast/blockchain/what-is-blockchain/

[28] D. Puthal and S. P. Mohanty, "Proof of authentication: Iot-friendly blockchains," *IEEE Potentials*, vol. 38, no. 1, pp. 26–29, 2018.

[29] [Online]. Available: https://coinmarketcap.com/

[30] X. F. Liu, X.-J. Jiang, S.-H. Liu, and C. K. Tse, "Knowledge discovery in cryptocurrency transactions: A survey," *IEEE Access*, vol. 9, pp. 37 229–37 254, 2021.

[31] K. Toyoda, T. Ohtsuki, and P. T. Mathiopoulos, "Identification of high yielding investment programs in bitcoin via transactions pattern analysis," in *GLOBECOM 2017-2017 IEEE Global Communications Conference*. IEEE, 2017, pp. 1–6.

[32] W. Chen, Z. Zheng, J. Cui, E. Ngai, P. Zheng, and Y. Zhou, "Detecting ponzi schemes on ethereum: Towards healthier blockchain technology," in *Proceedings of the 2018 world wide web conference*, 2018, pp. 1409–1418.

[33] M. Bartoletti, S. Carta, T. Cimoli, and R. Saia, "Dissecting ponzi schemes on ethereum: identification, analysis, and impact," *Future Generation Computer Systems*, vol. 102, pp. 259–277, 2020.

[34] "Bitcointalk." [Online]. Available: https://bitcointalk.org/

[35] T. B. P. 2014-2021. [Online]. Available: https://badbitcoin.org/thebadlist/

[36] A. Holub and J. O'Connor, "Coinhoarder: Tracking a ukrainian bitcoin phishing ring dns style," in *2018 APWG Symposium on Electronic Crime Research (eCrime)*. IEEE, 2018, pp. 1–5.

[37] M. Conti, A. Gangwal, and S. Ruj, "On the economic significance of ransomware campaigns: A bitcoin transactions perspective," *Computers & Security*, vol. 79, pp. 162–189, 2018.

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

[39] Kaspersky, "Cryptolocker virus definition," Mar 2022. [Online]. Available: https://usa.kaspersky.com/resource-center/definitions/cryptolocker

[40] Y. Hu, S. Seneviratne, K. Thilakarathna, K. Fukuda, and A. Seneviratne, "Characterizing and detecting money laundering activities on the bitcoin network," *arXiv preprint arXiv:1912.12060*, 2019.

[41] S. Farrugia, J. Ellul, and G. Azzopardi, "Detection of illicit accounts over the ethereum blockchain," *Expert Systems with Applications*, vol. 150, p. 113318, 2020.

[42] R. F. Ibrahim, A. M. Elian, and M. Ababneh, "Illicit account detection in the ethereum blockchain using machine learning," in *2021 International Conference on Information Technology (ICIT)*. IEEE, 2021, pp. 488–493.

[43] R. M. Aziz, M. F. Baluch, S. Patel, and A. H. Ganie, "Lgbm: a machine learning approach for ethereum fraud detection," *International Journal of Information Technology*, pp. 1–11, 2022.

[44] V. Aliyev, "Ethereum fraud detection dataset," Jan 2021. [Online]. Available: https://www.kaggle.com/datasets/vagifa/ethereum-frauddetection-dataset

[45] N. Oza and S. Russell, "Online bagging and boosting," *Proceedings of Artificial Intelligence and Statistics*, 01 2001.

[46] "Cryptoscamdb." [Online]. Available: https://cryptoscamdb.org/

[47] "Etherscan." [Online]. Available: https://etherscan.io/

[48] "Ethplorer." [Online]. Available: https://ethplorer.io/

[49] "Etherchain." [Online]. Available: https://beaconcha.in/

[50] "Etherscan api." [Online]. Available: https://etherscan.io/apis

Appendices

Appendix A

Data Preprocessing

The initial data was composed of 9841 rows and 50 columns. The first two rows Index and Address are omitted at the beginning. The dataset was loaded and the object type data were converted into category type and features that are not useful were discarded.

```

# Load dataset
dataframe = pandas.read_csv('../input/ethereum-frauddetection-dataset/
    transaction_dataset.csv', index_col=0)

# Ommit first two columns (Index, Adress)
dataframe = dataframe.iloc[:,2:]

# Turn object variables into 'category' dtype for more computation efficiency
categories = dataframe.select_dtypes('O').columns.astype('category')

# Drop the two categorical features
dataframe.drop(dataframe[categories], axis=1, inplace=True)
```

Listing A.1: Load and Prepare Dataset

```

pyplot.figure(figsize=(12,6))
seaborn.heatmap(dataframe.isnull(), cbar=False)
pyplot.show()
```

Listing A.2: Visualize The Missing Pattern of The Dataset

```

dataframe.fillna(dataframe.median(), inplace=True)
```

Listing A.3: Fill in Missing Values With Median

Appendix B

Feature Reduction

In this aection we checked the correlation matrix of the resulting dataset. Highly correlated features were dropped. We also investigated the frequency distribution of the features using box plots. Two features that contained zero in most of the records were also dropped and finally remaining 18 features were used for model training and testing.

```
1
2 # Filtering the features with 0 variance
3 zero_variance = dataframe.var() == 0
4
5 # Drop features with 0 variance --- these features will not help in the
      performance of the model
6 dataframe.drop(dataframe.var()[zero_variance].index, axis = 1, inplace = True)
```

Listing B.1: Discard Features With Zero Variance

```
1
2 correlatin_matrix = daraframe.corr()
3
4 mask = np.zeros_like(correlation_matrix)
5
6 mask[np.triu_indices_from(mask)]=True
7 with seaborn.axes_style('white'):
8     figure, axis = pyplot.subplots(figsize=(18,10))
9     seaborn.heatmap(correlation_matrix,  mask=mask, annot=False, cmap='CMRmap',
       center=0, linewidths=0.1, square=True)
```

Listing B.2: Checking the Correlation matrix

```

highly_correlated_features = ['total transactions (including tnx to create
    contract', 'total ether sent contracts', 'max val sent to contract', 'ERC20
     avg val rec', 'ERC20 avg val rec', 'ERC20 max val rec', 'ERC20 min val rec
    ', 'ERC20 uniq rec contract addr', 'max val sent', 'ERC20 avg val sent','
    ERC20 min val sent', 'ERC20 max val sent', 'Total ERC20 tnxs', 'avg value
    sent to contract', 'Unique Sent To Addresses', 'Unique Received From
    Addresses', 'total ether received', 'ERC20 uniq sent token name', 'min
    value received', 'min val sent', 'ERC20 uniq rec addr']

dataframe.drop(drop, axis=1, inplace=True)
```

Listing B.3: Droping one of those highly correlated features

```

figure, axises = pyplot.subplots(6, 3, figsize=(14, 14), constrained_layout =
    True)
pyplot.subplots_adjust(wspace = 0.7, hspace=0.8)
pyplot.suptitle("Distribution of features",y=0.95, size=18, weight='bold')

axis = seaborn.boxplot(axis = axises[0,0], data=dataframe, x=columns[1])
axis.set_title(f'Distribution of {columns[1]}')

axis1 = seaborn.boxplot(axis = axises[0,1], data=dataframe, x=columns[2])
axis1.set_title(f'Distribution of {columns[2]}')

axis2 = seaborn.boxplot(axis = axises[0,2], data=dataframe, x=columns[3])
axis2.set_title(f'Distribution of {columns[3]}')

axis3 = seaborn.boxplot(axis = axises[1,0], data=dataframe, x=columns[4])
axis3.set_title(f'Distribution of {columns[4]}')

axis4 = seaborn.boxplot(axis = axises[1,1], data=dataframe, x=columns[5])
axis4.set_title(f'Distribution of {columns[5]}')

axis5 = seaborn.boxplot(axis = axises[1,2], data=dataframe, x=columns[6])
axis5.set_title(f'Distribution of {columns[6]}')

```

```
axis6 = seaborn.boxplot(axis = axises[2,0], data=dataframe, x=columns[7])
axis6.set_title(f'Distribution of {columns[7]}')

axis7 = seaborn.boxplot(axis = axises[2,1], data=dataframe, x=columns[8])
axis7.set_title(f'Distribution of {columns[8]}')

axis8 = seaborn.boxplot(axis = axises[2,2], data=dataframe, x=columns[9])
axis8.set_title(f'Distribution of {columns[9]}')

axis9 = seaborn.boxplot(axis = axises[3,0], data=dataframe, x=columns[10])
axis9.set_title(f'Distribution of {columns[10]}')

axis10 = seaborn.boxplot(axis = axises[3,1], data=dataframe, x=columns[11])
axis10.set_title(f'Distribution of {columns[11]}')

axis11 = seaborn.boxplot(axis = axises[3,2], data=dataframe, x=columns[12])
axis11.set_title(f'Distribution of {columns[12]}')

axis12 = seaborn.boxplot(axis = axises[4,0], data=dataframe, x=columns[13])
axis12.set_title(f'Distribution of {columns[13]}')

axis13 = seaborn.boxplot(axis = axises[4,1], data=dataframe, x=columns[14])
axis13.set_title(f'Distribution of {columns[14]}')

axis14 = seaborn.boxplot(axis = axises[4,2], data=dataframe, x=columns[15])
axis14.set_title(f'Distribution of {columns[15]}')

axis15 = seaborn.boxplot(axis = axises[5,0], data=dataframe, x=columns[16])
axis15.set_title(f'Distribution of {columns[16]}')

axis16 = seaborn.boxplot(axis = axises[5,1], data=dataframe, x=columns[17])
axis16.set_title(f'Distribution of {columns[17]}')

axis17 = seaborn.boxplot(axis = axises[5,2], data=dataframe, x=columns[18])
axis17.set_title(f'Distribution of {columns[18]}')

```

```
60 pyplot.show()
```

Listing B.4: Investigate The Distribution of Our Features Using Boxplots

Appendix C

Data Normalization and Augmentation

The feature matrix and target class were extracted from the dataset.

```

y = dataframe.iloc[:, 0]
X = dataframe.iloc[:, 1:]
```

Listing C.1: Split the features and target

```

# Split into training (80%) and testing set (20%)
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.2,
    random_state = 123)
```

Listing C.2: Split into training and testing set

We normalized the dataset to remove potential biasness during trainnig

```

norm = PowerTransformer()
norm_train_f = norm.fit_transform(X_train)
```

Listing C.3: Normalize the training features

Upon investigating, it was found that records of non-fraudulent transaction dominate our dataset.

```

# Inspect distribution

pie, axis = pyplot.subplots(figsize=[15,10])
labels = ['Non-fraud', 'Fraud']
colors = ['#f9ae35', '#f64e38']
pyplot.pie(x = dataframe['FLAG'].value_counts(), autopct='%.2f%%', explode
    =[0.02]*2, labels=labels, pctdistance=0.5, textprops={'fontsize': 14},
    colors = colors)
pyplot.title('Target distribution')
pyplot.show()
```

Listing C.4: Training the data using LR

We oversampled our dataset to remove the biasness using SMOTE technique.

```

oversample = SMOTE()

x_tr_resample, y_tr_resample = oversample.fit_resample(norm_train_f, y_train)
print(f'Shape of the training after SMOTE: {x_tr_resample.shape, y_tr_resample.
    shape}')
```

Listing C.5: Training the data using LR

Appendix D

Training

```

LR = LogisticRegression(random_state=42)
LR.fit(x_tr_resample, y_tr_resample)

# Transform test features
norm_test_f = norm.transform(X_test)

predictions = LR.predict(norm_test_f)
```

Listing D.1: Training the data using Logistic Regression

```

RF = RandomForestClassifier(random_state=42)
RF.fit(x_tr_resample, y_tr_resample)
preds_RF = RF.predict(norm_test_f)
```

Listing D.2: Training the data using Random Forest Classifier

```

xgb_c = xgb.XGBClassifier(random_state=42)
xgb_c.fit(x_tr_resample, y_tr_resample)
preds_xgb = xgb_c.predict(norm_test_f)
```

Listing D.3: Training the data using XGBoost Classifier

```

scaler = StandardScaler()

cols = X_test.columns
x_tr_resample = pandas.DataFrame(x_tr_resample, columns=[cols])
x_test = pandas.DataFrame(X_test, columns=[cols])
x_tr_resample.describe()

svc = SVC(gamma=0.001, C=100., kernel = 'linear')

svc.fit(x_tr_resample, y_tr_resample)
```

Listing D.4: Training the data using SVM Classifier

```

knn = KNeighborsClassifier(n_neighbors=3)
knn.fit(x_tr_resample, y_tr_resample)
```

Listing D.5: Training the data using KNN classifier

Appendix E

Testing and Evaluation

```

print(classification_report(y_test, predictions))
print(confusion_matrix(y_test, predictions))
plot_confusion_matrix(LR, norm_test_f, y_test)
```

Listing E.1: Evaluate the data using Logistic Regression

```

print(classification_report(y_test, preds_xgb))
print(confusion_matrix(y_test, preds_xgb))
plot_confusion_matrix(xgb_c, norm_test_f, y_test)
```

Listing E.2: Evaluate the data using RF

```

print(classification_report(y_test, preds_xgb))
print(confusion_matrix(y_test, preds_xgb))
plot_confusion_matrix(xgb_c, norm_test_f, y_test)
```

Listing E.3: Evaluate the data using XGB

```

x_test_norm = norm.transform(X_test)
plot_confusion_matrix(svc, x_test_norm, y_test))
```

Listing E.4: Evaluate the data using SVM

```

plot_confusion_matrix(knn, x_test_norm, y_test)
```

Listing E.5: Evaluate the data using KNN