Anomaly Detection in Blockchain Transactions using Machine Learning with Explainability Analysis

Mohammad Hasan

Department of Computer Science and Engineering

United International University

A thesis submitted for the degree of

MSc in Computer Science & Engineering

March 2024

Declaration

I, Mohammad Hasan, declare that this thesis titled, Anomaly Detection in Blockchain Transactions using Machine Learning with Explainability Analysis and the work presented in it are my own. I confirm that:

- This work was done wholly or mainly while in candidature for an MSc degree at United International University.
- Where any part of this thesis has previously been submitted for a degree or any other qualification at United International University or any other institution, this has been clearly stated.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work.
- I have acknowledged all main sources of help.
- Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself.

Signed: _____

Date: _____

(Mohammad Hasan)

Certificate

I do hereby declare that the research works embodied in this thesis entitled Anomaly Detection in Blockchain Transactions using Machine Learning with Explainability Analysis is the outcome of an original work carried out by Mohammad Hasan under my supervision.

I further certify that the dissertation meets the requirements and the standard for the degree of MSc in Computer Science and Engineering.

1. Supervisor:
Signed: _____

Date: _____

(Dr. Iqbal H. Sarker)

Postdoctoral Research Fellow, Centre for Securing Digital Futures, Edith Cowan University, Australia.

2. Co-Supervisor:
Signed: _____

Date: _____

(Dr. Mohammad Shahriar Rahman)

Professor, Department of Computer Science and Engineering, United International University (UIU), Dhaka, Bangladesh.

Abstract

In the era of growing cryptocurrency adoption, Blockchain has emerged as a leading player in the digital payment landscape. However, this widespread popularity also brings forth an array of security challenges, including the need to safeguard against malicious activities. One of the paramount challenges in this regard is the detection of anomalous transactions within the realm of Bitcoin data, a task that significantly influences the trust and security of digital payments. Yet, it's a formidable challenge given the relatively low occurrence of anomalous Bitcoin transactions. Although several studies have been conducted in the field, a limitation persists: the lack of explanations for the model's predictions. This study aims to address this limitation by combining eXplainable Artificial Intelligence (XAI) techniques and anomaly rules with tree-based ensemble classifiers. While deep learning techniques have demonstrated their prowess in anomaly detection, there remains a scarcity of studies exploring their potential, particularly in the context of Bitcoin. This study also aims to fill that gap, focusing on our 1D Convolutional Neural Network (CNN) model. To understand how our model works and explain its decisions, we use the Shapley Additive exPlanation (SHAP) method, which measures each feature's impact. We also deal with data imbalance by exploring various methods to balance anomalous and non-anomalous Bitcoin transaction data. Additionally, we have introduced an under-sampling algorithm named XGBCLUS, designed to balance anomalous and non-anomalous transaction data. This algorithm is compared against other commonly used under-sampling and over-sampling techniques. Our experimental results demonstrate that: (i) XGBCLUS enhances TPR and ROC-AUC scores compared to state-of-the-art under-sampling and over-sampling techniques, and (ii) our proposed ensemble classifiers outperform traditional single tree-based machine learning classifiers

in terms of accuracy, TPR, and FPR scores, and (iii) our proposed 1D CNN model attains elevated accuracy with a concurrent reduction in the False Positive Rate (FPR).

With the deepest wellspring of gratitude and the warmest affection that words can convey, I lovingly dedicate this work to my Honorable Father, to my beloved Mother, and to my precious wife and children, whose boundless love and encouragement have filled my world with immeasurable joy and purpose.

Published Papers

Work relating to the research presented in this thesis has been published/ submitted by the author in the following peer-reviewed journals and conferences:

1. Mohammad Hasan, Mohammad Shahriar Rahman, Helge Janicke, and Iqbal H. Sarker, "Detecting Anomalies in Blockchain Transactions using Machine Learning Classifiers and Explainability Analysis ", Blockchain: Research and Applications, Elsevier. [**Under 3rd Revision**]
2. Mohammad Hasan, Mohammad Shahriar Rahman, and Iqbal H. Sarker, "CNN Based Deep Learning Modeling with Explainability Analysis for Detecting Fraudulent Blockchain Transactions", Cyber Security and Applications, Elsevier. [**Under Review**]

Acknowledgements

I begin by expressing gratitude to Almighty Allah for granting me the strength and courage to complete the work and achieve the goal.

I extend my heartfelt gratitude to Dr. Iqbal H. Sarker, Postdoctoral Research Fellow, Centre for Securing Digital Futures, Edith Cowan University, Australia, who served as my academic supervisor during my MSc journey. His unwavering guidance, motivation, and consistent support have been invaluable. Dr. Sarker's insights and wisdom have significantly influenced my approach to research, leaving a lasting impact that will undoubtedly shape my future endeavors. The completion of this thesis owes much to his instructive supervision, without which it would not have come to fruition.

I express profound gratitude to Dr. Mohammad Shahriar Rahman, Professor in the Department of Computer Science and Engineering at United International University, Dhaka, Bangladesh, for serving as my co-supervisor in this research endeavor. I extend sincere thanks to him for his invaluable reviews and corrections, which played a crucial role in the successful completion of this thesis.

I also sincerely thank Dr. Jabed Morshed Chowdhury, Lecturer at La Trobe University, Australia, for his input to the discussion.

I sincerely appreciate the esteemed members of the examination board of the Department of Computer Science and Engineering at UIU for their valuable suggestions. I also recognize and thank the entire faculty at the Department of Computer Science and Engineering, Premier University, Chattogram, for their unwavering support. Special gratitude goes to Premier University for granting me the opportunity to pursue my MSc degree in Computer Science and Engineering as a part-time student.

A heartfelt acknowledgment is due to my family, including my parents and wife, whose unconditional love and profound emotional support have been invaluable throughout this journey.

Contents

List of Figures

List of Tables

List of Algorithms

Chapter 1

Introduction

Blockchain, a chain of blocks that contain the history of several transactions or records of other applications in a public ledger, has been considered an emerging technology both in academic and industrial areas since the last decade [1]. Bitcoin, the first digital cryptocurrency, was proposed in 2008 and then successfully implemented by Satoshi Nakamoto [2]. Although Blockchain was created to support the popular bitcoin currency, transactions of other digital crypto-currencies such as Ethereum, Ripple, Litecoin, etc., health records, transportation, IoT applications, etc. [3] are stored in the blocks in a decentralized manner and are managed without the help of a third party organization. Prominent attributes like trustworthiness, verifiability, decentralization, and immutability have rendered Blockchain an effective integration partner for various Information and Communication Technology (ICT) applications. Nevertheless, this technology remains susceptible to an array of challenges, encompassing security breaches, privacy concerns, energy consumption, regulatory policies, and issues like selfish mining [4]. Despite its growing popularity in digital payments, Bitcoin remains susceptible to a range of attacks, encompassing temporal attacks, spatial attacks, and logical-partitioning attacks [5]. To ensure the effective implementation of blockchain technology, the timely detection of malicious behavior or transactions within the network, or the identification of novel instances in the data, is imperative [6].

The Bitcoin network, as the pioneering blockchain in the financial realm, has confronted numerous challenges associated with illicit activities. This thesis endeavors to identify and flag anomalous or suspicious transactions within the Bitcoin network. Swift and appropriate measures must be taken to mitigate potential risks by incorporating

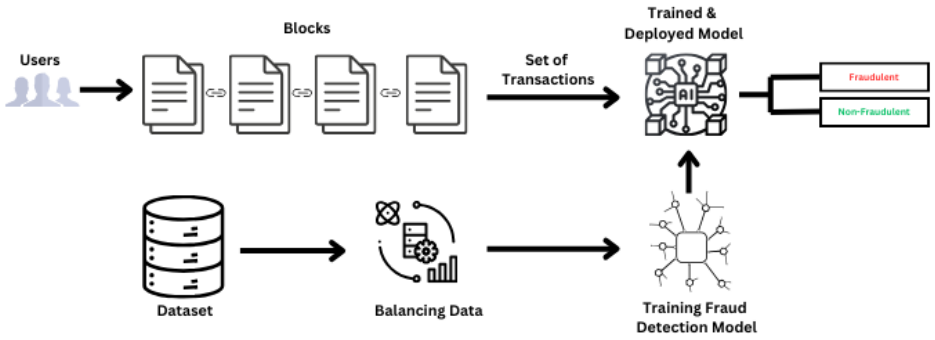


Figure 1.1: A System Architecture of the Blockchain Anomaly Detection System

a real-time anomaly detection system as shown in Figure 1.1. A real-time anomaly detection system plays a crucial role in identifying and preventing potential fraudulent activities or transactions conducted digitally on the blockchain by users, thereby preserving the blockchain system's integrity [7]. However, the activities in the Bitcoin network can be categorized into two main groups: user activities and transaction activities. This thesis primarily focuses on analyzing transaction activities, particularly in identifying and investigating suspicious behavior.

While this study primarily targets anomaly detection within the Bitcoin transaction network, its scope extends to encompass fraud detection across various financial transaction blockchain systems, including Ethereum, BNB, Bitcoin Cash, and Lite Coin, among others. Furthermore, the research broadens its focus to address anomaly detection within diverse blockchain networks, spanning sectors like healthcare services, transportation, IoT applications, and beyond. Thus, this thesis delves into the broader challenge of anomaly detection within blockchain technology, exploring its applicability across a spectrum of use cases and contexts.

1.1 Motivation

Within the context of blockchain systems, anomaly detection assumes paramount importance. Since the dataset contains significantly fewer anomalous Bitcoin transactions compared to legal transactions, the classifiers are biased towards the majority class (non-anomalous transactions). Various over and under-sampling techniques have been employed to address these imbalanced data challenges in diverse domains [8]. One

common challenge is the potential omission of significant instances that greatly impact model training [9]. Additionally, issues such as overfitting, sensitivity to noise, and the introduction of bias into the dataset can reduce the performance of machine learning models [10].

After balancing the data, selecting a machine learning classifier is another challenging task. Tree-based machine learning classifiers have been used in many studies to classify malicious activities [11] since faster training can be performed on tree-based classifiers [12]. However, several studies show that the ensemble method can perform better in the case of large-scale data e.g. Bitcoin transactions than a single machine learning algorithm [13].

In recent times, Deep Learning (DL) algorithms have emerged as a promising approach for addressing the anomaly detection challenge [14],[15], [16]. Deep Learning, a subset of Machine Learning (ML), has gained prominence in predictive analysis [17]. In recent times, the 1D Convolutional Neural Network (1D CNN) has garnered attention for its swift feature extraction capabilities.

After performing all these great tasks for anomaly detection, a question can arise "Should we trust the prediction of the Black-Box model?". XAI, Explainable Artificial Intelligence, is a field of interest to find the answer to this question. This latest AI technique helps to increase the explainability and transparency of the black-box AI models by making complex interpretable decisions [18]. Two popular XAI techniques e.g. 'Local Interpretable Model-Agnostic Explanations' (LIME) [19] and 'SHapely Additive exPlanations' (SHAP) [20] have been used by researchers to prove the explainability and transparency of the AI models.

1.2 Objectives of the Thesis

This study aims to create an anomaly detection system for identifying anomalous Bitcoin transactions. To accomplish this objective, the following goals have been outlined.

- To investigate the suitability of under and over-sampling techniques in balancing the highly imbalanced Bitcoin transaction data.
- To develop an ensemble machine learning model utilizing tree-based classifiers for the detection of anomalous Bitcoin transactions.

- To design a deep learning architecture using 1D CNN, capable of extracting significant features from Bitcoin transaction data to accurately identify anomalous transactions.
- To identify the crucial features that contribute significantly to the detection of anomalous Bitcoin transactions.

1.3 Thesis Contributions

The notable contributions of this study include:

- We introduce an under-sampling algorithm based on eXtreme Gradient Boosting (XGBoost) called XGBCLUS, and we compare it with state-of-the-art methods.
- We also explore various over-sampling and combined sampling techniques for balancing Bitcoin transactions.
- Further, we compare the effectiveness of both under-sampling and over-sampling techniques.
- Additionally, we propose tree-based ensemble classifiers to classify anomalous Bitcoin transactions.
- We also compare the tree-based ensemble classifiers with the individual ML classifiers for detecting anomalous Bitcoin transactions.
- We propose a 1D CNN model designed to identify anomalous Bitcoin transactions and assess its performance in comparison to state-of-the-art Machine Learning (ML) algorithms.
- We explain the predictions of both the ensemble model and CNN model using SHAP (an eXplainable Artificial Intelligence technique) and identify the crucial features that exert the most influence on classifying Bitcoin transactions.
- Lastly, we present a set of rules derived from a tree-based model to provide explanations for anomalous transactions.

1.4 Organization of the Thesis

The remainder of the thesis is structured as follows:

Chapter 2 introduces initial concepts relevant to this thesis and conducts a thorough review of existing literature on anomaly detection in several cryptocurrencies, identifying notable research gaps.

Chapter 3 outlines our proposed anomaly detection approach, employing a combination of ensemble machine learning and deep learning models, integrated with both under-sampling and over-sampling techniques.

Chapter 4 provides a comprehensive comparative analysis of results between machine learning and deep learning models, including the explainability and anomaly rules.

Chapter 5 wraps up the thesis by summarizing key discoveries and discussing on future research scopes.

Chapter 2

Background Study and Related Works

In this chapter, foundational concepts are introduced to provide a comprehensive understanding of the thesis. Key topics covered include Bitcoin anomalies, machine learning, and deep learning. Additionally, recent efforts in anomaly detection are explored, shedding light on their limitations and underscoring the research gaps that this thesis seeks to address.

2.1 Bitcoin Transactions

Bitcoin is a decentralized digital currency that operates on a peer-to-peer network, allowing users to send and receive payments without the need for a central authority or intermediary [21]. Its significance in digital payments lies in offering a secure, transparent, and borderless means of transferring value globally. As a pioneer in the world of cryptocurrencies, Bitcoin has garnered widespread attention for its potential to revolutionize traditional financial systems.

However, the Bitcoin blockchain operates under specific rules, notably the limit of 21 million bitcoins. Illustrated in Figure 2.1, each transaction undergoes encryption using the sender's private key. This encrypted transaction is then accompanied by a digital signature and the public key of the receiver before being forwarded to the recipient's address. Such encryption ensures secure delivery to the intended recipient and facilitates verification using the digital signature. Moreover, the anonymity of

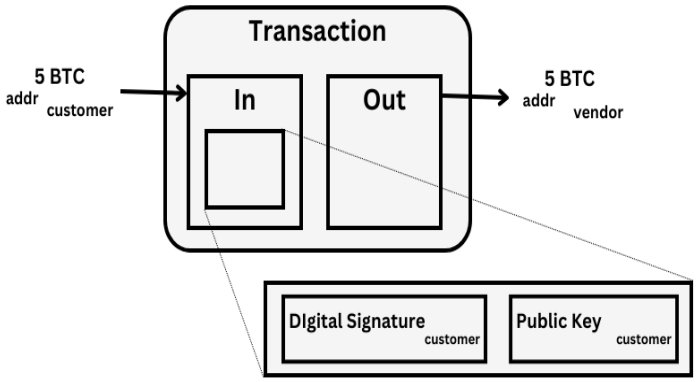


Figure 2.1: Basic Bitcoin Transaction

the sender and receiver is maintained, as they can generate new addresses for each transaction. Bitcoin transactions may include multiple inputs and outputs, with users specifying the destination address and amount sent, encapsulated within the transaction output to prevent double-spending.

Figure 2.2 presents a sample sub-network extracted from network T. Transaction t1, executed on May 1, 2011, involves one input and one output. One of t1's outputs entails transferring 1.2 BTC (Bitcoins) to a user identified by public-key pk1. The figure does not depict the public keys. Transaction t2, conducted on May 5, 2011, comprises two inputs and two outputs. Among t2's outputs is a transfer of 0.12 BTC to a user identified by public-key pk2. Transaction t3, added to the ledger on the same day as t2, also includes two inputs and one output. The inputs of t3 are linked to the outputs of both t1 and t2. T3 itself has only one output, t4, which transferred 1.32 BTC on May 5, 2011.

Recent statistics on Bitcoin payments underscore its growing adoption. The increasing number of merchants and businesses accepting Bitcoin reflects a shift toward recognizing it as a legitimate form of payment. Additionally, the rise in the total value of Bitcoin transactions and the expanding user base contribute to its prominence in the digital economy [22]. However, with the surge in popularity, challenges such as security concerns, regulatory issues, and the potential for illicit activities have emerged [23].

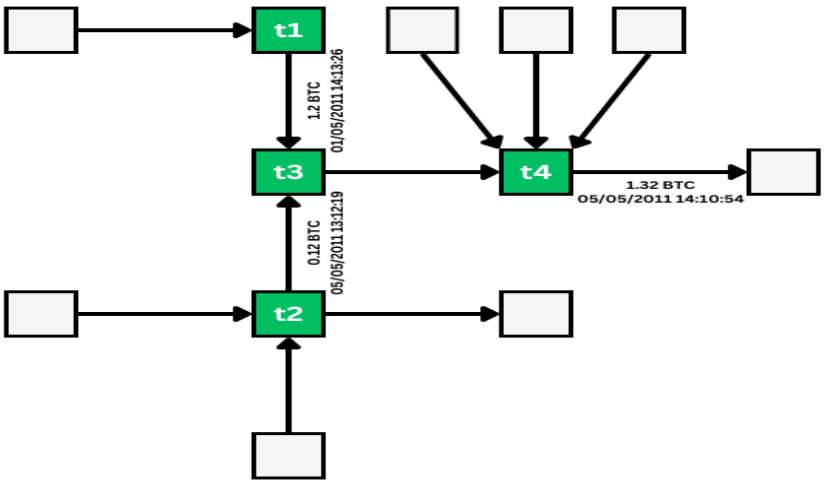


Figure 2.2: A sub-network representing Bitcoin transactions

2.2 Anomalies and Their Effects in Blockchain Technology

Anomalies in Bitcoin transactions refer to atypical or irregular patterns in the flow of digital currency, deviating from expected behaviors. These anomalies may signify potential fraudulent activities, security breaches, or other irregularities in the blockchain network [24]. Blockchains are heralded for their promise of privacy and security in financial architecture. Yet, despite the perception of being impregnable, certain individuals have managed to circumvent this supposedly foolproof infrastructure. These perpetrators typically aim to conduct illicit activities covertly, either by covering their tracks or by completely deceiving the system to appear legitimate.

While many smaller-scale incidents may go unreported, larger ones often make headlines. Bitcoin, as the pioneering and oldest financial blockchain, has faced its share of challenges related to illegal activities. For instance, Reid and Harrigan et al. documented a notable Bitcoin theft dubbed 'All In Vain,' where approximately 25,000 bitcoins were stolen [25]. Additionally, a victim known as Stone-Man [26] shared their harrowing experience on a Bitcoin forum, recounting the loss of 8,999 bitcoins due to the exploitation of their original private key. This victim had initially purchased 9,000 bitcoins from an exchange, transferring them to a disc and also backing them up on

a USB flash drive. Curiously, the victim also initiated a single Bitcoin transfer to another address for unknown reasons. Upon confirming and securing all wallet data, the victim discovered an unauthorized transaction of 8,999 bitcoins to an unfamiliar address, which they had not authorized.

Moreover, The most significant financial debacle in the Bitcoin world occurred with the collapse of Mt. Gox, the leading Bitcoin exchange at the time, in 2014 [27]. Led by CEO Mark Karpelès, a French national operating from Japan, Mt. Gox served as the primary platform for buying and selling Bitcoins from its inception in 2010 until February 2014. However, in a startling announcement, Mt. Gox revealed that 850,000 bitcoins had disappeared, presumably stolen by hackers. At the prices prevailing in early 2014, these bitcoins were valued at approximately $450 million. Today, their worth would soar to $8.5 billion. In January 2015, Bitstamp, a well-known Bitcoin exchange, disclosed a loss of approximately 19,000 bitcoins, valued at about $5 million at the time [28]. Despite the setback, the exchange managed to weather the attack and continues to maintain its position as one of the leading Bitcoin exchanges today. In August 2016, Bitfinex, a Bitcoin exchange, revealed that hackers had absconded with $77 million worth of bitcoins [29]. To mitigate the impact, the company transferred the financial burden onto its users, compelling them to accept a 36-percent reduction in the value of their deposits.

These instances exemplify certain behaviors that raise suspicion. Therefore, we can identify similar activities and behaviors as anomalies and aim to develop a model capable of detecting them. Bitcoin has gained notoriety for its association with illicit activities on the dark web, including money laundering, drug trafficking, and arms dealing. Individuals have exploited the technology to facilitate transactions for illegal goods and services on platforms like the online black market 'Silk Road,' which operated on the dark web since 2011. According to Christin et al., [30], Silk Road generated a monthly revenue of approximately 1.2 million USD.

According to the Crystal Hacks Report [31], the yearly average number of security breaches and frauds involving cryptocurrencies remained relatively low, at less than 12, from 2011 to 2018. However, the total stolen funds saw a significant increase, averaging about $11.6 million per year during the same period. The situation took a drastic turn in 2019, with 26 reported incidents resulting in $3.5 billion in losses. Subsequent years witnessed further escalations, with 32 incidents and $1.49 billion in losses in 2020, 94

incidents and $4.63 billion in losses in 2021, and 120 incidents and $2.14 billion in losses in 2022. These rising instances of fraudulent activities and hacking pose a threat to the cryptocurrency ecosystem, potentially dampening investor confidence and impacting cryptocurrency pricing and trading.

Recent statistics show that the detection of these anomalies is crucial for maintaining the integrity and security of Bitcoin transactions, with advancements in machine learning and deep learning techniques playing a pivotal role in enhancing the accuracy and efficiency of anomaly detection systems. As the popularity of Bitcoin continues to grow, addressing and understanding these anomalies becomes increasingly vital for ensuring the trustworthiness of digital payment systems. This thesis leverages the transactional data from a widely accessible financial blockchain, Bitcoin, due to its availability and extensive academic literature. The data undergoes thorough preprocessing, followed by the application of various modeling techniques to detect transactions associated with theft, heists, or money laundering.

2.3 Labelling of Anomalous Bitcoin Transactions

The raw data obtained from the Bitcoin client underwent a filtering process based on the year parameter to extract a specific subset. Utilizing a Python script, approximately 29,000,000 transactions spanning the years 2011 to 2013 were isolated from the dataset to be used in this thesis. The rationale behind selecting this particular timeframe stems from the presence of anomalous Bitcoin transaction data during this period. Notably, certain transactions within these years were flagged as anomalous, as documented by regulatory entities such as the Bitcoin Forum. This platform maintains a comprehensive record of BTC fraud transactional activities, including incidents categorized as BTC Thefts, BTC Hacks, and BTC Losses. Each flagged transaction contains pertinent details such as the date of occurrence and the amount of BTC involved.

Finally, The dataset of anomalies was curated by extracting data from the Bitcoin Forum (2014) using a custom Python crawler. This process enabled the compilation of a dataset specifically focused on anomalous Bitcoin transactions. To accurately label malicious transactions within the dataset, a tagging technique was employed. Specifically, any transaction identified as malicious resulted in all subsequent transactions

involving the same BTC being labeled as malicious as well. This cascading effect ensured comprehensive coverage of potentially fraudulent activities associated with the initial malicious transaction.

2.4 Tree-Based ML Models

2.4.1 Decision Tree

A Decision Tree classifier is a popular machine learning algorithm used for both classification and regression tasks. It builds a tree-like structure of decisions based on features to make predictions [32]. The decision tree starts with a root node, representing the entire dataset. It then splits the data into subsets based on feature values to create child nodes. Each node represents a decision or test on a feature such as *indegree, in_btc, out_btc, total_btc, mean_in_btc,* and *mean_out_btc*. Decision Trees use different criteria to determine the best feature and value to split the data at each node. Two common criteria are Gini Impurity and Information Gain (Entropy). The tree continues to split the data into subsets, creating child nodes until a stopping condition is met. This condition could be a predefined tree depth or a minimum number of samples required in a node. When a stopping condition is reached, the final nodes are called leaf nodes or terminal nodes. Each leaf node represents a class label i.e. Anomalous or non-Anomalous. To make predictions, data samples traverse the tree from the root node, following the decisions at each node, until they reach a leaf node. The class label associated with the leaf node is the prediction.

2.4.2 Gradient Boosting

The Gradient Boosting classifier is an ensemble Machine Learning technique used for both classification and regression tasks. It is powerful and often achieves high predictive accuracy. This tree-based classifier combines the predictions of multiple weak learners, typically decision trees, to create a strong predictive model [33]. Gradient Boosting starts with an initial base learner, often a decision tree, to make predictions. It calculates the residuals (the differences between the actual and predicted values) for each data point in the training set using the current model's predictions. A new weak learner (usually another decision tree) is trained to predict the residuals. The goal is to fit this new learner to the residuals to capture the errors made by the previous

model. The predictions of the new learner are weighted and added to the predictions of the previous model. The weights are determined by a learning rate that scales the contribution of each new model. Residual calculation and weighted addition steps are repeated for a predefined number of iterations or until a certain performance metric is met. The final prediction is obtained by summing up the predictions of all the models, effectively creating a weighted combination of the weak learners' predictions. Gradient Boosting aims to minimize a loss function e.g. Cross-Entropy Loss for Bitcoin anomaly classification during the iterative process.

2.4.3 Extreme Gradient Boosting

Extreme Gradient Boosting (XGBoost) is an advanced ensemble machine learning algorithm known for its high predictive accuracy and efficiency. Its efficient implementation and ability to handle missing values, feature importance analysis, and early stopping make it a popular choice for various Machine Learning tasks [34]. XGBoost optimizes an objective function that combines a loss function (log loss for classification) and a regularization term. The model in XGBoost is represented as an ensemble of decision trees, where each tree is a weak learner. The final prediction is the weighted sum of predictions from all the trees. It employs a gradient boosting approach for constructing trees. At each iteration, it fits a new tree to the negative gradient of the loss function, which is a measure of how far off the current predictions are from the true values. Trees are added sequentially, with each new tree correcting the errors made by the previous ones. It also provides regularization techniques to prevent overfitting. Trees are pruned during training if they do not provide sufficient reduction in the objective function. This helps to avoid overfitting and contributes to model efficiency. This classifier uses gradient boosting to minimize the objective function. The gradients of the loss function with respect to the model predictions are calculated for each data point, and the trees are constructed to reduce these gradients.

2.4.4 Adaptive Boosting

AdaBoost's strength lies in its ability to focus on the most challenging data points by assigning higher weights to misclassified samples. It creates a strong classifier by boosting the performance of weak learners and is less prone to overfitting [35]. Initially, each data point in the training set is assigned an equal weight. These weights represent

the importance of each data point in the classification. AdaBoost trains a base learner (a weak classifier) on the training data. The goal of the base learner is to perform slightly better than random guessing. The base learner's performance is evaluated based on the weighted classification error. High-performing base learners are assigned higher weights, indicating their importance in the ensemble. AdaBoost updates the weights of the training samples to focus more on the misclassified samples. AdaBoost combines the predictions of all base learners by weighted majority voting. The final prediction is the result of a weighted sum of the base learners' predictions.

2.4.5 Random Forest

The Random Forest classifier is one of the most popular and widely used ensemble machine learning algorithms that is widely used for both classification and regression tasks. The strength of Random Forest lies in its ability to reduce overfitting, handle noisy data, and provide estimates of feature importance. The ensemble of decision trees, each trained on a different subset of data and features, collectively improves the model's predictive performance and generalization [36]. This classifier starts by creating multiple decision trees. Each tree is trained on a random subset of the original dataset, sampled with replacement. This process is called bootstrapped sampling, and it helps create diverse and unique trees. In addition to sampling data points, Random Forest randomly selects a subset of features for each tree. This randomness ensures that each tree focuses on different subsets of data and features, reducing the risk of overfitting. Each decision tree in the Random Forest is constructed using a portion of the data and a subset of features. The trees are grown recursively by selecting the best feature and split point to partition the data. This process continues until a stopping criterion, such as an Anomalous or non-anomalous leaf node, is reached. For the classification of Bitcoin transaction data, the Random Forest combines the predictions of individual trees through a majority voting mechanism.

2.5 Convolution Neural Network(CNN)

A Convolutional Neural Network (CNN) was originally crafted for processing two-dimensional data, specifically tailored for tasks like image processing. Its fundamental

architecture encompasses essential components such as convolutional, pooling, activation, and fully connected layers. A 1D Convolutional Neural Network (CNN) is used to extract hierarchical features from sequential data.

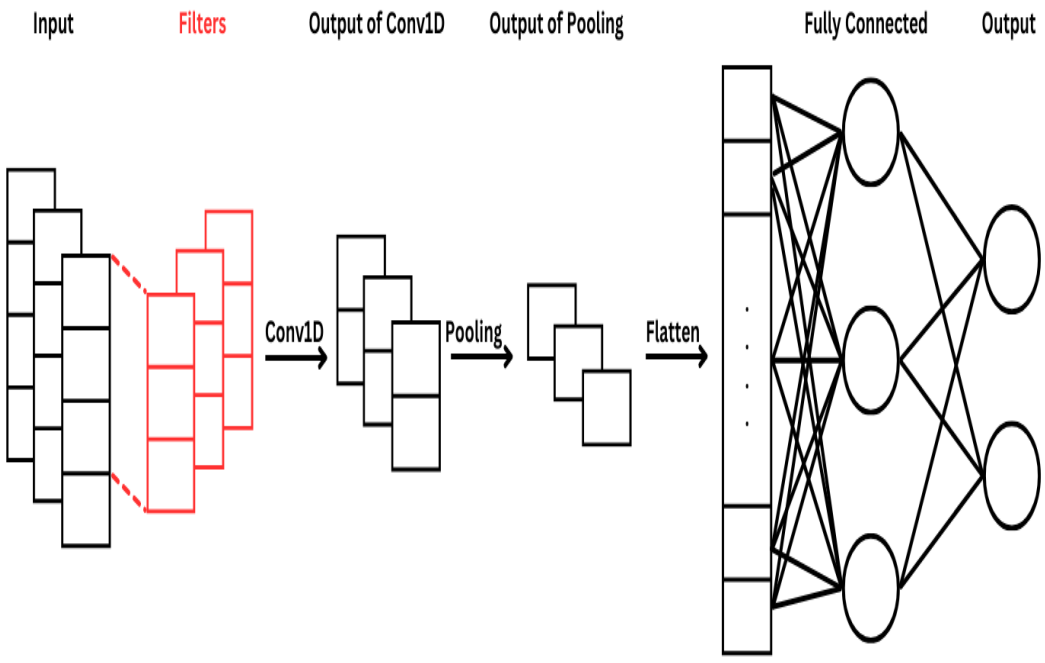


Figure 2.3: Architecture of 1D CNN

2.5.1 Convolutional Layer

The pivotal component in a CNN is the convolutional layer, responsible for the majority of computational tasks within the network. Input images undergo transformation into tensors with a given shape, serving as input to this layer. The convolutional layer conducts a dot product operation with a 2D matrix known as a kernel, filter, or feature detector, traversing receptive fields across the input. The kernel, characterized by learnable parameters or weights, is smaller in spatial dimensions than the input image and is updated during training. Moving with a fixed stride, the kernel performs dot products, generating output matrices referred to as feature maps or activation maps through the convolution process. These feature maps, representing convolved features,

are then transmitted to subsequent layers. Additional convolutional layers, including the initial one, can be incorporated. Following each convolution, a Rectified Linear Unit (ReLU) activation function is applied to the feature map, introducing nonlinearity to the model. For an input of size $(W_i \times H_i)$ and a kernel of size $(W_k \times H_k)$ with stride S, the size of the output matrix W_c can be determined by the following formula:

$$W_c = \left(\frac{W_i - W_k}{S} + 1, \frac{H_i - H_k}{S} + 1\right)$$

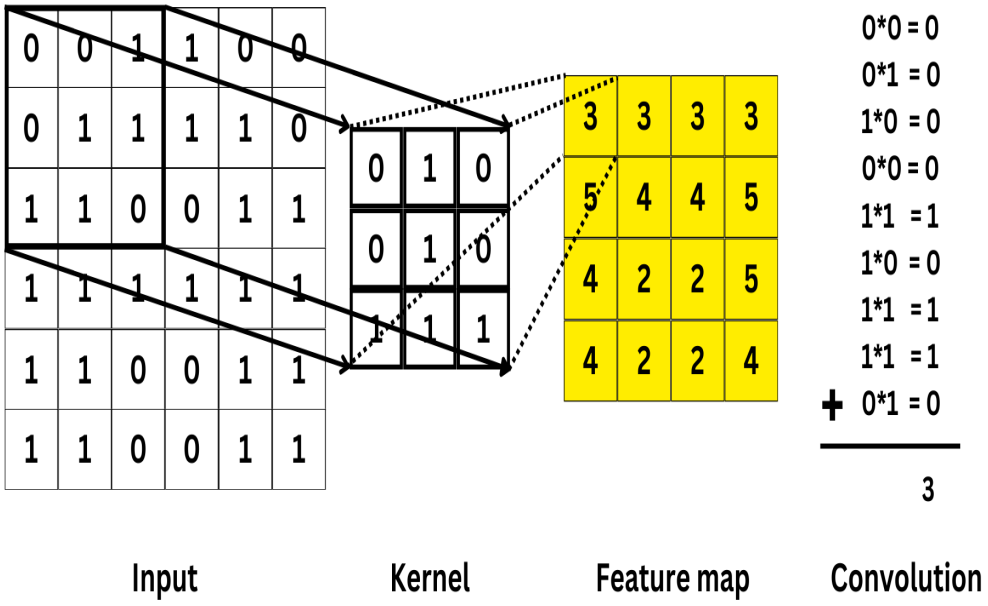


Figure 2.4: Convolution operation using 3X3 filter.

2.5.2 Pooling Layer

Pooling layers play a crucial role in reducing dimensionality, mitigating computational load, and preventing overfitting in CNNs. Their function involves downsampling or reducing the spatial dimensions of feature maps, serving as an effective noise suppressor. Similar to convolution, pooling operations utilize a kernel that traverses the height and

width of the input, but unlike convolution, this kernel lacks weights. Two common types of pooling are employed: max pooling and average pooling. Max pooling extracts the maximum pixel value from each partition of the input image covered by a kernel, while average pooling calculates the average values from each partition. These pooling operations contribute to dimensionality reduction and noise suppression in the overall network architecture.

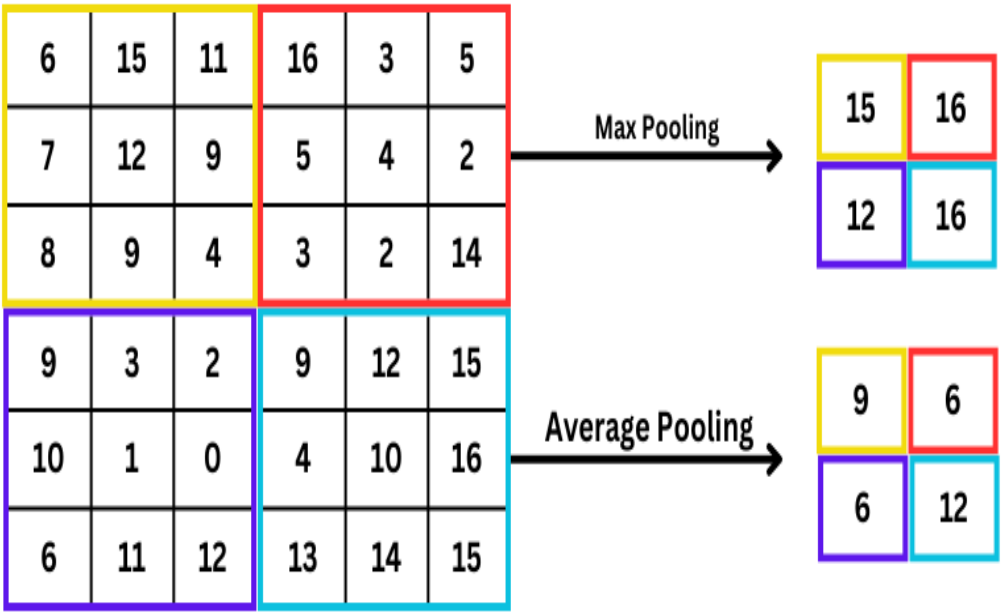


Figure 2.5: Max Pooling and Average Pooling using 2X2 filter

2.5.3 Fully Connected Layer

The fully connected layer receives the flattened output matrix from the previous layer, establishing connections between every neuron in the preceding layer and those in the subsequent layer, akin to multilayer perceptrons. This mapping helps refine the representation between the input and output. In the output layer, a Softmax activation function is applied for input classification, assigning probability scores ranging from 0 to 1 to different classes.

2.6 Explainable AI

Explainable Artificial Intelligence (XAI) is a field of study aimed at making machine learning models more transparent and interpretable [18]. It enables users to understand how models make predictions, thus building trust and aiding in decision-making processes. Two popular methods used for XAI are SHAP (SHapley Additive exPlanations) and LIME (Local Interpretable Model-agnostic Explanations).

SHAP is a game-theoretic approach based on Shapley values from cooperative game theory. It assigns each feature in a prediction a Shapley value, indicating its contribution to the final prediction [20]. By analyzing these values, users can understand which features have the most significant impact on model predictions and how they interact with each other.

On the other hand, LIME is a model-agnostic method that explains individual predictions by fitting a simpler interpretable model locally around them [19]. It generates perturbed samples around the instance of interest and observes how the model behaves with these perturbations. By analyzing the changes in predictions, LIME provides insights into why a model made a particular prediction for a given instance.

Both SHAP and LIME offer valuable insights into the inner workings of machine learning models, allowing users to understand complex model decisions and identify potential biases or errors. These methods play a crucial role in ensuring the transparency and accountability of AI systems across various applications and domains.

Despite numerous studies in the field of anomaly detection in Blockchain transactions, a significant limitation remains: the absence of explanations for the predictions made by the models. This study endeavors to overcome this limitation by integrating eXplainable Artificial Intelligence (XAI) techniques. These techniques are crucial for providing interpretable insights into the decision-making process of anomaly detection models operating within Blockchain transactions. By leveraging XAI methodologies, this research seeks to enhance transparency and understanding, thus contributing to more reliable and accountable anomaly detection systems in the realm of Blockchain technology.

2.7 Related Studies

Researchers have focused on detecting or predicting anomalous transactions using the concept of blockchain intelligence [37]. This involves the incorporation of Artificial Intelligence (AI) for anomaly detection and fraud detection within blockchain transaction data. Both Machine Learning (ML) and Deep Learning (DL) techniques, along with a range of balancing methods, have been employed to identify fraudulent transactions within blockchain transaction data. While previous studies have primarily focused on using supervised and unsupervised algorithms for classifying illicit transactions within Bitcoin transaction data, a limited number of studies have addressed data balancing methods. However, the issue of class imbalance presents challenges when it comes to classifying or detecting anomalies in areas related to illicit activities in digital currencies, money laundering, etc. [38]. To address this concern, researchers have put forth various undersampling and oversampling methods to assess their impact on improving evaluation metrics.

2.7.1 Machine Learning Techniques

In their study [39], the authors delved into the Bitcoin ecosystem to assess the extent of illicit activities, including ransomware attacks, scams, and the illegal trade of goods, among others. They aimed to categorize these activities and determine the predominant share of cybercrime within the Bitcoin ecosystem. Additionally, the study sought to identify the number of fraudulent addresses and quantify the volume of Bitcoin transactions associated with these illicit activities. The research highlighted Bagging and Gradient Boosting as the most effective models for classifying the five categories of cybercrime activities. Furthermore, the experimental results were presented visually. However, it's worth noting that one of the key limitations of their work was the absence of proper balancing techniques to fine-tune the models. Singh et al. [40] utilized SVM, Decision Tree, and Random Forest classifiers to identify anomalies within the Ethereum network. When operating based on the dataset's ground truth, they successfully detected 47 anomalous transactions out of 50 positive cases. However, upon considering indegree nodes as fraudulent cases, they were able to identify 49 positive cases. Additionally, they included outdegree nodes as positive cases but could only identify a single anomalous case. It's important to note that a notable limitation of

this study is that it examined only a small subset of the dataset and did not employ any balancing techniques. The authors in [41] employed five distinct supervised machine learning classifiers, namely Random Forest, Adaptive Boosting, MLP, SVM, and KNN, to identify instances of Bitcoin transaction theft. Among these classifiers, the Random Forest (RF) classifier demonstrated the highest performance, achieving an F1 value of 0.952. Furthermore, the authors asserted that their top-performing model surpassed the performance of other unsupervised algorithms, specifically Mahalanobis Distance, LOF, and OCSVM. Notably, they enhanced their experimental results by introducing balance to the training data through the utilization of the SMOTE oversampling method. In the study [42], the authors employed active learning tools to identify illicit activities within Bitcoin transaction data using machine learning classifiers. Remarkably, by utilizing only 5% of the labels, they asserted that their proposed method had surpassed state-of-the-art unsupervised methods in the detection of anomalous behavior. Additionally, the authors conducted a comparative study, as outlined in [43], to differentiate between non-anomalous and anomalous transactions within the Bitcoin ecosystem. Their findings revealed that ensemble-based methods outperformed common machine learning models in terms of accuracy and F1-score evaluation metrics. In their paper [44], the authors employ multiple Machine Learning models to identify anomalous transactions in various digital currency markets. Their findings indicate that supervised learning techniques produce promising results, while unsupervised learning techniques face more challenges when it comes to classification. In [45], the authors have conducted a comparative analysis of Bitcoin and Ethereum transaction data using various under and over-sampling techniques. Notably, their customized nearest-neighbor under-sampling method has achieved an impressive 99% accuracy, outperforming several SMOTE-based over-sampling techniques. They also have applied popular supervised algorithms to classify anomalies in both transaction datasets. In a separate study detailed in [46], the authors have explored ensemble-based classifiers for detecting fraudulent activities in bank transactions. They have investigated the effectiveness of numerous up-and-down sampling techniques. Among these techniques, SVM SMOTE for dataset balancing, combined with the Random Forest classifier, has emerged as the most promising combination. It's important to note that both studies primarily aimed to evaluate the performance of different data sampling methods in their respective contexts.

2.7.2 Deep Learning Techniques

The paper [47] discusses the limitations of traditional machine learning techniques, such as One-Class Support Vector Machine and Isolation Forest, in identifying anomalies in Ethereum transactions. It points out that these techniques struggle to capture the internode or account relationship information within transactions. To address this issue, the authors propose the utilization of a One-Class Graph Neural Network-based anomaly detection framework tailored for Ethereum blockchain network analysis. Empirical evaluations conducted in the study illustrate that this proposed method outperforms traditional non-graph-based machine learning algorithms in terms of anomaly detection accuracy. In [48], the authors introduce a novel approach for Bitcoin transaction analysis to detect potential money laundering anomalies. It leverages a combined method, integrating random forests with information from a graph convolutional network. The model's results suggest the presence of possible shadow transactions, estimated at 2-3% of the total market. The study in [49] presents a deep learning-based anomaly detection model in the financial sector, employing Long Short-Term Memory (LSTM), Gated Recurrent Unit (GRU), and one-dimensional Convolutional Neural Network (1dCNN) algorithms. Hyperparameter optimization is performed using the grid search method, and the methods are applied to Tesla's stock market and Ethereum cryptocurrency datasets. The comparative analysis reveals that the GRU algorithm achieves the highest prediction score in both datasets, while the 1dCNN algorithm performs the least effectively. Furthermore, graphical representations of anomaly values using the GRU algorithm are showcased for both datasets. The authors in [50] address the identification of illegal transactions in the Bitcoin network by extracting nineteen features and proposing a deep learning-based graph neural network model. The model is compared to graph attention network (GAT2) and extreme gradient boosted decision tree (XGBOOST) techniques, utilizing a dataset comprising 13,310,125 transactions from 2,059 entities across 3,152,202 Bitcoin account addresses and 28 user categories. Two experiments are conducted: binary classification for legal or illegal transactions and multi-class classification to determine the transaction originator's category. The proposed models achieve up to 92% accuracy in both tasks, demonstrating their effectiveness for real-world deployment. A one-dimensional CNN model for network anomaly detection in cybersecurity is introduced by the authors

in [51]. It categorizes network traffic data into TCP, UDP, and OTHER protocols, addressing class imbalance through over-sampling after feature selection using the Chi-square technique. The model achieves weighted average F-scores of 0.85, 0.97, 0.86, and 0.78 for TCP, UDP, OTHER, and ALL categories, respectively, and is tested on the UNSW-NB15 dataset. In [52], the authors have devised an encoder-decoder-based deep learning model to detect anomalous activities within the Ethereum transaction network. Their work marks a pioneering effort in employing deep learning techniques, along with some feature engineering, to identify illicit activities within the Ethereum network. On the other hand, the authors in [53] employ both clustering and role detection methods to identify suspicious users within Bitcoin transaction data. K-means is utilized for clustering, while RoIX is employed for role detection. However, it's worth noting that their approach primarily categorizes anomalous activities without providing a quantitative measure for them.

While previous studies focused on employing supervised and unsupervised algorithms solely for classifying illicit transactions in blockchain transaction data, limited attention has been given to addressing class imbalance. This imbalance poses challenges in detecting anomalies in areas such as illicit activities in digital currencies, money laundering, and various single or multi-class classifications [38]. To tackle this issue, researchers have introduced various undersampling and oversampling methods aimed at enhancing evaluation metrics. In studies [54] to [55], SMOTE over-sampling balancing methods have been applied to classify illegal activities in credit card transactions. Additionally, studies by the authors of [56] and [57] have explored various under-sampling techniques to distinguish between normal and illegal activities in credit card transactions. In a different context, the authors [58] have addressed the challenge of handling highly imbalanced datasets by employing several undersampling techniques for early product back-order prediction.

2.8 Research Gap

In many instances, the frequency of anomalous data points significantly pales in comparison to that of non-anomalous data points, thereby yielding imbalanced datasets. Such skewed data distribution can exert an adverse impact on the efficacy of anomaly

detection algorithms. These algorithms, often biased towards the majority class (normal data), grapple with difficulties in accurately discerning the minority class (anomalous data). Several Over and Under-sampling techniques such as Synthetic Minority OverSampling Technique (SMOTE), Adaptive Synthetic (ADASYN), Random Under Sampling (RUS), Near-Miss, etc. have been used to handle imbalanced data in various domains[8]. A major problem in most of the under-sampling algorithms is that significant instances which have a great impact on model training may be missed [9]. Tree-based machine learning classifiers have been used in many studies to classify malicious activities [11] since faster training can be performed on tree-based classifiers [12]. However, several studies show that the ensemble method can perform better in the case of large-scale data e.g. Bitcoin transactions than a single machine learning algorithm [13]. In recent times, the 1D Convolutional Neural Network (1D CNN) has garnered attention for its swift feature extraction capabilities, making it a preferred choice for anomaly detection [59], intrusion detection [60], and fault detection [61] across various domains. Despite its proven efficacy in these applications, there is a notable scarcity of studies harnessing the potential of the 1D CNN model for anomaly detection within Blockchain transaction data. Finally, a key limitation is the lack of explanations for the model's predictions and human interpretability.

2.9 Summary

This chapter provides an overview of various approaches documented in the literature for detecting or classifying anomalies in Blockchain, emphasizing their limitations and identifying potential research gaps. The subsequent chapter delves into our proposed methodology, leveraging both machine learning and deep learning techniques. It details the experimental outcomes and engages in a comprehensive discussion of the results.

Chapter 3

Proposed Method

The initial phase involved dataset collection and subsequent preprocessing. Following this, the dataset was split into training and test data partitions. Notably, data sampling was exclusively applied to the training data, while the test data remained independent. The sampled data was utilized for training both machine learning classifiers and the Proposed deep learning model. Ultimately, independent test data was employed to validate the models, utilizing a range of evaluation metrics including Accuracy, TPR, FPR, and ROC-AUC score. Furthermore, the study incorporates a comparative analysis, complemented by eXplainable Artificial Intelligence (XAI) techniques and rules from Decision Tree, which are elaborated upon in the results section. The overall methodology of this study is illustrated in Figure 3.1.

3.1 Dataset

The Bitcoin transaction data utilized in this study was sourced from the IEEE Data Portal [1]. The period from 2011 to 2013 was chosen due to the accessibility of anomalous Bitcoin transaction data during this timeframe. The Bitcoin Forum contains a wealth of information on fraudulent BTC transactional activities, including incidents categorized as BTC Thefts, BTC Hacks, and BTC Losses. Each case highlights red-flagged transactions, providing insights into the date and the amount of BTC stolen or lost.

The raw Bitcoin data undergoes a transformation process to imbue it with meaning and structure, with fraudulent transactional cases serving as labels. Parsing the

[1] https://ieee-dataport.org/open-access/.bitcoin-transactions-data-2011-2013

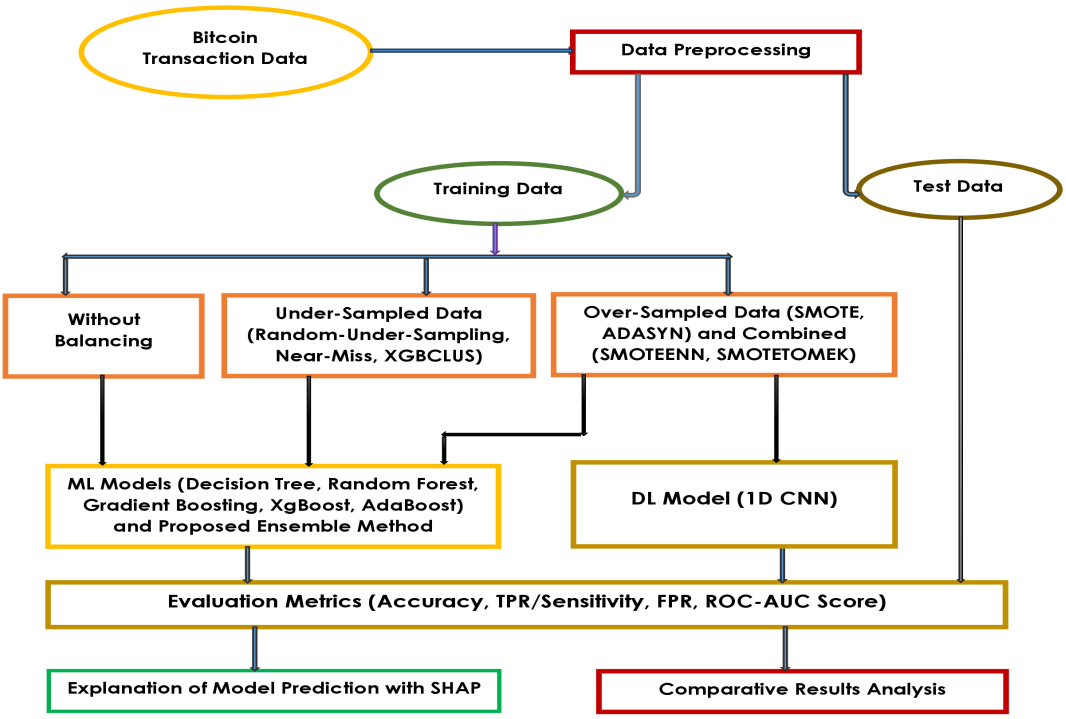


Figure 3.1: Methodology of this study

transactional data from these raw files results in a representation resembling a directed graph structure. As outlined by [25], this data is logically interconnected and can be visualized as a transaction network, denoted as T. In this network, each vertex represents a transaction, while directed edges between a source and destination encapsulate details such as the number of bitcoins and the timestamp of the transaction. These directed edges originate from the input of the transaction and extend towards the corresponding target output. The authors [62] utilized a previously generated CSV file along with the transaction graph to construct a directed acyclic graph (DAG). Within this DAG, each vertex symbolized a transaction, characterized by distinct indegree and outdegree values, as well as specific quantities of bitcoins (BTC) inflowing and outflowing. Traversing the DAG facilitated the extraction of features, which were subsequently integrated into a dataset. During the traversal of the DAG, each transaction underwent labeling as either anomalous or non-anomalous, thereby culminating in the creation of a final dataset comprising metadata extracted from the DAG.

The features in Table 3.2 were extracted from the DAG. Indegree represents the

count of transactions from which a specific transaction receives Bitcoins and Outdegree represents the count of transactions for which a specific transaction sends Bitcoins. The total amount of bitcoins received through incoming edges to a transaction is denoted as in_btc whereas out_btc denotes the total amount of bitcoins sent through outgoing edges from a transaction. out_and_tx_malicious indicates whether a transaction is flagged as malicious or not. All other features such as total_btc, mean_in_btc, mean_out_btc, etc. were generated from these basic features applying summation or average or counting.

This dataset comprises a total of 30,248,134 samples, with the majority, specifically 30,248,026, being labeled as negative samples, denoting non-malicious transactions. In stark contrast, there are only 108 samples labeled as malicious. This dataset exhibits a significant class imbalance, emphasizing the scarcity of malicious instances. Exploratory Data Analysis (EDA) was conducted to gain insights into the dataset's characteristics. It encompasses 12 attributes, with each transaction being assigned a label indicating its status as anomalous (labeled as 1) or non-anomalous (labeled as 0). A correlation matrix of the features is shown in Figure 3.2. Features such as

Table 3.1: The T-values and P-values for all attributes

Attribute	t value	p value
indegree	-14.013838	0.000000
outdegree	0.842249	0.399648
in_btc	-17.229753	0.000000
out_btc	-16.469202	0.000000
total_btc	-16.864202	0.000000
mean_in_btc	-8.727102	0.000000
mean_out_btc	-16.014732	0.000000
in_malicious	-68.869826	0.000000
out_malicious	-5432.702805	0.000000
is_malicious	-3878.622465	0.000000
all_malicious	-1866.899584	0.000000

in_btc, out_btc, total_btc, mean_in_btc, and *mean_out_btc* exhibit strong positive correlations. Conversely, the *indegree* and *outdegree* features display weak correlations. While *in_malicious, out_malicious, is_malicious,* and *all_malicious* features are notably correlated with the output feature *out_and_tx_malicious*, no substantial correlations

are observed between these features and *indegree, outdegree, in_btc, out_btc, total_btc, mean_in_btc,* and *mean_out_btc*. Furthermore, a hypothesis test for feature selection

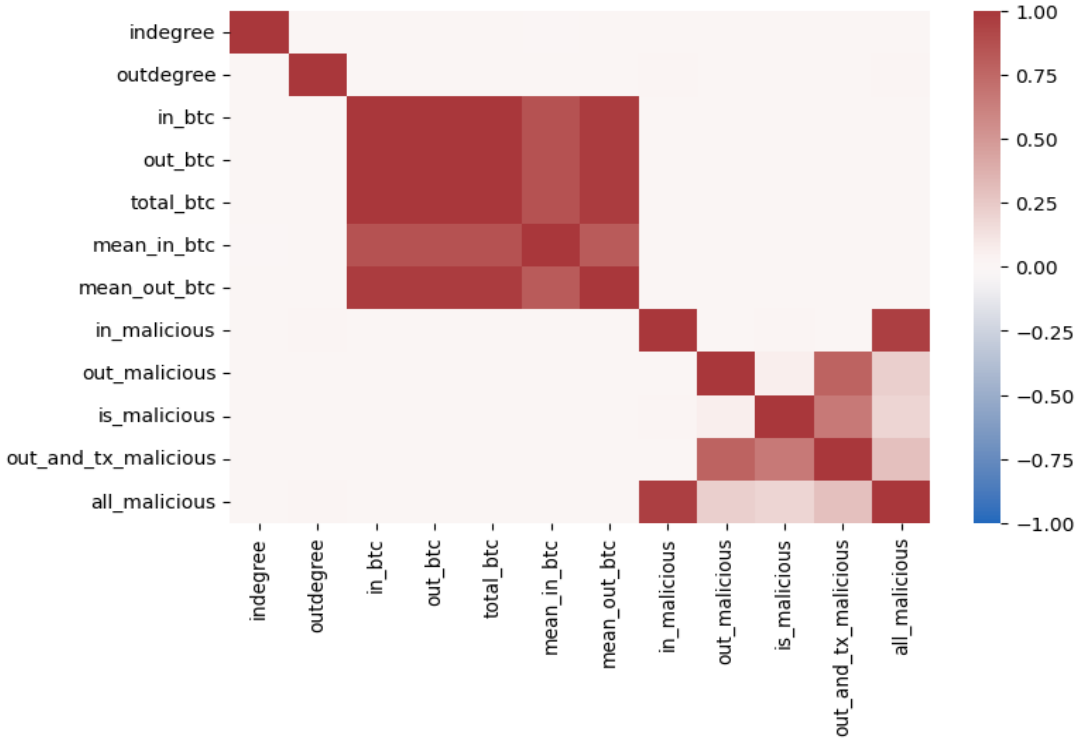


Figure 3.2: Correlations among the features using Heatmap

was conducted using the T-test to explore the correlations between positive (anomalous Bitcoin transactions) and negative (non-anomalous Bitcoin transactions) samples. The T-test assesses whether a significant difference exists between the means of the positive and negative samples. T-statistic values and corresponding p-values were computed for each attribute, as presented in Table 3.1. Notably, all attributes exhibited significance with p-values below 0.01, except for *outdegree*. It's also important to mention that all features (*in_malicious, out_malicious, is_malicious,* and *all_malicious*) share the same value range as the target feature *out_and_tx_malicious*. Consequently, this similarity poses a challenge for ML classifiers in effectively distinguishing Bitcoin transactions, leading to the exclusion of these four features and *outdegree*. As a result, a set of seven features, including the target feature, was selected for classification. A summary of the selected features is presented in Table 3.2. After completing the feature selection

Table 3.2: Summary of the selected features

Feature Name	description
Indegree	No. of inputs for a given transaction
in_btc	No. of Bitcoins on each incoming edge to a given transaction
out_btc	No. of Bitcoins on each outgoing edge from a given transaction
total_btc	Total number of Bitcoins for a given transaction
mean_in_btc	Average number of Bitcoins on each incoming edge to a given transaction
mean_out_btc	Average number of Bitcoins on each outgoing edge from a given transaction
out_and_tx_malicious	Status of a given Bitcoin transaction if it is malicious or not

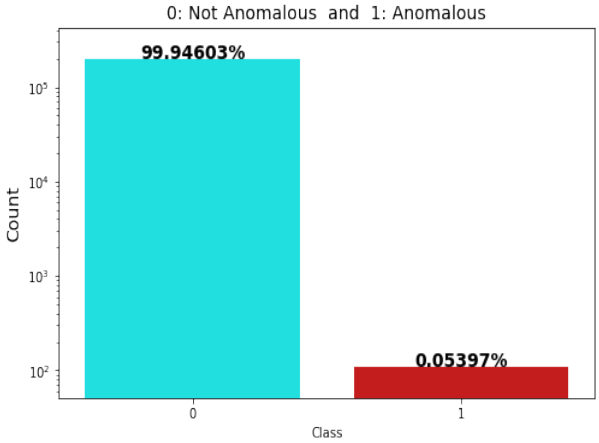


Figure 3.3: Class ratio

process, the dataset was partitioned into negative and positive samples, with a specific focus on eliminating duplicate entries exclusively from the negative samples. To manage computational complexity, a decision was made to retain only 200,000 negative samples, while maintaining the 108 positive samples. However, it's important to note that this choice still resulted in a significantly high imbalance ratio, as depicted in Figure 3.3.

3.2 Imbalanced Data Handling

In Bitcoin transactions, the number of illicit transactions is significantly lower than that of normal transactions, leading to an imbalanced dataset. Consequently, machine learn-

ing classifiers tend to exhibit bias toward the majority class [63]. While classification accuracy might appear satisfactory in many cases, a notable discrepancy between True Positive and False Positive values often arises—indicating that the models struggle to accurately classify anomalies. To address this, it is crucial to balance the dataset using built-in or customized sampling techniques prior to training the classification models. In scenarios involving anomaly detection, fraud identification, or money laundering, positive cases are typically scarce. As such, undersampling techniques can prove effective in rebalancing the dataset while prioritizing accurate identification of positive cases. However, in instances where the number of positive cases or anomalies in the minority class is exceedingly low, ML classifiers may be trained on a limited dataset generated by the undersampling technique. On the other hand, over-sampling methods aim to increase the instance count of the minority class to match that of the majority class. Despite generating artificial data based on a combination of majority and minority samples, these methods can be effective. In our study, we introduce an under-sampling algorithm named XGBCLUS, and we also investigate established under-sampling techniques such as Random Under Sampling (RUS) and Near-Miss. Furthermore, we explore popular over-sampling techniques including SMOTE, ADASYN, as well as combined approaches like SMOTEENN and SMOTETOMEK. We present a comparison between over-sampling and under-sampling methods in the Result Analysis Section.

3.2.1 Under-Sampling Techniques

We have investigated two under-sampling methods e.g. Random Under Sampling and Near-Miss along with our proposed under-sampling method which is described in Section 3.2.1.1. Random Under Sampling (RUS) is a simple technique used to handle class imbalance in datasets. It randomly selects a subset of instances from the majority class. The size of this subset is determined based on the desired balance ratio between the minority and majority classes. The balance ratio α_{us} is defined by Equation 3.1.

$$\alpha_{us} = \frac{N_m}{N_{rm}} \tag{3.1}$$

where N_m is the number of samples in the minority class and N_{rm} is the number of samples in the majority class after resampling. Then, the instances from both the minority class and the randomly selected subset of the majority class instances are

combined to form a balanced dataset. Another under-sampling technique namely Near-Miss is also used for balancing the dataset. It reduces the imbalance by retaining a subset of instances from the majority class that are close to instances from the minority class. This down-sampling technique selects instances based on their proximity to the minority class, making it possible to preserve important samples. For each instance in the minority class, Near-Miss calculates its distances to all instances in the majority class using various distance metrics such as Euclidean distance or Manhattan distance. After that, this algorithm identifies the k instances from the majority class that are closest to each instance in the minority class. The value of k is typically set as a hyperparameter and determines the degree of under-sampling. We set the value as 1 and hence we call it Near-Miss-1. Finally, it combines the instances from both the minority class and the selected instances from the majority class to form an under-sampled dataset.

3.2.1.1 Proposed XGBCLUS Algorithm

XGBCLUS (eXtreme Gradient Boosting-based Clustering) operates by merging clusters (accomplished by selecting random instances from the majority class in the training data, equal in number to the positive instances from the minority class) and the Extreme gradient Boosting algorithm. The algorithm starts with splitting the whole dataset into train and test data. The test data is kept independent and the positive samples, $\boldsymbol{P}$, are counted from the training set. Then, the number of iterations, $\boldsymbol{k}$, is calculated by dividing the total number of negative samples in the training data by the positive samples, $\boldsymbol{P}$.

In each iteration, the algorithm arbitrarily selects $\boldsymbol{n}$ negative samples equal to $\boldsymbol{P}$ and a new training set is prepared to train the Xgboost model. Using the independent test set, the model predicts the True Positive (TP) and False Positive (FP) values. The values of TP and FP values are compared with TMAX and FMIN, respectively. TMAX and FMIN are initialized with arbitrary values where the TMAX represents the maximum true positive value and the minimum false positive value is defined by FMIN. If the TP value is greater than the TMAX value and the value of FP is less than the FMIN value, then both TMAX and FMIN are updated with the TP and FP values respectively. At the same time, current $\boldsymbol{n}$ samples are updated in the *Selected_Samples* set. Otherwise, no changes are made in the current iteration.

Algorithm 1 XGBCLUS algorithm

Input: Imbalanced Training samples, *DATA*; Number of iterations, k; Number of positive samples,P; The independent test data and The XGBoost algorithm.

Output: Selected Under-Sampled data

0: Initialize $TMAX$ and $FMIN$
0: Initialize an empty set *Selected_Samples*
0: **for** $i = 1$ to k **do**
0: Select n negative samples arbitrarily equal to P and Prepare the Train data
0: Train the model and predict using the test samples
0: Calculate True Positive (TP) and False Positive (FP) values
0: **if** $TP > TMAX$ and $FP < FMIN$ **then**
0: Set $TMAX = TP$ and $FMIN = FP$
0: Update current n samples in *Selected_Samples*
0: **end if**
0: **end for**
0: **if** *Selected_Samples* is empty **then**
0: **Goto** step 3 and repeat the steps 3 - 13 after changing $TMAX$ and $FMIN$ values
0: **end if**
0: **Return** *Selected_Samples*;
=0

After the ***k*** iterations are finished, the *Selected_Samples* set is checked. If the set is empty, the algorithm should be run again with new TMAX and FMIN values. Otherwise, the samples in the *Selected_Samples* set are the under-sampled data returned by the algorithm. The XGBClus algorithm is shown in Algorithm 1.

3.2.2 Over-Sampling Techniques

In our investigation, we have explored two commonly used over-sampling techniques to address class imbalance in Bitcoin transaction data. Between these two methods, the Synthetic Minority Over-sampling Technique (SMOTE) stands out as a widely adopted approach for mitigating the class imbalance problem [64], particularly in the context of Bitcoin Transactions for anomaly detection. The core objective of SMOTE is to rectify the class distribution imbalance by creating synthetic instances of the minority class. This process helps alleviate bias issues and enhances the generalization of the model. For each minority sample X_i, SMOTE randomly selects k nearest neighbors from the same class. Subsequently, a new sample X_n is generated using one of the nearest neighbors X_{zi} from the set of k neighbors. The generation of the new sample follows the equation 3.2.

$$X_n = X_i + \lambda * (X_{zi} - X_i) \tag{3.2}$$

Where λ represents a random number ranging from 0 to 1. Consequently, a new synthetic instance is created within the feature space. This process iterates until the number of samples in both the majority and minority classes becomes equal. Ultimately, the original minority instances are combined with the newly generated synthetic instances to establish a balanced dataset.

ADASYN [65] (Adaptive Synthetic Sampling) employs the same formula as mentioned earlier for generating new samples, with the exception being the selection of X_i. ADASYN focuses on enhancing the density of synthetic instances in regions that pose greater classification challenges, offering a more nuanced approach to addressing class imbalance. For each minority instance, ADASYN initially calculates the number of its k nearest neighbors that belong to the majority class, providing an indication of the proximity of the minority sample to the majority class. Subsequently, it calculates the imbalance ratio α_{os} for each minority instance using the equation 3.3.

$$\alpha_{os} = \frac{N_{rm}}{N_m} \tag{3.3}$$

Here, N_{rm} represents the number of samples in the minority class after resampling, and N_m denotes the number of samples in the majority class. This imbalance ratio is employed to determine the desired count of synthetic instances to be generated for the current minority instance. Additionally, a difficulty ratio is calculated to assess the level of difficulty. When this ratio is high, more neighbors are considered for generating synthetic instances. By interpolating feature values between the minority instance and its selected neighbors, a new synthetic instance is generated. This process continues for all minority instances until the number of samples in both the majority and minority classes is equalized. Ultimately, the original minority instances are merged with the newly generated synthetic instances to construct a balanced dataset.

3.2.3 Combined-Sampling Techniques

Additionally, we have explored two combined sampling strategies to address class imbalance in Bitcoin transaction data. SMOTEENN [66] is a combination of two resampling techniques: SMOTE (Synthetic Minority Over-sampling Technique) and Edited Nearest Neighbors (ENN). This approach is designed to address the class imbalance by initially generating synthetic samples using SMOTE and subsequently enhancing the dataset quality through the application of ENN. The primary goal of SMOTEENN is to provide a more sophisticated approach to balancing imbalanced datasets while concurrently improving dataset quality by eliminating potential noise. Following the generation of synthetic instances via SMOTE, ENN operates as follows: for each instance in the dataset, ENN identifies its k nearest neighbors. If the instance's class differs from the majority class of its neighbors, ENN removes the instance from the dataset, effectively eliminating noisy or misclassified instances.

Another combined resampling technique is SMOTETOMEK [67], where synthetic instances are produced using SMOTE, and the under-sampling technique TOMEK Link identifies pairs of instances from different classes that are closest to each other. For each pair of instances identified as Tomek links, the majority class instance is removed. This undersampling method assists in eliminating instances that are in close proximity to the decision boundary and may be prone to misclassification. The outcome is a more balanced, discriminative, and effective dataset for training machine learning models.

3.3 Proposed Ensemble Model

The meta-classification ensemble method based on stacked generalization is a Machine Learning (ML) approach used to improve the accuracy of predictions by combining multiple models [68]. The stacking-based ensemble model is formed by two classifiers. One is the base classifier and the other is the meta classifier. It starts with training a set of base models using several classifiers. A new dataset is found from the base-level classifiers and then the meta-classifier, also known as a combiner or a blender, is trained using the new dataset. After that, the learned meta-classifier is used to predict the independent test dataset. The architecture of the proposed stacked-ensemble model is shown in Figure 4.

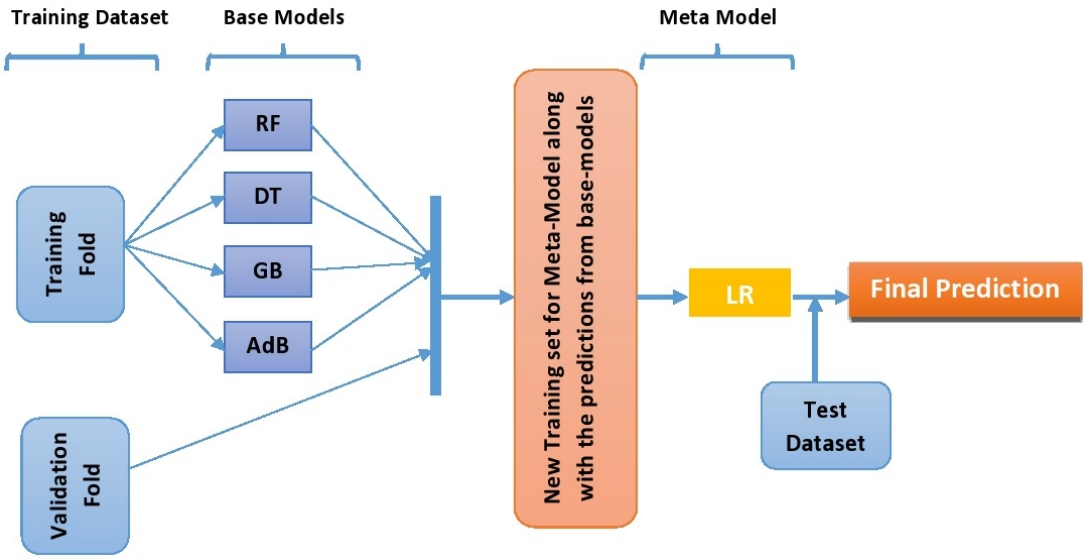


Figure 3.4: Stacked-Ensemble Model architecture

In our proposed stacking-based ensemble model, Random Forest (RF), Decision Tree (DT), Gradient Boosting (GB), and Adaptive Boosting (AdB) have been used as the base models. The training dataset is used to train the base models and the outputs of the four base models along with the validation fold are combined to create a new dataset. Then Logistic regression (LR) [69], which is the meta-classifier in our proposed model, receives the newly formed dataset by combining the predictions of the base classifiers as input and learns on that input set. Finally, the test dataset has been used to predict anomalous and non-anomalous transactions.

On the Other side, the Voting Classifier is constructed using only tree-based models, which are a family of machine learning algorithms known for their robustness and interpretability. The architecture of a Voting Classifier that incorporates tree-based models like Decision Trees (DT), XGBoost (XGB), Gradient Boosting (GB), Random Forest (RF), and AdaBoost (ADB) is shown in Figure 5. The Voting Classifier takes the training set as input and the ensemble is constructed by combining the predictions of several individual tree-based models. Each model in this context refers to a unique instantiation of the tree-based algorithm with a specific set of hyperparameters or configurations. After the Voting Classifier has been trained and evaluated, it is used to make predictions on test data. The Voting Classifier aggregates the predictions of each individual tree-based model using a voting mechanism. The voting can be either "hard" or "soft". In hard voting, each model in the ensemble casts a single vote for the predicted class label, and the majority class receives the final prediction. For soft voting, the probabilities (confidence scores) of each model's predicted classes are averaged, and the class with the highest average probability is chosen as the final prediction.

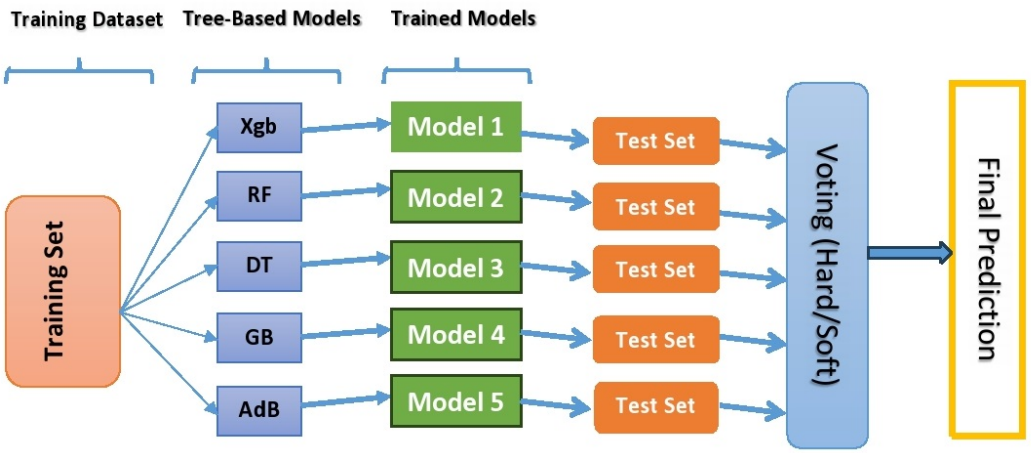


Figure 3.5: Voting-Ensemble Model architecture

3.4 Proposed 1D CNN

The 1D CNN architecture, as shown in Figure 3.6, is designed to extract and transform features from Bitcoin transaction data through convolutional layers, normalize activa-

tions with batch normalization, reduce spatial dimensions with max-pooling, and make a binary classification decision in the output layer. It encompasses various layers and techniques to balance model complexity and performance while preventing overfitting, ultimately serving as a robust tool for detecting fraudulent transactions within Bitcoin data.

The feature extraction process begins with Conv1D layers, two in total, each serving a unique purpose. These layers apply convolutional operations to the input sequences, enabling them to capture intricate patterns and representations. The first Conv1D layer is responsible for extracting important features from the input data (6, 1) where 6 represents the length of the input sequence and 1 for the dimension of each feature (usually 1 for scalar values). It uses 32 filters to scan the input data with a kernel size of 3, allowing it to capture local patterns within the input sequence. It is also followed by a Rectified Linear Unit (ReLU) activation function to introduce non-linearity. The second Conv1D layer follows a similar pattern but utilizes 64 filters, thereby enabling the extraction of increasingly complex patterns from the data.

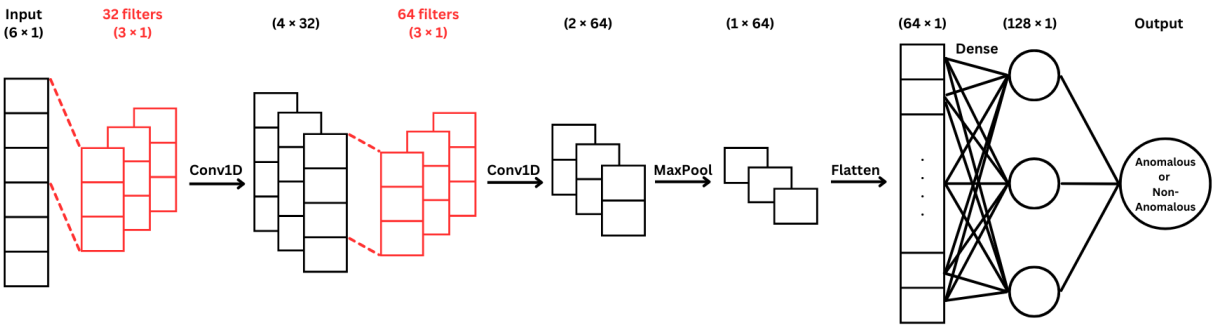


Figure 3.6: Architecture of proposed 1D CNN model

Batch Normalization layers are strategically placed after each Conv1D layer to stabilize and accelerate the training process. These layers normalize the activations produced by the convolutional layers, ensuring they have a mean of 0 and a standard deviation of 1. This normalization is pivotal in improving training stability and convergence.

To reduce the spatial dimensions of the feature maps while retaining essential information, a MaxPooling1D layer is incorporated. This layer effectively halves the sequence length, thus reducing computational complexity and retaining the most significant information present in the data. The flattened layer follows the MaxPooling1D

layer, reshaping the 3D tensor output into a 1D vector. This transformation prepares the data for processing by the fully connected layers, ensuring compatibility between the convolutional and dense layers.

Layer	Kernel Size	No. of Filters	Activation	Output Shape	Parameters
Conv1D	3	32	relu	(None, 4, 32)	128
BatchNormalization				(None, 4, 32)	128
Conv1D	3	64	relu	(None, 2, 64)	6,208
BatchNormalization				(None, 2, 64)	256
MaxPoolling1D	2			(None, 1, 64)	0
Flatten				(None, 64)	0
Dense			relu	(None, 128)	8,320
Dense			sigmoid	(None, 1)	129
Total Parameters					15,169

Table 3.3: Details of the Layers of the Proposed CNN Model

The architecture employs two dense (fully connected) layers for classification. The first dense layer consists of 128 neurons and incorporates the ReLU activation function. Additionally, it applies dropout regularization with a 50% dropout rate, which helps mitigate overfitting. The second dense layer functions as the output layer, housing a single neuron, and uses the sigmoid activation function for binary classification. This layer is responsible for making the final prediction regarding the nature of the Bitcoin transaction. Details of the Layers of the Proposed model are given in Table 3.3.

Learning Rate	0.0001
Optimizer	Adam
Batch Size	32
Loss Function	binary_crossentropy

Table 3.4: Parameters for Model Compilation

For optimization, we employed the Adam stochastic gradient descent method with a learning rate of 0.0001. During training, binary cross-entropy was utilized to compute the loss. A batch size of 32 was employed, implying that 32 samples were used per gradient update. A list of parameters used for model compilation is given in Table 3.4.

Given the approximately 320,000 samples in the training set, this resulted in (320,000 ÷ 32) = 10,000 (approximately) steps needed per epoch to iterate over the entire training set. Initially, we used 100 epochs for training our proposed CNN model and monitored the accuracy and loss updates for both the training and validation sets. The number of epochs was determined using the early stopping technique with a patience of 10 epochs to restore the model's best weights to mitigate overfitting.

3.5 Evaluation metrics

Given that accuracy alone is insufficient to gauge the performance of an anomaly detection system, it becomes crucial to employ additional metrics such as True Positive Rate (TPR) to assess the accurate identification of anomalous transactions and False Positive Rate (FPR) to evaluate the correct identification of non-anomalous transactions. This aligns with the primary objective of our study. The evaluation metrics e.g. accuracy, True Positive Rate (TPR), and False Positive Rate (FPR) have been used to compare the performance of the single models without and with balancing the data against the proposed ensemble models. Additionally, we also use the feature importance score to show the hierarchy of the features. We have also considered the Receiver Operating Characteristic (ROC) score, which compares the True Positive Rate (TPR) against the False Positive Rate (FPR). The performance metrics are defined below:
TP = True Positive: an anomalous transaction is correctly identified as anomalous
TN = True Negative: a non-anomalous or normal transaction is correctly identified as non-anomalous
FP = False Positive: a non-anomalous transaction is incorrectly identified as anomalous
FN = False Negative: an anomalous transaction is incorrectly identified as non-anomalous

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{3.4}$$

$$TPR = Sensitivity = \frac{TP}{TP + FN} \tag{3.5}$$

$$TNR = Specificity = \frac{TN}{TN + FP} \tag{3.6}$$

$$FPR = \frac{FP}{TN + FP} \tag{3.7}$$

AUC is calculated as the Area Under the $Sensitivity(TPR)$ - $(1-Specificity)(FPR)$ Curve.

3.6 Summary

This chapter provides an overview of our proposed methodology, leveraging both machine learning and deep learning techniques. The subsequent chapter delves into details of the experimental outcomes and engages in a comprehensive discussion of the results.

Chapter 4

Experimental Analysis

In this chapter, we assess the performance of the proposed ensemble models using both under-sampled and over-sampled data. Additionally, a comprehensive comparative analysis between single classifiers and ensemble classifiers is presented herein. Additionally, we evaluate the proposed 1D CNN model, considering both oversampled and combined sampled data. Furthermore, we present a comprehensive comparative analysis between the performance of machine learning classifiers and the proposed 1D CNN model. We commence by configuring the experimental environment and subsequently present outputs that scrutinize various facets of the model's performance concerning Bitcoin anomaly detection.

4.1 Environment Setup

We employed the Python programming language for implementation and utilized Jupyter Notebook for executing all modules. The computational tasks benefited from the capabilities of Google Colab, which leveraged 12.68 GB of RAM and 225.83 GB of disk space for enhanced hardware support. A suite of libraries and packages played integral roles in diverse tasks: Pandas facilitated data extraction, Matplotlib enabled visualization and plotting, Numpy performed various mathematical functions, Scikit-learn handled tasks such as binarizing output, dataset splitting, and generating classification reports and confusion matrices, Keras was instrumental in building models using the TensorFlow library, Seaborn was employed for plotting confusion matrices, and shap contributed to explaining the model's output.

4.2 Effects of Under-Sampling in Classification with ML

We have kept 20% data for the independent test set and the remaining 80% has been used for training the models. To prove the data imbalanced problem, the classifiers have been trained without balancing the train set. The ML classifiers become biased to the majority samples and result in a high true negative value. However, the ML classifiers can not identify the positive samples correctly that's why the true positive rate is very low and in some cases, it is zero. Table 4.9 shows the comparison of accuracy, True Positive (TP), and roc-auc score of the Decision Tree (DT), Gradient Boosting (GBoost), Random Forest (RF), and Adaptive Boosting (AdaBoost) classifiers. The TP values are zero for DT and RF classifiers which indicates that no anomalous transactions are correctly identified and all transactions are classified as normal transactions. Although the accuracy seems to be good enough for the classifiers, the TP score tends to zero i.e. anomalous transactions are not identified because of the biases of models to the majority of transactions. Given that detecting anomalous transactions is the primary objective of our study, classifiers may struggle to identify those transactions without balanced data. Therefore, we explored several balancing techniques to enhance the True Positive Rate (TPR) and decrease the False Positive Rate (FPR) values.

Table 4.1: Comparison among the classifiers without balancing the data

Classifiers	Accuracy	TPR	AUC-Score
DT	0.99	0.0	0.55
GBoost	0.99	0.09	0.62
RF	0.99	0.0	0.72
AdaBoost	0.99	0.05	0.82

In Figure 4.1, the confusion matrices illustrate the performance of the ensemble classifier under different under-sampling methods, namely Random Under Sampling (RUS), Nearmiss1, and XGBClus. Notably, the True Positive (TP) values exhibit an increase compared to scenarios without balancing, where TP values are consistently zero. However, it is essential to discern that, despite the improvements in TP, the False Positive (FP) values show variations among the under-sampling methods. Specifically, Nearmiss1 displays a relatively higher FP count compared to RUS and XGBClus,

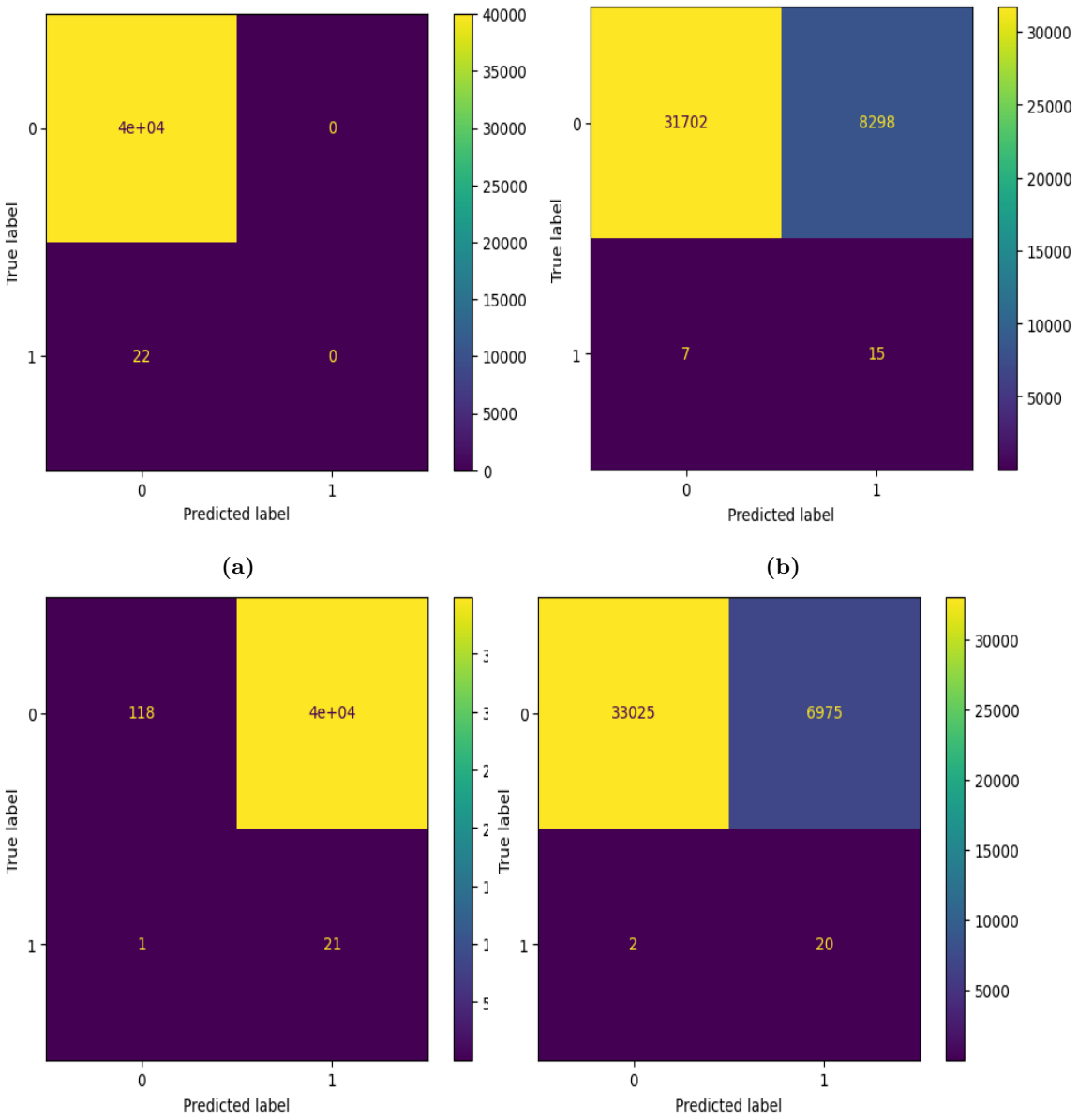


Figure 4.1: Confusion Matrix for (a) Without Balancing, (b) Ensemble classifier with RUS, (c) Ensemble classifier with NearMiss1, (d) Ensemble classifier with XGBClus

even though the FP is zero in the absence of balancing. The XGBClus undersampling method proposed in our study outperforms the existing method by achieving the highest True Positive (TP) value along with relatively low False Positive (FP) values.

This superiority stems from our algorithm's approach of considering all instances in downsampling, whereas existing algorithms randomly select instances, leading to the omission of important cases.

Given that the TPR or sensitivity signifies the count of correctly classified positive transactions, down-sampling techniques were explored to equalize the numbers of normal and anomalous transactions. We employed XGBCLUS, our proposed under-sampling method, in conjunction with other established techniques for downsampling. Figure 4.2 illustrates that the sensitivity values for all single and ensemble ML classifiers witnessed an increase, except for the NearMiss-1 under-sampling algorithm. Notably, the sensitivity values for both single and ensemble classifiers utilizing the XGBCLUS algorithm stand at 0.82, 0.86, 0.86, 0.81, 0.86, 0.81, and 0.91, respectively. These values exceed the sensitivity values obtained without balancing and those from random under-sampling techniques.

While the NearMiss-1 undersampling technique yields a higher sensitivity value compared to the XGBCLUS method, the corresponding FPR value is markedly high, as demonstrated in Table 4.14. As the False Positive Rate (FPR) decreases, the True Negative Rate (TNR) increases, indicating the correct identification of non-anomalous transactions. The NearMiss-1 undersampling technique exhibits a higher FPR, suggesting its limitation in accurately identifying non-anomalous transactions. In contrast, the random undersampling method produces average FPR values, although they are higher than the FPR values of XGBClus. In terms of TPR and FPR values, XGBClus outperforms other undersampling techniques.

Table 4.2: FPR of ML classifiers after Under-Sampling

Classifiers	Random Undersampling	NearMiss 1	XGBCLUS
DT	0.26	0.99	0.18
GBoost	0.26	0.99	0.19
RF	0.20	0.99	0.16
AdaBoost	0.26	0.99	0.21
Ensemble (Stacked)	0.22	0.99	0.15
Ensemble (Hard-Voting)	0.21	0.99	0.14
Ensemble (Soft-Voting)	0.21	0.99	0.17

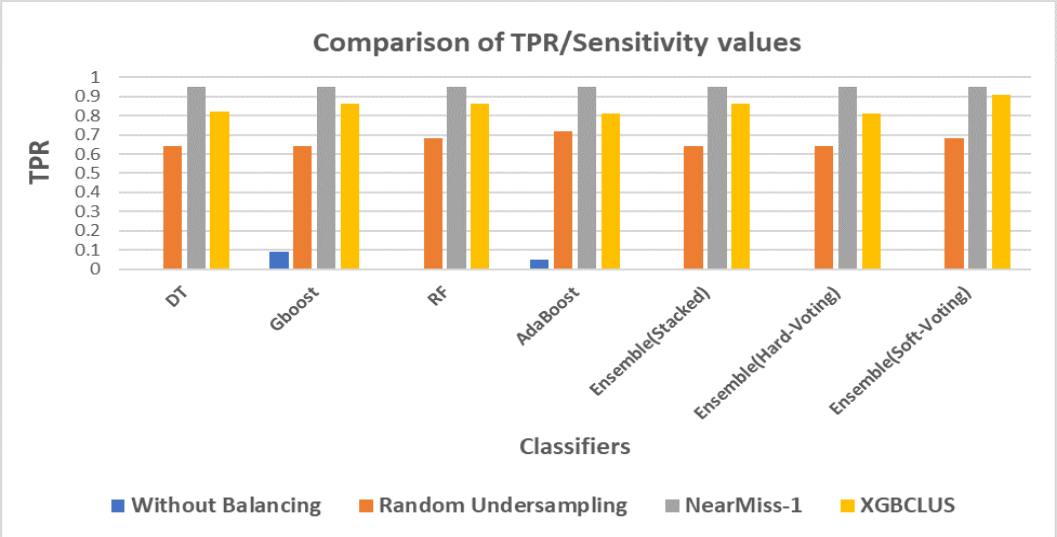


Figure 4.2: Comparison of TPR or sensitivity values using under-sampling methods

Figure 4.3 illustrates the enhanced ROC-AUC scores obtained through the utilization of under-sampled data. Notably, the proposed XGBCLUS undersampling technique outperforms RUS and Near-Miss in terms of ROC-AUC scores. The Gradient Boosting (GBoost) classifier achieves the highest ROC-AUC score of 0.92, which stands as the peak among all single and ensemble classifiers. Furthermore, the remaining classifiers achieve ROC-AUC scores ranging from 0.85 to 0.91. This range indicates that the true positive and false positive rates adhere to their expected levels.

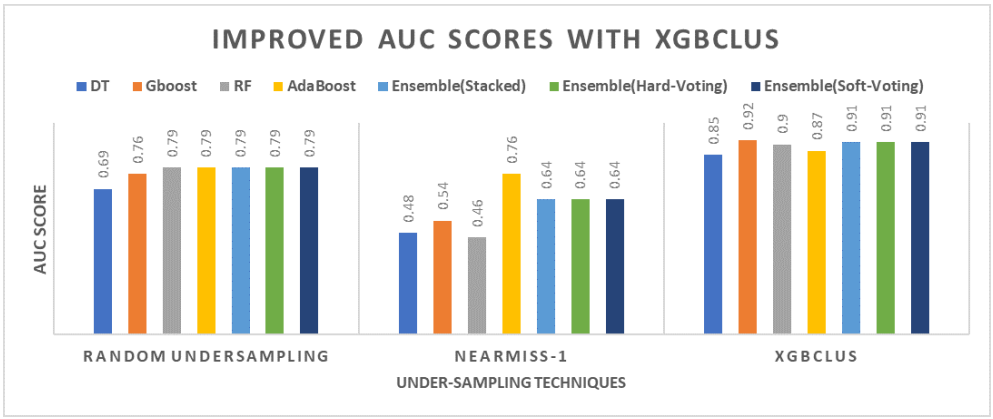


Figure 4.3: Improved ROC-AUC value using XGBClus

4.3 Effects of Over-Sampling in Classification with ML

Although the TPR scores have increased for all ML classifiers following the under-sampling of data, the FPR values remain unsatisfactory. The objective is to elevate the True Positive (TP) rate while reducing the FP rate. To achieve this balance, various over-sampling and combined methods have been applied to balance the training data. In Figure 4.4, the confusion matrices depict the performance of the ensemble classifier when employing various over-sampling methods, including SMOTE, ADASYN, SMOTEENN, and SMOTETOMEK. Notably, the True Positive (TP) values exhibit a decrease compared to the results obtained with under-sampling techniques. Despite this reduction in TP values, it is crucial to observe that the False Positive (FP) values have also shown a decrease when contrasted with the figures achieved through under-sampling methods.

Table 4.3 demonstrates that FP rates have decreased for both individual and ensemble ML classifiers. However, the attained TP rates have reduced. Among all classifiers, AdaBoost (AdB) attains the highest TPR score of 0.59 with the SMOTETOMEK sampling technique, while the Ensemble Hard-Voting (EHV) classifier secures the lowest FPR value of 0.03 i.e. only 3% across all over-sampling methods. The remaining ML classifiers, including Ensemble-Stacked (ES) and Ensemble Soft-Voting (ESV), achieve FPR scores of either 0.04 or 0.05, except for the Decision Tree (DT), Gradient Boosting (GB), and AdaBoost which record the higher FPR values between 0.06 and 0.08. Moreover, All the ML classifiers have achieved an average TPR value of about 50% except the Random Forest (RF) which scores the lowest TP rate of 0.23.

Turning to ROC-AUC scores with oversampling techniques, Table 4.4 displays the performance of all ML classifiers. Among all classifiers, AdaBoost secures the highest ROC-AUC values, ranging from 0.80 to 0.85. Conversely, the DT classifiers exhibit the lowest ROC-AUC scores with oversampled data. The ADASYN over-sampling technique demonstrates superior performance compared to other sampling methods, yielding favorable ROC-AUC values for all ML classifiers within the range of 0.71 to 0.83. overall, the ADASYN over-sampling technique shows a better performance compared to other over and combined sampling methods.

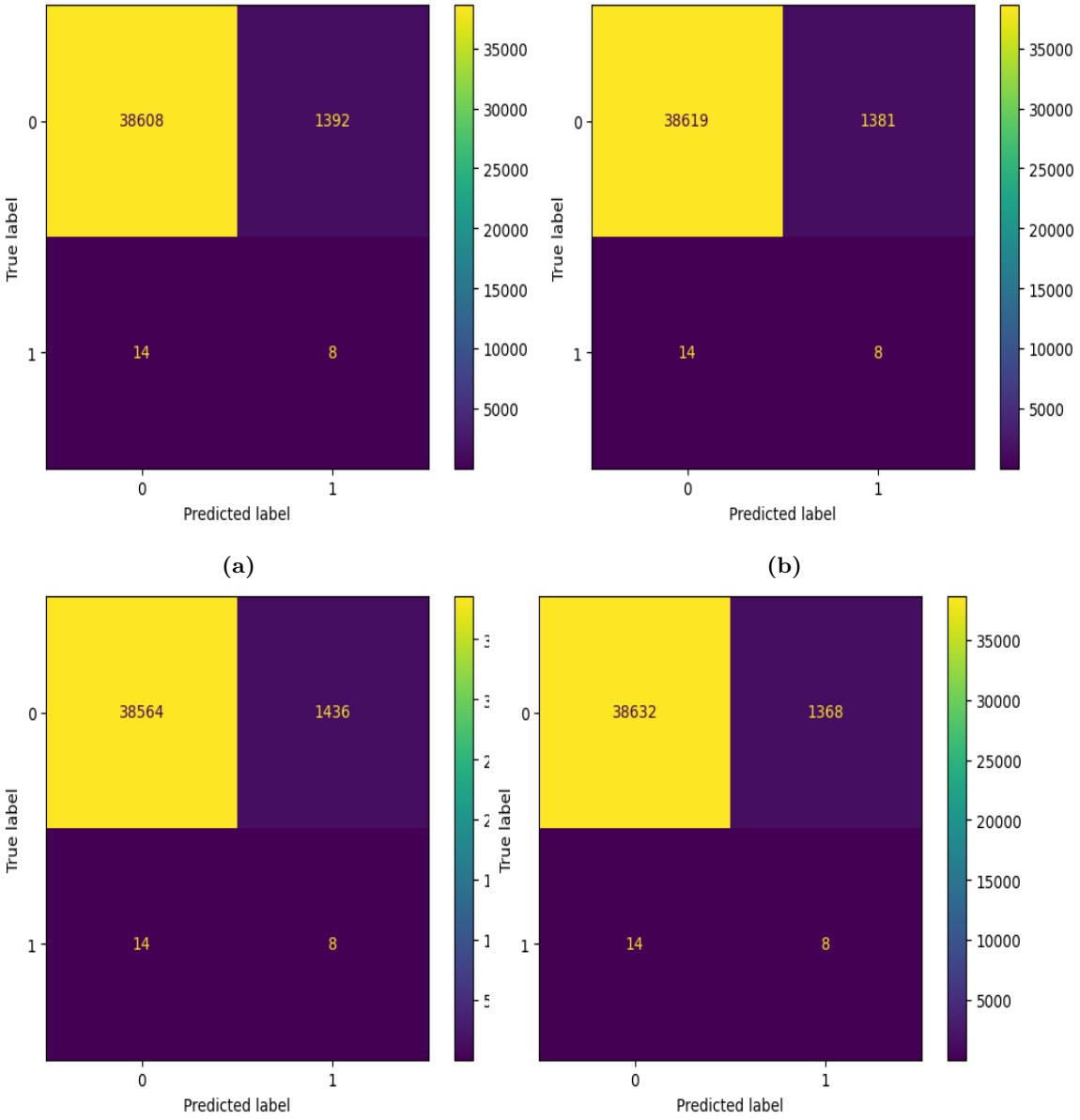


Figure 4.4: Confusion Matrix for (a) Ensemble classifier with SMOTE, (b) Ensemble classifier with ADASYN, (c) Ensemble classifier with SMOTEENN, (d) Ensemble classifier with SMOTETOMEK

Table 4.3: Comparison between TPR and FPR values of ML classifiers after Over-Sampling

	SMOTE		ADASYN		SMOTEENN		SMOTETOMEK	
	TPR	FPR	TPR	FPR	TPR	FPR	TPR	FPR
DT	0.41	0.06	0.41	0.06	0.36	0.06	0.41	0.05
GB	0.45	0.07	0.55	0.06	0.59	0.07	0.50	0.07
RF	0.23	0.04	0.32	0.04	0.36	0.04	0.23	0.04
AdB	0.59	0.08	0.59	0.08	0.55	0.08	0.59	0.08
ES	0.32	0.04	0.41	0.05	0.36	0.04	0.23	0.04
EHV	0.36	0.03	0.36	0.03	0.36	0.04	0.36	0.03
ESV	0.27	0.04	0.36	0.04	0.36	0.04	0.32	0.04

Table 4.4: ROC-AUC Scores after Over-Sampling the data

Classifiers	SMOTE	ADASYN	SMOTEENN	SMOTETOMEK
DT	0.59	0.58	0.56	0.59
GBoost	0.77	0.79	0.78	0.84
RF	0.71	0.71	0.71	0.71
AdaBoost	0.83	0.83	0.80	0.85
Ensemble (Stacked)	0.63	0.75	0.75	0.65
Ensemble (Hard-Voting)	0.63	0.75	0.75	0.6
Ensemble (Soft-Voting)	0.63	0.75	0.75	0.6

4.4 Comparative Analysis between Undersampling and Oversampling methods with ML classifiers

Both under-sampling and over-sampling techniques exert distinct effects on Blockchain anomaly detection. Under-sampling methods address the minority class instances by selecting an equivalent number of instances from the majority class through various distance-based techniques [70]. Conversely, over-sampling methods originate from the majority class instances and craft an equal number of synthetic instances from the minority class using different distance techniques [71]. Both sampling techniques yield varying impacts on the classification of Bitcoin transaction data. A comprehensive comparative analysis is detailed from Table 4.5 to Table 4.8. The confusion matrices in figure 4.1 and 4.4 suggest a nuanced trade-off between true positives and false positives, emphasizing the importance of a comprehensive evaluation of the model's performance

under different sampling strategies.

Table 4.5 showcases that TPR scores are relatively higher for under-sampling methods such as Random Under Sampling (RUS) and the proposed XGBCLUS method, compared to over-sampling and combined sampling methods. Conversely, over-sampling and combined techniques surpass under-sampling methods in terms of FPR scores. Among under-sampling methods, XGBCLUS secures the highest TPR value of 0.91, which also stands as the pinnacle across all under-sampling, over-sampling, and combined techniques. Among over and combined balancing methods, the highest TPR value is 0.59 achieved through SMOTETOMEK. Under-sampling methods prove more effective than over-sampling methods in increasing True Positive (TP) rates. With Under-sampling methods, Machine Learning classifiers can accurately identify a larger portion of anomalous transactions.

In terms of FPR scores as shown in Table 4.6, the best value of 0.03 is achieved across all over-sampling and combined methods. However, under-sampling methods yield a modest FPR score of 0.14 with XGBCLUS. Over-sampling methods prove more effective than under-sampling methods in reducing False Positive (FP) rates. With over-sampling methods, Machine Learning classifiers can accurately identify a larger portion of non-anomalous transactions.

Figure 4.5 depicts that all ML classifiers exhibit improved ROC-AUC scores using the XGBCLUS under-sampling method, outperforming the scores achieved through the ADASYN over-sampling technique. This suggests that under-sampling methods excel over over-sampling methods in terms of ROC-AUC scores.

Table 4.5: TPR or Sensitivity of ML classifiers after Under-Sampling and Over-Sampling

	Under Sampling		Over Sampling		Combined Sampling	
	RUS	XGBCLUS	SMOTE	ADASYN	SMOTEENN	SMOTETOMEK
DT	0.64	0.82	0.41	0.41	0.36	0.41
GB	0.64	0.86	0.45	0.55	0.59	0.55
RF	0.68	0.86	0.23	0.32	0.36	0.23
AdB	0.72	0.81	0.59	0.59	0.55	0.64
ES	0.64	0.86	0.32	0.41	0.36	0.23
EHV	0.64	0.81	0.36	0.36	0.36	0.36
ESV	0.68	0.91	0.27	0.36	0.36	0.32

Table 4.6: FPR of ML classifiers after Under-Sampling and Over-Sampling

	Under Sampling		Over Sampling		Combined Sampling	
	RUS	XGBCLUS	SMOTE	ADASYN	SMOTEENN	SMOTETOMEK
DT	0.26	0.18	0.06	0.06	0.06	0.05
GB	0.26	0.19	0.07	0.06	0.07	0.07
RF	0.20	0.16	0.04	0.04	0.04	0.04
AdB	0.26	0.21	0.08	0.08	0.08	0.08
ES	0.22	0.15	0.04	0.05	0.04	0.04
EHV	0.21	0.14	0.03	0.03	0.04	0.03
ESV	0.21	0.17	0.04	0.04	0.04	0.04

Table 4.7: Accuracy of Single and Ensemble classifiers after Under-Sampling and Over-Sampling

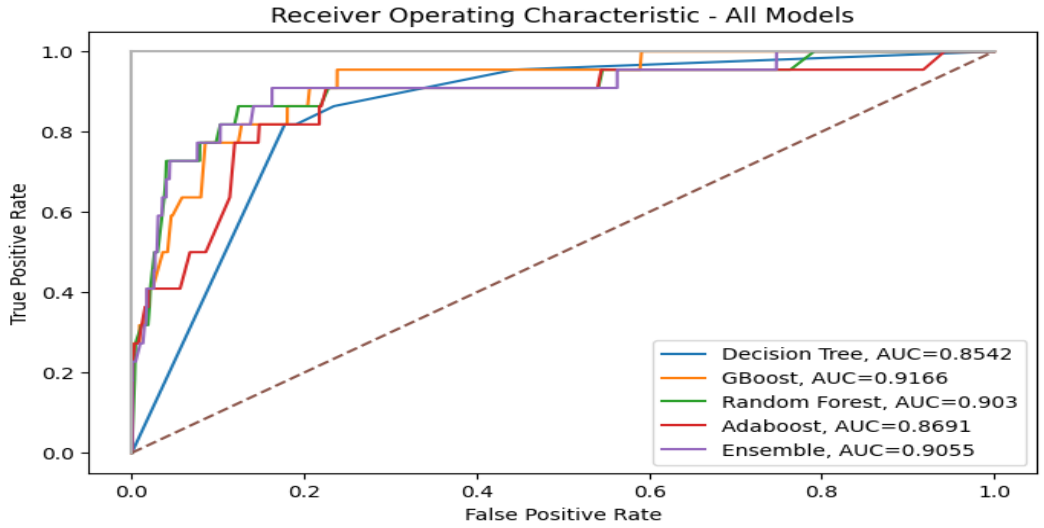
	Under Sampling		Over Sampling		Combined Sampling	
	RUS	XGBCLUS	SMOTE	ADASYN	SMOTEENN	SMOTETOMEK
DT	0.74	0.82	0.94	0.94	0.94	0.94
GB	0.74	0.81	0.93	0.94	0.93	0.93
RF	0.80	0.83	0.96	0.96	0.96	0.96
AdB	0.74	0.79	0.92	0.92	0.92	0.92
ES	0.78	0.85	0.96	0.95	0.96	0.96
EHV	0.79	0.86	0.96	0.97	0.96	0.97
ESV	0.79	0.83	0.96	0.96	0.96	0.96

4.5 Effects of Ensemble Classifiers

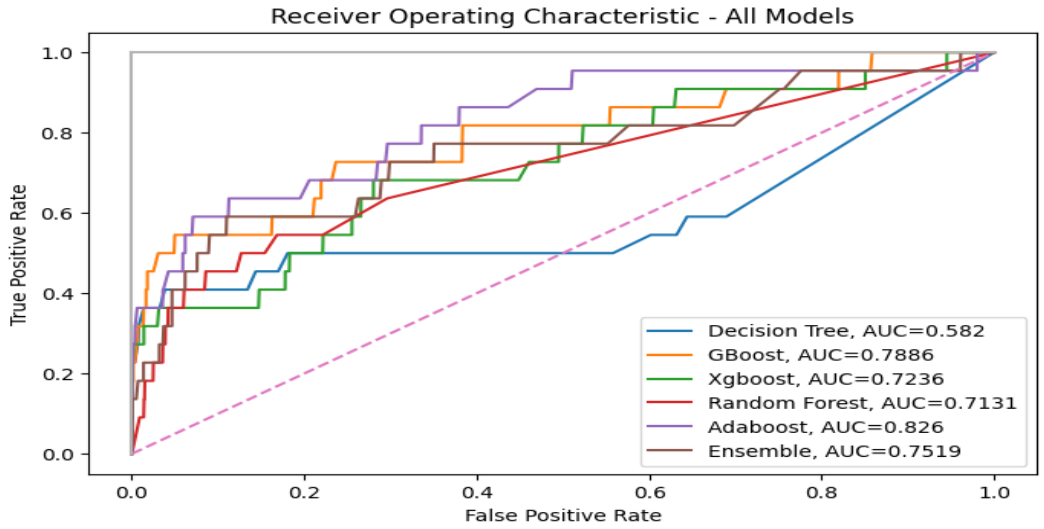
While a single tree-based ML classifier can identify both anomalous and non-anomalous transactions, stacked and voting classifiers have been implemented to mitigate issues related to errors and overfitting. As demonstrated by the experimental results in Table 4.8, ensemble classifiers exhibit superior performance compared to individual tree-based models, particularly for accuracy and FPR values.

Table 4.7 showcases the enhanced test accuracy of the three ensemble methods in comparison to single classifiers. The voting (hard) classifier attains the highest accuracy of 97%, which stands as the peak value across all single classifiers. Among the ensemble classifiers, the voting (soft) classifier secures the highest TPR or sensitivity value of 0.91 when utilizing under-sampled data from the XGBCLUS method. Concerning the FP rate, the voting (hard) classifier achieves the best value of 0.03, capitalizing on the XGBCLUS under-sampling technique.

With oversampled data, although the TPR values for the three ensemble classifiers exhibit a minor decline, the voting (hard) classifier attains the best FPR value of 0.03.



(a)



(b)

Figure 4.5: ROC-AUC curves for all models after (a) Under-Sampling with XGBCLUS, (b) Over-Sampling with ADASYN

Furthermore, ensemble classifiers prove effective in reducing the False Positive (FP) rate, thereby enhancing the correct identification of non-anomalous transactions. The voting (hard) classifier outperforms the other two ensemble methods by securing the highest accuracy of 0.97, alongside a commendable ROC-AUC score of 0.75.

Table 4.8: Evaluation metrics after Under-Sampling and Over-Sampling with test data

	Under Sampling				Over Sampling			
Classifier	Acc	TPR/Sensitivity	FPR	ROC-AUC	Acc	TPR/Sensitivity	FPR	ROC-AUC
DT	0.82	0.82	0.18	0.85	0.94	0.41	0.05	0.58
GB	0.81	0.86	0.19	**0.92**	0.94	0.55	0.06	0.79
RF	0.83	0.86	0.16	0.90	0.96	0.36	0.04	0.71
AdB	0.79	0.81	0.21	0.87	0.92	**0.59**	0.08	**0.83**
ES	**0.86**	**0.91**	**0.14**	0.91	**0.97**	0.41	**0.03**	0.75

4.6 Impact of sampling in detection with 1D CNN

We also trained our proposed CNN model on the unbalanced training set, leading to a bias towards the majority class. Consequently, the model displayed a high true negative rate but faced challenges in correctly identifying positive samples i.e. the Bitcoin anomalous transactions, resulting in zero true positives.

While keeping the test set independent, we balanced the training set using various sampling techniques, including SMOTE, ADASYN oversampling methods, as well as SMOTEENN and SMOTETOMEK combined sampling techniques. Subsequently, the CNN model was trained on each of the sampled datasets. For optimization, we employed the Adam stochastic gradient descent method with a learning rate of 0.0001. During training, binary cross-entropy was utilized to compute the loss. A batch size of 32 was employed, implying that 32 samples were used per gradient update. Given the approximately 320,000 samples in the training set, this resulted in (320,000 $\div$ 32) = 10,000 (approximately) steps needed per epoch to iterate over the entire training set. Initially, we used 100 epochs for training our proposed CNN model and monitored the accuracy and loss updates for both the training and validation sets. The number of epochs was determined using the early stopping technique to mitigate overfitting.

Table 4.9 provides a comparison of accuracy, TPR or Sensitivity, and ROC-AUC score for the Decision Tree (DT), Gradient Boosting (GBoost), Extreme Gradient Boosting (XGBoost), Random Forest (RF), Adaptive Boosting (AdaBoost), and 1d CNN classifiers. Notably, the TPR scores were zero for DT, XgBoost, RF, and 1D CNN, indicating that no fraudulent transactions were correctly identified; instead, all transactions were classified as non-fraudulent. Although GBoost and AdaBoost classifiers displayed a low TPR, their scores approached zero. This suggests that fraudulent

transactions remained undetected due to the models' bias towards the majority class i.e. non-fraudulent transactions.

Table 4.9: Comparison among the classifiers without balancing the data

Classifiers	Accuracy	TPR	AUC-Score
DT	0.99	0.0	0.55
GBoost	0.99	0.09	0.62
XgBoost	0.99	0.0	0.85
RF	0.99	0.0	0.72
AdaBoost	0.99	0.05	0.82
1D CNN	0.99	0.0	0.49

As we trained our proposed CNN model on the unbalanced training set, the model displayed a high true negative rate but faced challenges in correctly identifying positive samples i.e. the Bitcoin fraudulent transactions, resulting in zero true positive rate. While keeping the test set independent, we balanced the training set using various sampling techniques, including SMOTE, ADASYN oversampling methods, as well as SMOTEENN and SMOTETOMEK combined sampling techniques. Subsequently, the CNN model was trained on each of the sampled datasets.

Table 4.10 illustrates that the CNN model's accuracy remains consistent across different sampled datasets. Among these datasets, the CNN model with SMOTE oversampling exhibits the highest True Positive rate (0.36), emphasizing its effectiveness in identifying fraudulent Bitcoin transactions. In contrast, the CNN model with SMOTEEEN demonstrates the best FPR score (0.0008), indicating its capability to detect all non-fraudulent transactions i.e. True Negative rate is almost 100%. Remarkably, the ROC-AUC scores for the CNN model remain consistent across all sampled datasets. These results are achieved after certain epochs based on early stopping. Overall, the effectiveness of the proposed CNN model is evident in reducing the False Positive Rate (FPR) with both over and combined sampling methods, despite the True Positive Rate (TPR) not meeting the desired level.

While early stopping was implemented to mitigate overfitting issues, we additionally conducted a run of the CNN model for 100 epochs to assess potential improvements in the results. Table 4.11 presents the evaluation metrics, including accuracy, FPR,

Table 4.10: Comparison of evaluation metrics between Over and Combined sampling with 1D CNN

Model	Accuracy	TPR/Sensitivity	FPR	AUC-Score
CNN+SMOTE	0.98	0.36	0.02	0.76
CNN+ADASYN	0.99	0.27	0.005	0.83
CNN+SMOTEENN	0.99	0.27	0.0008	0.76
CNN+SMOTETOMEK	0.98	0.32	0.02	0.78

and TPR, for both cases. Notably, the TPR or Sensitivity has increased in all models, except for CNN+SMOTETOMEK, which remains constant when trained for 100 epochs. However, there is a significant increase in False Positive Rate when using 100 epochs, indicating a trade-off between FPR and TPR. The validation accuracy has generally decreased, with the exception of CNN+SMOTETOMEK, which exhibits a slight increase after 100 epochs. This behavior suggests that the model oscillates between focusing on non-fraudulent and fraudulent transactions, leading to increased False Positive (FP) values despite improvements in True Positive (TP) values. The models stopped training at different epochs: 48, 36, 41, and 31 for CNN+SMOTE, CNN+ADASYN, CNN+SMOTEENN, and CNN+SMOTETOMEK, respectively.

Table 4.11: Comparison of the results with and without callback

	Early Stopping			100 epochs		
Model	Accuracy	FPR	TPR	Accuracy	FPR	TPR
CNN+SMOTE	0.98	0.02	0.36	0.95	0.05	0.45
CNN+ADASYN	0.99	0.005	0.27	0.92	0.08	0.45
CNN+SMOTEENN	0.99	0.0008	0.27	0.97	0.03	0.41
CNN+SMOTETOMEK	0.98	0.02	0.32	0.99	0.01	0.32

4.7 Performance Analysis of the Proposed CNN model

A comprehensive comparative analysis was carried out to evaluate the performance of machine learning (ML) classifiers and the proposed 1D CNN model using over-sampled and combined-sampled data. The results, as shown in the table 4.12, demonstrate that our 1D CNN model consistently outperforms all tree-based classifiers, achieving an impressive 100% accuracy across all sampled data. In contrast, the tree-based classifiers

exhibit accuracies ranging from 0.92 to 0.96. However, the accompanying table 4.13 reveals that the ML classifiers consistently achieve higher True Positive Rates when compared to the 1D CNN model. Notably, among the ML models, Adaptive Boosting stands out with the highest TPR value of 0.59 when using SMOTEENN sampled data.

However, In table 4.14, the 1D CNN achieves the best False Positive Rate of 0.0008 with SMOTEENN sampled data, while the ML methods achieve FPR scores ranging from 0.04 to 0.07. Although the CNN model exhibits slightly lower TPR, its robust FPR score leads to the correct identification of most of the non-fraudulent transactions. In addition, Figure 4.6 depicts that our proposed 1D CNN model exhibits improved ROC-AUC scores using the ADASYN over-sampling technique, outperforming the scores achieved by the tree-based ML classifiers. This suggests that our proposed CNN model also performs better in terms of ROC-AUC scores.

Table 4.12: Comparison of accuracy between ML and DL models

Classifiers	SMOTE	ADASYN	SMOTEENN	SMOTETOMEK
DT	0.94	0.94	0.94	0.94
GBoost	0.93	0.94	0.93	0.93
XgBoost	0.96	0.96	0.96	0.96
RF	0.96	0.96	0.96	0.96
AdaBoost	0.92	0.92	0.92	0.92
Ensemble	0.96	0.97	0.96	0.97
CNN	**0.98**	**0.99**	**0.99**	**0.98**

In a comparative analysis with two prior studies of [62] and [72] that focused on blockchain anomaly detection using the Bitcoin dataset as ours, Table 4.15 illustrates a detailed juxtaposition. Notably, our proposed ensemble method and 1D CNN method showcase superior performance across all metrics. While the studies share the primary objective of identifying malicious transactions, our emphasis extends to the accurate detection of non-malicious transactions as well. In this regard, our study endeavors to mitigate false positive rates, resulting in an enhanced true positive rate. Shafiq et al. delved into various unsupervised Machine Learning approaches to identify anomalous transactions, ultimately finding that the ensemble method outperformed other

Table 4.13: Comparison of TPR/Sensitivity between ML and DL models

Classifiers	SMOTE	ADASYN	SMOTEENN	SMOTETOMEK
DT	0.41	0.41	0.36	0.41
GBoost	0.45	0.55	0.59	0.50
XgBoost	0.36	0.36	0.36	0.36
RF	0.23	0.32	0.36	0.23
AdaBoost	**0.59**	**0.59**	**0.55**	**0.59**
Ensemble	0.36	0.41	0.36	0.36
CNN	0.36	0.27	0.27	0.32

Table 4.14: Comparison of FPR between ML and DL models

Classifiers	SMOTE	ADASYN	SMOTEENN	SMOTETOMEK
DT	0.06	0.06	0.06	0.05
GBoost	0.07	0.06	0.07	0.07
XgBoost	0.05	0.04	0.04	0.04
RF	0.04	0.04	0.04	0.04
AdaBoost	0.08	0.08	0.08	0.08
Ensemble	0.03	0.03	0.04	0.03
CNN	**0.02**	**0.005**	**0.0008**	**0.02**

explored classifiers. Conversely, Sayadi et al. investigated the use of K-means clustering and OCSVM for classifying anomalous Bitcoin transactions. However, both studies encountered challenges, as they exhibited low accuracy and a high False Positive Rate. The proposed ensemble method exhibits substantial enhancements in both true positive and false positive values, leveraging preprocessed sampled data obtained through various preprocessing steps, including feature selections. Our ensemble method excels in accuracy and notably the FPR value. Notably, our proposed 1D CNN excels in accuracy and FPR values. A significant distinction among the studies is that our research incorporates explainability, along with the inclusion of decision rules, a component absent in their investigation.

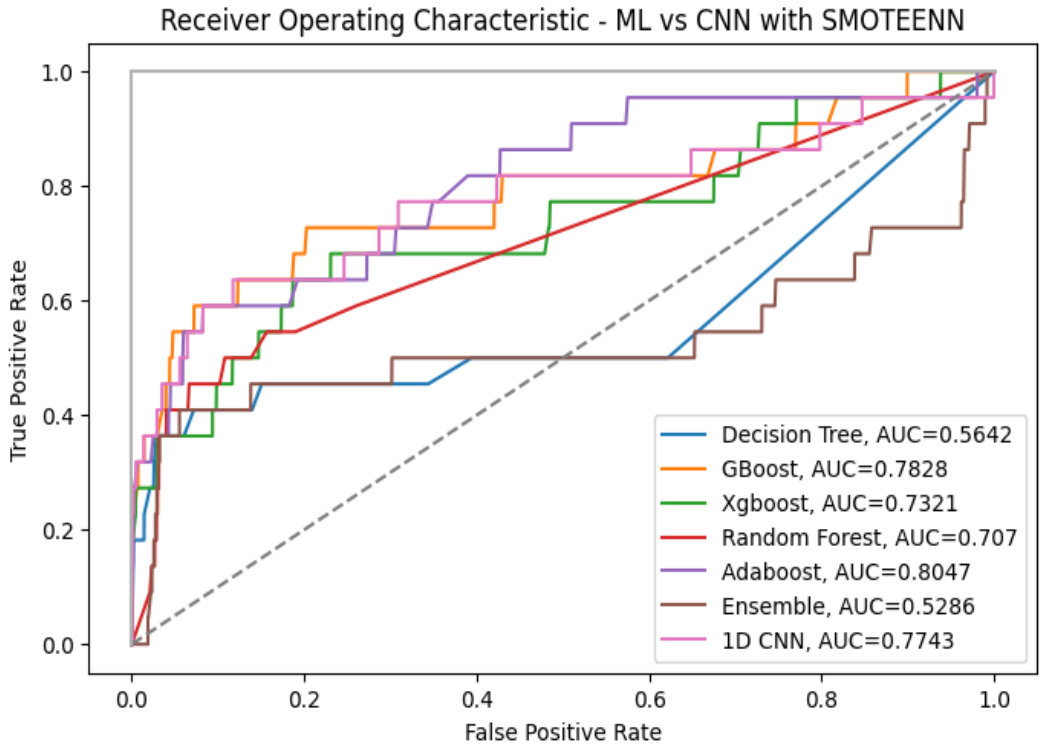


Figure 4.6: Comparison of ROC-AUC values with ML and DL models

Table 4.15: Comparison between Proposed and Existing Work for Bitcoin anomaly detection

References	Model	Accuracy	FPR	Explainability	Anomaly Rules
Shafiq et al.[62]	Ensemble Classifiers	0.94	0.05	No	No
Sayadi et al.[72]	OCSVM	0.90	0.09	No	No
Our Study	Ensemble Classifiers	**0.97**	**0.03**	**Yes**	**Yes**
Our Study	1D CNN	**0.98**	**0.0008**	**Yes**	-

4.8 Contribution of the features to anomaly detection with SHAP

SHAP is a potent technique employed to elucidate the predictions of machine learning models. It offers a means to assign the contribution of each feature to a particular prediction, enhancing our comprehension of why a model reached a specific decision.

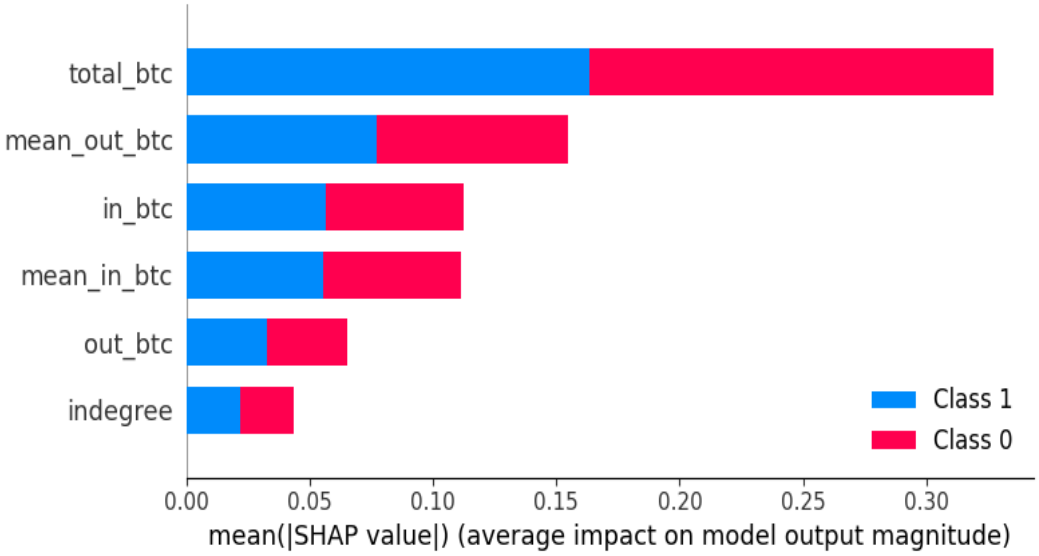


Figure 4.7: Hierarchy of the features contributing to classification

This study leverages the SHAP KernelExplainer method to calculate SHAP values, elucidating the behavior of both ML and DL models for individual instances. Utilizing the model-agnostic SHAP approach, we applied it to both Machine Learning (ML) and Deep Learning (DL) models. SHAP values serve the purpose of explaining how a specific feature's value influences the model's prediction compared to what the prediction would be when that feature adopts a baseline value. At first, it establishes a baseline prediction, typically by using the average prediction of the model for a set of instances. This serves as a reference point. For each feature permutation, SHAP predicts the model's output and calculates the difference between the baseline prediction and the current prediction. Then, the method averages these differences across all permutations to determine the SHAP value for the feature being evaluated.

Figure 4.7 provides a hierarchical summary of features, ordered by their contributions to the model's output. The magnitude of the SHAP values indicates the strength of the feature's influence. Features with larger absolute SHAP values have a more substantial impact on the anomaly prediction. Notably, the feature "total_btc" exerts the most substantial average impact on the model's classification, whereas "indegree" contributes the least. Furthermore, the second-highest mean SHAP value is attributed to "mean_out_btc," playing a significant role in anomaly detection. Conversely, "out_btc"

contributes the second least to the classification. "in_btc" and "mean_in_btc" exhibit nearly equivalent contributions to the classification of Bitcoin transactions.

Figure 4.8 provides a visual comparison of SHAP values, showcasing the contributions of features for four randomly selected anomalous and normal transactions, as predicted by both the CNN and ML models. In Figure 4.8, the baseline value for expected anomalous transactions is 0.27 for the CNN model and 0.50 for the ML models. Remarkably, the predicted scores for anomalous Bitcoin transactions are 0.89 with the CNN model and 0.86 with the ML models, which are notably higher than the baseline values. The SHAP value representations in Figures 4.8a and 4.8b illuminate the features contributing the most to increasing the score. These figures highlight that "total_btc" holds the highest importance for the CNN model and the second highest for ML models, followed by "in_btc" or "mean_out_btc," "out_btc". On the other hand, with the same baseline value, the predicted scores for non-anomalous Bitcoin transactions are 0.03 with the CNN model and 0.40 with the ML models, which are lower than the baseline values. The SHAP value representations in Figures 4.8c and 4.8d elucidate the features contributing the most to decreasing the score. These figures emphasize that "total_btc" holds the highest importance for the CNN model and the second highest for ML models, followed by "out_btc," or "indegree".

Although "mean_in_btc" and "in_btc" hold the highest positions for ML models in anomalous and non-anomalous cases, respectively, "total_btc" contributes the most in both cases. Comparing all four cases in Figure 4.8, it's evident that the "total_btc" values for anomalous instances are higher than those for normal instances, making them distinguishable to human observation. Hence, it can be concluded that SHAP values are effective in explaining both normal and anomalous Bitcoin transactions.

Comparing all four cases in Figure 4.8, it's evident that the feature values for anomalous instances are higher than those for normal instances, making them discernible to human observation. To facilitate a better human understanding, a comparison between actual anomalous and normal data instances is presented in Table 4.16. This table indicates that the features identified by SHAP values in Figure 10 exhibit significant disparities in actual transaction data. Hence, it can be concluded that SHAP values are effective in explaining both normal and anomalous Bitcoin transactions.

Table 4.16: Comparison of actual values of an anomalous and normal Bitcoin transaction

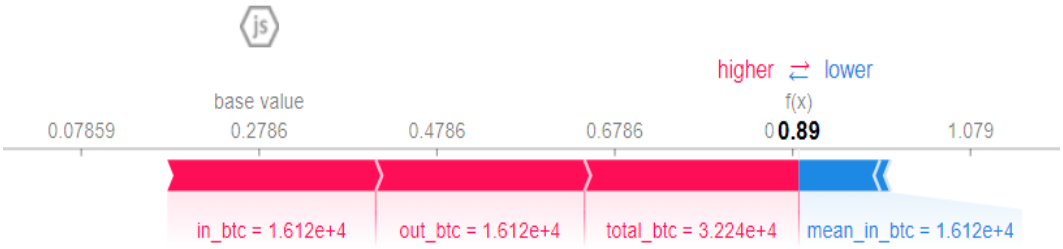
Feature Name	Anomalous	Normal
Indegree	7	2
in_btc	2902	15.96
out_btc	2902	15.96
total_btc	5804	31.92
mean_in_btc	414.6	7.98
mean_out_btc	1451	5.32

4.9 Anomaly Rule Generation and Interpretability Analysis

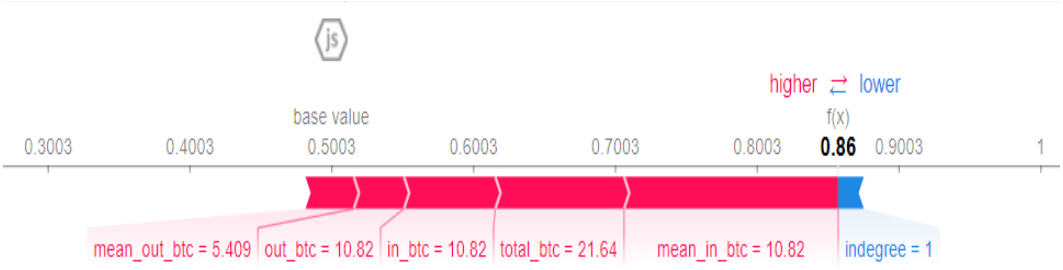
Table 4.17: Feature Importance values

Feature Name	Importances
total_btc	0.666268
mean_in_btc	0.124338
mean_out_btc	0.092236
in_btc	0.046185
out_btc	0.036751
Indegree	0.034222

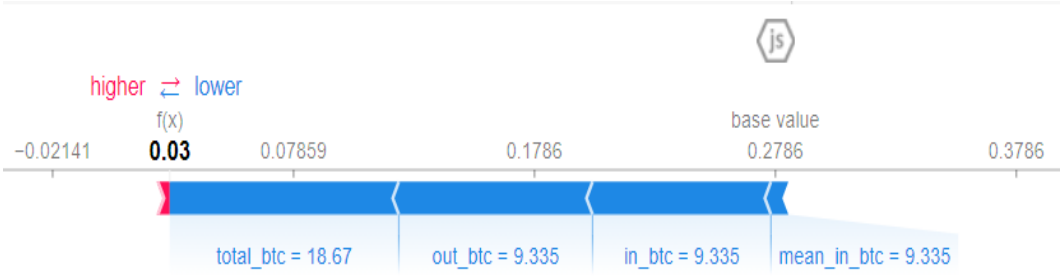
Interpreting anomaly rules using tree representations can help in understanding why certain instances are classified as anomalies. It provides a step-by-step breakdown of the decision process and highlights which features and thresholds played a crucial role in making the decision. In this study, having explored tree-based machine learning classifiers for detecting anomalous Bitcoin transactions, tree visualization offers a structured and interpretable approach to comprehending how the model arrives at decisions using input features. A visualization of the decision tree with max-depth 10 is shown in Figure 4.9. The top node of the tree is called the root node, and it represents the entire dataset. Each subsequent node represents a decision point based on a specific feature and threshold. Moving down the sub-trees, each node represents a feature and a



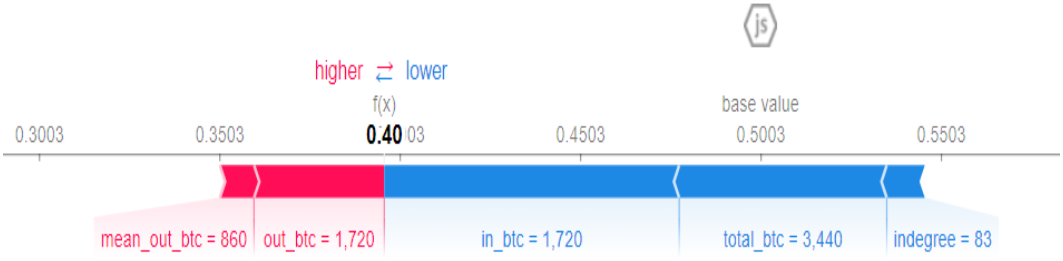
(a)



(b)



(c)



(d)

Figure 4.8: Features contributing to (a) detect an anomalous transaction with CNN model, (b) detect an anomalous transaction with ML model, (c) detect a non-anomalous transaction with CNN, (d) detect a non-anomalous transaction with ML model.

corresponding threshold. Instances are directed to different branches based on whether their feature values satisfy the given threshold. Each leaf node corresponds to a specific prediction class, in this case, anomalous or non-anomalous.

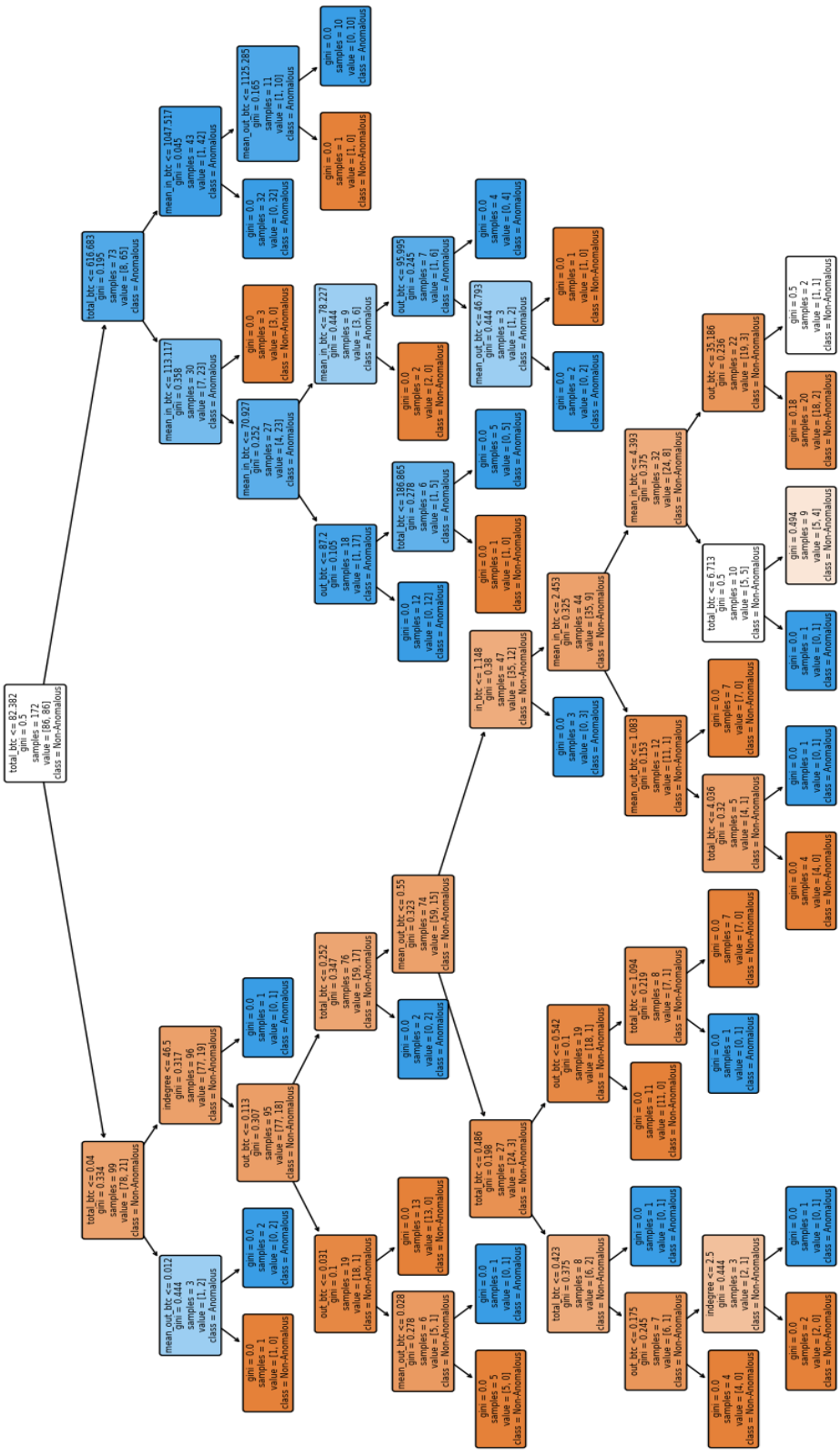


Figure 4.9: Visualization of DT of depth 10 for generating anomalous rules

Table 4.18: Significant Rules for being anomalous Bitcoin transaction

Anomaly rule	Class	Total Samples	Correctly Identified	Confidence(%)
1. If (total_btc is greater than 616.683 and mean_in_btc is less than or equal to 1047.517) then	Anomalous	32	32	100
2. If (total_btc is greater than 616.683 and mean_in_btc is greater than 1047.517, and mean_out_btc is greater than 1125.285) then	Anomalous	10	10	100
3. If (total_btc is greater than 82.38 and total_btc is less or equal to 616.683 and mean_in_btc is less or equal to 70.927, and out_btc is less or equal to 87.2) then	Anomalous	12	12	100
4. If (total_btc is greater than 82.38 and total_btc is less or equal to 616.683 and mean_in_btc is less or equal to 70.927) then	Anomalous	18	17	94
5. If (total_btc is greater than 616.683 and mean_in_btc is greater than 1047.517) then	Anomalous	11	10	91

In Table 4.17, the descending order of feature importance is presented, with numerical values representing the total normalized reduction in Gini Impurity achieved by splitting the respective feature throughout the tree. Unsurprisingly, 'total_btc,' the feature utilized to split at the root node, holds the highest importance. This implies that 'total_btc,' followed by 'mean_in_btc',' are the most crucial features in determining the anomaly status of a Bitcoin transaction.

By traversing the decision tree from the root node to a specific leaf node, we can extract the sequence of decisions (feature and threshold combinations) that lead to an anomaly prediction. These decisions essentially form the "anomaly rules" for that instance. In Table 4.18, we have a set of rules along with the confidence scores that lead to an anomalous node regarding the tree in Figure 4.9. Decision trees also provide a measure of feature importance. Features closer to the root of the tree have a more significant influence on the overall decision-making process. So total_btc is the most important feature that contributes the highest to deciding whether a transaction is anomalous or not. Moreover, both mean_in_btc and mean_out_btc have valuable effects on classifying anomalous transactions. However, in some cases, the indegree may also contribute to detecting anomalous transactions. The confidence score reflects the model's confidence or certainty in its prediction that the transaction is anomalous based on the specified conditions in the rules. Rules 1 to 3 give the highest confidence scores i.e. the highest probability for being anomalous. However, rules 4 to 5 have an average probability for the Bitcoin transactions to be anomalous. From Table 4.18, the more thorough the examination of decision rules, the greater the confidence in accurately identifying anomalous transactions. Although we've specifically crafted anomaly rules for our Bitcoin transaction data, this methodology holds promise for creating effective decision rules in diverse domains, particularly within the realm of Blockchain.

4.10 Summary

This chapter provides the details of the experimental outcomes and engages in a comprehensive discussion of the results. The subsequent chapter shows the summary of the study and concludes with some feature directions.

Chapter 5

Discussions, Conclusions, and Future Work

5.1 Discussions

Our main goal is to enhance the accuracy of fraudulent as well as non-fraudulent transactions detection. Detecting fraudulent transactions within highly imbalanced Blockchain transaction data is a complex task. The severe data imbalance can make them lean towards non-fraudulent transactions, resulting in low true positive rates, as shown in Table 4.9. Since the convolution neural network can learn features easily from large instances, we've explored the use of Deep Learning, particularly the 1D CNN model, for identifying fraudulent Bitcoin transactions. Nevertheless, both machine learning (ML) and deep learning (DL) techniques exhibit distinct impacts on the detection of fraudulent activity depending on the circumstances.

Tree-based machine learning models prove to be effective in identifying fraudulent transactions, particularly in scenarios with a relatively low number of data instances—both non-fraudulent and fraudulent transactions. Given the capability of ML models to discern transactions in datasets with limited instances, we introduced a novel undersampling method named XGBCLUS, comparing its performance against state-of-the-art methods. Furthermore, we extracted anomaly rules from these tree-based models, aiming for enhanced human understanding. Thus, we posit that ML models demonstrate efficacy in detecting anomalies or fraud in blockchain transactions when data instances are limited, balanced through under-sampling techniques, and the

interpretation of anomaly rules is deemed essential for human comprehension.

Conversely, deep learning models, particularly those based on convolutional neural networks (CNNs), prove to be effective in detecting fraud or anomalies in blockchain transactions, especially when dealing with a substantial volume of data instances. CNN-based models excel at learning features from large datasets, providing a robust framework for fraud detection. Additionally, in the context of data balancing, over-sampling techniques demonstrate effectiveness, as deep learning models can adeptly glean insights from extensive data. This study specifically explores the application of the 1D CNN deep learning method for fraud detection in blockchain transactions, employing various over-sampling methods for effective data balancing. While utilizing the same dataset for both studies, our future work aims to expand the scope by collecting additional blockchain transaction data to develop an even more robust real-time fraud detection system, as shown in Figure 1.1, for blockchain transactions.

Since Machine learning and deep learning models are often considered as "Black Boxes", to make these models more understandable, we turn to Explainable AI using SHAP, as discussed in section 4.8. By analyzing SHAP values across all dataset instances, we identify the most crucial feature, which, in this case, is "total_btc" for both ML and DL models. This feature's significance is clear to human observers. SHAP's analysis emphasizes the importance of "total_btc" in identifying fraudulent Bitcoin transactions. SHAP offers a clear and interpretable way to comprehend how each feature influences the fraudulent transaction detection process. This aids data analysts and domain experts in recognizing patterns, connections, and potential features in the data that the model uses for its predictions. In addition, Interpreting anomaly rules using tree representations can help in understanding why certain instances are classified as anomalies. It provides a step-by-step breakdown of the decision process and highlights which features and thresholds played a crucial role in making the decision. This transparency is especially valuable in domains where explainability is important, allowing stakeholders to validate the model's decisions and

The findings of our study bear significant implications for enhancing both blockchain security and anomaly detection. The efficacy of the proposed ensemble model in detecting anomalous transactions in Bitcoin signifies a promising approach to enhance the security of blockchain systems. Additionally, the incorporation of eXplainable Artificial Intelligence (XAI) techniques, including SHAP analysis, along with anomaly rules

for interpretability analysis, contributes transparency and clarity to the anomaly detection process of blockchain technologies. Implementing such explainability measures can contribute to robust anomaly detection systems and, consequently, fortify the overall security of blockchain technologies. Additionally, the exploration of various sampling techniques, including the proposed under-sampling and combined-sampling methods, contributes to the development of strategies for handling highly imbalanced datasets in the blockchain domain. This has broader implications for anomaly detection in other contexts where imbalanced data is a common challenge.

5.2 Conclusions

In this study, we have conducted an extensive comparative analysis aimed at detecting anomalies within Blockchain transaction data with both ML and DL techniques. While numerous studies have been conducted in this field, a prevailing limitation has been the absence of explanations for model predictions. To address this shortcoming, our study endeavors to combine eXplainable Artificial Intelligence (XAI) techniques and anomaly rules with tree-based ensemble classifiers using Bitcoin transactions. Notably, the Shapley Additive exPlanation (SHAP) method plays a pivotal role in quantifying the contribution of each feature toward predicting the model's output. Moreover, the anomaly rules help to interpret whether a Bitcoin transaction is anomalous or not. Consequently, one can readily identify which features play a pivotal role in anomaly detection.

To further enhance our methodology, we have developed an under-sampling algorithm termed XGBCLUS. This algorithm facilitates the balance between anomalous and non-anomalous transaction data, and its performance has been compared against other well-known under-sampling and oversampling techniques. Our findings unequivocally demonstrate that our proposed under-sampling method, XGBCLUS, has yielded improvements in TPR and ROC-AUC scores.

Subsequently, the results derived from various single tree-based classifiers are juxtaposed against those obtained from stacking and voting ensemble classifiers. Our proposed ensemble classifiers have exhibited superior performance in comparison to popular single ML classifiers. Furthermore, we have presented a 1D CNN model designed to identify anomalous Bitcoin transactions and assess its performance in compar-

ison to state-of-the-art Machine Learning algorithms. Our rigorous evaluation, pitting various tree-based classifiers against our 1D CNN model, resoundingly endorses the latter. It shines with the highest accuracy and best FPR values. In sum, this study underscores that combining Machine Learning or Deep learning methods with adept balancing techniques yields remarkable results in the realm of Blockchain Anomaly detection.

5.3 Future Work

In future research endeavors, there is potential to delve into the efficacy of our proposed under-sampling techniques, alongside over-sampling methods and the proposed CNN techniques, for anomaly detection in different Blockchain domains such as Ethereum transactions [73], credit card fraud detection [74], and money laundering [75]. The exploration of a data-driven approach [76] could lead to the development of real-time blockchain anomaly detection systems. In future research, we plan to gather additional data related to blockchain and perform further experimental analyses. Additionally, combining machine learning and deep learning algorithms, such as CNN+SVM, could be explored to identify the best model for digital transaction anomaly detection. This could significantly improve the accuracy and effectiveness of fraud detection methods.

Bibliography

[1] M. Nofer, P. Gomber, O. Hinz, and D. Schiereck, "Blockchain," *Business & Information Systems Engineering*, vol. 59, no. 3, pp. 183–187, 2017. 1

[2] S. Nakamoto, "Bitcoin: A peer-to-peer electronic cash system," *Decentralized Business Review*, p. 21260, 2008. 1

[3] D. Yaga, P. Mell, N. Roby, and K. Scarfone, "Blockchain technology overview," *arXiv preprint arXiv:1906.11078*, 2019. 1

[4] A. A. Monrat, O. Schelén, and K. Andersson, "A survey of blockchain from the perspectives of applications, challenges, and opportunities," *IEEE Access*, vol. 7, pp. 117 134–117 151, 2019. 1

[5] M. Saad, V. Cook, L. Nguyen, M. T. Thai, and A. Mohaisen, "Partitioning attacks on bitcoin: Colliding space, time, and logic," in *2019 IEEE 39th international conference on distributed computing systems (ICDCS)*. IEEE, 2019, pp. 1175–1187. 1

[6] M. U. Hassan, M. H. Rehmani, and J. Chen, "Anomaly detection in blockchain networks: A comprehensive survey," *IEEE Communications Surveys & Tutorials*, 2022. 1

[7] M. Signorini, M. Pontecorvi, W. Kanoun, and R. Di Pietro, "Advise: anomaly detection tool for blockchain systems," in *2018 IEEE World Congress on Services (SERVICES)*. IEEE, 2018, pp. 65–66. 2

[8] V. Ganganwar, "An overview of classification algorithms for imbalanced datasets," *International Journal of Emerging Technology and Advanced Engineering*, vol. 2, no. 4, pp. 42–47, 2012. 2, 22

[9] M. Saripuddin, A. Suliman, S. Syarmila Sameon, and B. N. Jorgensen, "Random undersampling on imbalance time series data for anomaly detection," in *Proceedings of the 2021 4th International Conference on Machine Learning and Machine Intelligence*, 2021, pp. 151–156. 3, 22

[10] R. A. Alsowail, "An insider threat detection model using one-hot encoding and near-miss under-sampling techniques," in *Proceedings of International Joint Conference on Advances in Computational Intelligence: IJCACI 2021.* Springer, 2022, pp. 183–196. 3

[11] I. H. Sarker, A. Kayes, S. Badsha, H. Alqahtani, P. Watters, and A. Ng, "Cybersecurity data science: an overview from machine learning perspective," *Journal of Big data*, vol. 7, pp. 1–29, 2020. 3, 22

[12] M. Rashid, J. Kamruzzaman, T. Imam, S. Wibowo, and S. Gordon, "A tree-based stacking ensemble technique with feature selection for network intrusion detection," *Applied Intelligence*, vol. 52, no. 9, pp. 9768–9781, 2022. 3, 22

[13] Y. Zhou, G. Cheng, S. Jiang, and M. Dai, "Building an efficient intrusion detection system based on feature selection and ensemble classifier," *Computer networks*, vol. 174, p. 107247, 2020. 3, 22

[14] G. Pang, C. Shen, L. Cao, and A. V. D. Hengel, "Deep learning for anomaly detection: A review," *ACM computing surveys (CSUR)*, vol. 54, no. 2, pp. 1–38, 2021. 3

[15] R. Chalapathy and S. Chawla, "Deep learning for anomaly detection: A survey," *arXiv preprint arXiv:1901.03407*, 2019. 3

[16] R. Wang, K. Nie, T. Wang, Y. Yang, and B. Long, "Deep learning for anomaly detection," in *Proceedings of the 13th international conference on web search and data mining*, 2020, pp. 894–896. 3

[17] M. Kang, "Machine learning: Anomaly detection," *Prognostics and health management of electronics: fundamentals, machine learning, and the internet of things*, pp. 131–162, 2018. 3

[18] I. R. Ward, L. Wang, J. Lu, M. Bennamoun, G. Dwivedi, and F. M. Sanfilippo, "Explainable artificial intelligence for pharmacovigilance: What features are important when predicting adverse outcomes?" *Computer Methods and Programs in Biomedicine*, vol. 212, p. 106415, 2021. 3, 17

[19] M. T. Ribeiro, S. Singh, and C. Guestrin, "" why should i trust you?" explaining the predictions of any classifier," in *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, 2016, pp. 1135–1144. 3, 17

[20] S. M. Lundberg and S.-I. Lee, "A unified approach to interpreting model predictions," *Advances in neural information processing systems*, vol. 30, 2017. 3, 17

[21] L. Schilling and H. Uhlig, "Some simple bitcoin economics," *Journal of Monetary Economics*, vol. 106, pp. 16–26, 2019. 6

[22] R. Böhme, N. Christin, B. Edelman, and T. Moore, "Bitcoin: Economics, technology, and governance," *Journal of economic Perspectives*, vol. 29, no. 2, pp. 213–238, 2015. 7

[23] E. Zaghloul, T. Li, M. W. Mutka, and J. Ren, "Bitcoin and blockchain: Security and privacy," *IEEE Internet of Things Journal*, vol. 7, no. 10, pp. 10 288–10 313, 2020. 7

[24] C. Natoli and V. Gramoli, "The blockchain anomaly," in *2016 IEEE 15th international symposium on network computing and applications (NCA)*. IEEE, 2016, pp. 310–317. 8

[25] F. Reid and M. Harrigan, *An analysis of anonymity in the bitcoin system*. Springer, 2013. 8, 24

[26] "Bitcoin loss," https://bitcointalk.org/index.php?topic=782.5, accessed: 2024-03-03. 8

[27] "Hackers bring down the world's then-largest exchange," https://arstechnica.com/tech-policy/2017/12/a-brief-history-of-bitcoin-hacks-and-frauds/, accessed: 2024-03-03. 9

[28] "Bitstamp exchange is hacked," https://arstechnica.com/tech-policy/2017/12/a-brief-history-of-bitcoin-hacks-and-frauds/, accessed: 2024-03-03. 9

[29] "Another exchange loses 120,000 bitcoins to hackers," https://arstechnica.com/tech-policy/2017/12/a-brief-history-of-bitcoin-hacks-and-frauds/, accessed: 2024-03-03. 9

[30] N. Christin, "Traveling the silk road: A measurement analysis of a large anonymous online marketplace," in *Proceedings of the 22nd international conference on World Wide Web*, 2013, pp. 213–224. 9

[31] Y.-L. Chen, Y. T. Chang, and J. J. Yang, "Cryptocurrency hacking incidents and the price dynamics of bitcoin spot and futures," *Finance Research Letters*, p. 103955, 2023. 9

[32] Priyanka and D. Kumar, "Decision tree classifier: a detailed survey," *International Journal of Information and Decision Sciences*, vol. 12, no. 3, pp. 246–269, 2020. 11

[33] C. Bentéjac, A. Csörgő, and G. Martínez-Muñoz, "A comparative analysis of gradient boosting algorithms," *Artificial Intelligence Review*, vol. 54, pp. 1937–1967, 2021. 11

[34] T. Chen, T. He, M. Benesty, V. Khotilovich, Y. Tang, H. Cho, K. Chen, R. Mitchell, I. Cano, T. Zhou *et al.*, "Xgboost: extreme gradient boosting," *R package version 0.4-2*, vol. 1, no. 4, pp. 1–4, 2015. 12

[35] P. Sornsuwit and S. Jaiyen, "A new hybrid machine learning for cybersecurity threat detection based on adaptive boosting," *Applied Artificial Intelligence*, vol. 33, no. 5, pp. 462–482, 2019. 12

[36] N. M. Abdulkareem and A. M. Abdulazeez, "Machine learning classification based on radom forest algorithm: A review," *International journal of science and business*, vol. 5, no. 2, pp. 128–142, 2021. 13

[37] Z. Zheng, H.-N. Dai, and J. Wu, "Blockchain intelligence: When blockchain meets artificial intelligence," *arXiv preprint arXiv:1912.06485*, 2019. 18

[38] Y. Li, Y. Cai, H. Tian, G. Xue, and Z. Zheng, "Identifying illicit addresses in bitcoin network," in *Blockchain and Trustworthy Systems: Second International Conference, BlockSys 2020, Dali, China, August 6–7, 2020, Revised Selected Papers 2*. Springer, 2020, pp. 99–111. 18, 21

[39] H. S. Yin and R. Vatrapu, "A first estimation of the proportion of cybercriminal entities in the bitcoin ecosystem using supervised machine learning," in *2017 IEEE international conference on big data (Big Data)*. IEEE, 2017, pp. 3690–3699. 18

[40] A. Singh, "Anomaly detection in the ethereum network," *A thesis for the degree of Master of Technology/Indian Institute of Technology Kanpur*, 2019. 18

[41] B. Chen, F. Wei, and C. Gu, "Bitcoin theft detection based on supervised machine learning algorithms," *Security and Communication Networks*, vol. 2021, 2021. 19

[42] J. Lorenz, M. I. Silva, D. Aparício, J. T. Ascensão, and P. Bizarro, "Machine learning methods to detect money laundering in the bitcoin blockchain in the presence of label scarcity," in *Proceedings of the First ACM International Conference on AI in Finance*, 2020, pp. 1–8. 19

[43] I. Alarab, S. Prakoonwit, and M. I. Nacer, "Comparative analysis using supervised learning methods for anti-money laundering in bitcoin," in *Proceedings of the 2020 5th international conference on machine learning technologies*, 2020, pp. 11–17. 19

[44] K. Martin, M. Rahouti, M. Ayyash, and I. Alsmadi, "Anomaly detection in blockchain using network representation and machine learning," *Security and Privacy*, vol. 5, no. 2, p. e192, 2022. 19

[45] I. Alarab and S. Prakoonwit, "Effect of data resampling on feature importance in imbalanced blockchain data: Comparison studies of resampling techniques," *Data Science and Management*, vol. 5, no. 2, pp. 66–76, 2022. 19

[46] S. Taneja, B. Suri, and C. Kothari, "Application of balancing techniques with ensemble approach for credit card fraud detection," in *2019 International Conference on Computing, Power and Communication Technologies (GUCON)*. IEEE, 2019, pp. 753–758. 19

[47] V. Patel, L. Pan, and S. Rajasegarar, "Graph deep learning based anomaly detection in ethereum blockchain network," in *International conference on network and system security*. Springer, 2020, pp. 132–148. 20

[48] K. Kolesnikova, O. Mezentseva, and T. Mukatayev, "Analysis of bitcoin transactions to detect illegal transactions using convolutional neural networks," in *2021 IEEE International Conference on Smart Information Systems and Technologies (SIST)*. IEEE, 2021, pp. 1–6. 20

[49] K. Yıldız, S. Dedebek, F. Y. Okay, and M. U. Şimşek, "Anomaly detection in financial data using deep learning: A comparative analysis," in *2022 Innovations in Intelligent Systems and Applications Conference (ASYU)*. IEEE, 2022, pp. 1–6. 20

[50] P. Nerurkar, "Illegal activity detection on bitcoin transaction using deep learning," *Soft Computing*, vol. 27, no. 9, pp. 5503–5520, 2023. 20

[51] M. K. Hooshmand and D. Hosahalli, "Network anomaly detection using deep learning techniques," *CAAI Transactions on Intelligence Technology*, vol. 7, no. 2, pp. 228–243, 2022. 21

[52] F. Scicchitano, A. Liguori, M. Guarascio, E. Ritacco, and G. Manco, "A deep learning approach for detecting security attacks on blockchain." in *ITASEC*, 2020, pp. 212–222. 21

[53] J. Hirshman, Y. Huang, and S. Macke, "Unsupervised approaches to detecting anomalous behavior in the bitcoin transaction network," *Technical report, Stanford University*, 2013. 21

[54] B. Prasetiyo, M. Muslim, N. Baroroh *et al.*, "Evaluation performance recall and f2 score of credit card fraud detection unbalanced dataset using smote oversampling technique," in *Journal of Physics: Conference Series*, vol. 1918, no. 4. IOP Publishing, 2021, p. 042002. 21

[55] W. Yang, Y. Zhang, K. Ye, L. Li, and C.-Z. Xu, "Ffd: A federated learning based method for credit card fraud detection," in *Big Data–BigData 2019: 8th International Congress, Held as Part of the Services Conference Federation, SCF*

2019, San Diego, CA, USA, June 25–30, 2019, Proceedings 8. Springer, 2019, pp. 18–32. 21

[56] F. Itoo and S. Singh, "Comparison and analysis of logistic regression, naïve bayes and knn machine learning algorithms for credit card fraud detection," *International Journal of Information Technology*, vol. 13, pp. 1503–1511, 2021. 21

[57] S. Xuan, G. Liu, Z. Li, L. Zheng, S. Wang, and C. Jiang, "Random forest for credit card fraud detection," in *2018 IEEE 15th international conference on networking, sensing and control (ICNSC)*. IEEE, 2018, pp. 1–6. 21

[58] F. Ahmed, M. Hasan, M. S. Hossain, and K. Andersson, "Comparative performance of tree based machine learning classifiers in product backorder prediction," in *International Conference on Intelligent Computing & Optimization*. Springer, 2022, pp. 572–584. 21

[59] A. T. Assy, Y. Mostafa, A. Abd El-khaleq, and M. Mashaly, "Anomaly-based intrusion detection system using one-dimensional convolutional neural network," *Procedia Computer Science*, vol. 220, pp. 78–85, 2023. 22

[60] E. U. H. Qazi, A. Almorjan, and T. Zia, "A one-dimensional convolutional neural network (1d-cnn) based deep learning system for network intrusion detection," *Applied Sciences*, vol. 12, no. 16, p. 7986, 2022. 22

[61] I. Mitiche, A. Nesbitt, S. Conner, P. Boreham, and G. Morison, "1d-cnn based real-time fault detection system for power asset diagnostics," *IET Generation, Transmission & Distribution*, vol. 14, no. 24, pp. 5766–5773, 2020. 22

[62] O. Shafiq, "Anomaly detection in blockchain," Master's thesis, 2019. 24, 53, 55

[63] N. Rout, D. Mishra, and M. K. Mallick, "Handling imbalanced data: a survey," in *International Proceedings on Advances in Soft Computing, Intelligent Systems and Applications: ASISA 2016*. Springer, 2018, pp. 431–443. 28

[64] P. Jeatrakul, K. W. Wong, and C. C. Fung, "Classification of imbalanced data by combining the complementary neural network and smote algorithm," in *Neural Information Processing. Models and Applications: 17th International Conference,*

ICONIP 2010, Sydney, Australia, November 22-25, 2010, Proceedings, Part II 17. Springer, 2010, pp. 152–159. 31

[65] C. Lu, S. Lin, X. Liu, and H. Shi, "Telecom fraud identification based on adasyn and random forest," in *2020 5th International Conference on Computer and Communication Systems (ICCCS)*. IEEE, 2020, pp. 447–452. 31

[66] M. Muntasir Nishat, F. Faisal, I. Jahan Ratul, A. Al-Monsur, A. M. Ar-Rafi, S. M. Nasrullah, M. T. Reza, and M. R. H. Khan, "A comprehensive investigation of the performances of different machine learning classifiers with smote-enn oversampling technique and hyperparameter optimization for imbalanced heart failure dataset," *Scientific Programming*, vol. 2022, pp. 1–17, 2022. 32

[67] S. Sams Aafiya Banu, B. Gopika, E. Esakki Rajan, M. Ramkumar, M. Mahalakshmi, and G. Emil Selvan, "Smote variants for data balancing in intrusion detection system using machine learning," in *International Conference on Machine Intelligence and Signal Processing*. Springer, 2022, pp. 317–330. 32

[68] S. Rajagopal, P. P. Kundapur, and K. S. Hareesha, "A stacking ensemble for network intrusion detection using heterogeneous datasets," *Security and Communication Networks*, vol. 2020, pp. 1–9, 2020. 33

[69] J. E. King, "Binary logistic regression," *Best practices in quantitative methods*, pp. 358–384, 2008. 33

[70] X.-Y. Liu, J. Wu, and Z.-H. Zhou, "Exploratory undersampling for class-imbalance learning," *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 39, no. 2, pp. 539–550, 2008. 46

[71] A. Gosain and S. Sardana, "Handling class imbalance problem using oversampling techniques: A review," in *2017 international conference on advances in computing, communications and informatics (ICACCI)*. IEEE, 2017, pp. 79–85. 46

[72] S. Sayadi, S. B. Rejeb, and Z. Choukair, "Anomaly detection model over blockchain electronic transactions," in *2019 15th international wireless communications & mobile computing conference (IWCMC)*. IEEE, 2019, pp. 895–900. 53, 55

[73] S. Tikhomirov, "Ethereum: state of knowledge and research perspectives," in *Foundations and Practice of Security: 10th International Symposium, FPS 2017, Nancy, France, October 23-25, 2017, Revised Selected Papers 10*. Springer, 2018, pp. 206–221. 67

[74] V. N. Dornadula and S. Geetha, "Credit card fraud detection using machine learning algorithms," *Procedia computer science*, vol. 165, pp. 631–641, 2019. 67

[75] Z. Chen, L. D. Van Khoa, E. N. Teoh, A. Nazir, E. K. Karuppiah, and K. S. Lam, "Machine learning techniques for anti-money laundering (aml) solutions in suspicious transaction detection: a review," *Knowledge and Information Systems*, vol. 57, pp. 245–285, 2018. 67

[76] I. H. Sarker, "Data science and analytics: an overview from data-driven smart computing, decision-making and applications perspective," *SN Computer Science*, vol. 2, no. 5, p. 377, 2021. 67