University of Reading

Department of Computer Science

Detecting Fraud on Ethereum: A Machine Learning and Blockchain Analytics Approach

Anand Patel

Supervisor: Atta Badii

A report submitted in partial fulfilment of the requirements of
the University of Reading for the degree of
Bachelor of Science in *Computer Science*

April 23, 2024

Declaration

I, Anand Patel, of the Department of Computer Science, University of Reading, confirm that this is my own work and figures, tables, equations, code snippets, artworks, and illustrations in this report are original and have not been taken from any other person's work, except where the works of others have been explicitly acknowledged, quoted, and referenced. I understand that if failing to do so will be considered a case of plagiarism. Plagiarism is a form of academic misconduct and will be penalised accordingly.

I give consent to a copy of my report being shared with future students as an exemplar.

I give consent for my work to be made available more widely to members of UoR and public with interest in teaching, learning and research.

Anand Patel
April 23, 2024

Abstract

Blockchain technology, exemplified by Ethereum—the world's second-largest blockchain by market capitalisation—has transformed digital transactions with its promise of enhanced transparency and security. However, as the technology spreads, it also becomes an abundant ground for increasingly sophisticated fraudulent activities, highlighting the urgent need for effective fraud detection mechanisms. This dissertation contributes to the discourse on blockchain security by offering actionable insights for law enforcement and stakeholders in the Ethereum network through a comprehensive analysis of various machine learning models. It evaluates the effectiveness of traditional models such as logistic regression and decision trees, alongside more advanced approaches including neural networks and ensemble methods. The study employs key performance metrics like accuracy, precision, recall, and the area under the receiver operating characteristic curve to assess the models. Notably, the tuned XGBoost model emerges as the most effective, achieving an impressive accuracy of 0.98. This research not only advances the application of machine learning for fraud detection but also enhances the reliability and integrity of blockchain transactions, thereby supporting the continued evolution of this critical technology. The findings provide a thorough evaluation of the models' performance and offer practical applications for their use in real-world scenarios, marking a significant step forward in the battle against blockchain fraud.

Keywords: Blockchain Technology, Ethereum, Fraud Detection, Machine Learning, XGBoost

Report's total word count:

$$\approx 18500$$

Acknowledgements

I extend my deepest gratitude to my sisters and father, whose unwavering support and encouragement have been instrumental throughout my university journey. Their belief in my capabilities and constant guidance have been pillars of strength for me. I also wish to express my sincere thanks to my supervisor, Atta Badii, for his invaluable guidance and insights. This work is dedicated to my mother, whose memory inspires me daily to strive for excellence. Her enduring love continues to guide me in all my endeavors.

Contents

List of Figures

List of Tables

List of Abbreviations

DApps	Decentralised Applications
DeFi	Decentralised Finance
EVM	Ethereum Virtual Machine
EOA	Externally Owned Accounts
CA	Contract Accounts
ERC-20	Ethereum Request for Comment 20
ICOs	Initial Coin Offerings
ML	Machine Learning
AI	Artificial Intelligence
API	Application Programming Interface
ROC	Receiver Operating Characteristic
AUC	Area Under the Curve
SMOTE	Synthetic Minority Over-sampling Technique
PCA	Principal Component Analysis
t-SNE	t-Distributed Stochastic Neighbor Embedding
RF	Random Forest
XGB	Extreme Gradient Boosting
LR	Logistic Regression
CNN	Convolutional Neural Network
GNN	Graph Neural Network
LGBM	Light Gradient Boosting Machine
TP	True Positives
TN	True Negatives
FP	False Positives
FN	False Negatives
HTTP	Hypertext Transfer Protocol
SSL	Secure Sockets Layer
TCP/IP	Transmission Control Protocol/Internet Protocol

Chapter 1

Introduction

1.1 Background

Blockchain technology represents a paradigm shift in digital transactions and decentralised data management. This technology underpins cryptocurrencies and many other distributed systems and applications like the two largest Blockchains by market capitalisation, Bitcoin and Ethereum. Bitcoin, created by anonymous developer Satoshi Nakamoto in 2009, authored a white paper titled *Bitcoin: A Peer-to-Peer Electronic Cash System* and was posted to a cryptography mailing list. Nakamoto implemented the Bitcoin software as open-source code and released it in January 2009. Bitcoin introduced the world to a decentralised currency that operates independently of a central authority, using blockchain to maintain a secure public ledger of transactions.

The success and limitations of Bitcoin paved the way for the development of Ethereum, launched in 2015 by Vitalik Buterin and co. Ethereum expanded on Bitcoin's foundational technology by introducing programmable smart contracts. These are self-executing contracts dictated by the lines of code that describe them, facilitating not only transactions but also complex and automated agreements such as decentralised applications (DApps). Ethereum distinguished itself by offering a platform where developers can create DApps beyond sending tokens between participants and evolved into a breeding ground for Decentralised Finance (DeFi) applications which allow the borrowing, lending and exchanging of assets on the blockchain.

1.2 Problem Statement

Despite the robust security features to prevent the corruptibility of the ledger inherent in blockchains, they are not immune to fraudulent activities. The complexity of Ethereum with its smart contract functionality presents unique vulnerabilities and challenges. Fraudulent activities, ranging from Ponzi schemes to smart contract exploits can result in significant financial losses but also erode trust in the ecosystem. The decentralised and pseudonymous nature of Ethereum complicates the task of fraud detection, rendering traditional centralised monitoring mechanisms less effective.

1.3 Objectives

This study aims to leverage machine learning and visualisation techniques to enhance the detection of fraudulent activity on the Ethereum blockchain. The objectives include:

- Comprehensive Analysis of the Ethereum Platform:
  - Examine the Ethereum architecture, including its consensus mechanism, smart contract functionality, and the Ethereum Virtual Machine (EVM).
  - Catalogue and analyse rampant types of fraud within the Ethereum ecosystem, such as phishing attacks, Ponzi schemes, smart contract vulnerabilities, and other malicious exploits. This analysis will involve reviewing case studies, existing literature, and incident reports to develop an understanding of Ethereum fraud.
  - Identify key indicators or features within Ethereum transaction data that may signal fraudulent activity, considering factors like transaction frequency, value, gas prices, and patterns of interaction between accounts.
- Evaluation of Machine Learning Models for Fraud Detection:
  - Implement and assess various machine learning models to determine their effectiveness in detecting fraudulent activities on Ethereum. This will include traditional models (like logistic regression and Random Forest) for their interpretability and advanced models (like neural networks and ensemble methods) for their predictive power.
  - Explore feature engineering techniques to enhance model performance, focusing on extracting meaningful insights from the chosen dataset
  - Conduct a comparative analysis of the models based on performance metrics such as accuracy, precision, recall, F1 score, and the area under the ROC curve. This analysis will help in identifying the most suitable model or combination of models for detecting fraud on Ethereum.
- Development of a Machine Learning-Based Fraud Detection Framework:
  - Design and develop a comprehensive machine-learning framework tailored to Ethereum's unique characteristics. This framework should integrate the best-performing models and techniques identified in the previous objective. Additionally, incorporate a transaction visualiser to aid in the analysis and interpretation of complex transaction patterns and interactions, enhancing the understanding of potential fraudulent activities.
  - Ensure that the framework is adaptable, allowing for the incorporation of new data, the updating of models, and the adjustment of parameters in response to the evolving landscape of blockchain fraud. The transaction visualiser should be capable of scaling and updating its visualisation features to accommodate new types of transactions and fraud tactics as they emerge.
  - Validate the framework using a dataset of Ethereum transactions, which should include a mix of legitimate and fraudulent activities. This validation process should test the framework's effectiveness in detecting known types of fraud and its ability to flag new and emerging fraudulent patterns. The transaction visualiser will serve as a critical tool in this process, providing visual insights that enhance the detection capabilities of the framework.

- Contribution to the Field and Practical Application:
  - Synthesise the findings from the research to contribute to the academic field of blockchain security, offering new insights and methodologies for fraud detection on Ethereum. This synthesis will involve the integration of theoretical perspectives with empirical data derived from the machine-learning framework to propose novel solutions that can address existing vulnerabilities in Ethereum fraud detection.
  - Develop practical guidelines and recommendations for implementing the machine learning-based fraud detection framework in real-world settings, such as cryptocurrency exchanges, wallet services, and individual users. These guidelines will include step-by-step procedures for deploying the framework, maintaining system integrity, and ensuring continuous improvement based on operational feedback and emerging security threats.
  - Evaluate the broader implications of the research findings, discussing how enhanced fraud detection can impact user trust, market stability, and the future development of blockchain technologies. This evaluation will explore the potential for widespread adoption of the proposed methodologies and their effect on the perception and value of Ethereum in the global financial ecosystem.

Chapter 2

Literature Review

2.1 Literature Review

2.1.1 Decentralised Finance and Blockchain Technology

The theoretical foundations for decentralised financial systems were established long before their practical application was realised in the late 2000s. In 2008, the emergence of blockchain technology and Bitcoin marked a significant milestone in the history of digital currencies, introducing a transformative technology to the world Vujičić et al. (2018), Huh et al. (2017). Currently, the digital economy is largely dependent on centralised authorities, requiring transactions to be processed through established intermediaries. This dependency requires that end users trust these entities to execute transactions securely and privately, applicable across both financial and non-financial domains. Blockchain technology revolutionises this model by enabling a decentralised consensus mechanism within the digital realm Dannen (2017). It supports a public ledger—a comprehensive, append-only record of all digital transactions accessible to all participants. This ledger is maintained across a peer-to-peer network, ensuring that data once entered is immutable, thanks to the integration of cryptographic hash functions and consensus algorithms Atzei et al. (2016), Aste et al. (2017). This setup prevents any alteration of the ledger unless control of over 50% of the network is obtained by a single entity or group, a scenario known as a 51% attack. The integrity of the ledger provides a reconciled and historical account of all transactions, establishing a base level of trust among users, enhanced by the security of public/private key encryption without third-party intervention. Moreover, blockchain effectively addresses the issue of double spending, where the same digital tokens could otherwise be spent more than once. This is achieved by automating transaction verification and record-keeping, preventing the duplication of transaction entries in the ledger. The advantages of adopting blockchain are multifaceted: it enables the deployment of algorithms to automate and streamline business processes through smart contracts, facilitates secure and trustworthy digital transactions, and maintains an immutable database of all records and assets. These features collectively enhance the reliability of digital transactions, paving the way for a more decentralised and transparent financial ecosystem.

2.1.2 Ethereum and Its Evolution in Blockchain Technology

Ethereum, developed by Vitalik Buterin and launched on July 30, 2015, represents a significant advancement in blockchain technology, offering increased functionality and broader applications beyond Bitcoin's initial scope. While Bitcoin primarily serves as a peer-to-peer system for electronic payments and ownership tracking of its own currency, Ethereum introduces a more versatile platform by incorporating a Turing-complete programming language Dannen (2017).

This capability addresses several limitations of Bitcoin by enabling the execution of complex contracts and autonomous, decentralised applications.

Ether, the native cryptocurrency of Ethereum, functions as the primary medium for transaction fee payments within the network. Denominated in "Wei," the smallest unit of Ether, the system allows for precise and granular transactions, facilitating a wide range of computation and state management operations that extend the basic architecture of blockchain technology.

Ethereum can be conceptualised as a sophisticated layer atop the foundational blockchain framework. It enables users to create custom rules for ownership, transaction protocols, and the execution of state transition functions through "smart contracts." These contracts operate under specific conditions, executing agreed-upon terms autonomously when predefined criteria are met Buterin et al. (2014).

The structure of Ethereum is organised around accounts, each identifiable by a unique 20-byte address with prefix "0x". Accounts on Ethereum are categorised into two types: Externally Owned Accounts (EOA), controlled by private keys, and Contract Accounts (CA), governed by their embedded code. EOAs initiate transactions through signatures, while CAs activate their code upon receiving messages, facilitating further interactions like contract creation or internal state modifications.

Ethereum transactions differ from Bitcoin in several key respects. They can originate from either an EOA or a CA, include additional data fields, and, in the case of CAs, can trigger executable responses. Transactions in Ethereum comprise multiple fields: the recipient and sender addresses, the amount of Ether being transferred, and optional data. They also specify "Startgas" and "Gasprice," which cap the computational effort and transaction cost, respectively, preventing system abuse through resource-intensive operations Luu et al. (2017).

Transaction fees within the Ethereum ecosystem serve as a crucial regulatory mechanism, maintaining the integrity of the network by incentivising validators through a proof-of-stake (PoS) consensus mechanism. The transition from proof-of-work (PoW) to PoS was formalised through Ethereum Improvement Proposal (EIP) 3675, marking a significant stride towards reducing the network's environmental footprint and enhancing scalability. Under PoS, the likelihood of earning rewards is proportionate to validators' staked holdings, as opposed to the PoW model which emphasises computational power. Validators may choose to combine their stakes together, thus forming a pool that improves the probability of successfully validating transactions and securing rewards, whilst simultaneously reducing the risks associated with individual validation Bhatt and Pandey (2024).

2.1.3 ERC-20

ERC-20 is a technical standard used for smart contracts on the Ethereum blockchain for deploying tokens. ERC means Ethereum Request for Comment and the number 20 represents the unique proposal identifier. This standard provides a set of rules that all Ethereum-based tokens must follow, allowing for the seamless interaction between tokens and decentralised applications (dApps) within the Ethereum ecosystem.

With the inception of Ethereum, there was a need for a standardised protocol to streamline the token implementation process to ensure interoperability among various Ethereum tokens and applications. The ERC-20 standard, proposed in November 2015 by Fabian Vogelsteller, defines a common list of rules that an Ethereum token has to implement, giving developers the ability to program how new tokens will function within the Ethereum ecosystem. This standardisation simplifies the task of integrating new tokens on various services like wallets and exchanges.

The ERC-20 standard's accessibility has lowered the barriers to entry for token creation, allowing for a proliferation of new tokens on Ethereum. While this democratisation has spurred innovation, it has also led to a surge in fraudulent activity, notably in the form of initial coin offerings (ICOs). ICOs have been used to launch new projects and tokens usually with the promise of high returns for early investors. However, without proper regulation and oversight in "The new digital wild west of finance" as noted by Robinson, many ICOs have turned out to be scams, with promoters absconding with investors' funds after the token sale Robinson (2017), Catalini and Gans (2020).

Moreover, the ERC-20 framework does not inherently address security concerns, placing the onus on the developers to ensure the smart contract code's integrity. Attackers can exploit security vulnerabilities in smart contract implementations of ERC-20 tokens. For instance, the infamous "batchOverflow" and "proxyOverflow" bugs have been exploited to generate an astronomically large supply of tokens, leading to substantial financial losses and market manipulation Atzei et al. (2017).

Phishing campaigns have also targeted ERC-20 tokens, where attackers create fraudulent tokens or websites to deceive users into disclosing private keys or sending funds to the wrong addresses. These phishing attempts often exploit the trust and eagerness of users seeking to participate in new token offerings Chen et al. (2018). In response to such fraudulent activities, the community has taken steps to enhance due diligence and security measures. This includes third-party audits of smart contract code by companies such as Certik who charge a fee for such services, improved security standards beyond ERC-20, and community-driven efforts to educate investors about the risks associated with token investments Perez et al. (2021).

2.1.4 Overview of Cryptocurrency Fraud

The advent of cryptocurrencies has not only introduced a new era in financial technology but has also introduced novel forms of financial deceit. Cryptocurrencies, like Ethereum, have become prime targets for an array of sophisticated fraudulent schemes, exploiting both technological complexities and regulatory gaps. According to Trozze et al. (2022), a systematic review that explored the landscape of cryptocurrency fraud highlighted the existence of 47 unique types of fraud. This review categorised various schemes such as Ponzi schemes, where returns to older investors are paid out of new incoming funds; high yield investment programs, which promise unsustainable returns; pump-and-dump schemes that involve artificially inflating the price of an asset before selling off; and ransomware attacks demanding ransoms in cryptocurrency forms.

These fraudulent activities not only underscore the vulnerabilities inherent in digital financial systems but also highlight the urgent need for sophisticated detection mechanisms. These systems need to be dynamic, capable of evolving quickly to keep pace with the rapidly advancing fraudulent strategies that leverage the unique properties of blockchain technologies Trozze et al. (2022).

2.1.5 Specific Detection Techniques in Ethereum

The inherent features of Ethereum, such as its Turing-complete smart contracts and decentralised architecture, while innovative, have opened up new vectors for fraudulent activities. As highlighted by Liu et al. (2023), a survey on blockchain abnormal transaction detection emphasised the development of machine learning models tailored to identify Ethereum-specific frauds. These models are designed to scrutinise the unique structure and transaction data intrinsic to Ethereum, such as unusual smart contract executions and anomalous transaction flow patterns, which are indicative of schemes like Ponzi schemes and honeypot contracts.

This survey indicates a significant trend towards the development of specialised machine learning models that can effectively leverage Ethereum's transactional data to detect and prevent fraudulent activities. The continuous evolution of these detection techniques is crucial as they must adapt to counteract the sophisticated strategies employed by fraudsters who are constantly exploring new ways to exploit the system Liu et al. (2023).

2.1.6 Application of Advanced Machine Learning Models

Advanced machine learning models have shown promising results in identifying and categorising Ethereum fraud:

- **Graph Neural Networks (GNN)**: Tan et al. (2024) proposed a framework utilising Graph Convolutional Neural Networks (GCN) to analyse Ethereum transactions. By constructing a transaction network and employing a behavior-based network embedding algorithm, their system achieved an impressive 96% accuracy in distinguishing between legal and fraudulent transactions. This approach highlights the potential of GNNs to capture the complex interconnections and patterns typical of Ethereum transactions Tan et al. (2024).

- **Light Gradient Boosting Machine (LGBM)**: Research comparing various machine learning classifiers found that LGBM, alongside Logistic Regression and Random Forest, offers a competitive approach for fraud detection in Ethereum. These models are valued for their efficiency and effectiveness in handling large datasets, which are characteristic of blockchain technologies Aziz et al. (2022).

2.1.7 Feature Set Construction and Correlation Analysis

The accuracy and efficacy of machine learning (ML) and artificial intelligence (AI) models heavily depend on the quality of the dataset used for training, particularly the relevance and selection of features Guyon et al. (2008). A well-constructed feature set is vital as it should not only represent the data accurately but also encapsulate essential characteristics that are significant for the model's intended tasks.

Feature Set Construction

Feature set construction transforms raw data into a structured format usable by machine learning models. This process, termed 'feature engineering,' requires substantial domain expertise to capture the underlying patterns and relationships pertinent to the specific domain. Techniques such as normalisation, which scales input variables to a common scale, and aggregation, which combines several features into one, can derive new insights and reveal complex patterns. For example, creating interaction terms that explore the interactions between variables can uncover non-linear relationships that simple models might miss Hastie et al. (2009).

Correlation Analysis in Feature Selection

Correlation analysis is crucial for determining the linear relationships between pairs of features, using metrics such as the Pearson correlation coefficient, which ranges from -1 (perfect negative correlation) to +1 (perfect positive correlation). A common practice is to use a threshold, typically 0.7, to filter out highly correlated features to prevent multicollinearity, which can destabilise the model and lead to over fitting James et al. (2013). However, other types of correlation measures, such as Spearman or Kendall, are used when the data does not meet the normality assumption, providing a more robust analysis in non-linear data scenarios.

The selection of the correlation threshold is not rigid and can be adjusted based on the specific modelling technique or domain requirements. For instance, less stringent thresholds might be appropriate in domains where data is scarce, and more nuanced distinctions between features are necessary.

Practical Applications and Case Study

Kuhn and Johnson (2013) illustrates the significant role of feature set construction and correlation analysis for predictive modelling. By efficiently managing feature sets through the elimination of highly correlated clinical indicators has proven to enhance the accuracy of predictive models. They demonstrate that such careful feature selection not only preserves, but can actually enhance the clinical validity of the predictions, making the models more reliable without sacrificing detail or accuracy. This case exemplifies the critical importance of precision in feature engineering to improve model performance across high-stakes domains.

Advanced Techniques and Challenges

Beyond basic correlation checks, advanced techniques like recursive feature elimination and regularisation methods (such as L1 and L2 penalties) offer more sophisticated ways to manage feature sets by penalising the inclusion of less important features, thus reducing overfitting Tibshirani (2018). These methods not only help in feature selection but also enhance the model's generalisation capabilities.

Feature set construction and correlation analysis are fundamental processes in the development of powerful AI and ML models. By carefully selecting, engineering, and analysing features, data scientists can enhance model accuracy, prevent over fitting, and ensure that models perform well on unseen data. The continuous evolution of feature engineering and selection methodologies highlights the dynamic nature of the field, underscoring the importance of staying updated with the latest research and techniques Guyon et al. (2008), Dormann et al. (2013).

2.1.8 Tree Models: Decision Trees, Random Forest, and XGBoost in Ethereum Fraud Detection

Introduction to Tree Models

Tree-based models are a cornerstone of machine learning, characterised by their hierarchical decision-making structure. Deployed across a diverse range of applications, from anomaly detection in experimental physics to sophisticated financial systems, these models are notably effective in Ethereum fraud detection due to their ability to simplify complex decision-making processes Chen and Guestrin (2016).

Decision Trees

At the heart of tree-based modelling is the Decision Tree, which uses a set of binary rules to progressively divide data into smaller subsets, forming a tree with decision nodes (branches) and outcomes (leaves). The decision at each node is typically based on a threshold value, represented mathematically as:

$$\text{if } x_i \leq \text{threshold, go to left child node, otherwise go to right child node}$$

where x_i is a feature. This model's simplicity and interpretability make it invaluable for preliminary data analysis and exploring underlying data structures ?.

Random Forest

Building on Decision Trees, the Random Forest model enhances decision making by creating an ensemble of decision trees. It aggregates their predictions through majority voting or averaging, represented as:

$$\hat{y} = \frac{1}{B}\sum_{b=1}^{B} T_b(x)$$

where $T_b(x)$ is the prediction of the b-th tree and B is the number of trees. This method improves accuracy and performance, effectively identifying complex fraudulent patterns in Ethereum Breiman (2001).

XGBoost and Its Advancements

XGBoost (Extreme Gradient Boosting) optimises both model performance and computational efficiency. It includes an objective function with a regularisation term to prevent overfitting, given by:

$$\text{Obj}(\Theta) = \sum_{i=1}^{n} l(y_i, \hat{y}_i^{(t-1)} + f_t(x_i)) + \Omega(f_t)$$

where l is a differentiable loss function, $\hat{y}_i^{(t-1)}$ is the prediction at the $(t-1)$-th step, and Ω is the regularisation term. XGBoost's capabilities make it particularly effective for detecting subtle fraud patterns in Ethereum transactions Chen and Guestrin (2016).

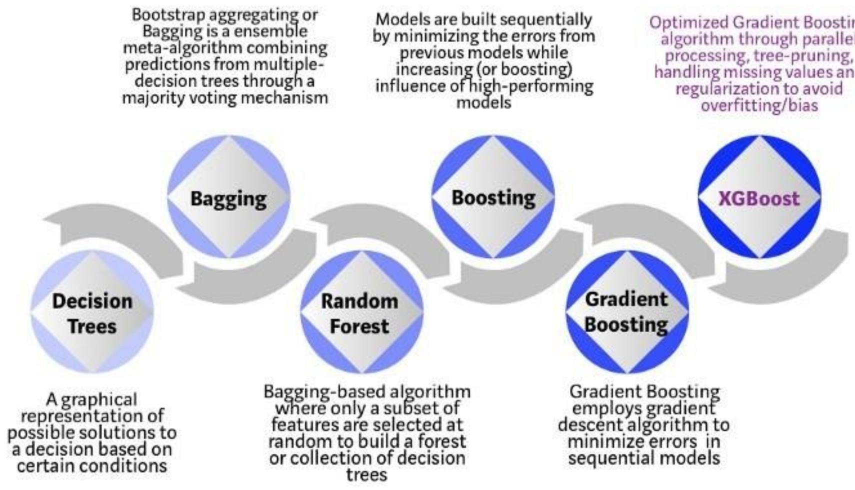
Bootstrap aggregating or Bagging is a ensemble meta-algorithm combining predictions from multiple-decision trees through a majority voting mechanism
Models are built sequentially by minimizing the errors from previous models while increasing (or boosting) influence of high-performing models
Optimized Gradient Boosting algorithm through parallel processing, tree-pruning, handling missing values and regularization to avoid overfitting/bias
Bagging
Boosting
XGBoost
Decision Trees
Random Forest
Gradient Boosting
A graphical representation of possible solutions to a decision based on certain conditions
Bagging-based algorithm where only a subset of features are selected at random to build a forest or collection of decision trees
Gradient Boosting employs gradient descent algorithm to minimize errors in sequential models

Figure 2.1: The Evolution of Decision Trees

2.1.9 Application in Ethereum Fraud Detection

In Ethereum fraud detection, the ability of tree models like XGBoost to accurately classify and predict fraudulent activities is crucial, given the substantial financial stakes involved. Empirical studies, such as those conducted by Paschen (2022), have demonstrated that XGBoost, with its advanced gradient boosting framework, is exceptionally proficient at detecting fraudulent patterns and anomalies that other models might miss. Its capacity to process large amounts of data and adaptively refine its learning makes it a powerful tool against sophisticated fraud tactics in the blockchain domain Paschen (2022).

2.2 Comparative Literature and Results

In comparison to existing literature on detecting illicit activities within cryptocurrency networks, recent research by Farrugia et al. (2020) introduces a comprehensive approach specifically tailored for the Ethereum blockchain. While prior studies have acknowledged the prevalence of illegal activities such as money laundering, fraud, and phishing within blockchain ecosystems, Farrugia et al. (2020) provide a nuanced investigation focusing on over 400 million transactions on the Ethereum network.

Their methodology revolves around the utilisation of a machine learning classifier called XGBoost, to analyse transaction histories of flagged illicit accounts alongside normal ones. Through 10-fold cross-validation, their model achieves an impressive average accuracy of 0.963 ($\pm$ 0.006) and an average Area Under the Curve (AUC) of 0.994 ($\pm$ 0.0007). Notably, the study identifies key features crucial for identifying illicit accounts, including the time difference between first and last transactions, total Ether balance, and minimum value received.

The results of Farrugia et al. (2020) underscore the efficacy of their proposed approach in distinguishing illicit activities within the Ethereum network. Their contribution enhances the understanding and detection of illegal transactions in blockchain environments.

2.2.1 Logistic Regression in Ethereum Fraud Detection

Logistic Regression (LR) is a powerful statistical method primarily used for binary classification. It estimates probabilities using a logistic function, particularly effective in scenarios where the decision boundary between classes is linear or nearly linear. The strength of LR lies in its simplicity, interpretability, and the ease with which it can be updated with new data Hosmer et al. (2013).

In the context of Ethereum fraud detection, LR can be particularly effective in quickly assessing and classifying transactions based on predefined indicators of fraud. For instance, a logistic model might analyse variables such as transaction amount, frequency, and the novelty of the receiving account to calculate the likelihood of a transaction being fraudulent. Its computational efficiency also ensures that it can be applied to large datasets typical of blockchain transactions, providing a fast first-level filter for potentially fraudulent activities Zięba et al. (2016).

The logistic regression model is mathematically represented as follows:

$$P(Y = 1|\mathbf{X}) = \frac{1}{1 + e^{-(\beta_0 + \beta_1 X_1 + \beta_2 X_2 + \cdots + \beta_n X_n)}}$$

where $P(Y = 1|\mathbf{X})$ is the probability of the event occurring, $\beta_0, \beta_1, \beta_2, ..., \beta_n$ are the coefficients, and $X_1, X_2, ..., X_n$ are the features.

2.3 Convolutional Neural Networks for Complex Pattern Recognition in Ethereum Data

Convolutional Neural Networks (CNNs) are widely recognised for their success in image recognition and are capable of capturing spatial and temporal patterns, making them ideal for applications beyond mere image classification. Their suitability extends not just to traditional image-based applications but also to analysing complex data structures such as transaction sequences in blockchain environments Krizhevsky et al. (2012). It is also essential to discuss the theoretical basis for using CNNs in this context, supported by literature from related fields where CNNs have been applied to similar types of data problems, such as financial time series analysis ?.

2.3.1 Application to Ethereum Fraud Detection

The unique architecture of CNNs enables them to excel where simpler models might falter. Ethereum transactions, which can be complex and interlinked, present a type of data that is ripe for CNN analysis. By treating transaction sequences as temporal data or converting transaction data into a grid-like structure akin to an image, CNNs can effectively identify intricate patterns characteristic of fraudulent activities:

- **Detection of Layering in Money Laundering**: CNNs can analyse sequences of transactions to detect the layering phase of money laundering, where illicit funds are moved through a complex series of transfers to obscure their origin.
- **Identification of Anomalous Transaction Fees**: CNNs can also pinpoint anomalies in transaction fee structures, such as unusually high gas fees that might indicate manipulative practices or attempts to prioritise fraudulent transactions.

2.3.2 Enhancements from CNN Application

The deployment of CNNs in detecting Ethereum fraud introduces several advancements:

- **Pattern Recognition**: Leveraging their deep learning capabilities, CNNs can recognise and learn from patterns across large and unstructured datasets, providing insights into the operational dynamics of the Ethereum blockchain that other models might miss.
- **Adaptability to New Frauds**: The flexible nature of CNNs in handling different data representations makes them adaptable to evolving fraudulent schemes as criminals change tactics.

2.3.3 Theoretical and Practical Implications

By comparing the performance of CNNs with traditional models like Logistic Regression, Random Forest, and XGBoost, the review could highlight specific cases where CNNs provide superior detection capabilities. This comparative analysis will be crucial in substantiating whether CNNs can offer a significant advantage over traditional methods for Ethereum fraud detection, potentially leading to more effective and adaptive fraud prevention mechanisms.

2.3.4 Mathematical Background of CNNs

The fundamental operation in a CNN is the convolution operation, mathematically represented as:

$$S(i,j) = (I * K)(i,j) = \sum_{m} \sum_{n} I(m,n) K(i-m, j-n)$$

where I is the input image, K is the kernel, and S is the feature map. This operation slides the kernel over the image, computing elementwise multiplications and summing them up, thus capturing pattern information from the data.

2.3.5 1D Convolutional Neural Networks in Ethereum Fraud Detection

As transaction data does not represent an image, more like a one dimensional array, 1D Convolutional Neural Networks will be explored as they are particularly effective for analysing sequential data, such as transaction data in Ethereum. By applying a 1D convolution operation, these networks can detect complex fraud patterns embedded in the grid like structure that is transaction data. The mathematical operation for a 1D convolution is given by:

$$S(t) = (I * K)(t) = \sum_{a=-\infty}^{\infty} I(a) \cdot K(t-a)$$

where $S(t)$ represents the output feature map, I denotes the input data sequence, K is the convolutional kernel, and t indexes the output time steps. This convolution extracts important features from the transaction data, allowing the model to learn from temporal dependencies and anomalies indicative of fraudulent activities. The ability of 1D ConvNets to adapt to the dynamic nature of Ethereum transactions makes them a powerful tool for fraud detection, capable of identifying subtle patterns that may be indicative of fraudulent behaviour.

2.3.6 Ensemble and Hybrid Models

Considering the complexity and evolving nature of Ethereum fraud, ensemble methods that combine multiple machine learning techniques are increasingly being explored. These methods benefit from the strengths of individual models while mitigating their weaknesses, thus enhancing overall detection capabilities. This approach reflects a growing recognition of the need for adaptable and agile solutions to keep pace with sophisticated fraudsters in the Ethereum ecosystem.

2.3.7 BERT4ETH: Advanced Transformer Technology in Ethereum Fraud Detection

The paper "BERT4ETH: A Pre-trained Transformer for Ethereum Fraud Detection" by Hu (2023) introduces an innovative application of transformer technology, specifically designed for the complex task of detecting fraud within Ethereum transactions. BERT4ETH leverages the pre-trained Transformer encoder, meticulously tailored to address the unique challenges posed by Ethereum's transaction environment, characterised by high repetitiveness, skew distribution, and heterogeneity.

Key Features and Innovations of BERT4ETH

BERT4ETH fundamentally transforms the approach to Ethereum fraud detection by utilising deep learning techniques to analyse transaction data. This model is adept at capturing dynamic sequential patterns inherent in transaction data, a crucial feature given the temporal and often intricate nature of transaction logs in blockchain technologies.

The developers of BERT4ETH have implemented three strategic technological enhancements to optimise the model's performance:

- **Repetitiveness Reduction**: This strategy focuses on minimising redundancy in data processing, crucial for enhancing the efficiency of the model in real-time applications.
- **Skew Alleviation**: By addressing the skewness in the distribution of transaction data, BERT4ETH ensures that the model remains accurate across diverse and unevenly distributed datasets.

- **Heterogeneity Modeling**: This involves tailoring the model to effectively handle the diverse nature of transaction types and patterns within the Ethereum blockchain, ensuring comprehensive fraud detection capabilities.

Empirical Performance and Impact

Empirical evaluations highlight BERT4ETH's superior performance, where it consistently outperforms state-of-the-art methods in critical tasks such as phishing account detection and de-anonymisation. These results not only validate the effectiveness of BERT4ETH in practical scenarios but also underscore its potential to revolutionise Ethereum fraud detection strategies.

The introduction of BERT4ETH represents a significant advance in the field of cryptocurrency fraud detection, offering a highly specialised and effective tool tailored to the nuanced demands of Ethereum transactions. As blockchain technology continues to evolve and expand, tools like BERT4ETH are essential for maintaining the integrity and security of digital transactions.

2.3.8 Comprehensive Approaches and Theoaretical Models

In addition to specialised detection techniques, there is a growing emphasis on the development of comprehensive models and theoretical frameworks to combat the evolving landscapes of financial crime. Zarpala and Casino (2021) contribute significantly to this field with their work on a blockchain-based forensic model tailored for investigating financial crimes. Their study presents a taxonomy of financial investigation techniques, adaptable to the Ethereum blockchain environment. This holistic approach facilitates a deeper comprehension of the intricate patterns underlying blockchain fraud, thereby enabling the formulation of more effective countermeasures. This model not only enhances the understanding of blockchain dynamics but also provides actionable insights for regulators and law enforcement agencies aiming to curtail financial malpractices facilitated through blockchain technology.

2.4 Conclusion

The literature review has extensively covered the dynamic and multifaceted approaches to detecting and preventing fraud within the Ethereum blockchain ecosystem. From the foundational technologies that underpin blockchain's unique capabilities to the cutting-edge applications of machine learning models and hybrid techniques, the depth and breadth of research demonstrate a vigorous and evolving response to the challenges posed by financial crime in digital finance.

The integration of traditional methodologies with innovative machine learning techniques, such as Decision Trees, Random Forest, XGBoost, and Convolutional Neural Networks, highlights the industry's responsibility to refining fraud detection processes. The emergence of advanced models like BERT4ETH further exemplifies the ongoing adaptation of deep learning to specifically address the complexities of Ethereum transactions, showcasing significant strides in the development of specialised tools tailored for the crypto environment.

Moreover, the exploration of comprehensive theoretical models and frameworks, as discussed in works by Zarpala and Casino, provides a structured approach to understanding and combating fraud. These contributions are crucial in shaping a strong defence mechanism that is not only reactive but also proactive in anticipating and mitigating potential threats.

As blockchain technology continues to mature and integrate into global financial systems, the importance of these research endeavours cannot be overstated. The collaborative efforts of academia, industry, and regulatory bodies are essential to enable a secure, transparent, and resilient digital financial landscape. This literature review underscores the importance of continuous research and the need for innovative solutions to keep pace with the rapidly evolving tactics of fraudsters, ensuring the integrity and stability of Ethereum and other blockchain technologies.

In conclusion, while significant progress has been made in the detection and prevention of Ethereum fraud, the persistent advancement of fraudulent schemes demands ongoing research and development. The insights gained from this review not only contribute to academic knowledge but also inform practical applications, guiding future strategies in blockchain security and fraud mitigation.

Chapter 3

Methodology

3.1 Dataset

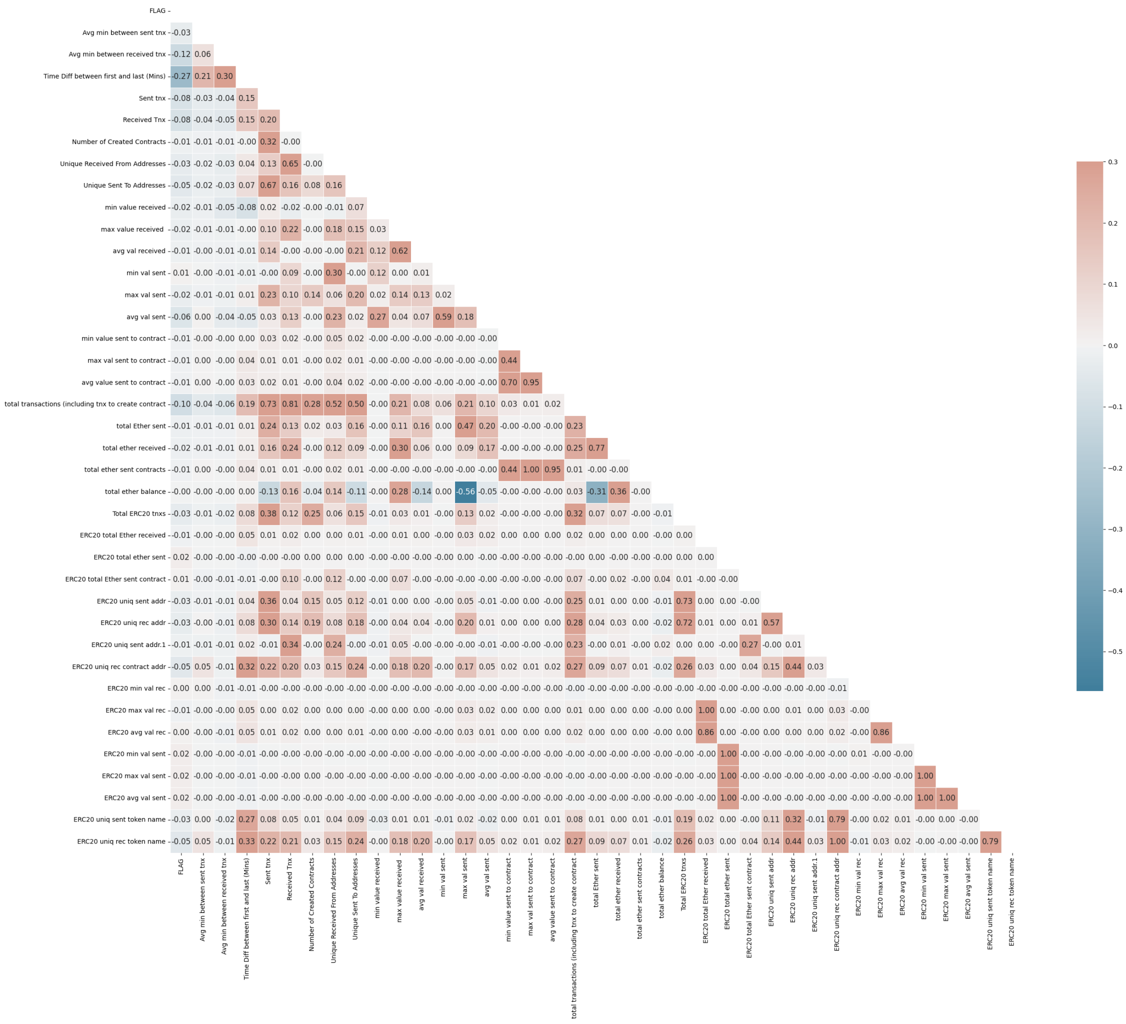

Figure 3.1: Initial correlation matrix of full unprocessed dataset

The dataset used in this study consists of 9,816 unique Ethereum account entries and is sourced from `Kaggle.com`. It includes a mix of 7,662 normal accounts and 2,179 fraudulent transactions. Each entry in the dataset contains a ‘FLAG‘ column, which is used to label the transactions, with ‘0‘ indicating normal and ‘1‘ indicating fraudulent accounts. Given the computational constraints of this research, the dataset size has been chosen to optimise training times. However, the dataset suffers from an imbalance between the normal and fraudulent transaction classes. To address this issue, data augmentation techniques such as the Synthetic Minority Over-sampling Technique (SMOTE) will be employed to balance the dataset effectively, ensuring more reliable and accurate machine learning analysis.

For a detailed view of the dataset columns prior to any processing, see Table A.1 in Appendix A.1: *Summary of Dataset Columns before Pre-processing*.

3.1.1 Data Pre-Processing

Dropping Categorical Variables

As many machine learning models compute solely with numerical figures, categorical columns are often removed. While categorical variables can be encoded into numerical values—for example, through one-hot encoding—this is only beneficial if the categorical features are relevant and contribute meaningful information. Removing irrelevant or less informative features simplifies the model, thus enhancing its performance and reducing the risk of overfitting. Including categorical variables, unless informative and appropriately encoded, may introduce unnecessary complexity and diminish both model interpretability and performance Hastie et al. (2009).

Handling Missing Values

Machine learning models typically require complete datasets. Missing values, which can arise due to errors in data collection or processing, need to be addressed to apply machine learning algorithms effectively. Median Imputation, a technique where missing values are replaced with the median value of the data, is particularly useful when the data is skewed or contains outliers. The median is less affected by outliers compared to the mean, making it a more stable measure of central tendency in such cases Little and Rubin (2002). This approach also helps maintain the overall distribution of the dataset, ensuring that the model is trained on data that represent the true nature of the underlying processes. The implementation of median imputation in this study was accomplished using the following Python code snippet:

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

Filtering Features with Zero Variance

A feature with zero variance indicates that it has the same value for all observations. Such a feature provides no information to help distinguish between different observations, rendering it useless for the model. Removing features with zero variance can reduce the dimensionality of the dataset, thereby enhancing computational efficiency and model simplicity Guyon and Elisseeff (2003).

Columns with Zero Variance

Below is a list of columns with zero variance that have been removed from the dataset:

Column Name	Variance
ERC20 avg time between sent tnx	0.0
ERC20 avg time between rec tnx	0.0
ERC20 avg time between rec 2 tnx	0.0
ERC20 avg time between contract tnx	0.0
ERC20 min val sent contract	0.0
ERC20 max val sent contract	0.0
ERC20 avg val sent contract	0.0

Table 3.1: Columns Removed Due to Zero Variance

The following code snippet demonstrates how these columns were removed from the data frame:

```
df.drop(df.var()[no_var].index, axis = 1, inplace = True)
print(df.var())
```

As of now, the dataset contains 38 columns and 9,841 rows of non-null data.

3.1.2 Dimensionality Reduction via Correlation Analysis

In the pursuit of constructing an optimised feature set for the development of machine learning models, dimensionality reduction plays a crucial role. One effective technique used in this study is correlation analysis, which involves the removal of features that are highly correlated with others, often using a threshold such as 0.7 or above. This threshold is guided by the work of Dormann et al. (2013), who discuss the issues arising from multicollinearity in statistical models.

Justification for Using a 0.7 Correlation Threshold

Multicollinearity between independent variables can distort the predictive model in several ways:

- **Parameter Estimates:** Multicollinearity can lead to unstable parameter estimates, making it challenging to assess the effect of individual features on the target variable O'Brien (2007).
- **Model Interpretability:** A high degree of correlation between features can reduce the interpretability of the model, as it becomes difficult to attribute the explanatory power to any one feature James et al. (2013).
- **Overfitting:** In the presence of multicollinearity, models may fit the noise in the training data too closely, leading to overfitting and poor generalisation on unseen data Farrar and Glauber (1967).

Addressing Multicollinearity

By removing features with a correlation coefficient above 0.7, the model's variance inflation factor (VIF) is controlled, ensuring that the predictive power is not unduly influenced by redundant data Kutner et al. (2004). This approach not only simplifies the model but also aids in the avoidance of overfitting, promoting a better generalisation of new data, which is paramount in the application of machine learning to fraud detection in the Ethereum blockchain.

Implementation in the Current Study

Features with a correlation coefficient exceeding 0.7 were removed. This decision was supported by the analysis of variance inflation factors and the aim to minimise model complexity while retaining predictive performance. The threshold of 0.7 was chosen based on empirical benchmarks commonly cited in the literature and was validated through cross-validation performance metrics as overfitting was observed when these highly correlated features remained in the dataset.

3.1.3 Features Excluded

The table (A.2) illustrates the pairs of features with high correlation coefficients, indicating significant multicollinearity, which necessitates their exclusion from the model. The refined set of 18 features, represented in the condensed correlation matrix, constitutes a more streamlined dataset. This dimensionality reduction should improve the performance of the predictive model. By reducing the number of features, computational demands are reduced and model interpretability is enhanced, with each feature exerting a more distinguishable influence on the model's predictions James et al. (2013). Moreover, the minimised feature set decreases the likelihood of overfitting, thus improving the model's ability to generalise to new, unseen data—an essential attribute in fraud detection within the Ethereum blockchain Farrar and Glauber (1967).

This approach aligns with established methodologies that advise prudent feature selection to manage multicollinearity and enhance model performance Dormann et al. (2013), O'Brien (2007), Kutner et al. (2004).

3.2 Dataset Preparation and Split

3.2.1 Training and Testing Set Partition

A fundamental step in the preparation of data for machine learning is dividing the dataset into subsets for training and testing purposes. This practice is crucial to evaluate the model's ability to generalise to new, unseen data, beyond the patterns learned during training. In this study, the dataset was split into two distinct sets: a training set, which comprised 80% of the data, and a testing set, which accounted for the remaining 20%. The split was executed using the `train_test_split` function from the scikit-learn library, a well-established tool for such tasks in the data science community Pedregosa et al. (2011).

The choice of an 80/20 split aligns with common conventions in machine learning research, which balances the need for a sufficiently large training dataset against the necessity for a robust testing framework Kohavi (1995). The training set is utilised to build and tune the model, allowing it to learn the underlying patterns and relationships present in the data. The testing set, on the other hand, acts as a proxy for real-world data, testing the model's predictions against outcomes it has not encountered during training.

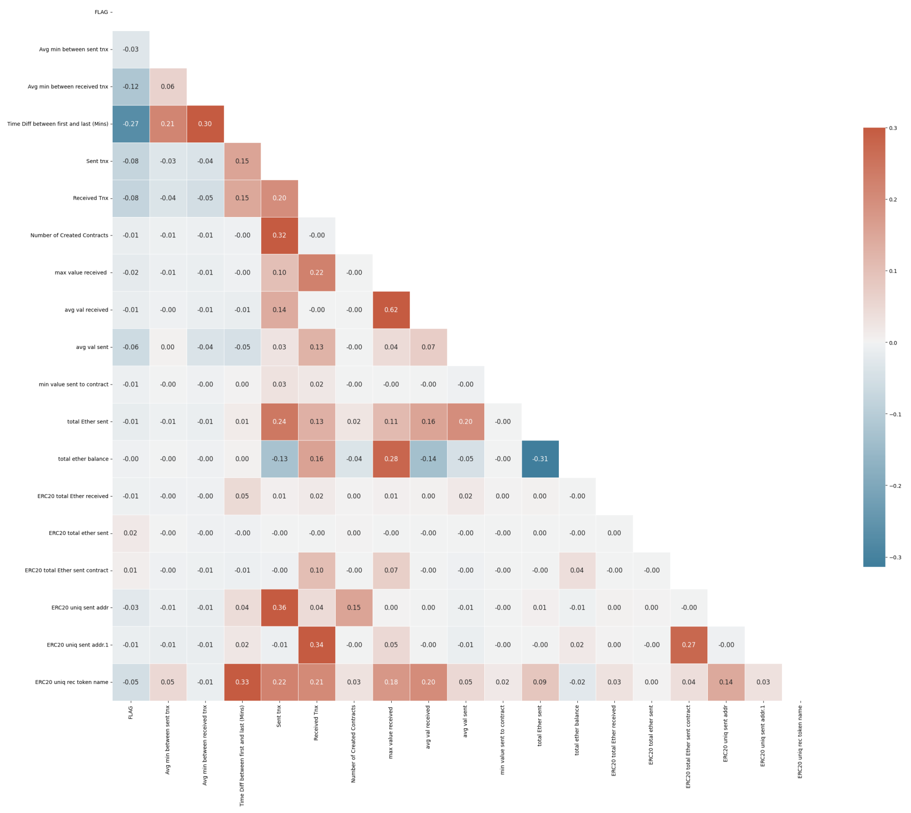

Figure 3.2: Updated smaller correlation matrix after data pre-processing

3.2.2 Random State for Reproducibility

To ensure the reproducibility of the results, a fixed `random_state` was set. This parameter serves as the seed to the random number generator used in partitioning the data, ensuring that the same split can be achieved consistently across different runs and by other researchers replicating the study.

3.2.3 k-fold Cross-validation

To ensure the generalisability of our predictive models, k-fold cross-validation will be employed as a key validation strategy. Cross-validation is a statistical method used to estimate the skill of machine learning models. It is commonly used to protect against overfitting in a predictive model, particularly when the data is limited in size Kohavi (1995).

In k-fold cross-validation, the data set is divided into k smaller sets or 'folds'. The model is then trained on k-1 of these folds, with the remaining fold used as the test set. This process is repeated k times, with each of the k folds used exactly once as the test set. The results from the k iterations are then averaged (or otherwise combined) to produce a single estimation James et al. (2013). The advantage of this method is that all observations are used for both training and validation, and each observation is used for validation exactly once.

While a comprehensive exploration of k-fold cross-validation will be discussed in detail in the results section, it is worth noting that this methodology allows us to maximise both our training and testing data. This is particularly important given the complex nature of fraud detection within Ethereum transactions, where every data point can contribute significantly to model performance.

3.3 t-Distributed Stochastic Neighbor Embedding (t-SNE)

t-Distributed Stochastic Neighbor Embedding (t-SNE) is a non-linear dimensionality reduction technique that excels at visualising high-dimensional data in two or three dimensions. Developed by van der Maaten and Hinton in 2008, t-SNE aids in the identification of inherent patterns in the data by projecting high-dimensional features onto a lower-dimensional space. In this representation, similar instances are modeled by nearby points, and dissimilar instances are modeled by distant points with high probability.

In this study, t-SNE has been employed to explore the structure of a complex Ethereum transaction dataset. By reducing the feature space to two and three dimensions, the t-SNE plots provide visual insights into the data, potentially revealing clusters that could signify underlying relationships or groupings relevant to fraudulent activity detection.

3.3.1 Two-dimensional t-SNE Visualisation

The two-dimensional t-SNE visualisation generated two distinct clusters, which suggests a degree of separability within the data. The formation of these clusters indicates that, when projected onto a two-dimensional plane, the data points exhibit natural groupings that could correlate with the characteristics of typical and atypical transaction patterns. Below is the visual representation of the two-dimensional t-SNE analysis:

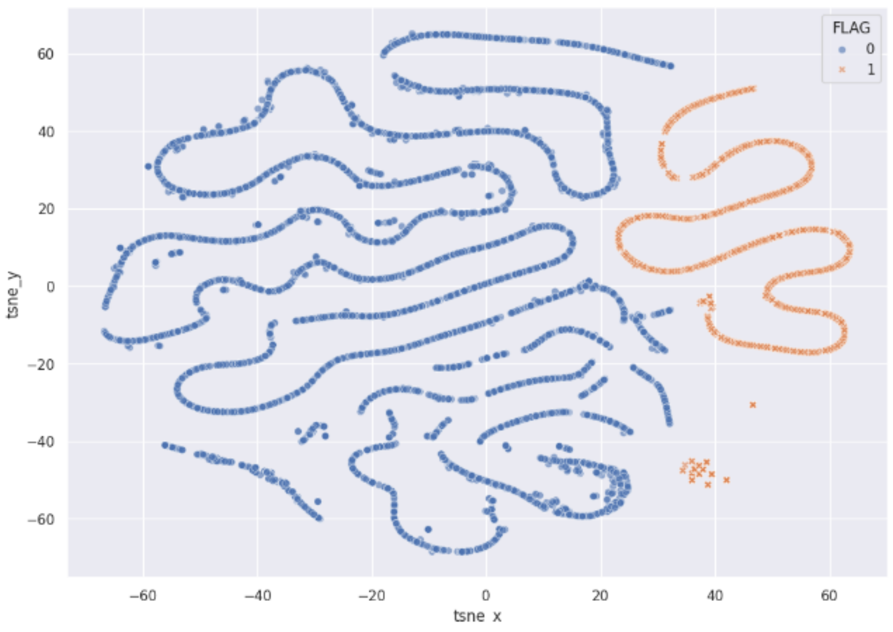

Figure 3.3: Two-dimensional t-SNE visualisation (Blue: normal account, Orange: illicit account).

3.3.2 Three-dimensional t-SNE Visualisation

Conversely, the three-dimensional t-SNE plot presents a more complex picture, with no distinct clusters immediately apparent. The additional dimension reveals subtleties and nuances of the dataset that are not captured in two dimensions. This complexity may reflect the intricate nature of Ethereum transactions, where simple groupings are insufficient to encapsulate the relationships present in the data.

The differences observed between the 2D and 3D visualisations underscore the importance of considering multiple perspectives when analysing high-dimensional data. While 2D projections are beneficial for identifying broad patterns, they may oversimplify the data and obscure important relationships. On the other hand, 3D projections, despite their complexity, provide a more comprehensive view of the data's structure. The lack of distinct clusters in the 3D plot suggest that the two classes are not linearly separable, thus fraud detection in Ethereum transactions require a more sophisticated, multi-faceted, approach that accounts for a higher level of detail and interaction within the data. These two figures highlight the necessity of machine learning techniques to help in distinguishing between normal and fraudulent accounts.

3.4 Dataset Class Imbalance

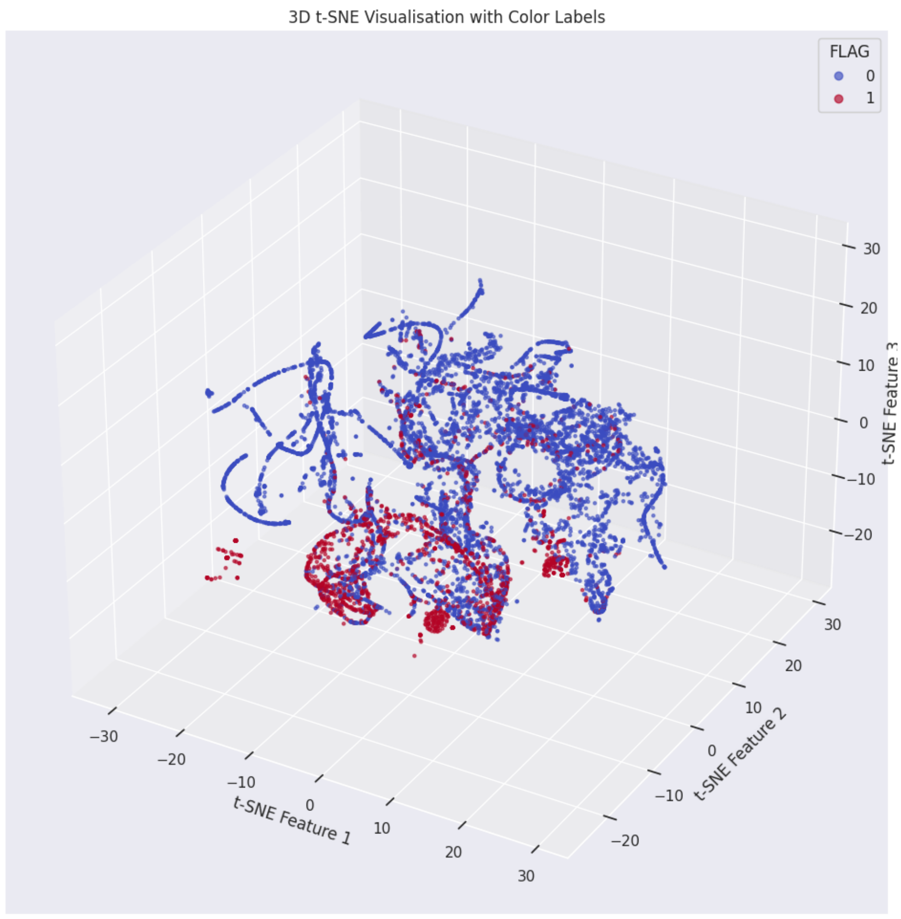

Figure 3.4: Three-dimensional t-SNE visualisation (Blue: normal account, Red: illicit account).

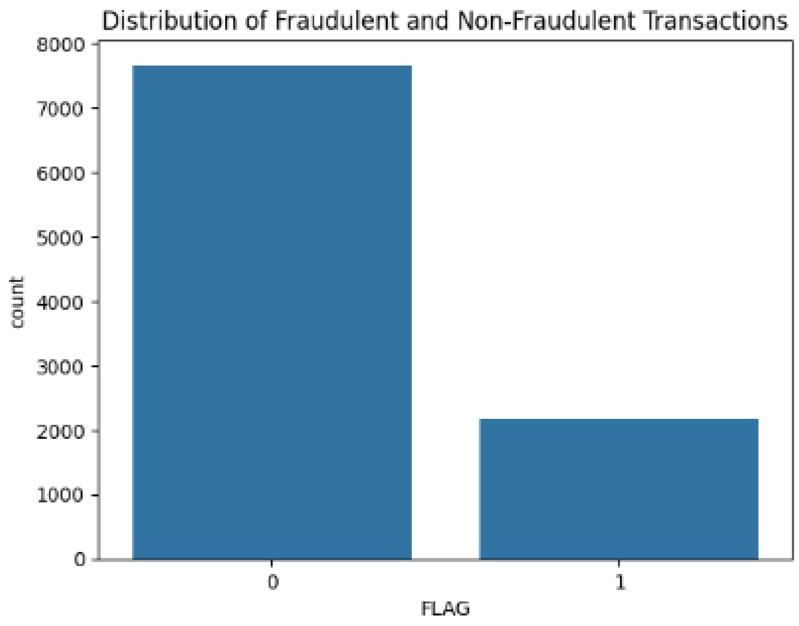

Figure 3.5: Class distribution before applying SMOTE.

Class imbalance can significantly impact the performance of machine learning models. Most algorithms assume an equal distribution of classes, and when one class dominates, the model may develop a bias towards the majority class, resulting in poor generalisation performance on the minority class He and Garcia (2009). Traditional evaluation metrics like accuracy can be misleading in imbalanced datasets. For instance, a model that consistently predicts 'non-fraudulent' could still achieve high accuracy despite failing to identify any fraudulent instances Jeni et al. (2013).

3.4.1 Strategies to Address Class Imbalance

Several strategies can be employed to address class imbalance:

- **Resampling Techniques:** Over-sampling the minority class or under-sampling the majority class can balance the class distribution. However, this comes with the risk of overfitting or losing valuable data, respectively Chawla et al. (2002).
- **Synthetic Data Generation:** Techniques like SMOTE (Synthetic Minority Over-sampling Technique) generate synthetic examples of the minority class to achieve balance without losing information Chawla et al. (2002).
- **Algorithmic Adjustments:** Some algorithms can be adjusted to compensate for imbalanced data by assigning higher weights to the minority class.
- **Anomaly Detection:** In cases where fraud is considered an anomaly, one-class classification or anomaly detection techniques may be more suitable than binary classification models Chandola et al. (2009).

Class imbalance is particularly pertinent in fraud detection domains, where fraudulent instances are typically rare compared to legitimate transactions. For this dataset, the imbalance is characterised by a majority of non-fraudulent instances (77.86%) and a significant minority of fraudulent instances (22.14%). This disproportion presents a risk of model bias towards predicting the majority class, which could result in an inadequate number of fraud detections—a critical failure for a fraud detection system.

Mitigating Class Imbalance

To address this imbalance, this study employs the Synthetic Minority Over-sampling Technique (SMOTE). SMOTE enriches the dataset without the loss of information typical of undersampling techniques, facilitating a more balanced class distribution and enabling the development of a model that is as sensitive to detecting fraudulent transactions as it is to confirming legitimate ones.

Dataset Distribution after SMOTE:

```
Shape of the training before SMOTE: ((7872, 18), (7872,))
Shape of the training after SMOTE: ((12230, 18), (12230,))
BEFORE OVERSAMPLING: Non-frauds: 6115, Frauds: 1757
AFTER OVERSAMPLING:  Non-frauds: 6115, Frauds: 6116
```

3.5 Model Development and Evaluation

3.5.1 Overview of Machine Learning Models Used

Logistic Regression (LR)

Logistic Regression is a fundamental statistical approach that models the probability of a binary outcome based on one or more predictor variables. Its simplicity, interpretability, and efficiency in binary classification problems make it a baseline model in many fraud detection studies Ngai et al. (2011). Whilst logistic regression might not capture complex relationships as effectively as more sophisticated algorithms, its outputs are highly interpretable, making it a valuable tool for initial analysis and a benchmark for more complex models.

Random Forest (RF)

Random Forest is an ensemble learning method that operates by constructing a multitude of decision trees at training time and outputting the class that is the mode of the classes (classification) of the individual trees. Random forests correct for the decision trees' tendency to overfit to their training set, making them effective for a variety of tasks, including fraud detection Carneiro et al. (2017). Their ability to handle large datasets with higher dimensionality and to identify the most important variables from a dataset makes them particularly suitable for detecting fraudulent activities in Ethereum transactions.

XGBoost Classifier

XGBoost is a decision-tree-based ensemble machine learning algorithm that uses a gradient boosting framework. In recent years, it has gained popularity due to its performance and speed in classification tasks. XGBoost has been particularly effective in various fraud detection scenarios because of its ability to handle imbalanced data, its flexibility to incorporate custom optimisation objectives and criteria, and its ability to capture complex non-linear relationships in data Chen and Guestrin (2016).

In the context of Ethereum fraud detection, the application of these models is promising yet exploratory. Ethereum transactions constitute a unique dataset with its own set of features and irregularities. Whilst the literature on using these specific models for Ethereum fraud detection may be developing, their proven effectiveness in similar domains suggests their potential utility. As such, this research aims to contribute to the body of knowledge by applying these established models to the novel domain of Ethereum fraud detection.

Convolutional Neural Network (CNN)

Though traditionally associated with image processing, Convolutional Neural Networks can be adapted for sequence data and are increasingly used in detecting patterns in time-series data, like financial transactions Fujimoto et al. (2019). Whilst not commonly used in Ethereum fraud detection, their capability to capture spatial and temporal anomalies in data could potentially offer novel insights, especially when transaction data is represented in a time-series or sequence format.

3.6 Convolutional Neural Network (CNN) Architecture

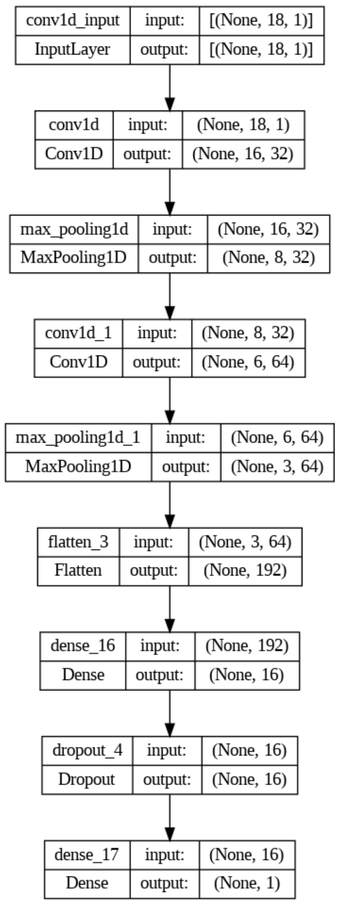
conv1d_input | input: | [(None, 18, 1)]
InputLayer | output: | [(None, 18, 1)]
conv1d | input: | (None, 18, 1)
Conv1D | output: | (None, 16, 32)
max_pooling1d | input: | (None, 16, 32)
MaxPooling1D | output: | (None, 8, 32)
conv1d_1 | input: | (None, 8, 32)
Conv1D | output: | (None, 6, 64)
max_pooling1d_1 | input: | (None, 6, 64)
MaxPooling1D | output: | (None, 3, 64)
flatten_3 | input: | (None, 3, 64)
Flatten | output: | (None, 192)
dense_16 | input: | (None, 192)
Dense | output: | (None, 16)
dropout_4 | input: | (None, 16)
Dropout | output: | (None, 16)
dense_17 | input: | (None, 16)
Dense | output: | (None, 1)

Figure 3.6: CNN Architecture

The model begins with a 1D Convolutional layer that has 32 filters and a kernel size of 3. This layer is designed to extract high-level features from the raw input data, which consists of 18 features extracted from the pre-processing steps representing various aspects of a single Ethereum account. The relu (rectified linear unit) activation function is used to introduce non-linearity, helping the model learn more complex patterns.

This is followed by a MaxPooling1D layer with a pool size of 2, which serves to reduce the dimensionality of the data, thereby condensing the information and reducing processing time while preserving important features.

Additional Convolutional and Pooling Layers

Another Conv1D layer follows, this time with 64 filters, allowing the network to build a more abstract representation of the input data. This layer uses the same kernel size and relu activation function, enhancing the model's ability to detect intricate patterns in the transaction data.

A subsequent MaxPooling1D layer with the same pool size further reduces the data dimensionality, streamlining the network for efficiency.

Flattening and Dense Layers

After the convolutional and pooling layers, the data structure is flattened to transform the 2D feature maps into a 1D vector. This step is crucial as it allows the transition from convolutional layers to the dense layers that follow.

A Dense layer consisting of 16 neurons with relu activation is then used to interpret the features extracted by the convolutional layers, providing a level of abstraction necessary for classification.

Dropout and Output Layer

A Dropout layer is incorporated with a rate of 0.5, which randomly sets input units to 0 at each update during training time. This layer is instrumental in preventing overfitting by reducing the model's complexity and promoting generalisation.

Finally, the output layer consists of a single neuron with a sigmoid activation function. This setup is suitable for binary classification tasks, as it outputs a probability indicating the likelihood of a transaction being fraudulent.

Model Compilation and Training

The model is compiled with a binary cross-entropy loss function, typical for binary classification problems. The optimiser used is Adam, a popular choice that provides an effective and efficient method for model training through adaptive learning rate optimisation.

The model is trained on the resampled training data (`x_tr_resample, y_tr_resample`) for 100 epochs with a batch size of 32. A validation split of 20% is used to monitor the model's performance on unseen data during training. This approach helps in tuning the model and avoiding overfitting.

Model Evaluation

Post-training, the model is evaluated on the scaled test data (`sc_test`), and its performance is quantified using a classification report and a confusion matrix. These evaluations provide insights into the model's predictive capabilities and its effectiveness in distinguishing between the two classes. The detailed results and analysis will be reviewed in the results section.

3.7 Model Selection Strategy for Hyperparameter Tuning

3.7.1 Initial Model Evaluation

To determine the most promising model for in-depth hyperparameter optimisation, an initial evaluation phase is conducted in which several candidate models are assessed based on standard machine learning metrics. This preliminary comparison encompasses models such as Logistic Regression, Random Forest, Convolutional Neural Networks, and XGBoost. The evaluation metrics used to compare the models include accuracy, precision, recall, F1-score, and the area under the ROC curve (AUC), ensuring a comprehensive assessment of each model's performance.

3.7.2 Criteria for Model Selection

The model demonstrating superior performance across these metrics, particularly emphasising recall and precision given the critical nature of fraud detection, will be selected for further refinement. In fraud detection, recall is crucial as it measures the model's ability to detect fraudulent transactions, whereas precision ensures that the model's predictions are reliable Powers (2011).

Once the best-performing model is identified, it will undergo a rigorous hyperparameter tuning process. This process aims to enhance the model's ability to generalise to unseen data while optimising its performance concerning the specific needs of Ethereum fraud detection. Hyperparameter tuning will be conducted using methodologies like GridSearchCV or RandomisedSearchCV, providing a systematic approach to exploring a range of parameter configurations and identifying the optimal set of hyperparameters Bergstra and Bengio (2012).

3.7.3 Evaluation of the Tuned Model

The final model, selected based on initial performance metrics and refined through hyperparameter optimisation, will be evaluated again on the testing set. This step ensures that the improvements from the tuning process translate into enhanced model performance on unseen data, reinforcing the model's generalisability and reliability.

3.7.4 Implications

This methodical approach ensures that computational resources are allocated efficiently, focusing on the model with the highest potential impact. It also aligns with best practices in machine learning, where model selection and tuning are guided by empirical evidence and strategic consideration of the problem domain Claesen and De Moor (2015).

3.8 Model Evaluation Metrics

The evaluation of machine learning models for Ethereum fraud detection relies on several performance metrics that are heavily utilised throughout academia, each offering insights into different aspects of model efficacy. These metrics include precision, recall, f1-score, accuracy, and the utilisation of a confusion matrix. Understanding the trade-offs between correctly identifying fraudulent transactions and avoiding the misclassification of legitimate transactions is critical in fraud detection applications.

3.8.1 Precision

Precision measures the accuracy of positive predictions, which in the case of fraud detection, reflects a model's ability to limit false positives—legitimate transactions incorrectly labelled as fraudulent. High precision is crucial in scenarios where the cost of falsely identifying legitimate activities as fraudulent is substantial, as it minimises inconvenience to users and reduces operational disruptions.

3.8.2 Recall

Recall, or sensitivity, quantifies the model's ability to identify all relevant instances within a dataset. For fraud detection systems, a high recall rate is vital as it ensures that most fraudulent transactions are detected, thus minimising the risk of financial loss and maintaining system integrity.

3.8.3 F1-Score

The f1-score is particularly useful in situations where the class distribution is imbalanced—a common scenario in fraud detection. It provides a balance between precision and recall by taking their harmonic mean, thus offering a single metric to assess model performance where both false positives and false negatives are crucial.

3.8.4 Accuracy

Accuracy measures the proportion of true results (both true positives and true negatives) among the total number of cases examined. It provides an overall indication of a model's effectiveness across all classes but can be misleading in imbalanced settings where the majority class dominates.

3.8.5 Confusion Matrix

The confusion matrix provides a visual and quantitative representation of a model's performance, delineating the number of true positives, false positives, true negatives, and false negatives. It offers comprehensive insight into the precision and recall of a model, facilitating detailed analysis of its predictive capabilities.

These metrics are systematically applied to each model during the evaluation phase. Each metric sheds light on different facets of model performance, aiding in a holistic assessment that is critical for deploying effective fraud detection systems in practice.

3.9 Transaction Visualiser: Augmenting Fraud Detection

In addition to the implementation of machine learning models for fraud detection, this study introduces a transaction visualiser—a dynamic and interactive tool designed to aid in the post-detection analysis of Ethereum transactions. This tool is particularly geared towards law enforcement officers and forensic analysts, enabling them to visually inspect and analyse the nature and sequence of transactions flagged as potentially fraudulent by the machine learning models. This offers real-world application from the output of the best-performing machine learning model.

3.9.1 Functional Overview

The transaction visualiser leverages the transparent nature of the Ethereum blockchain, extracting data directly from the network via the Etherscan API. It processes the transaction data associated with a specified Ethereum address, rendering an interactive graph that maps out the transactional relationships and flow of Ether (the cryptocurrency of Ethereum).

Features

- **Graphical Representation:** The visualiser creates nodes for each unique address and edges for transactions, with varying edge thickness to represent the transaction value.
- **Interactive Elements:** Users can interact with the graph to explore connections between different accounts, inspect transaction details, and trace the flow of funds.
- **Heuristic Analysis:** The tool incorporates heuristic indicators to flag patterns commonly associated with fraudulent behaviour, such as:
  - Unusually rapid succession of transactions, possibly indicative of automated scripts or money laundering activities.
  - Transactions involving large volumes of Ether, which may signify high-risk or high-value fraud.
  - Addresses that exhibit a high volume of activity within a short lifespan, potentially pointing to throwaway accounts used for deceptive purposes.

3.9.2 Application and Integration

The transaction visualiser serves as a complementary tool to the machine learning models, transitioning from detection to examination. Once a potential fraud is detected, investigators can utilise this tool to:

- **Explore Transaction Patterns:** Visualising complex chains of transactions to discern anomalous patterns that might not be apparent through traditional data analysis.
- **Contextualise Machine Learning Outputs:** Providing tangible context to the abstract predictions of machine learning models, aiding in the interpretation and justification of results.
- **Facilitate Investigative Workflows:** Offering an intuitive interface for non-technical stakeholders, such as law enforcement officers, to delve into blockchain analysis without the need for deep technical expertise in blockchain technology.

3.9.3 Implementation Details

Implemented in Python, the visualiser integrates requests to interact with the Etherscan API and the pyvis.network visualisation package to construct a Network object capable of producing the interactive graph. The system identifies and highlights significant transactions using heuristics based on time, value, and frequency of transactions, automatically flagged for further scrutiny. The output is an HTML file ("ethereum_transactions.html") that can be opened in any web browser, providing immediate access to the visualised network of transactions.

3.9.4 Integration with Machine Learning

This visualiser operates as an extension to the machine learning detection phase. Upon identification of suspicious activity by the models, the visualiser steps in to offer a granular, transaction-level view, facilitating a deeper investigation into the flagged activities. This approach not only bridges the gap between detection and investigation but also enhances the practical applicability of the research findings, rendering them actionable for real-world fraud detection and investigation scenarios.

3.10 Conclusion of Methodology

The methodology employed in this study represents a comprehensive and multifaceted approach to the detection of fraudulent transactions within the Ethereum blockchain. We began by meticulously detailing the dataset and executing thorough preprocessing steps to ensure the highest data quality and relevance. The application of dimensionality reduction techniques, such as correlation analysis and t-SNE visualisation, played a pivotal role in deconstructing the dataset's complexity and offering preliminary insights into its intricate structure.

We then carefully selected and implemented four promising machine learning models: Logistic Regression, Random Forest, Convolutional Neural Networks, and XGBoost. Each model was chosen for its established success in domains with similarities to our research context and for its capacity to provide diverse perspectives on the dataset's attributes and the latent patterns of fraudulent transactions.

To complement the analytical power of these models, we introduced an innovative transaction visualiser tool. This tool serves as a bridge between the detection capabilities of machine learning models and the practical investigative needs following a fraud alert. It provides an intuitive and interactive graphical interface that enables law enforcement officials and forensic analysts to visually dissect and analyse the nature of transactions identified as potential fraud.

Hyperparameter tuning was meticulously applied to the model with the strongest initial performance, demonstrating a strategic and resource-efficient approach to enhancing model precision. Adhering to established best practices, this step ensured that the chosen model was not only tailored to the complexities of Ethereum fraud detection but also fine-tuned to excel in generalising to new, unseen data.

The integration of the transaction visualiser into our methodology highlights the study's innovative spirit and its dedication to delivering actionable insights that extend beyond theoretical analysis. This tool augments the predictive strength of machine learning with real-world application, reinforcing the study's commitment to advancing both the academic field of blockchain security and the practical tools for financial safety.

In conclusion, the methodological rigour of this research contributes significantly to the field of Ethereum fraud detection, bridging the gap between data-driven detection and human-centric investigation.

Chapter 4

Results

4.1 Summary of Findings

In summary, the performance metrics across the models demonstrated a range of effectiveness, with the XGBoost (XGB) model outperforming the all the others. It exhibited the highest precision and recall rates, which are particularly significant in a domain where both detecting as many instances of fraud as possible (high recall) and maintaining a low rate of false alerts (high precision) are crucial. The Random Forest (RF) and Convolutional Neural Network (CNN) models also showed strong performances; however, the balanced metrics of the XGB model, especially its f1-score and accuracy, underline its suitability for further optimisation and application in the field of Ethereum fraud detection.

These findings illustrate the potential of sophisticated machine learning models to significantly enhance fraud detection systems in decentralised financial environments. The subsequent steps in this section will involve hyperparameter tuning of the XGB model to further refine its predictive capabilities.

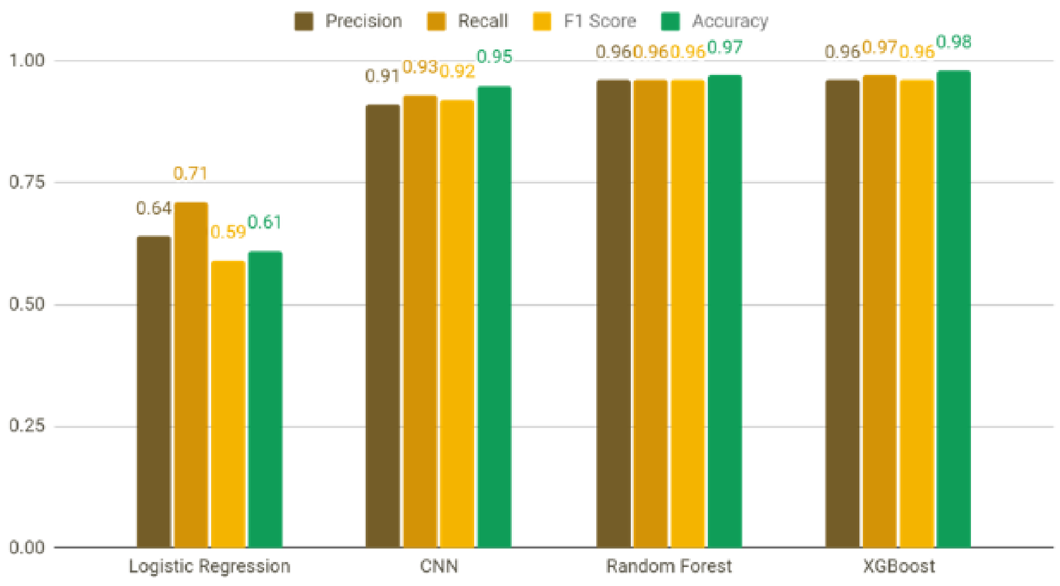

Figure 4.1: Comparative performance of the machine learning models explored in this study

4.2 Initial Comparative Analysis of Machine Learning Models for Ethereum Fraud Detection

The initial comparative analysis of machine learning models for Ethereum fraud detection provided comprehensive insights into the performance of each algorithm. This study evaluated four different models: Logistic Regression (LR), Random Forest (RF), XGBoost Classifier (XGB), and Convolutional Neural Network (CNN). These models were assessed based on their precision, recall, f1-score, and overall accuracy, offering a detailed perspective on their ability to classify fraudulent (labelled as '1') versus non-fraudulent (labelled as '0') transactions.

4.2.1 Logistic Regression (LR) Performance

The Logistic Regression model demonstrated a precision of 0.34 and a recall of 0.87 for detecting fraudulent transactions, resulting in an f1-score of 0.49. While the recall is relatively high, indicating a strong capability to detect most fraudulent transactions, the precision is considerably lower, suggesting a high number of false positives. The overall accuracy of the LR model was 61%, with the confusion matrix revealing that out of 422 fraudulent cases, 367 were correctly identified, while 55 were missed.

4.2.2 Random Forest (RF) Performance

The Random Forest model significantly outperformed the LR model, achieving a precision of 0.93 and a recall of 0.94 for fraudulent transactions, which resulted in an f1-score of 0.94. This model exhibited high accuracy at 97%, correctly identifying 398 out of 422 fraudulent transactions and only missing 24. For non-fraudulent transactions, it accurately classified 1517 out of 1547, with 30 false positives.

4.2.3 XGBoost Classifier (XGB) Performance

The XGBoost Classifier demonstrated the best performance among the models, with a precision of 0.94 and a recall of 0.95 for the fraudulent class, yielding an f1-score of 0.95. The model's accuracy was outstanding at 98%, correctly predicting 403 out of 422 fraudulent transactions and exhibiting a slightly better false-positive rate compared to the RF model.

4.2.4 Convolutional Neural Network (CNN) Performance

The Convolutional Neural Network showed promising results with a precision of 0.88 and a recall of 0.91 for the fraudulent class, leading to an f1-score of 0.90. The overall accuracy of the CNN model was 96%, slightly lower than that of the XGB model but still notably high. The CNN correctly identified 384 fraudulent transactions and incorrectly classified 38 as non-fraudulent. As shown in Figure A.1 in the Appendix, the training and validation loss curves display a converging nature, which indicates that the model was learning effectively and generalising well to unseen data. The closeness of the two curves by the end of training suggests that the model was not overfitting the dataset.

4.3 Model Selection Rationale

In the initial comparative analysis, several models were evaluated to determine their effectiveness in identifying fraudulent transactions within the Ethereum blockchain. While Logistic Regression (LR) provided a solid baseline, and both the Random Forest (RF) and Convolutional Neural Network (CNN) demonstrated promise, the XGBoost Classifier (XGB) emerged as the most suitable model for subsequent hyperparameter tuning. This choice was based on several factors that extend beyond the model's superior performance metrics:

- **Handling of Imbalanced Data:** XGBoost possesses built-in capabilities for managing imbalanced datasets, a prevalent issue in fraud detection where fraudulent transactions are significantly fewer than legitimate ones. The model's ability to prioritise the minority class enhances its performance in such scenarios.

- **Feature Importance Understanding:** XGBoost excels in providing interpretable results, including feature importance scores. This insight is invaluable in fraud detection, enabling investigators to understand which features most influence the model's decisions, thereby informing ongoing data collection and feature engineering efforts.

- **Scalability and Speed:** The scalability of XGBoost is crucial for processing large datasets. Optimised for both performance and speed, it can efficiently handle extensive transaction data, a critical factor given the voluminous nature of blockchain transactions.

- **Regularisation:** Incorporation of regularisation in XGBoost helps prevent overfitting—a vital feature in fraud detection, ensuring the model generalises effectively to new, unseen data.

- **Flexibility:** XGBoost's versatility in optimising various objective functions and accommodating different data structures makes it exceptionally adaptable. This flexibility is advantageous in the dynamic field of cryptocurrency, where transaction patterns and fraudulent techniques frequently evolve.

- **Gradient Boosting Framework:** As part of the gradient boosting model family, XGBoost is renowned for its high performance on structured data, such as transaction tables. Its advanced tree learning algorithms are adept at uncovering complex patterns that simpler models might overlook.

- **Cross-validation Support:** XGBoost's internal cross-validation mechanism permits performance evaluation without necessitating a separate validation set. This efficient data utilisation ensures that maximum data volume is available for training while still maintaining the model's capacity for self-validation.

In summary, the decision to advance with XGBoost for further hyperparameter tuning was motivated by its high performance, coupled with a suite of features that are well-suited to address the challenges of fraud detection within the Ethereum blockchain. The high performance, interpretability, and adaptability of the XGB model make it an ideal choice for this research's primary model of choice.

4.4 Hyperparameter Tuning

Hyperparameter tuning is an essential phase in the development of complex models such as XGBoost, which includes several parameters that can profoundly influence the model's effectiveness. To optimise these parameters, a systematic approach known as `GridSearchCV` was implemented. This method iteratively tests combinations of parameters to identify the most efficacious set. The parameters studied included `learning_rate`, `n_estimators`, `subsample`, `max_depth`, and `colsample_bytree`, with the primary goal to maximise the model's recall score. This ensures the highest number of fraudulent transactions are accurately identified, a crucial aspect in fraud detection.

The tuning process was executed on Google Colab's Python 3 compute engine, which provided the necessary computational resources to manage the extensive calculations involved. The procedure required approximately 22 minutes, highlighting the computational demands of exhaustive grid searches, particularly when combined with K-fold cross-validation. This extensive exploration is vital to ensure optimal model performance by discovering the best combination of parameters.

4.4.1 Considerations and Trade-offs in Parameter Selection

Learning Rate: Often termed as *eta*, this parameter controls the rate at which the model adapts to the problem. A range from 0.01 to 0.75 was selected to evaluate both gradual and aggressive learning strategies. While a lower learning rate might enhance generalisation and require more estimators, a higher rate expedites training but risks bypassing the optimal solutions.

Number of Estimators: This parameter determines the number of boosting rounds. The chosen range from 50 to 250 aims to balance model complexity against training efficiency, where too few estimators can lead to underfitting, and too many can cause overfitting.

Subsample: It specifies the fraction of samples used for fitting each tree. Values between 0.2 and 1.2 were considered to examine the effects of different levels of sample usage on the model's behaviour, including potential overfitting with rates above 1.

Max Depth: This controls the depth of each tree. A range from 3 to 6 allows the model to capture complexity without becoming overly specific to the training data, balancing the ability to model complex patterns and the potential to overfit.

Colsample_bytree: Sets the fraction of features used for each tree. The selected range from 0.3 to 1.2 tests the impact of feature sampling on model efficacy, aiming to increase computational speed and prevent overfitting by reducing feature usage.

These parameters influence the model's complexity, training duration, and performance. Complex models may yield better training accuracy but might overfit, whereas simpler models might not capture all data nuances but could be more potent on new data. The tuning process, therefore, navigates a balance between complexity and generalisability, with GridSearchCV facilitating a thorough exploration within the defined parameter ranges.

4.4.2 K-Fold Validation

K-fold cross-validation was utilised within the GridSearchCV framework to ensure the model's robustness and generalisability. By dividing the dataset into ten separate folds, this method allowed the model to be trained and validated ten times on different data subsets, enhancing the model's reliability and providing a comprehensive view of its expected performance in practical settings.

4.5 Tuned XGBoost Model Results

The tuning process resulted in identifying the best parameters for the XGBoost model which were found to significantly enhance the model's ability to detect fraudulent transactions.

Table 4.1: Optimal Hyperparameters found using exhaustive search via GridSearchCV

Hyperparameter	Value
ColSample by Tree	0.7
Learning Rate	0.1
Max Depth	6
Number of Estimators	250
Subsample	0.5

Tuned XGBoost Performance Utilising the optimised parameters, the XGBoost Classifier model achieved a recall score of 0.9885. This indicates that the model is capable of identifying approximately 98% of the fraudulent transactions. In the domain of fraud detection, such a high recall is crucial. Failing to detect fraudulent activities can lead to significant financial losses and undermine the trust in transaction systems. This performance underscores the model's efficacy in capturing nearly all fraudulent transactions, a key objective in the prevention of fraud within financial systems.

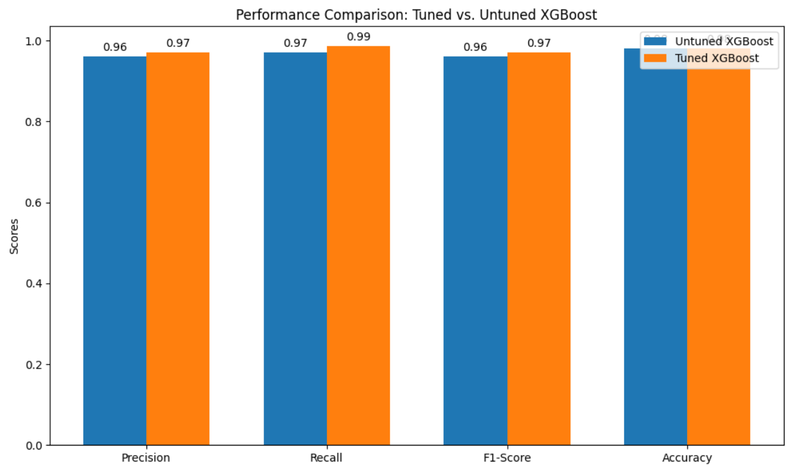

Figure 4.2: Performance comparison of Untuned (Blue) vs. Tuned (Orange) XGBoost models across various metrics such as Precision, Recall, F1-Score, and Accuracy. The optimisation is quite apparent here.

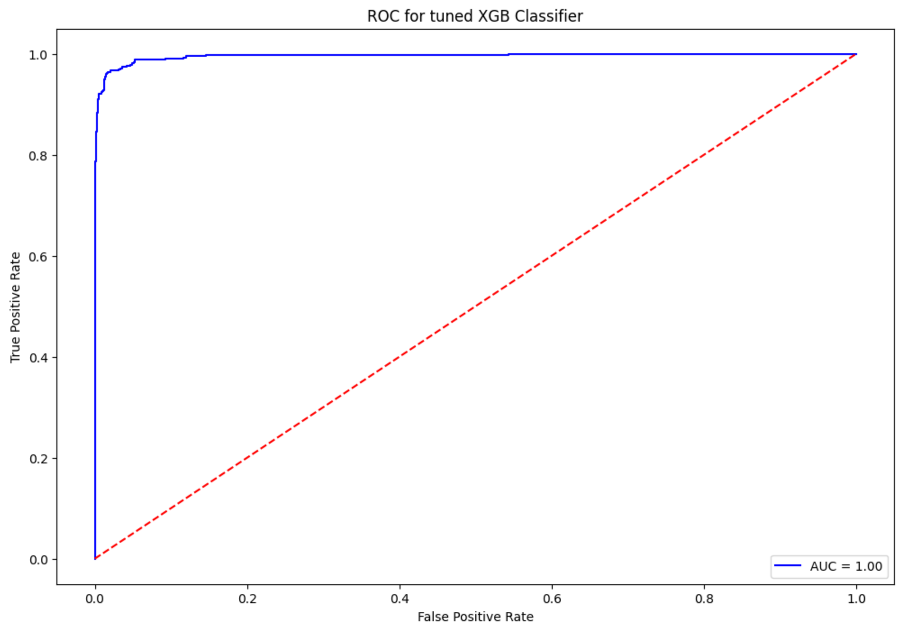

Figure 4.3: ROC curve of the tuned XGBoost model showing an AUC of 1.00.

ROC curve One of the key performance metrics, the area under the receiver operating characteristic (ROC) curve, reached an ideal score of 1.00, suggesting perfect classification by the model.

Figure 4.3 illustrates the ROC curve for the tuned XGBoost model. As evident from the curve, the model successfully discriminates between the fraudulent and non-fraudulent transactions with no overlap, indicating a high level of predictive accuracy.

The achievement of a 1.00 AUC score not only validates the effectiveness of the feature engineering and model optimisation strategies employed but also underscores the capability of advanced machine learning techniques in addressing complex challenges in blockchain-based systems.

4.6 Transaction Visualiser

The Transaction Visualiser developed for this dissertation offers an interactive, dynamic tool that has proven invaluable in dissecting the complex web of transactions associated with fraudulent activities on the Ethereum blockchain. Its intuitive interface and detailed visual representation provide a vivid examination of transaction flows, which is especially effective in scrutinising intricate fraud cases.

A particularly notable example involves the account `0xef57aa4b57587600fc209f345fe0a2687bc26985`, which our XGBoost model identified accurately as fraudulent. The visualiser elucidates the complex patterns of this account's transactions, notably connected with a questioned Initial Coin Offering (ICO) for WorldOfBattles (WoB). As shown in Figure 4.4, the account's transactions highlight victims (in red) and the perpetrators (in blue), with notably large transactions of 150 ETH each that were siphoned from investors, never to be returned.

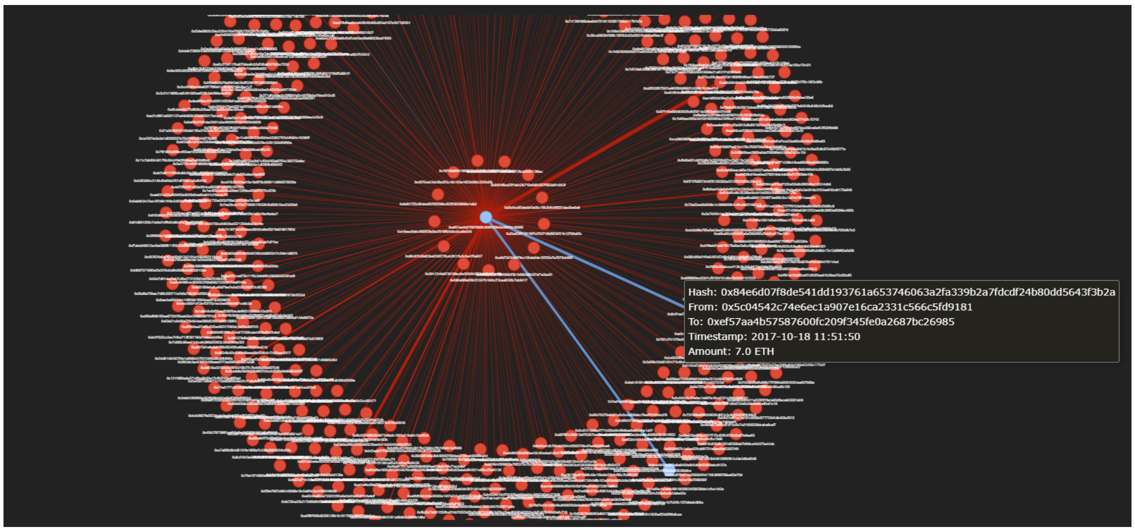
Hash: 0x84e6d07f8de541dd193761a653746063a2fa339b2a7fdcdf24b80dd5643f3b2a
From: 0x5c04542c74e6ec1a907e16ca2331c566c5fd9181
To: 0xef57aa4b57587600fc209f345fe0a2687bc26985
Timestamp: 2017-10-18 11:51:50
Amount: 7.0 ETH

Figure 4.4: Interactive visualisation of the fraudulent account's transactions

The visualiser's capabilities allow users to interact with the visualisation, enabling them to trace the path of each transaction, explore relationships between addresses, and identify clusters of activity that may indicate fraudulent behaviour. This tool not only aids in the immediate investigation of fraud reports but also helps refine predictive models by providing empirical evidence of fraud patterns.

Further investigation through the visualiser revealed that transactions originating from the flagged account were initially pooled and subsequently dispersed in a manner typical of money laundering schemes designed to obscure the origins of the funds. Particularly, two substantial transactions that failed to result in token distribution to investors are highlighted in Figure 4.5, marking a clear path of deceit and misappropriation of funds.

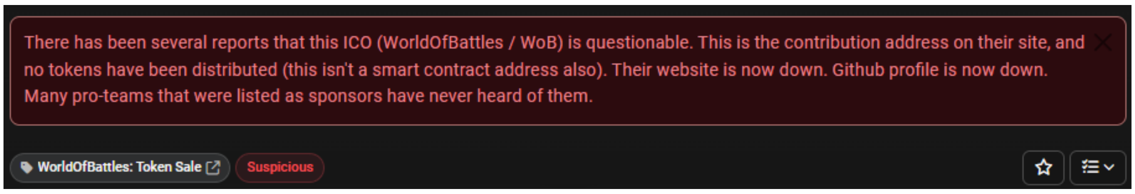
There has been several reports that this ICO (WorldOfBattles / WoB) is questionable. This is the contribution address on their site, and no tokens have been distributed (this isn't a smart contract address also). Their website is now down. Github profile is now down. Many pro-teams that were listed as sponsors have never heard of them.

Figure 4.5: Etherscan screenshot showing the fraudulent account

This visualiser thus serves not only as a tool for current investigations but also as a means to enhance the understanding and detection of fraudulent patterns, thereby supporting ongoing improvements in blockchain security.

4.7 Conclusion

In this study, the diligent application of machine learning models to the detection of fraudulent transactions within the Ethereum blockchain has produced significant insights. The XGBoost Classifier has emerged as the model with the best performance, exhibiting high precision and recall in identifying fraudulent transactions. Through the incorporation of hyperparameter tuning, the model's accuracy has been further refined, establishing a solid foundation for its practical deployment.

The Transaction Visualiser

Complementing the predictive models, the Transaction Visualiser has proven its merit as an analytical instrument. It provides vivid visual representations of the intricate networks of Ethereum transactions, proving indispensable not only in post-detection analysis but also in enhancing the interpretability and accessibility of complex blockchain data. This tool has shown to be particularly valuable for diverse stakeholders, including forensic analysts and law enforcement officials.

Integrated Approach to Fraud Detection

The integration of predictive analytics with visual tools, as illustrated throughout this dissertation, sets a new benchmark for fraud detection methodologies within the realm of digital finance. This combined approach underscores the transformative potential of machine learning and data visualisation technologies in combating fraud in the cryptocurrency domain. The insights gained and the methodologies developed here not only advance the academic field of blockchain security but also enhance practical applications in fraud detection.

Chapter 5

Discussion and Analysis

5.1 Discussion

5.1.1 Key Findings

This study conducted a meticulous examination of four sophisticated machine learning models—Logistic Regression (LR), Random Forest (RF), XGBoost Classifier (XGB), and Convolutional Neural Networks (CNN)—and their efficacy in detecting fraudulent accounts within the Ethereum blockchain. A significant finding was the superior performance of the XGBoost Classifier, which achieved the highest scores across several evaluation metrics including precision, recall, f1-score, and overall accuracy. This displays its capability and reliability in fraud detection scenarios.

An innovative aspect of this research was the development and application of the Transaction Visualiser tool. This tool has shown significant potential in augmenting analytical capabilities post-fraud detection, enabling an in-depth exploration of transaction patterns that may indicate fraudulent activities. It has proved especially useful in illustrating connections between accounts and transactions, providing an interactive means of tracing fund flows and potentially aiding in the understanding of complex fraud schemes, as demonstrated by the analysis of the fraudulent WoB account `0xef57aa4b57587600fc209f345fe0a2687bc26985`.

5.1.2 Implications for Practitioners and Policymakers

The findings from this study have meaningful implications for a range of stakeholders, particularly those involved in the security and regulatory aspects of blockchain technology.

For practitioners, the success of the XGBoost model underscores the importance of applying advanced machine learning techniques to the domain of fraud detection. Integrating such models could significantly enhance the efficiency and effectiveness of monitoring systems within blockchain networks. Furthermore, the Transaction Visualiser offers a practical tool that could be integrated into existing fraud detection systems to provide a more nuanced understanding of transactional data.

Policymakers could utilise these insights to shape regulations that promote the adoption of AI-driven tools, fostering a safer blockchain environment. The granular analysis facilitated by the Transaction Visualiser also emphasises the need for policies supporting transparency and traceability in blockchain transactions, which are crucial in deterring fraudulent activities.

Given the rapid evolution of fraudulent tactics, it is advised that practitioners and policymakers remain agile, continuously adapting and updating fraud detection mechanisms to stay ahead of potential threats. Additionally, collaboration between technology developers, regulatory bodies, and financial institutions is essential to establish comprehensive and proactive defences against fraud in the blockchain sector.

Chapter 6

Conclusions and Future Work

6.1 Conclusions and Future Work

The research provided valuable insights into the application of machine learning algorithms for detecting fraud within the Ethereum blockchain. The standout performance of the XGBoost model, in particular, has significant implications for the future of fraud detection in decentralised systems. The integration of the Transaction Visualiser tool further enhances the practicality of the research by providing a dynamic and detailed approach for analysing transaction patterns post-detection.

6.2 Major Contributions

This dissertation significantly advances the field of blockchain security by:

- Demonstrating the effectiveness of machine learning models, particularly the XGBoost Classifier, in detecting fraudulent transactions within the Ethereum blockchain. This model shows superior capability in both identifying fraudulent activities and operational efficiency.
- Developing and deploying the Transaction Visualiser—an innovative tool that analyses and interprets transaction data, thereby opening new avenues for post-fraud investigation and enhancing the understanding of transactional dynamics within the Ethereum network.
- Laying the groundwork for future research and practical applications in blockchain fraud detection, emphasising the necessity for adaptable and sophisticated tools to effectively combat financial fraud in increasingly complex blockchain ecosystems.

6.3 Recommendations for Future Research

This study highlights several key areas for further research that could substantially advance the field of fraud detection within blockchain technologies:

6.3.1 Enhanced Integration of Predictive Models and Visualisation Tools

Future studies should aim for seamless integration between the predictive capabilities of AI models and the visualisation tools for transaction analysis. Developing real-time fraud detection systems that automatically flag and visualise suspicious transaction patterns would greatly enhance efficiency and responsiveness. These systems should integrate the visualisation tool as a core component of the detection model, providing immediate and interactive analysis upon detection.

6.3.2 Real-time Fraud Detection

Advancing real-time fraud detection is crucial for minimising financial losses. Implementing streaming data models that can process transactions in real-time would allow for instantaneous responses, a necessity in the fast-paced environment of blockchain technologies.

6.3.3 Diversification of Data Sources

Broadening the data sources used in fraud detection models to include both on-chain and off-chain information could improve model accuracy. Future research should explore the integration of data such as IP addresses, transaction timings, and user behaviour patterns, while considering the ethical and privacy implications of such approaches.

6.3.4 Closer AI Integration with Transaction Visualisation

Integrating AI analysis directly into the transaction visualisation process would provide investigators with real-time, interactive, and intuitive visual feedback, enhancing decision-making processes during fraud investigations.

6.3.5 Autonomous Feedback Loops

Creating autonomous feedback loops where interactions with the visualisation tool directly inform model training could improve the accuracy and sophistication of fraud detection models over time.

6.3.6 Integration with Blockchain Platforms and Services

Engaging in collaborations with blockchain platforms and financial services can provide practical insights and ensure that the research remains relevant and grounded in real-world applications.

6.4 Limitations of the Study

The research faced several constraints that could impact the generalisability and scalability of the findings:

6.4.1 Computational Resources

Reliance on Google Colab for computational resources posed limitations on the complexity of the models trained and the scope of hyperparameter tuning, could have affected the refinement and optimisation of the models.

6.4.2 Dataset Size and Diversity

The relatively small size of the dataset used in this study may not adequately represent the diversity of the Ethereum transaction space. Future research would benefit from a larger and more varied dataset alongside added computation as complex AI models improve with more data and training resources available to them.

6.4.3 Generalisation to Other Blockchain Platforms

This study's focus was limited to the Ethereum blockchain; hence, the findings may not be applicable to other blockchain platforms, which could exhibit different transactional behaviours and fraud patterns.

6.4.4 Manual Integration Between Model Output and Visualisation

The current requirement for manual intervention to transition from model output to transaction visualisation reduces the process's efficiency and scalability. Automating this transition could significantly enhance operational effectiveness.

6.4.5 Dependence on External APIs

Dependence on external APIs like Etherscan poses risks related to data availability, rate limiting, and potential downtime, which could constrain the scalability and reliability of the Transaction Visualiser. Future work could host a Full Ethereum node and take transactions directly from the blockchain.

6.4.6 Scope of Transaction Analysis

Expanding the Transaction Visualiser to cover token transfers, smart contract interactions, and other complex transaction types could substantially increase its analytical capabilities possibly leading to better investigation outcomes

6.4.7 Data Privacy and Ethical Considerations

The study did not extensively address the data privacy and ethical considerations of utilising transaction data for fraud detection. Future research should address these concerns, ensuring compliance with privacy regulations and ethical standards.

Chapter 7

Reflection

7.0.1 Challenges in Data Acquisition

One of the main hurdles encountered during this dissertation was finding a high-quality labelled dataset. The lack of labelled data for fraudulent transactions within the blockchain significantly complicated the initial phases of the research. The complexities of blockchain data, coupled with the necessity for precise labelling to effectively train supervised machine learning models, emphasised the challenges in acquiring and utilising such datasets for academic research.

7.0.2 Data Preprocessing Demands

The dataset necessitated extensive preprocessing, which involved addressing a significant data imbalance. The predominance of non-fraudulent transactions necessitated the use of techniques like the Synthetic Minority Over-sampling Technique (SMOTE) to ensure the models did not develop a bias towards the majority class. This aspect of the project was both time-intensive as it needed to be heavily researched and meaningful, underscoring the importance of meticulous data preparation in achieving valid machine learning outcomes.

7.0.3 Integration of Interests

Despite these challenges, the project proved immensely rewarding as it merged my two fields of interest: Machine Learning and Blockchain Technology. It gave me the opportunity to apply theoretical knowledge in a practical context and deepened my understanding of both domains. The complexities of fraud detection on the blockchain presented a unique set of challenges that were intellectually stimulating and highly educational.

7.0.4 Learning and Skill Enhancement

This dissertation has substantially enhanced my skills in data handling, model tuning, and critical analysis. The hands-on experience with advanced machine learning techniques and blockchain analytics has been invaluable. It has also sharpened my ability to critically evaluate and employ technological solutions in addressing real-world problems.

7.0.5 Ethical and Privacy Concerns

Working with transaction data illuminated the critical importance of considering ethical and privacy issues in data science projects. This experience has heightened my awareness of the need to balance innovation with ethical responsibility, particularly when handling data that could impact individual privacy or financial security.

7.0.6 Future Aspirations

This project has inspired me to further explore the intersection of AI and blockchain technologies. Witnessing the practical impacts of my work has motivated me to continue researching and developing solutions that could make digital platforms safer and more transparent. This experience has solidified my interest in pursuing a career that contributes to advancing these technologies in an ethical and impactful manner.

Reflecting on this dissertation journey, the combination of challenges and learning opportunities has been profoundly transformative. It has not only shaped my academic and professional aspirations but has also instilled a deeper appreciation for the potential of combining machine learning with blockchain technology to address complex issues.

References

Aste, T., Tasca, P. and Di Matteo, T. (2017), 'Blockchain technologies: The foreseeable impact on society and industry', *Computer* **50**(9), 18–28.
URL: *https://discovery.ucl.ac.uk/id/eprint/10043048/1/Aste%5FBlockchainIEEE%5F600W%5Fv3.3%5FA.do*

Atzei, N., Bartoletti, M. and Cimoli, T. (2016), 'A survey of attacks on ethereum smart contracts', *IACR Cryptology ePrint Archive* **2016**, 1007.
URL: *https://link.springer.com/chapter/10.1007/978-3-662-54455-6%5F8*

Atzei, N., Bartoletti, M. and Cimoli, T. (2017), A survey of attacks on ethereum smart contracts (sok), *in* M. Maffei and M. Ryan, eds, 'Principles of Security and Trust', Springer Berlin Heidelberg.
URL: *https://link.springer.com/chapter/10.1007/978-3-662-54455-6%5F8#citeas*

Aziz, R., Baluch, M., Patel, S. et al. (2022), 'Lgbm: a machine learning approach for ethereum fraud detection', *International Journal of Information Technology* **14**, 3321–3331.
URL: *https://doi.org/10.1007/s41870-022-00864-6*

Bergstra, J. and Bengio, Y. (2012), 'Random search for hyper-parameter optimization', *Journal of Machine Learning Research* **13**, 281–305.
URL: *https://www.jmlr.org/papers/volume13/bergstra12a/bergstra12a.pdf?ref=broutonlab.com*

Bhatt, U. and Pandey, S. (2024), 'Empirical analysis of eip-3675: Miner dynamics, transaction fees, and transaction time'.

Breiman, L. (2001), 'Random forests', *Machine Learning* **45**(1), 5–32.
URL: *https://link.springer.com/article/10.1023/a:1010933404324*

Buterin, V. et al. (2014), 'A next-generation smart contract and decentralized application platform', *Ethereum white paper* **3**(37), 2–1.
URL: *https://ethereum.org/content/whitepaper/whitepaper-pdf/Ethereum%5FWhitepaper%5F-%5FButerin%5F2014.pdf*

Carneiro, N., Figueira, G. and Costa, M. (2017), 'A data mining based system for credit-card fraud detection in e-tail', *Decision Support Systems* **95**, 91–101.
URL: *https://doi.org/10.1016/j.dss.2017.01.002*

Catalini, C. and Gans, J. S. (2020), 'Some simple economics of the blockchain', **63**(7).
URL: *https://doi.org/10.1145/3359552*

Chandola, V., Banerjee, A. and Kumar, V. (2009), 'Anomaly detection: A survey', **41**(3).
URL: *https://doi.org/10.1145/1541880.1541882*

Chawla, N. V., Bowyer, K. W., Hall, L. O. and Kegelmeyer, W. P. (2002), ‘Smote: Synthetic minority over-sampling technique’, *Journal of Artificial Intelligence Research* **16**, 321–357.
URL: *https://doi.org/10.1613/jair.953*

Chen, T. and Guestrin, C. (2016), Xgboost: A scalable tree boosting system, Association for Computing Machinery, New York, NY, USA.
URL: *https://doi.org/10.1145/2939672.2939785*

Chen, W., Zheng, Z., Cui, J., Ngai, E., Zheng, P. and Zhou, Y. (2018), Detecting ponzi schemes on ethereum: Towards healthier blockchain technology, International World Wide Web Conferences Steering Committee.
URL: *https://doi.org/10.1145/3178876.3186046*

Claesen, M. and De Moor, B. (2015), ‘Hyperparameter search in machine learning’, *arXiv preprint arXiv:1502.02127* .
URL: *https://arxiv.org/abs/1502.02127*

Dannen, C. (2017), *Introducing Ethereum and Solidity*, Springer.
URL: *https://link.springer.com/book/10.1007/978-1-4842-2535-6*

Dormann, C. F., Elith, J., Bacher, S., Buchmann, C., Carl, G., Carré, G., García Marquéz, J. R., Gruber, B., Lafourcade, B., Leitão, P. J. et al. (2013), ‘Collinearity: a review of methods to deal with it and a simulation study evaluating their performance’, *Ecography* **36**(1), 27–46.
URL: *https://nsojournals.onlinelibrary.wiley.com/doi/pdf/10.1111/j.1600-0587.2012.07348.x*

Farrar, D. E. and Glauber, R. R. (1967), ‘Multicollinearity in regression analysis: The problem revisited’, *The Review of Economics and Statistics* pp. 92–107.
URL: *https://doi.org/10.2307/1937887*

Farrugia, S., Ellul, J. and Azzopardi, G. (2020), ‘Detection of illicit accounts over the ethereum blockchain’, *Expert Systems with Applications* **150**, 113318.
URL: *https://www.sciencedirect.com/science/article/pii/S0957417420301433*

Fujimoto, Y., Hayashi, T. and Saito, K. (2019), Anomaly detection for a water treatment system using unsupervised machine learning, *in* ‘2019 IEEE International Conference on Data Mining Workshops (ICDMW)’, pp. 461–468.
URL: *https://ieeexplore.ieee.org/document/8215783*

Guyon, I. and Elisseeff, A. (2003), ‘An introduction to variable and feature selection’, *Journal of Machine Learning Research* **3**, 1157–1182.
URL: *https://www.jmlr.org/papers/volume3/guyon03a/guyon03a.pdf?ref=driverlayer.com/web*

Guyon, I., Gunn, S., Nikravesh, M. and Zadeh, L. A. (2008), *Feature Extraction: Foundations and Applications*, Vol. 207, Springer.
URL: *https://books.google.co.uk/books?hl=en&lr=&id=FOTzBwAAQBAJ&oi=fnd&pg=PA1&dq=Feature+*

Hastie, T., Tibshirani, R. and Friedman, J. (2009), *The Elements of Statistical Learning*, Springer.
URL: *https://link.springer.com/book/10.1007/978-0-387-21606-5*

He, H. and Garcia, E. A. (2009), 'Learning from imbalanced data', *IEEE Transactions on Knowledge and Data Engineering* **21**(9), 1263–1284.
URL: *https://ieeexplore.ieee.org/document/5128907*

Hosmer, D. W., Lemeshow, S. and Sturdivant, R. X. (2013), *Applied Logistic Regression*, Wiley.
URL: *https://books.google.co.uk/books?hl=en&lr=&id=bRoxQBIZRd4C&oi=fnd&pg=PR13&dq=Applied+*

Hu, Sihao, e. a. (2023), 'Bert4eth: A pre-trained transformer for ethereum fraud detection'. https://arxiv.org/abs/2303.18138.
URL: *https://arxiv.org/abs/2303.18138*

Huh, S., Cho, S.-H. and Kim, S. (2017), Managing iot devices using blockchain platform, *in* '2017 19th International Conference on Advanced Communication Technology (ICACT)', IEEE, pp. 464–467.
URL: *https://ieeexplore.ieee.org/abstract/document/7890132?casa%5Ftoken=zqW4bK2%5FphoAAAAA:8ciV*

James, G., Witten, D., Hastie, T. and Tibshirani, R. (2013), *An Introduction to Statistical Learning*, Springer.
URL: *https://engineering.nyu.edu/sites/default/files/2021-02/CS-GY%206923%20Machine%20Learning%20-%20Kannan.pdf*

Jeni, L. A., Cohn, J. F. and De La Torre, F. (2013), Facing imbalanced data–recommendations for the use of performance metrics, *in* '2013 Humaine Association Conference on Affective Computing and Intelligent Interaction', pp. 245–251.
URL: *https://ieeexplore.ieee.org/document/6681438*

Kohavi, R. (1995), A study of cross-validation and bootstrap for accuracy estimation and model selection, *in* 'Ijcai', Vol. 14, pp. 1137–1145.
URL: *https://www.researchgate.net/profile/Ron-Kohavi/publication/2352264%5FA%5FStudy%5Fof%5FCross Validation%5Fand%5FBootstrap%5Ffor%5FAccuracy%5FEstimation%5Fand%5FModel%5FSelection/links/02 Study-of-Cross-Validation-and-Bootstrap-for-Accuracy-Estimation-and-Model-Selection.pdf*

Krizhevsky, A., Sutskever, I. and Hinton, G. E. (2012), Imagenet classification with deep convolutional neural networks, *in* F. Pereira, C. Burges, L. Bottou and K. Weinberger, eds, 'Advances in Neural Information Processing Systems', Vol. 25, Curran Associates, Inc.
URL: *https://proceedings.neurips.cc/paper%5Ffiles/paper/2012/file/c399862d3b9d6b76c8436e924a68c45b-Paper.pdf*

Kuhn, M. and Johnson, K. (2013), *Applied Predictive Modeling*, Springer.
URL: *https://link.springer.com/book/10.1007/978-1-4614-6849-3*

Kutner, M. H., Nachtsheim, C. J., Neter, J. and Li, W. (2004), *Applied Linear Statistical Models*, McGraw-Hill Irwin.
URL: *https://users.stat.ufl.edu/ winner/sta4211/ALSM%5F5Ed%5FKutner.pdf*

Little, R. J. and Rubin, D. B. (2002), *Statistical Analysis with Missing Data*, John Wiley & Sons.
URL: *https://toc.library.ethz.ch/objects/pdf03/e01%5F978-0-470-52679-8%5F01.pdf*

Liu, S., Cui, B. and Hou, W. (2023), A survey on blockchain abnormal transaction detection, *in* 'International Conference on Blockchain and Trustworthy Systems', Springer Nature Singapore, pp. 211–225.
URL: *https://link.springer.com/chapter/10.1007/978-981-99-8101-4%5F15*

Luu, L., Velner, Y., Teutsch, J. and Saxena, P. (2017), Smartpool: Practical decentralized pooled mining, *in* '26th {USENIX} Security Symposium ({USENIX} Security 17)', pp. 1409–1426.
URL: *https://www.usenix.org/conference/usenixsecurity17/technical-sessions/presentation/luu*

Ngai, E., Hu, Y., Wong, Y., Chen, Y. and Sun, X. (2011), 'The application of data mining techniques in financial fraud detection: A classification framework and an academic review of literature', *Decision Support Systems* **50**(3), 559–569.
URL: *https://doi.org/10.1016/j.dss.2010.08.006*

O'Brien, R. M. (2007), 'A caution regarding rules of thumb for variance inflation factors', *Quality & Quantity* **41**(5), 673–690.
URL: *https://doi.org/10.1007/s11135-006-9018-6*

Paschen, J. (2022), 'Application of machine learning algorithms in blockchain and cryptocurrency fraud detection: An empirical analysis', *European Journal of Criminology* .
URL: *https://doi.org/10.1007/s41870-022-00864-6*

Pedregosa, F. et al. (2011), 'Scikit-learn: Machine learning in python', *Journal of Machine Learning Research* **12**, 2825–2830.
URL: *https://www.jmlr.org/papers/volume12/pedregosa11a/pedregosa11a.pdf?ref=https:/*

Perez, D., Livshits, B. and Markov, I. (2021), 'Smart contract vulnerabilities: Does anyone care?', *arXiv preprint arXiv:2102.04444* .
URL: *https://allquantor.at/blockchainbib/pdf/perez2019smart.pdf*

Powers, D. M. (2011), 'Evaluation: from precision, recall and f-measure to roc, informedness, markedness & correlation', *Journal of Machine Learning Technologies* **2**(1), 37–63.
URL: *https://arxiv.org/abs/2010.16061*

Robinson, R. (2017), 'The new digital wild west: regulating the explosion of initial coin offerings', *Tenn. L. Rev.* **85**, 897.
URL: *https://heinonline.org/HOL/LandingPage?handle=hein.journals/tenn85&div=28&id=&page=*

Tan, Q., Zhang, X., Huang, X., Chen, H., Li, J. and Hu, X. (2024), 'Collaborative graph neural networks for attributed network embedding', *IEEE Transactions on Knowledge and Data Engineering* **36**(3), 972–986.
URL: *https://ieeexplore.ieee.org/abstract/document/10195214?casa%5Ftoken=e50M0OTABWoAAAAA:804v*

Tibshirani, R. (2018), 'Regression Shrinkage and Selection Via the Lasso', *Journal of the Royal Statistical Society: Series B (Methodological)* **58**(1), 267–288.
URL: *https://doi.org/10.1111/j.2517-6161.1996.tb02080.x*

Trozze, A. et al. (2022), 'Cryptocurrency and future financial crime: A systematic review', *Crime Science* **11**(1), 1.
URL: *https://crimesciencejournal.biomedcentral.com/articles/10.1186/s40163-021-00163-8*

Vujičić, D., Jagodić, D. and Randić, S. (2018), Blockchain technology, bitcoin, and ethereum: A brief overview, *in* '2018 17th International Symposium INFOTEH-JAHORINA (INFOTEH)', IEEE, pp. 1–6.
URL: *https://ieeexplore.ieee.org/abstract/document/8345547*

Zarpala, L. and Casino, F. (2021), 'A blockchain-based forensic model for financial crime investigation: the embezzlement scenario', *Digital Finance* **3**, 301–332.
URL: *https://doi.org/10.1007/s42521-021-00035-5*

Zięba, M., Świątek, J. and Świątek, P. (2016), 'The use of logistic regression in the detection of business cycles in poland', *Quality & Quantity* **50**, 509–524.
URL: *https://www.researchgate.net/profile/Tomasz-Pisula/publication/302588543%5FSTATISTICAL%5FME METHODS-OF-THE-BANKRUPTCY-PREDICTION-IN-THE-LOGISTICS-SECTOR-IN-POLAND-AND-SLOVAKIA.pdf*

Appendix A

Appendix

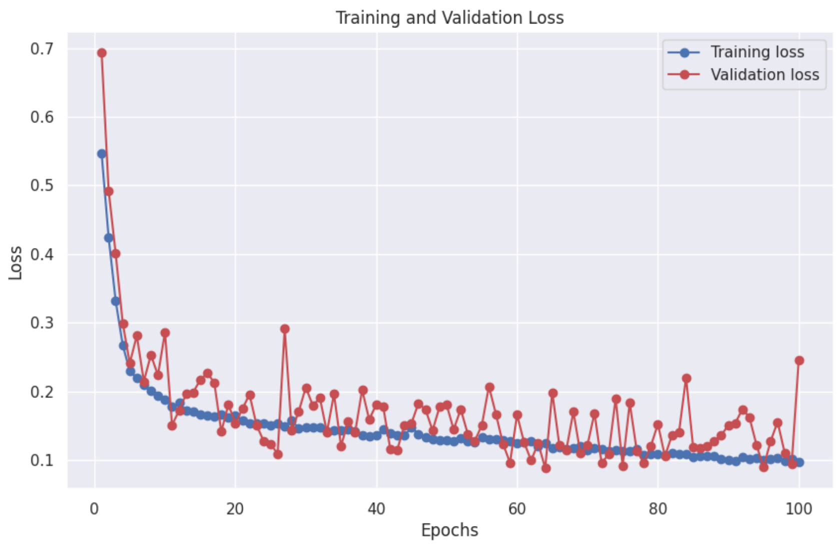

Figure A.1: Training and validation loss curves for the CNN model.

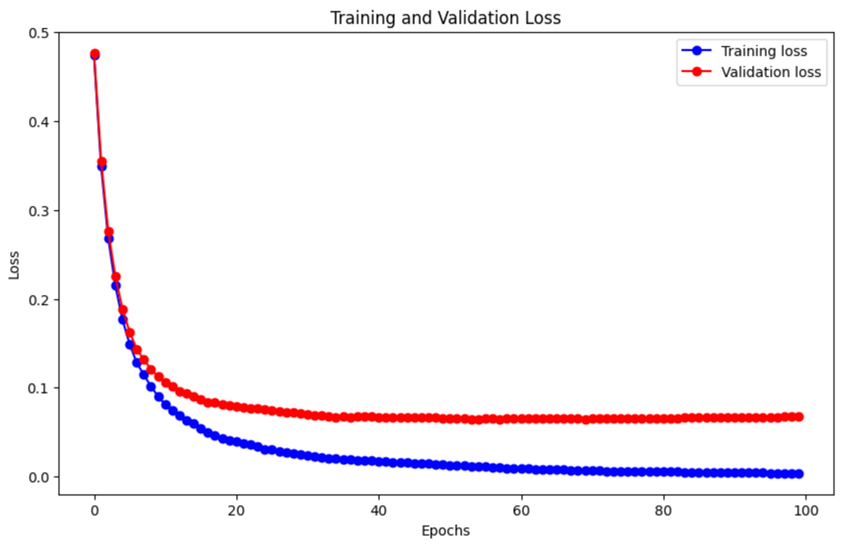

Figure A.2: Loss curves for the untuned XGBoost model.

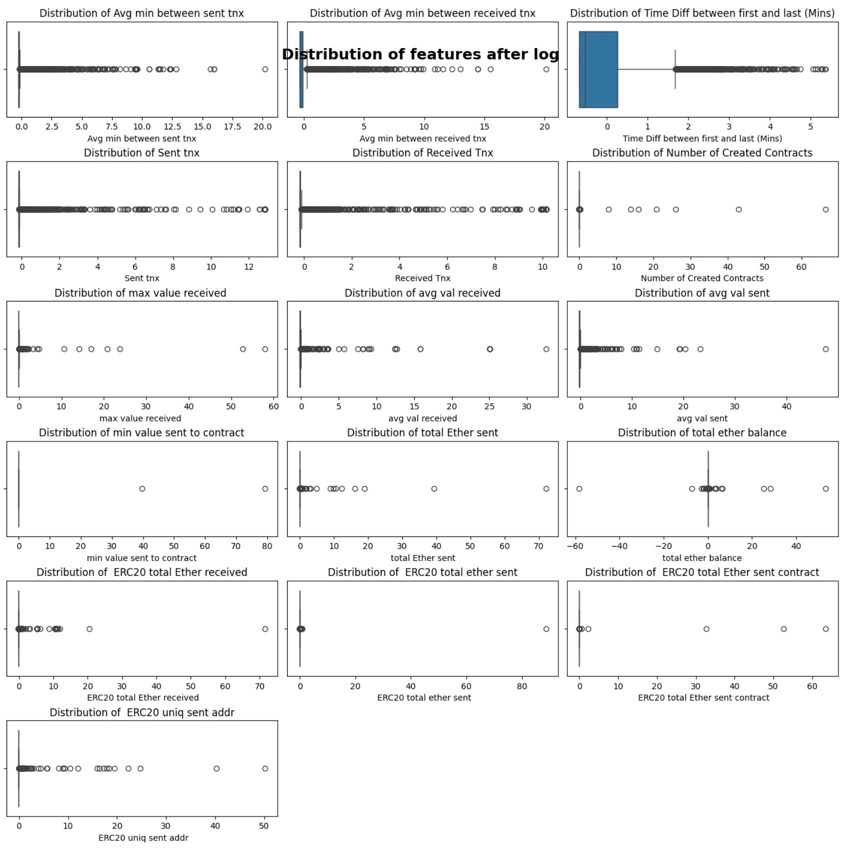

Figure A.3: Dataset distribution of features after log transformation.

Table A.1: Summary of Dataset Columns before Pre-processing

#	Column	Non-Null Count	Dtype	
—	——	—————	——	
0	Index	9841 non-null	int64	
1	Address	9841 non-null	object	
2	FLAG	9841 non-null	int64	
3	Avg min between sent tnx	9841 non-null	float64	
4	Avg min between received tnx	9841 non-null	float64	
5	Time Diff between first and last (Mins)	9841 non-null	float64	
6	Sent tnx	9841 non-null	int64	
7	Received Tnx	9841 non-null	int64	
8	Number of Created Contracts	9841 non-null	int64	
9	Unique Received From Addresses	9841 non-null	int64	
10	Unique Sent To Addresses	9841 non-null	int64	
11	min value received	9841 non-null	float64	
12	max value received	9841 non-null	float64	
13	avg val received	9841 non-null	float64	
14	min val sent	9841 non-null	float64	
15	max val sent	9841 non-null	float64	
16	avg val sent	9841 non-null	float64	
17	min value sent to contract	9841 non-null	float64	
18	max val sent to contract	9841 non-null	float64	
19	avg value sent to contract	9841 non-null	float64	
20	total transactions (including tnx to create contract	9841 non-null	int64	
21	total Ether sent	9841 non-null	float64	
22	total ether received	9841 non-null	float64	
23	total ether sent contracts	9841 non-null	float64	
24	total ether balance	9841 non-null	float64	
25	Total ERC20 tnxs	9012 non-null	float64	
26	ERC20 total Ether received	9012 non-null	float64	
27	ERC20 total ether sent	9012 non-null	float64	
28	ERC20 total Ether sent contract	9012 non-null	float64	
29	ERC20 uniq sent addr	9012 non-null	float64	
30	ERC20 uniq rec addr	9012 non-null	float64	
31	ERC20 uniq sent addr.1	9012 non-null	float64	
32	ERC20 uniq rec contract addr	9012 non-null	float64	
33	ERC20 avg time between sent tnx	9012 non-null	float64	
34	ERC20 avg time between rec tnx	9012 non-null	float64	
35	ERC20 avg time between rec 2 tnx	9012 non-null	float64	
36	ERC20 avg time between contract tnx	9012 non-null	float64	
37	ERC20 min val rec	9012 non-null	float64	
38	ERC20 max val rec	9012 non-null	float64	
39	ERC20 avg val rec	9012 non-null	float64	
40	ERC20 min val sent	9012 non-null	float64	
41	ERC20 max val sent	9012 non-null	float64	
42	ERC20 avg val sent	9012 non-null	float64	
43	ERC20 min val sent contract	9012 non-null	float64	
44	ERC20 max val sent contract	9012 non-null	float64	
45	ERC20 avg val sent contract	9012 non-null	float64	
46	ERC20 uniq sent token name	9012 non-null	float64	
47	ERC20 uniq rec token name	9012 non-null	float64	
48	ERC20 most sent token type	7144 non-null	object	
49	ERC20_most_rec_token_type	8970 non-null	object	
48	ERC20 most sent token type	7144 non-null	object	
49	ERC20_most_rec_token_type	8970 non-null	object	

Table A.2: Columns identified to be removed after correlation filtering

index	feature_1	feature_2	Absolute Correlat
15	ERC20 uniq rec contract addr	ERC20 uniq rec token name	1.0
19	ERC20 max val sent	ERC20 avg val sent	1.0
4	max val sent to contract	total ether sent contracts	1.0
18	ERC20 min val sent	ERC20 avg val sent	1.0
17	ERC20 min val sent	ERC20 max val sent	1.0
9	ERC20 total Ether received	ERC20 max val rec	1.0
11	ERC20 total ether sent	ERC20 min val sent	1.0
12	ERC20 total ether sent	ERC20 max val sent	1.0
13	ERC20 total ether sent	ERC20 avg val sent	1.0
3	max val sent to contract	avg value sent to contract	0.95
5	avg value sent to contract	total ether sent contracts	0.95
10	ERC20 total Ether received	ERC20 avg val rec	0.86
16	ERC20 max val rec	ERC20 avg val rec	0.86
1	Received Tnx	total transactions (including tnx to create contract	0.81
20	ERC20 uniq sent token name	ERC20 uniq rec token name	0.79
14	ERC20 uniq rec contract addr	ERC20 uniq sent token name	0.79
6	total Ether sent	total ether received	0.77
7	Total ERC20 tnxs	ERC20 uniq sent addr	0.73
0	Sent tnx	total transactions (including tnx to create contract	0.73
8	Total ERC20 tnxs	ERC20 uniq rec addr	0.72
2	min value sent to contract	avg value sent to contract	0.7

Model	**Precision (Class 1)**	**Recall (Class 1)**	**F1-Score (Class 1)**	**Accu**
Logistic Regression (LR)	0.34	0.87	0.49	61%
Random Forest (RF)	0.93	0.94	0.94	97%
XGBoost Classifier (XGB)	0.94	0.95	0.95	98%
Convolutional Neural Network (CNN)	0.88	0.91	0.90	96%

Table A.3: Comparative performance of machine learning models in Ethereum fraud detection.

Python Packages and APIs

The following Python packages used in the project:

```
import requests
from pyvis.network import Network
import datetime
import numpy as np
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Conv1D, MaxPooling1D, Flatten,
                                    Dense, Dropout
from tensorflow.keras.optimizers import Adam
from tensorflow.keras.utils import plot_model
from sklearn.metrics import classification_report, confusion_matrix
from sklearn.decomposition import PCA
from sklearn.manifold import TSNE
from sklearn.preprocessing import MinMaxScaler, StandardScaler
from sklearn.model_selection import train_test_split, GridSearchCV
from sklearn.ensemble import RandomForestClassifier
from sklearn.linear_model import LogisticRegression
from imblearn.over_sampling import SMOTE
from sklearn.metrics import confusion_matrix, roc_auc_score,
                             roc_curve, auc, classification_report
import matplotlib.pyplot as plt
import pandas as pd
import seaborn as sns
import plotly.express as px
import xgboost as xgb
import pickle
import warnings
```