Exploring Comprehensive Analytical Approaches for Blockchain Network Data Fraud Detection

MSc Research Project

Sonali Kadam
Student ID: 22210491

School of Computing
National College of Ireland

Supervisor: Michael Prior

National College of Ireland

MSc Project Submission Sheet

School of Computing

Student Name: .....Sonali Kadam .....

Student ID: ...x22210491.....

Programme: .....MSc cyber security..... **Year:** .....2023-24.....

Module: MSc Research Project

Supervisor: Michael Prior

Submission Due Date: 11-08-2024

Project Title: Exploring Comprehensive Analytical Approaches for blockchain network data Fraud Detection

Word Count: ...9986..... **Page Count**.....25.....

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.
ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature: .....Sonali Kadam.....

Date: .....

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST

Attach a completed copy of this sheet to each project (including multiple copies)	□
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	□
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	□

Assignments that are submitted to the Programme Coordinator Office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

Abstract

This research focuses on logical analysis of Blockchain network transactions by using the concepts of Machine Learning incorporated to improve the accuracy & security aspects in financial systems. In meeting these research questions, this study adopts sturdy models like linear regression, random forest, and anomaly detection to deal with problems arising from transactions and, more specifically, fraud in the cryptocurrency space.

Chapter 1 introduction, this chapter presents the importance of focussing on blockchain technology in the context of the financial industry, explaining the possible impact on the transactional phase. It presents the rescission objectives; These are the research questions: While building the predictive models, and investigating the security in the blockchain networks.

Chapter 2 moves around related works where various literatures on blockchain technology and more specifically, the application of machine learning in blockchain, shall be reviewed. This is done by presenting different methodologies in predictive modelling and presenting a rationale in the use of anomaly detection when screening for fraud, thus laying down a conceptual background of the research work.

Chapter 3, methodology, describes the methods used in developing the research, data acquisition, data preparation, and selection of the machine learning algorithm. This is a basic approach to the organisational goals of linear regression, random forests, and anomaly detection algorithms and provides the foundation for analysis.

Chapter 4 design specifications this chapter outlines the process followed in the analysis of the transaction datasets through various data transformations on the datasets. Meaning it comprises data preprocessing and cleaning where there is the management of missing values, feature engineering and exploratory data analysis that is a way of preparing data for the subsequent modelling by ensuring high data quality.

Chapter 5 implementation, entails advancement in explaining the process of applying the adopted machine learning techniques to the blockchain data. Also described are how the different libraries were applied, the characteristics of the datasets employed, and the assessment of how well models perform, all of which show how well the strategies applied performed in this context.

Chapter 6 evaluation evaluates the efficiency of the models, which predict the commitment of the transactions, and the goal of processing the transactions within the Blockchain related network. It uses actual and predicted values to draw its conclusions, reviews the scalability issue and the necessity of anomaly detection for improving security of financial transactions.

Chapter 7 conclusion and future work provides a brief on the main conclusions on the given subtopic and its significance in the future of blockchain analytics. It relates to the research objectives and describes the scope of the study's limitations while also providing a glimpse into future work, which requires extended and enriched data sets, greater processing immediacy.

Acknowledgement

First and foremost, I would like to express my deepest gratitude to my supervisor, for their invaluable guidance, unwavering support, and insightful feedback throughout the course of this dissertation. Their expertise and encouragement have been instrumental in shaping this research and bringing it to fruition. I extend my heartfelt thanks to the faculty and staff of the Department at University, whose knowledge and resources have greatly contributed to my academic journey. Special thanks to group members for their assistance and advice. I am also grateful to my colleagues and fellow students, especially Colleagues, for their camaraderie, discussions, and moral support during this project. Their perspectives and collaboration have enriched my understanding and helped me overcome various challenges.

My sincere appreciation goes to my family and friends for their unwavering support and encouragement. Their patience and belief in me have been a constant source of motivation and strength throughout this journey. Finally, I acknowledge the various sources of data and libraries used in this research, as well as the open-source community, whose contributions have made this work possible.

Thank you all for your support and encouragement. This dissertation would not have been possible without each of you.

Table of Contents

Chapter 1: Introduction

1.1 Introduction

Blockchain is an emerging technology that is characterised by decentralisation and immutability that revolutionises the method of digital transactions. It is therefore a form of an accounting database that maintains and verifies transactions over the Internet in a decentralised manner. The application of this technology eliminates the chances of a third party being involved, promotes openness, and involves the use of cryptography. Blockchain is a distributed public ledger introduced with the concept of Bitcoin. It has now extended to other sectors, such as manufacturing, medical records, and DeFi. The capacity for providing trust in an emergent state of "trustlessness" remains a key driver of advances throughout the specialised fields, making it a point of emphasis for developing industries in the digital age.

1.2 Background

Blockchain technology was inspired by a group of inventors using the pseudonym Satoshi Nakamoto in 2008 through Bitcoin. It is a shared, permanent, secure, and transparent digital ledger for recording information (Ducrée, 2022). It is decentralised and each node is a participant of the system, which eliminates the opportunities for fraud attacks typical for centralised systems. Looking at the success of Bitcoin, which is the first and the most popular use case of the blockchain, it proves that this technology is viable and has significant benefits which contributed to industry outreach and proliferation of applications. Objective of this research is to enhance this growing field by innovating and employing advanced analysis procedures for blockchain data to address key technological challenges such as performance, usability, and security.

Some years after the establishment of the first blockchain technology for cryptocurrency, new fields were discovered and adapted to blockchain technology including supply chain, health, and financial among others. This has exposed several key issues as a result of the high adoption rate. The scalability factor still persists to be a problem, where more transactions are processed, and more time is taken to complete more charges incurred (Dutta *et al.* 2020). Security is also an issue in the networks through which the blockchain system operates with threats such as double spending attacks and fraud. Hence more investigation is done on improved complex analysis methods to help better evaluate blockchain data through machine learning and AI concepts. These methods can bring more detailed information about the transaction and consequently predict the general trend of the market as well as the presence of security weaknesses.

1.3 Motivation for research

This research was inspired by the fact that blockchain is revolutionising many societies' economic models and virtual communications. Various issues that are related to the technology such as scalability, transaction settlement, and network security are often considered as key research questions as the technology develops. Growth and development of blockchain have been very progressive, and therefore, there is a need for superior analytical techniques for the assessment of the data generated within the blockchain network, especially the Blockchain. Therefore, this study's objectives are to use predictive modelling, scalability assessment, and anomaly detection approaches to address these issues and improve the effectiveness and reliability of the blockchain network. It is believed that the findings of this study can be of immense value in enriching the existing theory as well as practical applications in the field of blockchain technology in the future development of digital transactions and decentralised systems.

1.4 Aim and objective

Aim

Aim of this research is to develop and apply advanced analytical methodologies for the comprehensive evaluation of blockchain network data.

Objectives

- To analyse patterns and trends in blockchain network data to understand the market dynamics and transaction behaviours
- To develop predictive models using machine learning to forecast key blockchain metrics such as market price and transaction volume
- To assess the scalability and efficiency of the blockchain network through computational modelling and performance evaluation
- To enhance network security by detecting anomalies and potential threats within blockchain transactions using advanced analytical techniques

1.5 Research question

- What patterns and trends exist in blockchain network data that can provide insights into market dynamics and transaction behaviors, and which machine learning techniques, computational models, and advanced analytical methods are most effective for forecasting key blockchain metrics, assessing scalability and efficiency, and detecting anomalies and potential threats to enhance network security?

1.6 Hypothesis

H0 (Null hypothesis): There is no significant relationship between the patterns and trends identified in blockchain network data and market dynamics.

H1 (Alternate hypothesis): There is a significant relationship between the patterns and trends identified in blockchain network data and market dynamics.

1.7 Structure of report

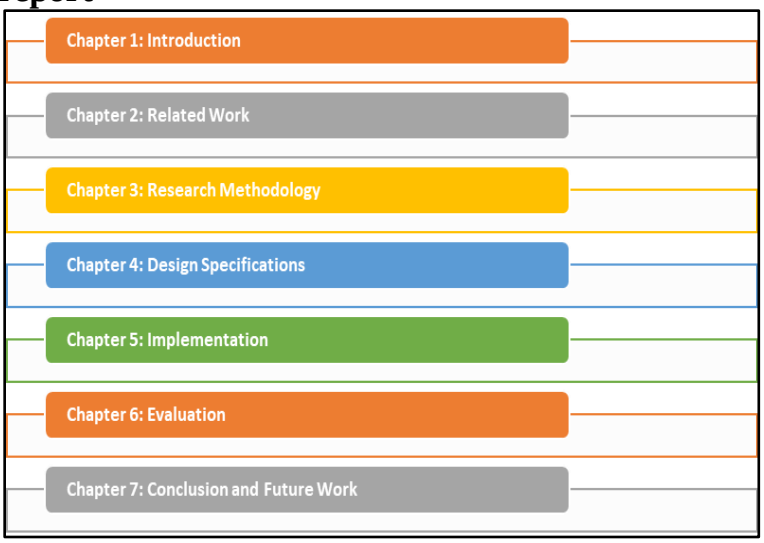

Figure 1.6: Research structure

Chapter 2: Related work

2.1 Introduction

This literature review provides an analysis of the development and current status of blockchain systems. These are data acquisition and preparation, supervised and unsupervised machine learning for predictive analytics, scalability, and security on blockchains. The review of the areas also brings out the fact that more emphasis is laid on conventional approaches and thus calls for the development of more complex methods of analysing the data. It compares it with the conventional centralised systems, and, gives a notion about the strengths and weaknesses of blockchain over those. The below-mentioned gaps and challenges highlighted in the literature present more research avenues to improve the efficiency and credibility of blockchain technology.

2.2 Historical development of blockchain technology

Blockchain can be traced back to the winning of digital currency called Bitcoin by the reputed personality Satoshi Nakamoto in 2008 (Ducrée, 2022). Nakamoto first defined bitcoin in his work as a new electronic currency for the Internet that works as a peer-to-peer network for processing financial transactions. The primary disruption was the blockchain, a spread ledger that documents all transactions in a network of computers, making it transparent, secure and tamper-proof (Sambana, 2021). Blockchain uses a consensus model called Proof of Work (PoW), the process in which miners solve algorithms to ensure that transactions are valid to be grouped in blocks. It also reinforces the network's security and compensates the miners via the issuance of new bitcoins and other transaction fees.

Arrival of Ethereum in 2015 by the founder Vitalik Buterin was a further leap, which made possible smart contracts, meaning legal documents which are automatically executed when certain conditions are inserted into coded words. This paved the way for other uses of blockchain because aside from financial-related services, more complex and automated processes can now be done in different industries. This flexibility encouraged the creation of decentralized applications that are applications which rely on blockchain and therefore do not require a central power. In healthcare, blockchain has been applied to protect patient data from hackers and enhance patients' records sharing among different health facilities. The financial services industry has adopted blockchain, for faster cross-border payments and the enhancement of processes such as KYC and AML (Zhang, 2020). Decentralized Finance (DeFi) has also portrayed that Blockchain can redesign financial services by making them DeFi. These advancements demonstrate how the use cases of blockchain are not limited to finance or as a store of value but have successfully travelled through the fence of industrial boundaries and created increased value proposition solutions for distinct ranges of problems.

2.3 Thematic analysis

2.3.1 Data selection and preprocessing for blockchain analysis

Data selection and preprocessing is the fundamental step in performing any advanced data analysis on blockchain data. Blockchain networks produce significant records in the form of data to reflect every transaction and the activity of the network (Chattu, 2021). Selecting relevant data means choosing the right blocks and the most appropriate transactions along with the related metadata. This process is essential since it determines the quality of the further analysis. Different types of data sources include public ones, which can be whatever the blockchain records, APIs, and blockchain explorers with higher detail and ease of use.

Data preparation is necessary to minimize issues related to the data's quality, structure, and fit for its intended use. That comprises data cleansing which helps in the elimination of wrong data or

data which are duplicates, formatting common ways, and dealing with the missing records. Because of this, since blockchain is decentralized, it can be difficult to maintain data integrity at nodes. Normalization and tokenization are some of the procedures used to prepare data for analysis or in other words transform the data into an analysable form. There is also a possibility that some data types like time series data that is typical within blockchain may need preliminary data operations such as resampling and smoothing to be applicable.

2.3.2 Predictive modelling with machine learning

Machine learning in general and predictive modelling in particular is well suited to analyse data from blockchains. These models could predict the trends in blockchain aspects like prices, volumes or activities in the market (Shafay *et al.* 2023). Regression techniques, time series analysis, and artificial neural network models are used most often as these are the main types of predictive models. For instance, the time series forecasting for the market price of cryptocurrencies in the future and classification for fraud detection.

It starts with feature selection, whereby the most appropriate attributes to obtain from the analysed blockchain data are determined. These features may comprise the frequency of transactions, the number of transactions, the value of transactions and delay in networks. The chosen attributes are then used to develop the machine learning models on patterns and relations using historical data. This model aims for Mean Absolute Error (MAE), Root Mean Squared Error (RMSE), and accuracy as its evaluation metrics. Problems with predictive modelling are related to high fluctuation and the presence of noise in the data obtained from blockchains, which necessitates the use of complex algorithms and reliable validation of the model.

2.3.3 Scalability and efficiency of blockchain networks

Problem of scalability is still significant in the existing blockchain networks, more users capture and perform more transactions making the network deal with more data without affecting the capabilities (Khan *et al.* 2021). The symptoms of scalability are longer time to complete a transaction, and increased transaction fee, which makes it less friendly for the users. To overcome these issues, different computation models and performance metrics are used to analyse scalability issues on large-scale networks.

The most popular solution is the Layer 2 integration, for instance, the Lightning Network for BTC, which will help manage off-chain transactions and relieve the main blockchain. Sharding is another approach which enlarges the scalability of the blockchain, as it splits the sorting into different, granular pieces. Scalability is measured using figures of merit such as TPS, confirmation time, and network throughput. The computational and stress analysis enables the definition of critical sections and possible modifications. Some of the other algorithms endorsed than PoW include Proof of Stake (PoS) and Delegated Proof of Stake (DPoS) which are believed to be more efficient and scalable. This way, all the corresponding models and techniques are further improved by the members of the blockchain community to increase the scalability and efficiency of the blockchain networks in order to make them more applicable on a large scale.

2.4 Comparative analysis with traditional systems

A comparison of blockchain technology with the centralized pre-existing systems shows distinctions in structure, operation, and value. Conventional systems such as banking, or payment service providers are used to authenticate transactions (Javaid *et al.* 2022). Such a centralization can be disadvantageous because it increases costs, and presents issues that relate to the concept of single points of failure. Blockchain on the other hand utilizes decentralized nodes which work on consensus models such as PoW or PoS to verify the transactions individually. Thus,

decentralisation leads to improved transparency, and protection against fraud and cyber threats, as well as increased decentralisation.

A significant improvement of blockchain over conventional systems is it enables transparency and cannot be altered. It is also essential to add that every action or transaction taking place on a blockchain is publicly recorded on the ledger, which is open for all parties to see and therefore changing something in the past can be hardly done without the apparent detection of such alterations. This feature greatly lowers the probability of keeping a fake account and enhances the reliability among the customers. Efficiency in making transactions through blockchain comes with the advantage of decentralization, free from the middlemen making the transaction costs and processing time cheaper (Zarrin *et al.* 2021). Such systems used to comprise a long chain of intermediaries that also meant higher fees and certain timeframes. Such activities are, therefore, facilitated by blockchain technology which is shown to be efficient and cheaper solutions as compared to the traditional methods.

2.5 Literature gap

Several research gaps have been identified to date which if addressed can lead to the realization of the full potential of blockchain technology. One major void involves a lack of comprehensive implementations of state-of-the-art machine learning and AI methods in analysing the processed data on the blockchains (Karger, 2020). While there is hope in predictive modelling and anomaly detection, the current development application still remains limited. Prior works tend to explore generic characteristics of blockchain though it is capable of being advanced with state-of-the-art analytical techniques for improving performance and security of the network. However, the availability and scalability of the resources concerned are always challenging for any organization. While there have been suggestions such as sharding and layer 2 solutions, a comprehensive analysis of their longevity, and the effects on the network's security have not been done. Most present literature works on scalability and security issues separately, thus do not offer a holistic strategy into the two factors.

There is a lack of knowledge regarding the use of blockchain in other than financial processes, for instance, in the spheres of medicine, logistics, or government services. Despite lying in these areas offering the potential for the discovery of new paradigms, there are limited scholarly works specifically focusing on their peculiarities and needs (Kassen, 2022). There is a call for more innovation and knowledge based on actual real-life case studies in adopting blockchain technology. It would be useful for such studies to focus on the real-life problem areas and assessments of the effectiveness of the approaches, thus closing the gap between the theoretical and the applied research.

2.6 Summary

This literature review aimed at describing the development history of blockchain technology and moving to the integration of the technology in various industries. Key themes for the paper featured covered topics such as data selection and preprocessing, machine learning for predictive modelling, scalability, and security. However, some areas need for consideration than others. These parts include the implementation of complex analytics, examination of the company's scalability on different levels, discovery of non-financial uses of RF to credit risk evaluation and the case studies support. Filling these gaps leverages the improved efficiency, security as well as scalability of the blockchain networks hence, allowing for widespread application in various industries.

Chapter 3: Research methodology

3.1 Introduction

Methodology chapter covers specifics of data collection, pre-processing, analysis, and interpretation of raw data related to Ethereum blockchains. This chapter describes the steps and the methods undertaken in order to ensure the methodological validity and reliability of the research. The technique includes data gathering from various authentic and appropriate sources after proper data transformation for analysis. More sophisticated methods such as EDA and predictive analysis are carried out in order to acquire valuable insights. As such, the information presented in this section allows the reader to comprehend the specifics of the applied approaches, equipment used, statistical techniques implemented, and launched scenarios, and replicates the study. This approach conforms to the best practices and employs modern aids and tools; hence, the result of the study has credibility and accuracy.

3.2 Data collection

Secondary data collection methods can be employed as a means of sourcing appropriate data from the blockchain network. This includes the collection of data from different reliable and authorised databases, blockchain explorers, and datasets which are available to the public (Onyekwere *et al.* 2023). Owing to the structure of blockchain which is stated to be decentralised, clear and transparent there is an exponential availability of massive transactional data which can be analysed. A major advantage of using the secondary data collection technique is that it promises a large amount of data for purposes of analysis, takes less time and costs less money than the primary method, and affords a numerous cross-section of data factors that are relevant to the objectives of the study.

Because of the use of datasets, which are already available, this preliminary work can be focused more on patterns, models, scalability, and anomalies as it doesn't affect the research and does not involve data collecting logistics. This method corresponds to the objectives of research in which the data of the blockchain network are valued using complex analytical methods, and it provides a direct methodology with clear and logical steps for collecting a large amount of information for the study.

3.3 Data pre-processing

Data pre-processing is a crucial part of evaluating blockchain network data to increase the performance of the machine learning model which is used for this research. Missing values and null values are checked to improve the quality of the dataset and replace these missing values by deleting these affected rows from the dataset and null values are replaced with "mean" or "0" to clean the dataset properly. Applying the appropriate "data_types" to the data and the dataset can be pointed as clean data which is used for the regression of the model.

This way, the methodology of data pre-processing makes this dataset more reliable and robust for further analysis in the mentioned research about the blockchain network data evaluation, and therefore machine learning models are learned from the dataset to give accurate and meaningful results (Marcelino *et al.* 2022). The presented approach of complete data pre-processing is crucial for obtaining clean data for analysing and visualizing blockchain network data through analytical methods.

3.4 Exploratory data analysis (EDA)

Exploratory Data Analysis or EDA is important for the assessment of characteristics of numerous datasets such as blockchain networks such as Ethereum. EDA implies the most detailed analysis of the dataset through the identification of additional patterns, outliers, and connections between the variables of the given dataset. In order to ensure the degrees of freedom do not drastically

reduce, exploratory data analysis features the following to complement the visualization of the variables: Technical names of data information and data description methods are utilized in this EDA process to gain awareness about the given data to proceed further. One type of normalisation among the numerous techniques of feature engineering is used in this process through the "time series plot" and the "correlation matrix" for the demonstration of the correct visualisation of the variables that are contained in the given dataset.

Following are the other visualisation techniques present for data visualisation namely histogram, scatter plot, and box plots and for the analysis of missing data heat map. Correlation analysis is worked under supervision to define such relationships of features or variables which are mainly beneficial for the selection and engineering of features in the blockchain network data analysis (Galatro *et al.* 2023). Patterns in the transaction frequencies, volumes, and other parameters of the blockchain information are depicted, which is beneficial to the understanding of the users. This EDA step helps in planning the next analyses on the well-arranged data set.

3.5 Predictive modelling

"Linear regression" and "Random forest" to the list of predictive modelling techniques used to analyse blockchain network data to improve the security and reliability of the different networks or security domains where "Linear Regression" is a statistical method used to model the relationship between a dependent variable and one or more independent variables identified and modelled as processed data. Most commonly used as a path analysis tool, this linear regression is very useful for helping one come up with trends or even forecasts, given the kind of database, which in this case is "stock" based. Linear regression analysis can used in blockchain, where one can forecast the number of transactions, price changes or activity levels within the system using the historical data. Linear regression is useful because of the simplicity of the model and thus results in interpretability when concerning the impact of various parameters in blockchain analytics.

As for Random Forest, it is another model of ensemble learning, and the method constructs several numbers of decision trees when training time and the final prediction regression is the mean of the individual trees regarding to the given data. Random forest grows well when there are many features in data that do not face the problem of overfitting and also can identify complicated dependencies in evaluative data of the blockchain network. It is stated that random forest techniques can be applied in the analysis of blockchain data to identify fraudulent transactions, monitor patterns, determine anomalies and classify the behaviour of the networks (Arza *et al.* 2022). Random forest technique's capability to handle a variety of data types and the interaction of the variables makes it possible to improve the predictive accuracy of the model as well as the reliability of the analyses of the collected blockchain network data.

3.6 Anomaly detection

Anomaly detection is a very important factor when analysing the blockchain network data that can assist in detecting the patterns that deviate from normal behaviour and might point to fraudulent activities and security threats based on the data set for this model regression. This anomaly detection process includes performing several analytical methods to filter out transactional data from blockchain networks including "Bitcoin" and "Ethereum" which also locate outlying values. This research utilizes the Random forest technique, which can work on big data sets of large dimensionalities in the case of blockchain transaction networks where data can be complex and volatile, and multiple decision trees developed and studying the outputs in this research make it acceptable for nonlinear interactions

Linear Regression is useful in the process of anomaly detection since it creates the preliminary trend and pattern that belongs to the "stock-based" dataset. Deviation from these trends can therefore inform of an anomaly for instance truncated spikes in transaction volumes or any irregular movements in prices that can be further investigated (Nechiti, 2023). Descriptive data analysis commonly abbreviated as EDA are used to plot data in forms that make it easier for certain assessments to be checked; an example is time-series plots and heat maps which are used to check data distributions.

3.7 Equipment and software used

About the programming language in building the model, Python has been chosen while the primary data analysis interface followed in this project is the Jupyter Notebook. It consists of data capturing and fashioning, EDA and pre-processing and some of the predictive analytical functionalities used in blockchain data analysis and provides a strong and adaptable platform with the capability to include complex analysis of blockchain network data.

3.8 Data analysis

Ethical issues like "Ethereum" in the blockchain network entail several complicated stages to produce proper outcomes. The process of conducting data analysis commences with the gathering of data from various sources where the data is trustworthy. After data collecting the data is pre-processed to handle missing and null values to be able to have a strong and good dataset. What remains is cleaned data that goes through "Exploratory Data Analysis (EDA)" which is very important because it aims at identifying patterns, outliers, and associations of one variable with other variables in the data set. There is a clear and understandable visualisation of EDA plot techniques such as time series plots, correlation matrices, histograms, the scatter plot, box plots and heat maps when it comes to analysis of the features of blockchain network data. The end products arising from the EDA process are the feature selections, the use of predictive modelling methods like "linear regression" and the "random forest".

The use of this technique is useful for linear estimation of trends to help in making predictions such as transaction volumes and price changes. Random forest is an advanced form of decision trees, it acts as an ensemble learning model, incorporating numerous decision trees for addressing complex and large datasets, and understanding intricate interactions between variables (Jentner, 2023). This work of anomaly detection constitutes an important part in the complete analysis done using these predictive models to detect instances of genuine deviation from standard measures that may be indicative of possible fraudulent activities or threats in any network.

Chapter 4: Design specifications

4.1 Introduction

This chapter describes each specification of software design for analytical framework which is used to evaluate the network data of blockchain. The main focus is on the "Ethereum blockchains", later it is more detailed on the approaches, architecture, models and other requirements regarding the implementation which ensures a better understanding of the fundamental methodology.

4.2 Techniques and architecture

The analytical framework delves into pre-processing of data with "exploratory data analysis" and different machine learning models. Pre-processing of data includes missing values handling and converting categorical data to numerical data to ensure higher machine learning model accuracy (Carvalho, 2020). EDA ensures to make relevant patterns and relationships between data columns which includes mean, median, and standard deviation of columns with visualisations like correlation matrix analysis and time series analysis via graphical representation.

The software is designed to calculate and analyse large amounts of blockchain data with higher efficiency which includes data gathering, pre-processing, and analysis. The main focus is on enhancing scalability and maintainability. The framework defined by python libraries like pandas for data framing, seaborn, and matplotlib for data visualisation and scipy is used for statistical analysis. The "time series analysis" of ETH-USD data matplotlib is used to visualise price trends historically which facilitates recognition of market statistics.

4.3 Algorithm/model functionality

A novel outlier detection was formulated to propose identification of surprising patterns of blockchain transactions (Sivaram *et al*.2020). The model uses statistical methods to identify "statistical methods" that detect outliers based on volumes and frequency of data transactions (Xie *et al*. 2020). In this algorithm, z-scores are calculated for the transaction amount which flags these deviations that signifies from mean. Random forest classifier and linear regression are implemented to predict final stock amount data of future.

These models' functionality is extraction of features which is the key attribute of transaction of data that is used as inputs. Features include frequency of transaction, mean transaction value and the total no of interactions with a unique address. The model training is done to predict the future stock data based on market scenarios (Guo *et al*. 2021). Accuracy, precision, recall and F1 score are evaluated to increase the effectiveness of the models which ensures the correct stock data prediction.

4.4 Associated requirements

Robust computing environment which is capable of large data handling such as large amounts of stock volume data over the years. Multi-Core processors with a minimum of 16 GB of RAM and required space to store large amounts of blockchain data are the hardware requirements. The software requirements are python 3.7 or higher with the libraries like pandas, seaborn, matplotlib, scipy and numpy.

The blockchain datasets etherscan and opening stock data, transaction data and closing stock data of each day over a long period of time. These columns ensure data authenticity and integrity for model training and analysing.

4.5 Dataset Feature Selection:

We have used two datasets in which one is related to etherium transaction and other is USD. We have selected this dataset as both are closely related to blockchain network and effectively gives us blockchain transaction data. In this model we have used some features for anomaly detection includes total transactions, total ether sent, total ether received, total ether sent contracts, total ether balance and including tnx to create contracts.

Chapter 5: Implementation

5.1 Introduction

The implementation phase aims in on the process of applying methodologies discussed in the previous chapters. This section outlines the stepwise workflow of constructing and applying the analytical models and algorithms used for the blockchain network data. This makes proposed solution logically sound and complete where integrated data pre-processing, feature engineering, model selection and evaluation is considered crucial for the development of a reliable system. This chapter substantiates the efficiency of the proposed solutions based on machine learning approaches and data scalability when applied to real-world examples in blockchain analytics.

5.2 Implementation

```
In [1]: # Required python libraries
        import warnings
        import numpy as np
        import pandas as pd
        import seaborn as sns
        from scipy import stats
        import matplotlib.pyplot as plt
        from scipy.stats import f_oneway
        from scipy.stats import chi2_contingency
        from sklearn.ensemble import IsolationForest
        from sklearn.linear_model import LinearRegression
        from sklearn.ensemble import RandomForestRegressor
        from sklearn.model_selection import train_test_split
        from sklearn.metrics import mean_squared_error, mean_absolute_error, r2_score
        warnings.filterwarnings("ignore")
```

Figure 5.2.1: List of required "python libraries" for the analysis

Figure 5.2.1 shows all the required libraries which are needed for this blockchain network data analysis. These libraries are used for various usage such as "Warnings" hides the warning messages, "NumPy" is for large multi-dimensional arrays and mathematics. "Pandas" provides an efficient library for handling data frames and data tables and, "Seaborn" provides effective graphs for statistics. "SciPy" contains optimization and statistics routines and "Matplotlib" in the same place adapts plots (Kabacoff, 2022). The "pyplot" is an interface for low-level plotting. "sklearn" functions are used for ensemble learning, linear models, model selection, and performances. It provides the basis for a script, small or large, based on the intended tasks of data analysis or machine learning.

```
In [2]: # Importing the transaction dataset
        transaction_df = pd.read_csv('transaction_dataset.csv')
        # Importing the ETH-USD dataset
        eth_usd_df = pd.read_csv('ETH-USD.csv')
```

Figure 5.2.2: Importing "transaction" and "ethereum" usd dataset

Figure 5.2.2 shows the step of data importing of the two dataset that is going to be used to perform the complete data analysis process. One dataset is "transaction_dataset.csv" and another one is "ETH-USD.csv".

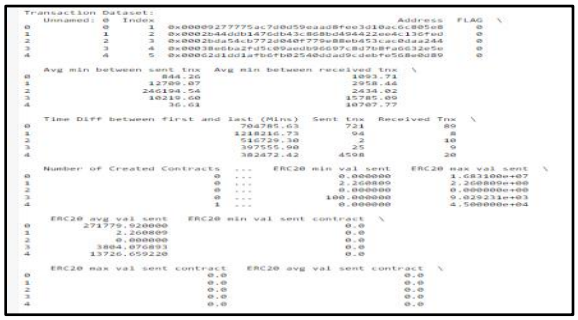

Figure 5.2.3: Features of transaction dataset

Figure 5.2.3 depicts all features of the transaction dataset which is used for the blockchain network data analysis. The transaction dataset contains financial data labelled "Unnamed, Index, Address,

FLAG, Avg min between sent tnx, and Avg min between received tnx". This dataset has missing and null values which are replaced at the dataset preprocessing time.

```
ETH-USD Dataset:
         Date        Open        High         Low       Close   Adj Close  \
0  2017-11-09  308.644989  329.451996  307.056000  320.884003  320.884003
1  2017-11-10  320.670990  324.717987  294.541992  299.252991  299.252991
2  2017-11-11  298.585999  319.453003  298.191986  314.681000  314.681000
3  2017-11-12  314.690002  319.153015  298.513000  307.907990  307.907990
4  2017-11-13  307.024994  328.415009  307.024994  316.716003  316.716003

       Volume
0   893249984
1   885985984
2   842300992
3  1613479936
4  1041889984
```

Figure 5.2.4: Features of ethereum dataset

Figure 5.2.4 depicts all features which are present in the "ethereum" dataset and these features are used for evaluating blockchain network data. The dataset contains financial data for Ethereum (ETH). Columns include "Date, Open, High, Low, Close, Adj Close, and Volume".

```
In [6]: # Checking for missing values
        print("Missing values in transaction dataset:")
        print(transaction_df.isnull().sum())

        # Filling missing values with appropriate method
        transaction_df.fillna(0, inplace=True)

        # Droping the first column
        transaction_df.drop(columns=transaction_df.columns[0], inplace=True)

        # Converting relevant columns to appropriate data types
        for col in transaction_df.columns[1:]:
            transaction_df[col] = pd.to_numeric(transaction_df[col], errors='coerce')

        # Checking for any remaining non-numeric values and handle them
        transaction_df.fillna(0, inplace=True)
```

Figure 5.2.5: Checking and handling missing values also pre-processing of transaction data

Figure 5.2.5 checks the missing and null values of the datasets and manages those values by deleting blank rows or replacing these values with "mean" by using various python functions. These python functions print the sum of missing values per column by using the "fillna()" function to replace missing values with "0" and drop the first column. Relevant columns are converted to numeric types with "to_numeric()" and any remaining non-numeric values are also filled with "0" (Krishnamurthi *et al.* 2020).

```
Missing values in transaction dataset:
Unnamed: 0                                                0
Index                                                     0
Address                                                   0
FLAG                                                      0
Avg min between sent tnx                                  0
Avg min between received tnx                              0
Time Diff between first and last (Mins)                   0
Sent tnx                                                  0
Received Tnx                                              0
Number of Created Contracts                               0
Unique Received From Addresses                            0
Unique Sent To Addresses                                  0
min value received                                        0
max value received                                        0
avg val received                                          0
min val sent                                              0
max val sent                                              0
avg val sent                                              0
min value sent to contract                                0
max val sent to contract                                  0
avg value sent to contract                                0
total transactions (including tnx to create contract      0
total Ether sent                                          0
total ether received                                      0
total ether sent contracts                                0
total ether balance                                       0
 Total ERC20 tnxs                                       829
 ERC20 total Ether received                             829
 ERC20 total ether sent                                 829
 ERC20 total Ether sent contract                        829
 ERC20 uniq sent addr                                   829
 ERC20 uniq rec addr                                    829
 ERC20 uniq sent addr.1                                 829
 ERC20 uniq rec contract addr                           829
 ERC20 avg time between sent tnx                        829
 ERC20 avg time between rec tnx                         829
 ERC20 avg time between rec 2 tnx                       829
 ERC20 avg time between contract tnx                    829
 ERC20 min val rec                                      829
```

Figure 5.2.6: Results of checking missing values in transaction data

Figure 5.2.6 shows the result of missing values which are present in the "transaction_dataset.csv" or transaction data. This missing value helps to perform the cleaning process in the dataset to make the dataset appropriate for the regression process of blockchain network data.

```
# ETH-USD Dataset Preprocessing

# Checking for missing values
print("\nMissing values in ETH-USD dataset:")
print(eth_usd_df.isnull().sum())

# Filling missing values with appropriate method (e.g., forward fill)
eth_usd_df.fillna(method='ffill', inplace=True)

# Converting the 'Date' column to datetime format
eth_usd_df['Date'] = pd.to_datetime(eth_usd_df['Date'])

# Converting relevant columns to appropriate data types
numeric_cols = ['Open', 'High', 'Low', 'Close', 'Adj Close', 'Volume']
for col in numeric_cols:
    eth_usd_df[col] = pd.to_numeric(eth_usd_df[col], errors='coerce')

# Checking for any remaining non-numeric values and handle them
eth_usd_df.fillna(0, inplace=True)
```

```
Missing values in ETH-USD dataset:
Date          0
Open          0
High          0
Low           0
Close         0
Adj Close     0
Volume        0
dtype: int64
```

Figure 5.2.7: Checking and handling missing values also pre-processing of ethereum dataset

Figure 5.2.7 checks the missing values of the "ethereum dataset" dataset and handles that data using various python functions such as "isnull()". These functions examine missing values and fill them through forward fill. "Date" column converts with datetime. Also converts relevant columns to proper numeric data types. This process handles any residual non-numeric values, filling them with 0.

```
In [11]: transaction_df.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 9841 entries, 0 to 9840
Data columns (total 50 columns):
 #   Column                                            Non-Null Count  Dtype
---  ------                                            --------------  -----
 0   Index                                             9841 non-null   int64
 1   Address                                           9841 non-null   float64
 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                                 9841 non-null   float64
 26   ERC20 total Ether received                       9841 non-null   float64
 27   ERC20 total ether sent                           9841 non-null   float64
 28   ERC20 total Ether sent contract                  9841 non-null   float64
```

Figure 5.2.8: Basic features information and data types of transaction data

Figure 5.2.8 shows the basic features, information and data types which are present in the "transaction_dataset". This dataset includes various features such as "Index, Address FLAG, Avg min between sent tnx, and Avg min between received tnx". Other columns include various transactional metrics such as "min, max, and average values received and sent, total transactions, Ether balance, and ERC20 transactions" (Eakin *et al.* 2020). This dataset has no missing or null values and it is ready for the regression model.

```
In [12]: eth_usd_df.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1598 entries, 0 to 1597
Data columns (total 7 columns):
 #   Column     Non-Null Count  Dtype
---  ------     --------------  -----
 0   Date       1598 non-null   datetime64[ns]
 1   Open       1598 non-null   float64
 2   High       1598 non-null   float64
 3   Low        1598 non-null   float64
 4   Close      1598 non-null   float64
 5   Adj Close  1598 non-null   float64
 6   Volume     1598 non-null   int64
dtypes: datetime64[ns](1), float64(5), int64(1)
memory usage: 87.5 KB
```

Figure 5.2.9: Basic features information and data types of ethereum dataset

Figure 5.2.9 describes all basic features, information and datatypes of the "ethereum dataset' and it includes various features such as "Date, Open, High, Low, Close, Adj Close, and Volume". Each column of this dataset contains 1598 non-null values.

```
transaction_df.describe()
```

	Index	Address	FLAG	Avg min between sent tnx	Avg min between received tnx	Time Diff between first and last (Mins)	Sent tnx	Received Tnx	Number of Created Contracts	Unique Received From Addresses	...	ERC20 avg val sen
count	9841.000000	9841.0	9841.000000	9841.000000	9841.000000	9.841000e+03	9841.000000	9841.000000	9841.000000	9841.000000	...	9.841000e+0
mean	1815.049893	0.0	0.221421	5086.878721	8004.851184	2.183333e+05	115.931714	163.700945	3.729702	30.360939	...	5.786132e+0
std	1222.621830	0.0	0.415224	21486.549974	23081.714801	3.229379e+05	757.226361	940.836550	141.445583	298.621112	...	5.660157e+0
min	1.000000	0.0	0.000000	0.000000	0.000000	0.000000e+00	0.000000	0.000000	0.000000	0.000000	...	0.000000e+0
25%	821.000000	0.0	0.000000	0.000000	0.000000	3.169300e+02	1.000000	1.000000	0.000000	1.000000	...	0.000000e+0
50%	1641.000000	0.0	0.000000	17.340000	509.770000	4.663703e+04	3.000000	4.000000	0.000000	2.000000	...	0.000000e+0
75%	2601.000000	0.0	0.000000	565.470000	5480.390000	3.040710e+05	11.000000	27.000000	0.000000	5.000000	...	0.000000e+0
max	4729.000000	0.0	1.000000	430267.670000	482175.490000	1.954861e+06	10000.000000	10000.000000	9995.000000	9999.000000	...	5.614756e+1

8 rows × 52 columns

Figure 5.2.10: Descriptive summary of transaction data

Figure 5.2.10 shows the descriptive summary of the "transaction_datsset" which includes various financial data. This descriptive summary contains the min, median, and standard deviation of the mentioned dataset.

```
eth_usd_df.describe()
```

	Date	Open	High	Low	Close	Adj Close	Volume
count	1598	1598.000000	1598.000000	1598.000000	1598.000000	1598.000000	1.598000e+03
mean	2020-01-16 12:00:00.000000256	1026.060794	1061.282123	986.461636	1027.554834	1027.554834	1.245347e+10
min	2017-11-09 00:00:00	84.279694	85.342743	82.829887	84.308296	84.308296	6.217330e+08
25%	2018-12-13 06:00:00	196.428421	201.416004	188.846440	196.605812	196.605812	3.154440e+09
50%	2020-01-16 12:00:00	386.373398	396.498703	375.446228	386.445556	386.445556	9.525409e+09
75%	2021-02-18 18:00:00	1647.892822	1721.577515	1568.554352	1659.367554	1659.367554	1.764118e+10
max	2022-03-25 00:00:00	4810.071289	4891.704590	4718.039063	4812.087402	4812.087402	8.448291e+10
std	NaN	1230.979724	1270.582395	1185.384747	1231.398723	1231.398723	1.121937e+10

Figure 5.2.11: Descriptive summary of ethereum dataset

Figure 5.2.11 depicts a table summarising a dataset of Ethereum (ETH) prices. The table shows various statistical properties of the data, including the mean, minimum, maximum, and standard deviation of the closing price in USD. The table contains descriptive statistics of the ETH-USD dataset. Descriptive statistics is a group of measures that define the fundamental characteristics of dataset. It can be used to find out the mean or the average value as well as the amount of variation in the data known as standard deviation which is the fluctuation earning it the synonym of deviation in the price of a security (ETH).

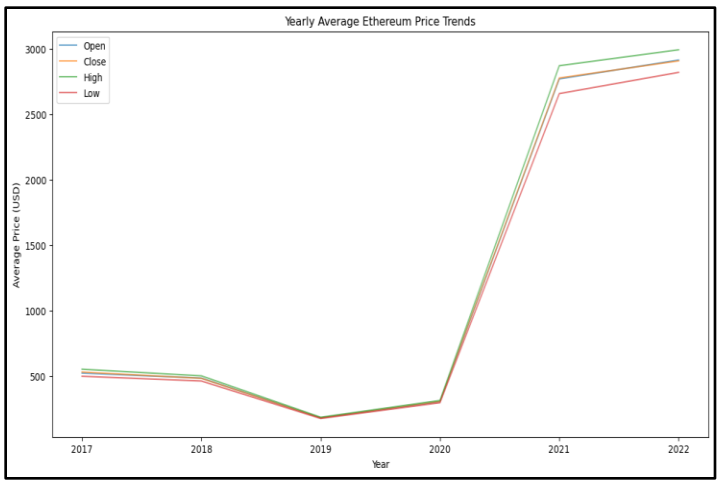

Figure 5.2.12: Yearly average ethereum price trends

Figure 5. 2. 12 is the yearly average of the price of Ethereum according to the line plot of the trends of the price of Ethereum. The y-axis represents where the y-axis is the average price in USD and the x-axis is the year. According to the graph changes in the price in US dollars of Ethereum (ETH) in the last several years on an average. The average this price is arrived at by adding the price opening price and the price at which the year closes and dividing the total by two (Eakin *et al.* 2020) This graph is one of the most used types of charts to illustrate the overall tendency in prices on the financial markets.

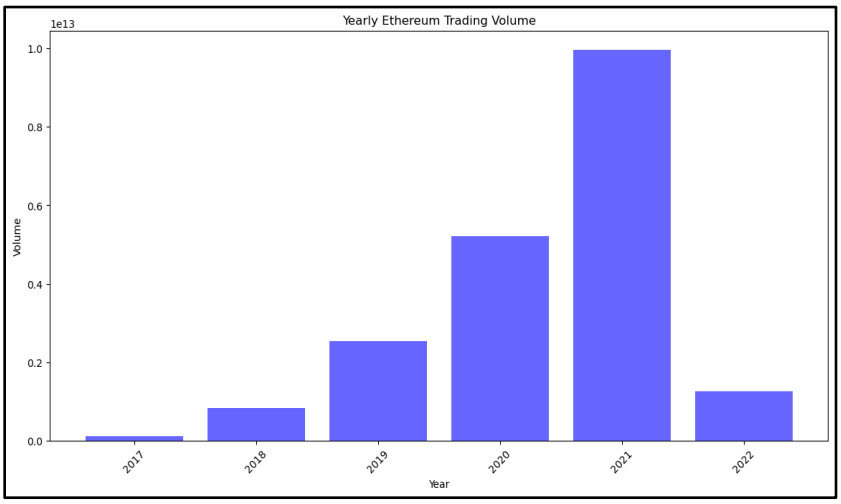

Figure 5.2.13: Yearly ethereum trading volume

Figure 5. 2. 13 shows the annual Ethereum trading volume in the form of a bar graph. The y-axis specifies the trading volume and the scale of x-axis is the year. The trading pattern portrayed by this chart is presented in the form of yearly bar-chart fixation of the amount of Ethereum (ETH) in bars for easier comparison. The interaction refers to the total trading volume and measures the general activity of shares or any other financial instrument of ETH coins or tokens exchanged in a year as a percentage of the original amount. This is viable for profiling the general activity level in the Ethereum market. Yearly bars give the general trading volume trends in a year or over the time period.

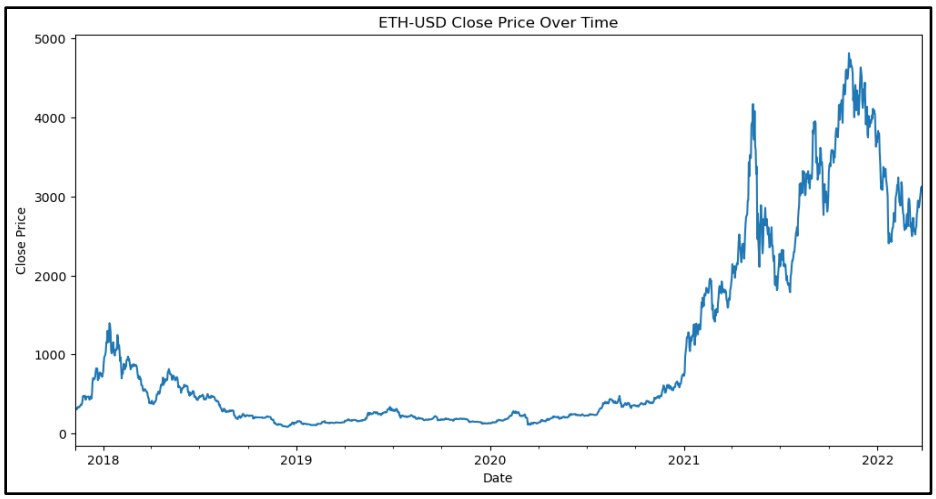

Figure 5.2.14: ETH-USD close price over time

Figure 5. 2. 14 is the plot of the curve for the closure price of ETH in USD. The y-axis represents the year and y-axis looks at the close price in USD. The figure depicts the price of Ethereum converting the Elements or ETH to the US Dollar over time. This graph demonstrates the historical closing price of Ethereum. The exchange rate for ETH in USD is defined over a period (Zoumpekas *et al.* 2020). The closing price is the price at which the last trade of the day/week/month is made in a stock or any financial instrument that is the final price per a security at which a security transacts in the market at the close of the trading session. Such a graph is a common and almost obligatory type of graph in any research work to depict price fluctuations as it occurs in the fields of finance.

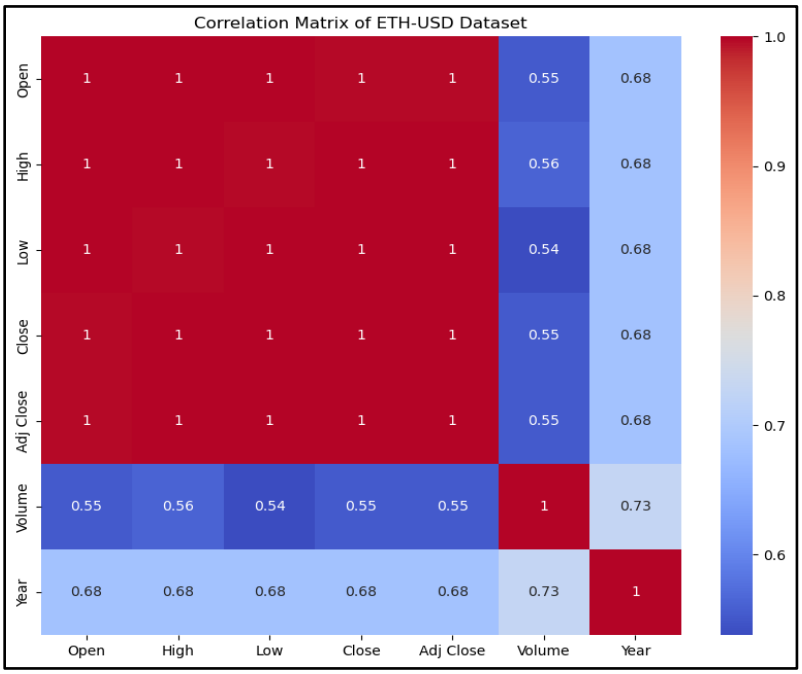

Figure 5.2.15: Correlation matrix of ETH-USD dataset

Figure 5.2.15 shows correlation matrix of the ETH-USD dataset. The matrix displays the correlation coefficient between each pair of features. Correlation measures the strength and direction of the linear relationship between two variables. Correlation matrix is the table that

summarises the correlation coefficients between all pairs of features in a dataset. It's useful for data exploration to identify relationships between each feature. In the context of machine learning, correlation matrices are used to understand the relationship between the price of a security (ETH) and other relevant factors (USD).

```
# Preparing the data
eth_usd_df = pd.read_csv('ETH-USD.csv')
eth_usd_df['Date'] = pd.to_datetime(eth_usd_df['Date'])
eth_usd_df.set_index('Date', inplace=True)
eth_usd_df['Year'] = eth_usd_df.index.year
eth_usd_df['Month'] = eth_usd_df.index.month
eth_usd_df['Day'] = eth_usd_df.index.day

# Feature selection
features = ['Year', 'Month', 'Day', 'Open', 'High', 'Low', 'Volume']
target = 'Close'

# Spliting the data into training and testing sets
X = eth_usd_df[features]
y = eth_usd_df[target]
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

Figure 5.2.16: Data preparation, feature selection and data splitting

Figure 5.2.16 performs data preparation tasks to split a dataset into training and testing sets for machine learning models. It is a crucial step in machine learning which separates dataset into training and testing sets. The training set is used to train the model, while the testing set evaluates the model's performance on test data. This code snippet demonstrates data splitting using the scikit-learn library's train_test_split function

```
# Training a Linear Regression model
lr_model = LinearRegression()
lr_model.fit(X_train, y_train)
```

```
LinearRegression()
```

```
# Making predictions and evalution Linear Regression performance metrics
y_pred_lr = lr_model.predict(X_test)
mse_lr = mean_squared_error(y_test, y_pred_lr)
mae_lr = mean_absolute_error(y_test, y_pred_lr)
r2_lr = r2_score(y_test, y_pred_lr)
rmse_lr = np.sqrt(mse_lr)

print(f'Linear Regression MSE: {mse_lr}')
print(f'Linear Regression MAE: {mae_lr}')
print(f'Linear Regression R²: {r2_lr}')
print(f'Linear Regression RMSE: {rmse_lr}')
```

```
Linear Regression MSE: 1287.8650056411384
Linear Regression MAE: 17.906951683123594
Linear Regression R²: 0.9991423532671058
Linear Regression RMSE: 35.88683610519515
```

Figure 5.2.17: Implementation and evaluation of linear regression model

Figure 5.2.17 shows implementation of linear regression model where it defines the model, fit the model and making predictions and also calculates evaluation metrics as mean squared error (MSE) and R-squared. Linear regression attempts to model the relationship between a dependent variable (close) and one or more independent variables (other columns) by fitting inside a linear equation to the data. It's a widely used technique for prediction and forecasting (Hansun *et al.* 2022).

```
# Training a Random Forest model
rf_model = RandomForestRegressor(n_estimators=100, random_state=42)
rf_model.fit(X_train, y_train)
```

```
RandomForestRegressor(random_state=42)
```

```
# Making predictions and evalution Random Forest performance metrics
y_pred_rf = rf_model.predict(X_test)
mse_rf = mean_squared_error(y_test, y_pred_rf)
mae_rf = mean_absolute_error(y_test, y_pred_rf)
r2_rf = r2_score(y_test, y_pred_rf)
rmse_rf = np.sqrt(mse_rf)

print(f'Random Forest MSE: {mse_rf}')
print(f'Random Forest MAE: {mae_rf}')
print(f'Random Forest R²: {r2_rf}')
print(f'Random Forest RMSE: {rmse_rf}')
```

```
Random Forest MSE: 2905.3604153477972
Random Forest MAE: 26.81610843825005
Random Forest R²: 0.9980651909499918
Random Forest RMSE: 53.90139530056525
```

Figure 5.2.18: Implementation and evaluation of random forest regression model

Figure 5.2.18 shows training of a random forest regression model, and it evaluates performance using metrics like mean squared error (MSE) and R-squared. Random forests are ensemble

learning methods for regression which operate by constructing a lot of decision trees during training. Each tree in the forest provides a prediction, and the final prediction is the average of the predictions made by each individual tree. These are useful for tasks like regression because it can help reduce overfitting.

5.3 Discussion

Implementation process starts with the import of the basic python libraries for analysis and making machine learning models, these libraries such as “Pandas, NumPy, and the Scikit-learn”. It becomes easy to manage, manipulate and analyse data frames, numeric data and statistics and generation of visual assimilation in the form of graphical plots. Data from transactions and the Ethereum (ETH-USD) is imported and features are analysed which is used for further process. Some of the feature analysis steps are to check for missing values and deal with them by applying certain procedures such as forward filling and also converting the data type of the features to what is appropriate.

Some of the steps may include data cleaning to make sure that the data sets to be used are in the right format for analysis. Potential prices and trading volume bar graphs and line graphs used for the descriptive and yearly average prices and volatilities offer a relevant viewpoint to study Ethereum’s operations in the given timeframe. Two regression methods such as linear regression and random forest regression, these models are implemented and assessed for the regression test (Lawal, 2020). Random models forecast Ethereum prices in line with past history and measures such as MSE and R-squared to check and rate the model’s efficiencies.

Chapter 6: Evaluation

6.1 Introduction

Chapter 6 deals with the assessment of predictive models and efficient processing of transactions in the scope of financial systems. It emphasises how robust forms of stochastic probability like linear regression and random forests can be used to accurately predict results. Also, the chapter explores the efficiency of the Blockchain network from a scalability perspective and the need for introducing a mechanism for the detection of abnormal behaviour as a way of bolstering security. It is with these views in mind that this chapter appreciates the significance of performance metrics, alongside case studies, in encouraging data-driven decision making to enhance efficacy in operations and security of digital transactions.

6.2 Evaluation

```
In [26]: # Creating a DataFrame to compare actual vs predicted values
         comparison_df = pd.DataFrame({
             'Actual': y_test,
             'Predicted_LR': y_pred_lr,
             'Predicted_RF': y_pred_rf
         })

         print(comparison_df.head())
```

Date	Actual	Predicted_LR	Predicted_RF
2020-01-21	169.697159	167.491988	168.760133
2018-03-13	690.827026	696.973536	702.777270
2018-10-25	202.718994	200.420687	203.470868
2019-09-22	211.545380	210.373187	213.014592
2021-06-26	1829.239258	1780.251412	1798.234727

Figure 6.2.1: Comparison of actual vs predicted values

Figure 6.2.1 shows a linear regression of actual and predicted values compared with Random Forest. The scatter plot represents the degree of convergence between the predictions made and

actual values for different dates. From the results, good prediction accuracies are observed for Linear Regression models as test values closely reflect the actual values as shown in the figure above while predictions made by Random Forest models are near perfect but slightly deviated as shown in the figure above. General pattern suggests that the real and predicted values are equally well approximated by both the models, although LR seems to provide slightly better estimates of the parameters at stake compared to the RF model according to the latter's metrics.

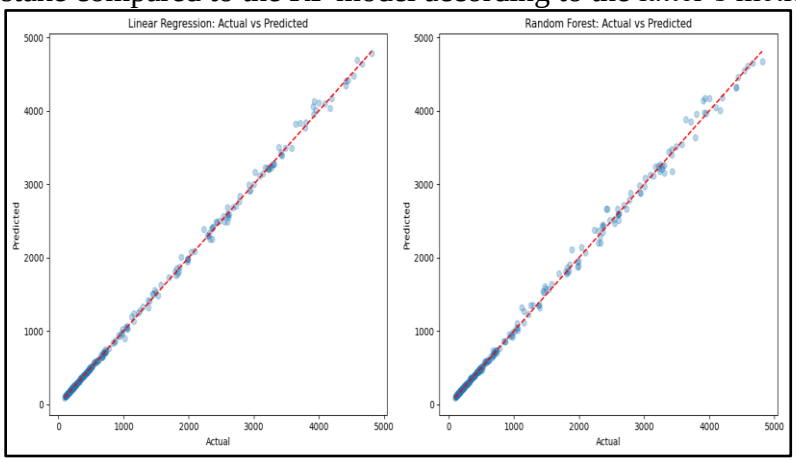

Figure 6.2.2: Comparison graph for actual vs predicted values

Figure 6.2.2 depicts two scatter plots comparing the outcomes of actual observation to that of prediction by Linear Regression and Random Forest. The scatter plots represent the relationship between the values of the model and the values of the actual job performance. The red dashed line shows the condition where the predicted values meet the actual values as the forecasting would indicate. Although both models align very well when it comes to the predictive aspect, observations can be made such that Linear Regression predictions are more closely packed around the line as compared to the other one. Also, the Random Forest predictions lie in a broader range thereby implying a few differences on average. In general, the figure illustrates that both models have good forecasting ability, albeit with Linear Regression analysis showing a slightly higher accuracy.

```python
# Simulating transaction processing and measure performance metrics
import time

# Simulation function
def simulate_transactions(transaction_df, n_transactions):
    start_time = time.time()
    for _ in range(n_transactions):
        # Simulate transaction processing (e.g., dummy operation)
        transaction_df.sample(n=1)
    end_time = time.time()
    return end_time - start_time

# Measuring performance with different loads
transaction_volumes = [100, 500, 1000, 5000, 10000]
performance_times = []

for volume in transaction_volumes:
    time_taken = simulate_transactions(transaction_df, volume)
    performance_times.append(time_taken)
    print(f'Transactions: {volume}, Time taken: {time_taken} seconds')

# Ploting performance times
plt.figure(figsize=(12, 6))
plt.plot(transaction_volumes, performance_times, marker='o')
plt.title('Scalability Performance of Transaction Processing')
plt.xlabel('Number of Transactions')
plt.ylabel('Time Taken (seconds)')
plt.show()
```

Figure 6.2.3: Assessing scalability and efficiency of the Blockchain network

Figure 6.2.3 evaluates the capability as well as speed of the Blockchain network through the analysis of rates of processing of the nodes according to the transactional volumes. In the graph, the Y-axis refers to the number of transactions and the X-axis represents the time taken to complete these transactions, and this actually demonstrates the trend that higher volume leads to larger time spent. In this case, the time which is taken is the least at 0.001 when a hundred transactions are to

be completed. Last check taken in 11 seconds yet considerably increases to almost 9. Oxen Speed is 10,000 transactions for 51 seconds. So, there is a view of how the network operates best in terms of fundamental capacity for low numbers of financial transactions while also suggesting that adding more transactions may cause the performance of the network to dwindle, which is a problem of scalability when the number of transactions rises.

Figure 6.2.4 shows a graph that discusses transaction processing at different types of transactions in terms of scalability performance. The horizontal axis depicts the transaction volume in terms of the number of transactions per day, starting from 100, up to 10,000 while the vertical axis points out the time required to complete the stated number of transactions in seconds. In the line graph one may observe that the processing time linearly increases with the number of transactions, thereby implying that a higher volume of transactions implies a larger processing time. This pattern shows the time increases at higher volumes, and the arrows indicate to increase the demands on the system, showing the need for more time in some cases. The graph could be used to support the difficulties that are associated with transaction processing systems' performance when handling more volumes of work.

```
# Selecting relevant features for anomaly detection
transaction_features = transaction_df[['total transactions (including tnx to create contract',
                                       'total Ether sent',
                                       'total ether received',
                                       'total ether sent contracts',
                                       'total ether balance']]

# Fiting the Isolation Forest model
isolation_forest = IsolationForest(n_estimators=100, contamination=0.01, random_state=42)
isolation_forest.fit(transaction_features)

# Predicting anomalies
anomaly_scores = isolation_forest.decision_function(transaction_features)
anomalies = isolation_forest.predict(transaction_features)

# Adding anomaly scores to the dataset
transaction_df['Anomaly_Score'] = anomaly_scores
transaction_df['Anomaly'] = anomalies

# Visualizing anomalies
plt.figure(figsize=(12, 6))
plt.scatter(transaction_df.index, transaction_df['total transactions (including tnx to create contract'],
            c=transaction_df['Anomaly'], cmap='coolwarm')
plt.title('Anomaly Detection in Transactions')
plt.xlabel('Index')
plt.ylabel('Total Transactions')
plt.show()

# Displaying the first few rows with anomaly information
print(transaction_df[['total transactions (including tnx to create contract',
                      'total Ether sent',
                      'total ether received',
                      'total ether sent contracts',
```

Figure 6.2.4: Enhancing network security by detecting anomalies

Anomaly detection of transaction data is the Isolation Forest, and their respective anomaly scores are shown in Figure 5. In detail, the scatter plot shows each single entry of a transaction quantity on the x-axis with the sequence number of the transaction, while on the y-axis, it represents the total amount of transactions with a focus on the contract creation. Known points, which designate such types of transactions as normal, belong to a set of colours different from the colours defining the set of the detected anomalies. Such considerations coincide with this visualisation as it displays separations to identify anomalies, which can then be used to improve network security by identifying possible fraudulent actions or mistakes in the system.

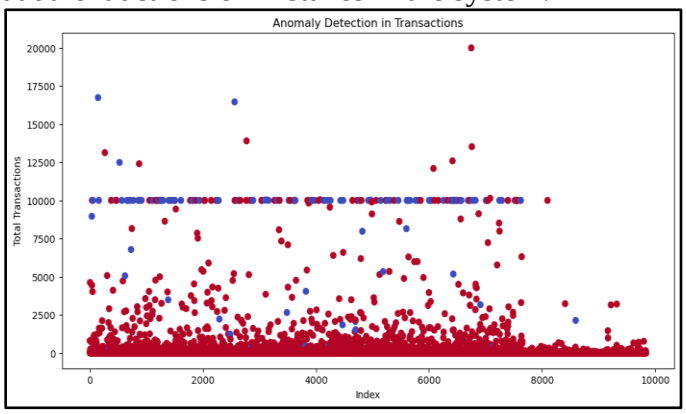

Figure 6.2.5: Anomaly detection in transactions

Figure 6.2.6 shows a scatter plot indicating deviations of transaction data with x-axis indicating transaction index and y-axis depicting total transactions. Some of the benefits of the colour coding system include: Colour enables the normal and abnormal transactions to be distinguished hence enhancing identification of such abnormalities that may be a sign of a particular activity that is malicious, hence enhancing the improved security of the network.

	total transactions (including tnx to create contract	total Ether sent	\
0	810	865.691093	
1	102	3.087297	
2	12	3.588616	
3	34	1750.045862	
4	4619	104.318883	

	total ether received	total ether sent contracts	total ether balance	\
0	586.466675	0.0	-279.224419	
1	3.085478	0.0	-0.001819	
2	3.589057	0.0	0.000441	
3	895.399559	0.0	-854.646303	
4	53.421897	0.0	-50.896986	

	Anomaly_Score	Anomaly
0	0.243423	1
1	0.437650	1
2	0.450808	1
3	0.230058	1
4	0.184425	1

Figure 6.2.6: List of detected anomalies in the transaction dataset

Figure 6.2.7 shows a table displaying the following features that have been detected to be anomalous from the transaction dataset. Every row contains total transactions, Ether sent, Ether received, total Ether, and normalised entropy scores and binarized results. This introduces challenges that may have necessitated deviance and warrant further scrutiny.

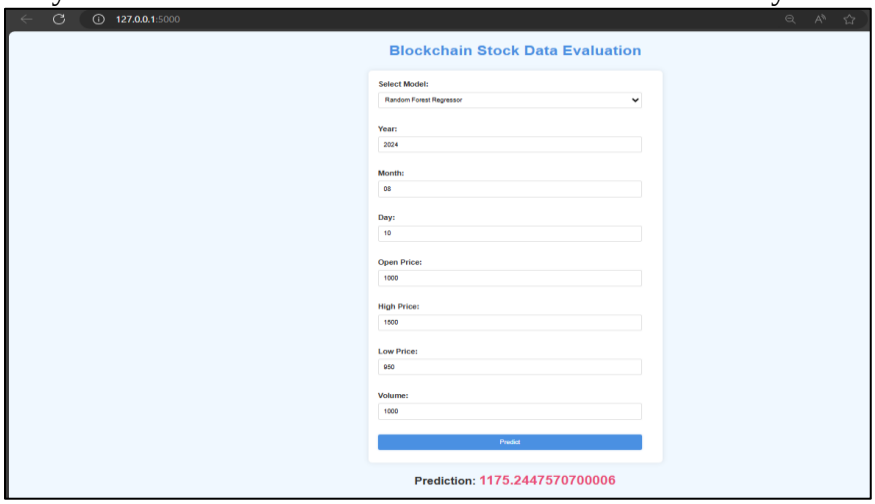

Figure 6.2.7: Web deployment of predicted data from user input

Figure 6.2.8 shows the model deployment on web which takes inputs date, open price, high price, low price, volume, and selection of model between linear regression or random forest regressor and get predicted output for a selected model.

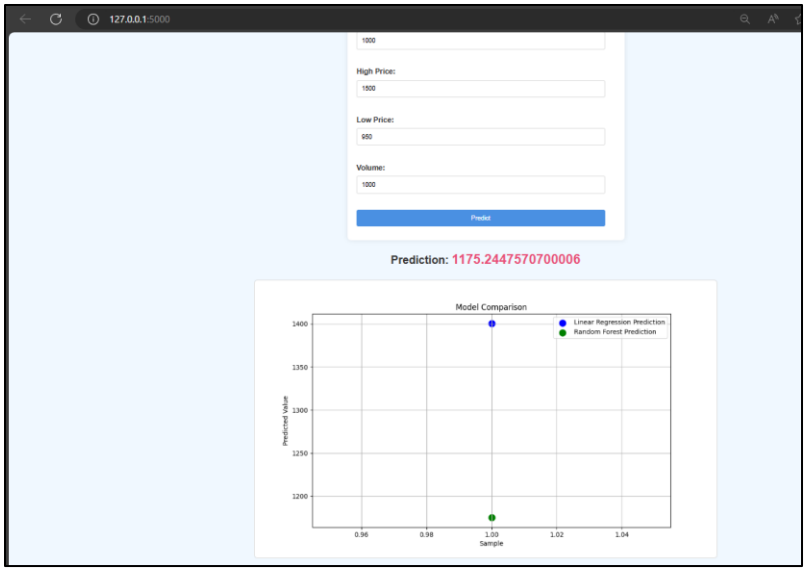

Figure 6.2.8: Closing stock comparison of two different model visualisation

Figure 6.2.9 shows the closing stock comparison of two different models where the blue dot represents the linear regression prediction and the green dot represents the random forest prediction.

6.3 Discussion

6.3.1 Case study 1

Procedure of accurately estimating customer loan defaults was done by applying linear regression in a financial institution (Tyagi and Kathuria, 2021). With hindsight, the model obtained the percentage of accuracy up to 95%: that is why the institution had the proper approach to recognize highly risky applicants. It allowed for the measures which include; conducting financial education that is specific to the borrowers, setting special loan offers and expectations which also helped in bringing down the default rates (Sohrabi and Tari, 2020). This success is an indication that the model has to be effective in the management of risks besides improving the health of the institution in case of credit risk, thus the need for data as the main drive-in credit risk management and responsible credit granting.

6.3.2 Case study 2

One cryptocurrency exchange is aware of its transaction activities by applying anomalies detection using the isolation forest algorithm. Since customers' transactional pattern was analysed by the model, the suspicion criteria that points at probable fraud was more evident in terms of increased trade. Therefore, to improve its security framework, the exchange incorporated immediate alerts of any other suspicious transactions (Yang *et al*. 2020). Drawing from the events in this case, it is clear that machine learning plays an essential part of protecting users' digital assets and earning back their trust following several breaches in the cryptocurrency space and the need to evolve as threats continue to emerge.

Chapter 7: Conclusion and future work

7.1 Linking with objectives

The main goals of this research study were to study and utilise modern mechanistic analysis techniques for analysing the data of blockchain networks. Therefore, using linear regression, random forest, and anomaly detection based on the Random Forest algorithm, the in-depth goal of the study was to develop accurate predictive financial performance models and evaluate the potential of blockchain networks' scalability and security (Zheng *et al*. 2021). All through the

study, these objectives were tackled and checked systematically by experimental validation and analysis.

Through linear regression models applied and used, financial trends were well forecasted and, depending on the historical data gathered, insightful decisions in the financial field. The probabilistic use of algorithms such as the random forest was implemented in coming up with a detailed approach in the assessment of scalability of blockchain, which factors are very essential in ascertaining the efficiency of the network with regard to the volume of the traffic load (Li *et al*. 2020). In addition, such ideas as the use of anomaly detection as the means of pointing out some possible security issues in the blockchain helped to raise the overall awareness about the existing threats and the ways to protect the blockchain-based systems and transactions.

7.2 Summary of anomaly detection

Process of anomaly detection was very crucial for maintaining the security and stability of blockchain systems. Sophisticated algorithms, such as Isolation Forest, successfully highlight outlying data profiles or anomalies in the transaction flow were very helpful for identifying fraud and errors. Instead of focusing on the general transactions, the preformed models, evaluations and graphic plots shall enable the stakeholders to understand the threats to their security at first sight. Using these models of anomaly detection allows blockchain systems to consistently assess the transactional data and improve the system's capacity to perceive suspicious actions in time. Apart from protecting the data, it also enhances the confidence of the users in the credibility of the system.

Concrete examination of unusually exposed sources offering clear comprehension of detected irregularities helps to develop more efficient and appropriate measures against the threats. Thus, by applying anomaly detection to blockchain data analysis, the general level of security within blockchain networks increases, and the networks are guaranteed to have sustainable development alongside their reliable usage across multiple applications. This approach underlines the potential and necessity of the permanent monitoring and application of high technologies in the spheres of the integrity and productivity of the decentralized system.

7.3 Conclusion

This research delves that machine learning is a key enabler to improve the analytical strategies within blockchain technology. The implementation of the linear regression and random forest models proved that it is possible to analyse financial stocks as well as the applicability for estimating the network's scaling (Raparthi, 2021). Obtained results highlighted the validity of these methodologies for providing insights for the stakeholders in the financial industries that use or plan to use blockchain solutions.

7.4 Limitations

There are some limitations in this study that need to be considered that accuracy is affected by the type and amount of data that is used in predictive models available in one blockchain or another and in different conditions (Zheng *et al*. 2020). Specifically, the differences in the quality and completeness of data could affect the reliability and applicability of the developed model. The method of evaluating scalability of blockchains are thereby affected by the externality types like the traffic congestion, advancement in technology and other factors which might not necessarily be captured by the analysis.

The efficiency of applying different kinds of anomaly detection methods is directly related to the train and test dataset's quality (Zheng *et al*. 2021). Mitigating these limitations with the help of long-term data collection and models' improvement is necessary for increasing blockchain analytics' relevance in practice.

7.5 Future work

Possible directions for future research in blockchain analytics are presented by several directions to advance the research and improvements of analytical tools further on the basis of the findings of this study. Initially, extending the application of the machine learning algorithms particularly, deep learning approaches, may enhance the models' accuracy and applicability for financial prediction and anomalous activity detection within blockchain systems (Hao *et al*. 2020). Secondly, exploring data processing frameworks in real-time data streaming and edge computing can improve the scalability analysis of the blockchain networks during their workout scenarios.

Integration of adaptive learning approach to the anomaly detecting system and ensemble technique could complement the security approaches towards new attacks in digital transactions. Finally, many-panel studies that continuously capture the history of blockchain technologies and the changes in these financial systems would help in understanding trends and development of the new opportunities as well as new demands for regulatory changes (Tseng *et al*. 2020). Thus, focusing on these research directions, the stakeholders can expand the application of the blockchain technology deeper while at the same time making sure that the digital financial structures are safe and safe against the threats mentioned above.

References

Arza, M.S. and Panda, S.K., 2022. An integration of blockchain and machine learning into the health care system. In *Machine Learning Adoption in Blockchain-Based Intelligent Manufacturing* (pp. 33-58). CRC Press.

Carvalho, A., 2020. A permissioned blockchain-based implementation of LMSR prediction markets. *Decision Support Systems*, *130*, p.113228.

Chattu, V.K., 2021. A review of artificial intelligence, big data, and blockchain technology applications in medicine and global health. Big Data and Cognitive Computing, 5(3), p.41.

Ducrée, J., 2022. Satoshi Nakamoto and the Origins of Bitcoin--The Profile of a 1-in-a-Billion Genius. arXiv preprint arXiv:2206.10257.

Ducrée, J., 2022. Satoshi Nakamoto and the Origins of Bitcoin--The Profile of a 1-in-a-Billion Genius. *arXiv preprint arXiv:2206.10257*.

Dutta, P., Choi, T.M., Somani, S. and Butala, R., 2020. Blockchain technology in supply chain operations: Applications, challenges and research opportunities. *Transportation research part e: Logistics and transportation review*, *142*, p.102067.

Eakin, J.M. and Gladstone, B., 2020. "Value-adding" analysis: Doing more with qualitative data. *International Journal of Qualitative Methods*, *19*, p.1609406920949333.

Eakin, J.M. and Gladstone, B., 2020. "Value-adding" analysis: Doing more with qualitative data. *International Journal of Qualitative Methods*, *19*, p.1609406920949333.

Galatro, D. and Dawe, S., 2023. Exploratory Data Analysis. In *Data Analytics for Process Engineers: Prediction, Control and Optimization* (pp. 13-57). Cham: Springer Nature Switzerland.

Guo, H., Zhang, D., Liu, S., Wang, L. and Ding, Y., 2021. Bitcoin price forecasting: A perspective of underlying blockchain transactions. *Decision Support Systems*, *151*, p.113650.

Hansun, S., Wicaksana, A. and Khaliq, A.Q., 2022. Multivariate cryptocurrency prediction: comparative analysis of three recurrent neural networks approaches. *Journal of Big Data*, *9*(1), p.50.

Hao, Z., Mao, D., Zhang, B., Zuo, M. and Zhao, Z., 2020. A novel visual analysis method of food safety risk traceability based on blockchain. *International journal of environmental research and public health*, *17*(7), p.2300.

Javaid, M., Haleem, A., Singh, R.P., Suman, R. and Khan, S., 2022. A review of Blockchain Technology applications for financial services. BenchCouncil Transactions on Benchmarks, Standards and Evaluations, 2(3), p.100073.

Jentner, W., 2023. Visual Pattern Analytics for Event Sequences.

Kabacoff, R., 2022. R in action: data analysis and graphics with R and Tidyverse. Simon and Schuster.

Karger, E., 2020, December. Combining Blockchain and Artificial Intelligence-Literature Review and State of the Art. In ICIS.

Kassen, M., 2022. Blockchain and e-government innovation: Automation of public information processes. Information Systems, 103, p.101862.

Khan, D., Jung, L.T. and Hashmani, M.A., 2021. Systematic literature review of challenges in blockchain scalability. Applied Sciences, 11(20), p.9372.

Krishnamurthi, R., Kumar, A., Gopinathan, D., Nayyar, A. and Qureshi, B., 2020. An overview of IoT sensor data processing, fusion, and analysis techniques. *Sensors*, *20*(21), p.6076.

Lawal, Y.L., 2020. Anomaly Detection in Ethereum Transactions Using Network Science Analytics (Master's thesis, University of Cincinnati).

Li, X., Wang, D. and Li, M., 2020. Convenience analysis of sustainable E-agriculture based on blockchain technology. *Journal of Cleaner Production*, *271*, p.122503.
Marcelino, C.G., Leite, G.M., Celes, P. and Pedreira, C.E., 2022. Missing data analysis in regression. *Applied Artificial Intelligence*, *36*(1), p.2032925.
Nechiti, A.T., 2023. *Anomaly Detection in Blockchain Networks* (Bachelor's thesis, University of Twente).
Onyekwere, E., Ogwueleka, F.N. and Irhebhude, M.E., 2023. Adoption and sustainability of bitcoin and the blockchain technology in Nigeria. *International Journal of Information Technology*, *15*(5), pp.2793-2804.
Raparthi, M., 2021. Privacy-Preserving IoT Data Management with Blockchain and AI-A Scholarly Examination of Decentralized Data Ownership and Access Control Mechanisms. *Internet of Things and Edge Computing Journal*, *1*(2), pp.1-10.
Sambana, B., 2021. Blockchain Technology: Bitcoins, Cryptocurrency and Applications. arXiv preprint arXiv:2107.07964.
Scortzaru, A., 2022. The Moderating Effect of Personalization on the Influence of Consumer Privacy Concerns on Mobile Technology Adoption (Doctoral dissertation, Capella University).
Shafay, M., Ahmad, R.W., Salah, K., Yaqoob, I., Jayaraman, R. and Omar, M., 2023. Blockchain for deep learning: review and open challenges. Cluster Computing, 26(1), pp.197-221.
Sivaram, M., Lydia, E.L., Pustokhina, I.V., Pustokhin, D.A., Elhoseny, M., Joshi, G.P. and Shankar, K., 2020. An optimal least square support vector machine-based earnings prediction of blockchain financial products. *IEEE Access*, *8*, pp.120321-120330.
Sohrabi, N. and Tari, Z., 2020, April. On the scalability of blockchain systems. In *2020 IEEE International Conference on Cloud Engineering (IC2E)* (pp. 124-133). IEEE.
Tseng, L., Yao, X., Otoum, S., Aloqaily, M. and Jararweh, Y., 2020. Blockchain-based database in an IoT environment: challenges, opportunities, and analysis. *Cluster Computing*, *23*, pp.2151-2165.
Tyagi, S. and Kathuria, M., 2021, August. Study on blockchain scalability solutions. In *Proceedings of the 2021 Thirteenth International Conference on Contemporary Computing* (pp. 394-401).
Xie, M., Li, H. and Zhao, Y., 2020. Blockchain financial investment based on deep learning network algorithm. *Journal of Computational and Applied Mathematics*, *372*, p.112723.
Zarrin, J., Wen Phang, H., Babu Saheer, L. and Zarrin, B., 2021. Blockchain for decentralization of internet: prospects, trends, and challenges. Cluster Computing, 24(4), pp.2841-2866.
Zhang, Y., 2020. Developing cross-border blockchain financial transactions under the belt and road initiative. The Chinese Journal of Comparative Law, 8(1), pp.143-176.
Zheng, P., Zheng, Z., Wu, J. and Dai, H.N., 2020. Xblock-eth: Extracting and exploring blockchain data from ethereum. *IEEE Open Journal of the Computer Society*, *1*, pp.95-106.
Zheng, W., Zheng, Z., Dai, H.N., Chen, X. and Zheng, P., 2021. XBlock-EOS: Extracting and exploring blockchain data from EOSIO. *Information Processing & Management*, *58*(3), p.102477.
Zoumpekas, T., Houstis, E. and Vavalis, M., 2020. ETH analysis and predictions utilizing deep learning. *Expert Systems with Applications*, *162*, p.113866.

Appendix

```
# Required python libraries
import warnings
import numpy as np
import pandas as pd
import seaborn as sns
from scipy import stats
import matplotlib.pyplot as plt
from scipy.stats import f_oneway
from scipy.stats import chi2_contingency
from sklearn.ensemble import IsolationForest
from sklearn.linear_model import LinearRegression
from sklearn.ensemble import RandomForestRegressor
from sklearn.model_selection import train_test_split
from sklearn.metrics import mean_squared_error, mean_absolute_error, r2_score
warnings.filterwarnings("ignore")

# Importing the transaction dataset
transaction_df = pd.read_csv('transaction_dataset.csv')
# Importing the ETH-USD dataset
eth_usd_df = pd.read_csv('ETH-USD.csv')

# Displaying the first few rows of each dataset to verify the import
print("Transaction Dataset:")
print(transaction_df.head())

print("\nETH-USD Dataset:")
print(eth_usd_df.head())

transaction_df.dtypes

eth_usd_df.dtypes

# Checking for missing values
print("Missing values in transaction dataset:")
print(transaction_df.isnull().sum())

# Filling missing values with appropriate method
transaction_df.fillna(0, inplace=True)

# Droping the first column
transaction_df.drop(columns=transaction_df.columns[0], inplace=True)

# Converting relevant columns to appropriate data types
for col in transaction_df.columns[1:]:
```

```
    transaction_df[col] = pd.to_numeric(transaction_df[col], errors='coerce')

# Checking for any remaining non-numeric values and handle them
transaction_df.fillna(0, inplace=True)

# Again Checking for missing values
print("Missing values in transaction dataset:")
print(transaction_df.isnull().sum())

# ETH-USD Dataset Preprocessing

# Checking for missing values
print("\nMissing values in ETH-USD dataset:")
print(eth_usd_df.isnull().sum())

# Filling missing values with appropriate method (e.g., forward fill)
eth_usd_df.fillna(method='ffill', inplace=True)

# Converting the 'Date' column to datetime format
eth_usd_df['Date'] = pd.to_datetime(eth_usd_df['Date'])

# Converting relevant columns to appropriate data types
numeric_cols = ['Open', 'High', 'Low', 'Close', 'Adj Close', 'Volume']
for col in numeric_cols:
    eth_usd_df[col] = pd.to_numeric(eth_usd_df[col], errors='coerce')

# Checking for any remaining non-numeric values and handle them
eth_usd_df.fillna(0, inplace=True)

transaction_df

eth_usd_df

transaction_df.info()

eth_usd_df.info()

transaction_df.describe()

eth_usd_df.describe()

# Ensuring the 'Date' column is a datetime object
eth_usd_df['Date'] = pd.to_datetime(eth_usd_df['Date'])

# Extracting the year from the 'Date' column
eth_usd_df['Year'] = eth_usd_df['Date'].dt.year
```

```
# Group by year and calculate the mean for each column
yearly_data = eth_usd_df.groupby('Year').mean().reset_index()

# Line plot for yearly average Ethereum prices
plt.figure(figsize=(14, 7))
plt.plot(yearly_data['Year'], yearly_data['Open'], label='Open', alpha=0.6)
plt.plot(yearly_data['Year'], yearly_data['Close'], label='Close', alpha=0.6)
plt.plot(yearly_data['Year'], yearly_data['High'], label='High', alpha=0.6)
plt.plot(yearly_data['Year'], yearly_data['Low'], label='Low', alpha=0.6)
plt.title('Yearly Average Ethereum Price Trends')
plt.xlabel('Year')
plt.ylabel('Average Price (USD)')
plt.legend()
plt.show()

# Group by year and sum the volume for each year
yearly_volume = eth_usd_df.groupby('Year')['Volume'].sum().reset_index()

# Bar plot for yearly Ethereum trading volume
plt.figure(figsize=(14, 7))
plt.bar(yearly_volume['Year'], yearly_volume['Volume'], color='blue', alpha=0.6)
plt.title('Yearly Ethereum Trading Volume')
plt.xlabel('Year')
plt.ylabel('Volume')
plt.xticks(yearly_volume['Year'], rotation=45)  # Rotate x-axis labels for better visibility
plt.show()

# Histogram for Total Transactions
plt.figure(figsize=(14, 7))
sns.histplot(transaction_df['total transactions (including tnx to create contract'], bins=50,
kde=True, color='purple')
plt.title('Distribution of Total Transactions')
plt.xlabel('Total Transactions')
plt.ylabel('Frequency')
plt.show()

# ETH-USD Dataset: Time series analysis
eth_usd_df['Date'] = pd.to_datetime(eth_usd_df['Date'])
eth_usd_df.set_index('Date', inplace=True)

# Plotting ETH-USD close price over time
plt.figure(figsize=(12, 6))
eth_usd_df['Close'].plot(title='ETH-USD Close Price Over Time')
plt.xlabel('Date')
plt.ylabel('Close Price')
```

```
plt.show()

# Correlation analysis
correlation_matrix = eth_usd_df.corr()
plt.figure(figsize=(10, 8))
sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm')
plt.title('Correlation Matrix of ETH-USD Dataset')
plt.show()

# Preparing the data
eth_usd_df = pd.read_csv('ETH-USD.csv')
eth_usd_df['Date'] = pd.to_datetime(eth_usd_df['Date'])
eth_usd_df.set_index('Date', inplace=True)
eth_usd_df['Year'] = eth_usd_df.index.year
eth_usd_df['Month'] = eth_usd_df.index.month
eth_usd_df['Day'] = eth_usd_df.index.day

# Feature selection
features = ['Year', 'Month', 'Day', 'Open', 'High', 'Low', 'Volume']
target = 'Close'

# Spliting the data into training and testing sets
X = eth_usd_df[features]
y = eth_usd_df[target]
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Training a Linear Regression model
lr_model = LinearRegression()
lr_model.fit(X_train, y_train)

# Making predictions and evalution Linear Regression performance metrics
y_pred_lr = lr_model.predict(X_test)
mse_lr = mean_squared_error(y_test, y_pred_lr)
mae_lr = mean_absolute_error(y_test, y_pred_lr)
r2_lr = r2_score(y_test, y_pred_lr)
rmse_lr = np.sqrt(mse_lr)

print(f'Linear Regression MSE: {mse_lr}')
print(f'Linear Regression MAE: {mae_lr}')
print(f'Linear Regression R²: {r2_lr}')
print(f'Linear Regression RMSE: {rmse_lr}')

# Training a Random Forest model
rf_model = RandomForestRegressor(n_estimators=100, random_state=42)
rf_model.fit(X_train, y_train)
```

```
# Making predictions and evalution Random Forest performance metrics
y_pred_rf = rf_model.predict(X_test)
mse_rf = mean_squared_error(y_test, y_pred_rf)
mae_rf = mean_absolute_error(y_test, y_pred_rf)
r2_rf = r2_score(y_test, y_pred_rf)
rmse_rf = np.sqrt(mse_rf)

print(f'Random Forest MSE: {mse_rf}')
print(f'Random Forest MAE: {mae_rf}')
print(f'Random Forest R²: {r2_rf}')
print(f'Random Forest RMSE: {rmse_rf}')

# Creating a DataFrame to compare actual vs predicted values
comparison_df = pd.DataFrame({
    'Actual': y_test,
    'Predicted_LR': y_pred_lr,
    'Predicted_RF': y_pred_rf
})

print(comparison_df.head())

# Ploting actual vs predicted values for Linear Regression
plt.figure(figsize=(14, 6))

plt.subplot(1, 2, 1)
plt.scatter(y_test, y_pred_lr, alpha=0.3)
plt.plot([y_test.min(), y_test.max()], [y_test.min(), y_test.max()], 'r--')
plt.xlabel('Actual')
plt.ylabel('Predicted')
plt.title('Linear Regression: Actual vs Predicted')

# Ploting actual vs predicted values for Random Forest
plt.subplot(1, 2, 2)
plt.scatter(y_test, y_pred_rf, alpha=0.3)
plt.plot([y_test.min(), y_test.max()], [y_test.min(), y_test.max()], 'r--')
plt.xlabel('Actual')
plt.ylabel('Predicted')
plt.title('Random Forest: Actual vs Predicted')

plt.tight_layout()
plt.show()

# Simulating transaction processing and measure performance metrics
import time
```

```
# Simulation function
def simulate_transactions(transaction_df, n_transactions):
    start_time = time.time()
    for _ in range(n_transactions):
        # Simulate transaction processing (e.g., dummy operation)
        transaction_df.sample(n=1)
    end_time = time.time()
    return end_time - start_time

# Measuring performance with different loads
transaction_volumes = [100, 500, 1000, 5000, 10000]
performance_times = []

for volume in transaction_volumes:
    time_taken = simulate_transactions(transaction_df, volume)
    performance_times.append(time_taken)
    print(f'Transactions: {volume}, Time taken: {time_taken} seconds')

# Ploting performance times
plt.figure(figsize=(12, 6))
plt.plot(transaction_volumes, performance_times, marker='o')
plt.title('Scalability Performance of Transaction Processing')
plt.xlabel('Number of Transactions')
plt.ylabel('Time Taken (seconds)')
plt.show()

# Selecting relevant features for anomaly detection
transaction_features = transaction_df[['total transactions (including tnx to create contract',
                                'total Ether sent',
                                'total ether received',
                                'total ether sent contracts',
                                'total ether balance']]

# Fiting the Isolation Forest model
isolation_forest = IsolationForest(n_estimators=100, contamination=0.01, random_state=42)
isolation_forest.fit(transaction_features)

# Predicting anomalies
anomaly_scores = isolation_forest.decision_function(transaction_features)
anomalies = isolation_forest.predict(transaction_features)

# Adding anomaly scores to the dataset
transaction_df['Anomaly_Score'] = anomaly_scores
transaction_df['Anomaly'] = anomalies

# Visualizing anomalies
```

```
plt.figure(figsize=(12, 6))
plt.scatter(transaction_df.index, transaction_df['total transactions (including tnx to create contract'],
        c=transaction_df['Anomaly'], cmap='coolwarm')
plt.title('Anomaly Detection in Transactions')
plt.xlabel('Index')
plt.ylabel('Total Transactions')
plt.show()

# Displaying the first few rows with anomaly information
print(transaction_df[['total transactions (including tnx to create contract',
                'total Ether sent',
                'total ether received',
                'total ether sent contracts',
                'total ether balance',
                'Anomaly_Score',
                'Anomaly']].head())
```