A Sustainable Bitcoin Architecture

by

Maruf Monem
20366005

A thesis submitted to the Department of Computer Science and Engineering
in partial fulfillment of the requirements for the degree of
M.Sc. in Computer Science

Department of Computer Science and Engineering
Brac University
April 2022

Declaration

It is hereby declared that

1. The thesis submitted is my own original work while completing degree at Brac University.
2. The thesis does not contain material previously published or written by a third party, except where this is appropriately cited through full and accurate referencing.
3. The thesis does not contain material which has been accepted, or submitted, for any other degree or diploma at a university or other institution.
4. We have acknowledged all main sources of help.

Student's Full Name & Signature:

Maruf Monem
20366005

Approval

The thesis titled "A Sustainable Bitcoin Architecture" submitted by

1. Maruf Monem (20366005)

Of Spring, 2022 has been accepted as satisfactory in partial fulfillment of the requirement for the degree of M.Sc. in Computer Science on 9th April, 2022.

Examining Committee:

External Examiner:
(Member)

Prof. Mohammad Shamsul Arefin, Ph.D.
Professor
Department of Computer Science and Engineering
Chittagong University of Engineering and Technology (CUET)
sarefin@cuet.ac.bd

Internal Examiner:
(Member)

Dr. Md Khalilur Rhaman
Associate Professor
Department of Computer Science and Engineering
Brac University
khalilur@bracu.ac.bd

Internal Examiner:
(Member)

Dr. Md. Ashraful Alam
Assistant Professor
Department of Computer Science and Engineering
Brac University
ashraful.alam@bracu.ac.bd

Supervisor:
(Member)

Dr. Md. Golam Rabiul Alam
Associate Professor
Department of Computer Science and Engineering
Brac University
rabiul.alam@bracu.ac.bd

Program Coordinator:
(Member)

Dr. Amitabha Chakrabarty
Associate Professor
Department of Computer Science and Engineering
Brac University
amitabha@bracu.ac.bd

Head of Department:
(Chair)

Sadia Hamid Kazi, Ph.D.
Associate Professor
Department of Computer Science and Engineering
Brac University
skazi@bracu.ac.bd

Abstract

Cryptocurrencies are the new form of trade that has revolutionized how we look into our financial institutions. Bitcoin dominates the industry with the highest market share among the hundreds of other cryptocurrencies. However, high energy consumption leading to increasing carbon emission, prioritizing high-value transactions, and long waiting times are some of the flaws preventing it from reaching its full potential. Due to the block rewards getting halved every four years, miners and researchers are fearful that this would be the breaking point of Bitcoin's success. One of the ways to tackle and hopefully reduce this problem while bringing wider adaptability is by ensuring faster transactions. Currently, Bitcoin has an average block size of 1MB, which many researchers and enthusiasts believe is insufficient. To tackle these limitations, we have proposed two different ideas. Our first concept proposes an industry 4.0 compliant next-generation Bitcoin architecture by introducing a dynamic and sustainable block concept. Using our improved knapsack algorithm, a priority-based 0/1 knapsack and advanced priority-based 0/1 knapsack, we can ensure a balanced transaction selection, quicker verification, higher transaction throughput, reduced carbon emission, and increased earnings for the miners. Moreover, with the addition of only one of our proposed sustainable blocks, we can cut down verification times by 50% and increase throughput by 2.56 times. We can also reduce carbon emissions per transaction by 62.318%, which would help reduce Bitcoins' large carbon footprint, enabling us to approach greener digital transactions.

In the second concept, we further try to improve the block sizes using the help of machine learning and artificial intelligence. Our proposed model analyzes the network's activity, such as incoming transaction frequency and other aspects, to adjust block sizes. The model can predict block sizes with 61.12% accuracy, and we can see a positive change in the amount of fees earned by miners (9.3%), transaction count and transaction per second (66.75%). With the help of our model, Bitcoin would be able to dynamically change the block size based on the transaction activity, resulting in shorter wait times, thus increasing wider adaptability and sustainability.

Keywords: Bitcoin, Blockchain, Knapsack, Sustainability, Machine Learning.

Acknowledgement

My heartfelt gratitude to my supervisor, **Dr. Md. Golam Rabiul Alam** Associate Professor, Department of Computer Science and Engineering, BRAC University, for his unwavering support throughout this research endeavor.

I would like to show my appreciation towards the reviewers from IEEE transactions on industrial informatics who have helped increase the quality of the paper with their valuable feedback. I would also like to thank my parents for their support. I would not be in this position now if it weren't for them.

Finally, I would like to express my gratitude to all of the faculty members, staff, students, and other Brac University stakeholders for their direct and indirect aid during various events of my education and research.

Table of Contents

List of Figures

List of Tables

Nomenclature

The next list describes several symbols & abbreviation that will be later used within the body of the document

AdaBoost Adaptive Boosting

$APBBK$ Advanced Priority Based Binary(0/1) Knapsack

BTC Bitcoin

CDD Coin Days Destroyed

$CO2$ Carbon Dioxide

DT Decision Tree

GBR Gradient Boosting Regressor

HPT High Priority Transactions

ICT Information and Communication Technology

IoT Internet of Things

$LGBM$ Light Gradient Boosting Machine

LPT Low Priority Transactions

$LSTM$ Long Short Term Memory

MAE Mean Absolute Error

MB Megabyte

MFS Mobile Financial Services

ML Machine Learning

MLR Multiple Linear Regression

MSE Mean Squared Error

$PBBK$ Priority Based Binary(0/1) Knapsack

PoA Proof of Activity

PoC Proof of Capacity

PoS Proof of Stake

PoW Proof of Work

$R2$ Coefficient of Determination

RFE Recursive Feature Elimination

RFR Random Forest Regressor

$RMSE$ Root Mean Squared Error

SB Sustainable Block

SBM Sustainable Block Model

SDG Sustainable Development Goals

TPS Transaction per second

$XGBoost$ Extreme Gradient Boosting

Chapter 1

Introduction

1.1 Background

Cryptocurrencies are rapidly gaining popularity throughout the world. To date, there are 10,846 cryptocurrencies globally with a total market capitalization of $1,388,386,220,001, where Bitcoin and Ethereum hold the first and second positions with 45.1% and 17.7% shares, respectively [23], [29].
Along with bitcoin, industry 4.0 is the latest buzzword that is changing the way we look into the use of technology in the industry. In summary, it refers to the fourth industrial revolution that is enabling gradual convergence between industries and information and communication technologies (ICT), including big data analytics, Internet of Things (IoT), and cloud computing platforms. The recent integration of Bitcoin technologies with Industry 4.0 is advancing the digital transformation of industries further and enabling unprecedented horizontal and vertical connectedness and collaboration across the global value chain [4]. Therefore, cryptocurrencies like Bitcoin are a vital part of industry 4.0 as it has paved a new path for a decentralized financial system. One of the targets of industry 4.0 is to make the industry connected to ensure sustainable development goals (SDG) [18], [21], [22]. Unfortunately, Bitcoin cannot comply with such goals due to its existing architecture. For example, a single block generation in Bitcoin emits 191 tons of CO2, which is 15 times more intensive than the equivalent amount of gold (in dollars) [39], [40]. So clearly, this needs to be managed as 2759 transactions (per block on an average) should not have such a significant impact on the environment [17], [39].
Bitcoin has other limitations, such as, on average, it takes 10 minutes for a transaction to be validated and added to a block [45]. However, a total of six (average 60 minutes) confirmations ensure that the transaction is entirely valid and would not be removed or tampered with [35]. Currently, Bitcoin can process 4.6 transactions per second (TPS), which is significantly less compared to its competitors, such as Visa, which can process 1700 transactions/second [17], [25]. Furthermore, in his paper [1] creator of Bitcoin, Satoshi Nakamoto mentioned that Bitcoin would no longer be generated when it has reached the 21 million threshold [1]. This gradual stopping of Bitcoin production is done by a process called halving, where every four years (since the starting of Bitcoin), the block fee would be halved. For example, in 2019, block rewards were 12.5 BTC, and currently, in 2022, it is 6.25 BTC. However, as the block rewards decrease and if the price of Bitcoin does not rise, at some point in time, there would not be enough miners to validate transactions [26] as it would

not be profitable for them. The only way to make it sustainable is to tackle such situations through transaction fees. However, the number of verified transactions produced every 10 minutes approximately is not enough to even balance out the resources used by the miners [31], [37], [38]. So researchers are working on finding ways to make Bitcoin sustainable even after reaching the 21 million mark.

1.2 Research Objective

To tackle the barriers mentioned earlier, we have proposed two different ideas in chapter 4 and 5. Both ideas have the same goal of making Bitcoin sustainable in the long run. The objectives of the research work are as follows:

Boost wider adoption of Bitcoin: There are numerous issues afflicting Bitcoin and its growth. Some are very obvious, like high energy consumption, and some are not so apparent from a high-level perspective. Among them, we have inclinations toward high-value transactions and long transaction verification times. These are creating a significant barrier to broader adoption. Miners earn from both the block reward and the transaction fees. As the process is very resource heavy, they focus on larger, more hefty transactions as it has more fees associated with them. As a result, we do not see smaller transactions in the network. Furthermore, due to this biasness, these transactions have to wait a very long time to get accepted, pushing people further away from using Bitcoin as other financial services seem more convenient.

Lowering energy consumption: It is no secret that Bitcoin's biggest problem is its high energy consumption. To tackle this, we need to change the core technologies being used in Bitcoin, namely the proof of work consensus algorithm. Many researchers have even proposed different ways to handle this energy consumption issue, but Bitcoin is still reluctant to incorporate those. So we need a different strategy to lessen this energy consumption and reduce carbon emissions.

Adaptability and modularity: This reluctance to adopt better ideas to the Bitcoin network is understandable as doing so would bring considerable changes to the protocol. So the target of our work is to propose a system that does not change the core ideas and technologies of Bitcoin and keeps the protocol untouched. Also, our objective is to make the ideas adaptive so that based on demand/congestion, they can scale up or down and bring modularity to the system.

1.3 Research Methodology

This research work would be able to make the world's largest cryptocurrency sustain in years to come, even after reaching the 21 million Bitcoin milestone. The proposed methodologies are not only making it sustainable in the adoption space but also addressing some environmental problems. To prove the system's capabilities, we needed real-world Bitcoin data flowing through different parts of the structure. Due to the nature of Bitcoin, data was readily available for anyone to use. The transparency aspect of Bitcoin even ensures that anyone can see transactions occurring

in real-time.
Luckily, we had access to a Bitcoin data dump which enabled us to collect the necessary data. A simple automation script helped to collect transaction information, mempool conditions, and all past block data generated every day [46]. These 1893 days of data assisted us in understanding the long-term behavior of Bitcoin, how the transactions navigate through the network, their types, sizes, patterns, and how they impact the block generation process.
The first proposed concept is based on the century-old knapsack algorithm. This algorithm is famously used for resource allocation, which is one of our targets as we want to have a balanced usage of the blocks. We had to tweak the algorithm accordingly to fit the Bitcoin network. For the second idea, we used 12 different widely used machine learning algorithms to see how they compare when predicting block sizes in real-world conditions.

1.4 Research Contributions

To achieve our goals and solve all the aforementioned problems, in chapter 4, we propose a sustainable Bitcoin model (SBM) and a balanced transaction selection process derived from a modified knapsack algorithm called priority-based 0/1 knapsack (PBBK) and advanced priority-based 0/1 knapsack (APBBK). Combined, three of these ideas can increase transactions throughput/TPS and reduce wait times. The proposed model can also make a balanced transaction selection, breaking the usual preference for high-value transactions, resulting in broader acceptability. Another remarkable outcome of the idea is its ability to reduce carbon footprint per transaction, thus making Bitcoin greener. One of the strengths of the model is its ability to adapt. It can change based on how the network or the moderators decide, meaning the network can use as many proposed sustainable blocks (SB) as they require. Even with the addition of only one SB, we see improvements all across the board. The key contributions of this research are:

- This research introduces an industry 4.0 compliant sustainable and dynamic block that ensures a balanced transaction selection.
- The proposed sustainable blocks (SB) use insignificant resources compared to the number of resources it takes to create the original block (OB) in the existing Bitcoin model. Thus significantly reducing carbon emissions for processing transactions.
- Modified knapsack algorithms are proposed to ensure quicker transaction verification and higher transaction throughput.
- This model enables the miners to earn more than the current network architecture due to the dynamic nature of the model.

Furthermore, we have also proposed another distinct idea in chapter 5, combining Bitcoin architecture with machine learning (ML) to predict the ideal block size. In this concept,

- The proposed model minimizes Bitcoin's long wait times by forecasting the optimal block size based on network congestion and adjusting it to demand. All while maintaining the same resource usage and ensuring blockchain-level security.
- With the adaptive block sizes recommended to miners on every block creation cycle, Bitcoin's crippling transaction throughput has increased, assuring higher profits for miners.
- Our proposed methodology addresses the lack of Adaptability in Bitcoin during peak hours by examining nine distinct data points and dynamically adjusting block sizes.

To summarize, both of the ideas would be able to bring a positive change to Bitcoin in the years to come.

1.5 Research Limitations

Among the many issues afflicting Bitcoin, we focus on some specific sectors of it. In doing so, we have faced some limitations. The first one is the lack of complete data for training and testing the model. The data we collected to understand the network's congestion and flow did not have the information for every block creation time. Instead, it had an average for the day, which we had to match and create a one-to-one relationship. Some features had considerable missing values to the point that it was not feasible to do preprocessing on them. If we had access to the complete data from the start of Bitcoin, we possibly could have much more accurate results. Moreover, because both proposed ideas are novel, there is not enough work in this space, especially when incorporating machine learning into Bitcoin, as most of these papers focus on price prediction and security.
Keeping our objectives in mind, we would not be modifying or replacing the current Bitcoin protocol as that does not fall within the scope of our work. Furthermore, another massive issue plaguing Bitcoin, the high transaction fees, is also beyond the scope of this research.

1.6 Document Outline

The first concept is organized in the paper as follows: in section 4.1, we elaborate SBM's architecture. We then move to section 4.2, focusing on the different transaction selection processes and their related findings. In section 4.3, we combine our transaction selection processes and ideas to generate the final model. Section 4.4 discusses our findings regarding the model with respect to actual data. Then we talk about the features and accomplishments of our model in section 4.5, and the following section discusses our limitations and future work.
The second proposed idea is organized as follows: in Section 5.1, we talk about our proposed model and the collected data. Section 5.2 shows the results for all of the prediction models and comparison between them. The following Section 5.3 shows us the results using real-world transaction data. In sections 5.4 , we discuss our limitations and future work.

Chapter 2

Contemporary Bitcoin Architecture

Bitcoin started back in 2009 as the first digital cryptocurrency with the creation of the genesis block. It uses a decentralized system, meaning no central entity controls the flow of transactions [1], [25]. This is how cryptocurrencies like Bitcoin and Ethereum differentiate themselves from other fiat currencies by removing intermediaries, as shown in Fig 2.1.

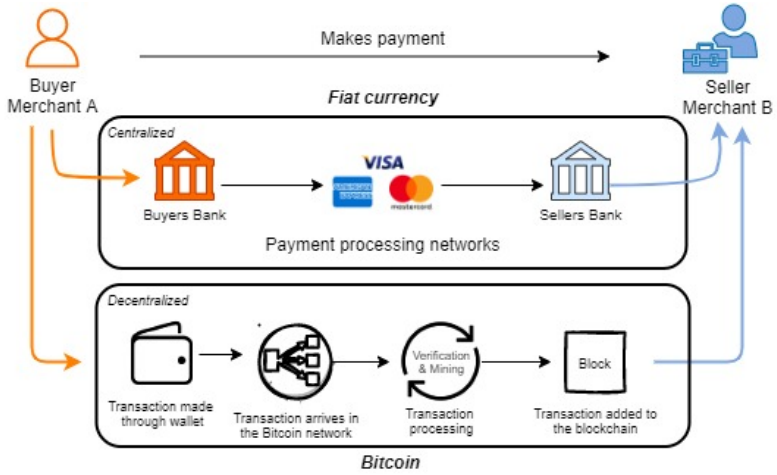

Figure 2.1: Comparison between centralized financial system (fiat currency) and decentralized system (Bitcoin).

Bitcoin uses blockchain technology which refers to a P2P distributed ledger that keeps track of transactions between participants in the network by storing the information in a sequence of blocks [41]. Blocks have their own hash value and contain the hash value of their previous blocks, which helps them keep connected [8]. These blocks contain information that is validated and protected through encryption mechanisms. [28]. Blockchain uses a unique data structure called the Merkle tree that helps to keep the history of all the transactions added to blocks.
The Bitcoin protocol uses the proof of work (PoW) consensus algorithm for mining coins which is the most widely used consensus algorithm in public blockchains [26]. This mining process can be done individually by a miner or collectively using a pool where multiple miners join together to solve the puzzle. The puzzle's difficulty is changed based on the mining capacity of the whole network and altered every 2016

block using the following formula,

$$T_{new} = T_{prev} \cdot \frac{P_{actual}}{B_c * BT_{avg}} \tag{2.1}$$

Here, T_{prev} represents the target value of the previous 2016 blocks, and P_{actual} represents the actual generation time of the previous 2016 blocks. T_{new} is the new target value that will be used to generate the following 2016 blocks. Furthermore, B_c represents the number of blocks (2016) before the difficulty is adjusted, and BT_{avg} represents the average block creation time.
A Miner adds values (nonce) to the hash to check if the hash value is below the threshold. If it is, then s/he is the winner; if not, the process goes on. When a miner creates a new candidate block, s/he adds a coinbase transaction to the top of the block. A coinbase transaction is a unique transaction considered as the block reward for the winning miner. Currently (2021), the coinbase transaction is 6.5 BTC. After solving the puzzle, at least 51% of nodes in the chain have to verify that it satisfies equation (2.2) [8], [25], [38]. Here, $B_{outputs}$, B_{inputs} and B_{reward} refer to the block outputs, block inputs, and the block reward the miner is getting for solving the puzzle, respectively. If any block does not comply with the equation, that means there are invalid transactions inside the block that a mischievous miner passed. Suppose such a situation occurs; this block is not propagated further and thus, not added to the chain.

$$\sum B_{outputs} \leq \sum B_{inputs} + B_{reward} \tag{2.2}$$

Bitcoin transactions work similarly to traditional cash payments. In simple terms, the users transfer and receive BTC using software-based wallets. Here they mention the input, which is the amount they are giving, and the outputs are the cost of the service or product and the amount in return. Finally, the fee and related Bitcoin addresses are mentioned. The system follows eq (2.3) to calculate the fees.

$$\sum B_{input} - \sum B_{output} = T_{fees} \tag{2.3}$$

One of Bitcoins' most prominent issues is scalability, as the system cannot adapt based on the demands such as increased transaction processing. Moreover, even though the average is 10 minutes, it does not ensure that the transaction is inside the next block. As the miners also earn from the transaction fees, they would first take the more significant transactions. This is making Bitcoin biased towards larger transactions. As a result, we rarely see small transactions in Bitcoin, whereas other competitions like banks regularly see all sizes of transactions.

Chapter 3

Related Work

Research has been going on to address the slow transaction verification, more balanced transaction selection, and increasing security in Bitcoin. Many blockchain-based consensus algorithms have addressed the problems with PoW and proposed their own such as proof stake (PoS), proof of activity (PoA), proof of capacity (PoC). Some cryptocurrencies are using them and have seen the light of success. Bitcoin's direct competitor is Ethereum, a second-generation cryptocurrency that also uses the concept of decentralization and an account-centric blockchain model. One of the key differentiating factors of Ethereum is that the block intervals take on an average of 15s and the transaction speeds are almost double of Bitcoin [11], [23], [24].
To tackle the faster transaction verification problem in 2021, Chris Larsen and Jed McCaleb introduced Ripple (XRP), a closed cryptocurrency ecosystem. Ripple is similar to the very famous Swift network used in Banking today as it can make transactions within roughly 4 seconds. Ripple has gone out of the traditional way of using PoW and PoS; instead, they use a consensus protocol to validate account balances and transactions on the system, which has the same goal of preventing double-spending and increasing integrity [34]. Currently, XRP holds the 7th position in market capital, costing 0.941$ per coin [30]. Furthermore, they are focusing on going public with their coin, marketing the concept of sustainability.
Using the concepts of XRP and PoS, researchers in their paper [6] proposed their own coin, which addressed the round trip time(RTT) to ensure transactions are validated within 3 seconds. They use their proposed algorithm known as Random checker proof of stake (RCPoS), which selects validators at random to ensure security. One of the critical features of their implementation is that every second a block is validated. Unlike PoS, where the amount of coin the miner holds determines who would validate the block, the system gives equal chances for all the nodes. P coin uses a 65% threshold, meaning at least 65% of the network has to agree that the block is valid. RCPoS selects checkers (validators validate the block and get the reward, checkers validate the newly created block) based on random bits, enabling them to undertake the 51% attack problem.
To benefit the miner's authors in paper [7] proposes a multi-leader multi follower scheme based on the Stackelberg game model. Their paper focused on increasing rewards for the miners (leaders and followers) and having a better resource distribution. They have also proven their theory by comparing it with the current mining pool strategies and outperforming them in simulated environments.
Authors in paper [16] discuss the optimal block size when we reach the 21,000,000

Bitcoin limit. According to them, the miners do not go out of their way to cheat the system as the resources and likeness of them winning is very low; however, when there would not be any block fees, miners are more likely to try to cheat the system. They give us a game-theoretic approach to a miner's payoff based on the block size. Due to how the system is designed, there can be multiple solutions for the puzzle, and there can be cases where two miners find the solution to the same problem at the same time, which can lead to forks. Which does not mean that the network will hold off on transactions until this gets fixed. Miners choose one of the branches and move on. The result is the longest branch wins, whereas the other gets reverted, and the transactions get pushed back into the mempool. However, the biggest concern is the size and propagation speed of the block. The bigger the size, the more time it would take to propagate. Now think of it in this way, if 2 deliver persons have to send an item to the same location, but one is 10kg, and the other is 100kg. The person carrying the 10kg load would do it faster. So increasing the block size to get more rewards and, most importantly, getting the consensus depends on the size and the number of transactions inside. The authors come to the agreement that block subsidiaries are the key to ensuring proper security. And, with the help of game theory, they recommend that a block size of 4MB would be ideal to avoid mischievous behaviors in the network [16].

Earning more from block creation has always been the goal of miners and researchers alike. In order to achieve higher transaction throughput while reducing confirmation times, payment channels have been proposed in the past that work on top of the main chain/layer. This reduces the stress on the network but also keeps the process safe. The payment channel concept was proposed to solve the scalability problem of bitcoin. They are focused on high frequency, and smaller transactions in general which can also be marked as micropayments or streaming payments [25].

In the paper [31], the authors discuss different ideas proposed by others on increasing block sizes to improve TPS. They conclude that going through this approach would result in larger block sizes requiring increased time to propagate throughout the network, causing an increase in forks. They propose using directed acyclic graphs (DAG) to address such problems and introduce parallelism to the system. They ensure that the original chain and the consensus protocol are not affected when such structures are implemented. They propose a smaller block with more relaxed puzzle requirements and variable rewards for the miners. As a result, this reduces the dependency on mining pools. Their implementation also reduces latency, thus increasing faster consensus and higher TPS.

RepuCoin is another proposed cryptocurrency that is resilient to 51% attacks as it relies on a miner's reputation, which they earn overtime. So it is basically how much validation work a miner has done in the network, and its regularity, which the authors named "miners integrated power." They also have safety guards in place in the sense that if a miner deviates from the typical system, his/her reputation will take a hit, ensuring that even powerful miners are kept on a leash. Their method increases security and increases transaction throughput to 10k/second, which is much higher than VISA or Mastercard. One of the critical features of RepuCoin is that as time goes on, the network becomes more secure [20].

With the introduction of the SegWit protocol, block size increased due to the removal of signature data from transactions. Digital signatures accounted for nearly 65% space of a single transaction. As the bulk of the information was separated from

the blocks, it opened up space for more transactions. This protocol upgrade's goal was to ensure the sustainability and higher levels of security of the Bitcoin network as it was gaining popularity. This upgrade theoretically increased the size to 2MB [34] however, in real life, that is not the case.
In the paper, [3] the authors discuss how different block sizes impact the performance and stability of the Bitcoin network. They initially show us previous works suggesting different methods proposed to change the block size and their impacts, such as increasing transactions to 8MB as soon as 75% of the miners agree. The author shows that increasing block sizes increase delays in the end-to-end transmission, which grows linearly depending on the block size. They also prove that larger block sizes have more chances of splits that would cause the consensus protocol to ditch a large number of blocks and also introduce scalability problems.
In another paper [13], the authors suggest changing the transactions information by removing wasteful and redundant transaction data to carry more transactions per block. The author reduced the output index to 1 byte and transaction hash information to 5 bytes from 4 and 32 bytes, respectively, ensuring a total size reduction of 11-16%.
Authors in paper [33] incorporate ML methodologies to increase the security in the Bitcoin network. Their research shows us the impact this model would have on the miner resource. Their testing on real-world data in their prototype testing environment shows that semi-supervised ML models perform worse than supervised ones, and logistic regression performs the best, impacting the lowest mining rate.
We have seen multiple papers using ML methodologies to predict Bitcoin prices. These studies compare ML models like the convolutional neural network, long short term memory, gated recurrent unit, generalized linear model, boosting, linear, logistic regression, and others. These price prediction papers show us that different aspects impact the price of Bitcoin, such as the network activity and daily trends [12] . Even attributes that do not seem prominent include tweets posted in the context of bitcoin [14], price of crude oil, stock market [44].
These studies have assured that people are concerned about the fate of Bitcoin, and research is going on to make it sustainable by various means while keeping its core features intact. However, throughout all these works, we see sacrifices to reduce the size to increase speed and vice versa or proposing new consensus algorithms. Moreover, no one addresses that the Bitcoin network is biased towards high-value transactions. Our models are developed, taking all these ideas and limitations into consideration. For the first concept shown in chapter 4 we have made the system dynamic to the point where it can go back to the current method without significant changes. Furthermore, we have not changed block size but were able to address both the long wait time and biasness issue, enabling our paper to be a novel idea. On the other hand, in chapter 5 we combine ML models to generate the ideal block size on every block creation based on the network activity. We have kept in mind the ideas and limitations of large block sizes and propagation speeds and generate the optimal block size every cycle, thus also making this a novel idea.

Chapter 4

A Sustainable Bitcoin Model for Industry 4.0

4.1 Proposed Sustainable Bitcoin Model (SBM)

The core idea of our proposed model is to make Bitcoin sustainable in the years to come while complying with industry 4.0 standards. As previously mentioned, Bitcoin is biased towards high-value transactions. As a result, we do not see its usage in typical transactions such as grocery shopping. These low-value transactions take a prolonged time to get accepted into the blockchain. So we propose a model that handles different types of transactions in their specific blocks. The OB structure would be handling large transactions with a miner-focused/fee-driven selection system. On the other hand, the proposed SB would be handling minor transactions where the customers/users require quick confirmations while maintaining the same chain. All of this is achieved while not increasing energy consumption.

4.1.1 SB Generation Process

One of the visions of our model is to keep the core features and ideas of Bitcoin intact while introducing a better sorting mechanism and creating specialized blocks. Currently, an OB is generated every 10 minutes on average, and they mostly contain very high-value transactions. Along with OB, we propose adding SBs in the middle based on a specific duration T_{SB}. The block creation threshold B_{time} would be between 1 to 9 minutes. We would have a 1-minute gap, so the block correctly propagates throughout the network, thus reducing the chances of orphan blocks. The interval is calculated using equations (4.1) & (4.2). Here C_{SB} refers to the count of SBs fixed by the network. T_{window} is calculated based on the subtracted value from average block creation time BT_{avg} and the propagation threshold $P_{threshold}$.

$$T_{SB} = T_{window}/C_{SB} \tag{4.1}$$

Where

$$T_{window} = BT_{avg} - P_{threshold} \tag{4.2}$$

If we look into an example concerning Fig. 4.1, we are targeting to create 3 SBs so C_{SB}=3. As previously discussed the propagation threshold would be $P_{threshold} = 1$

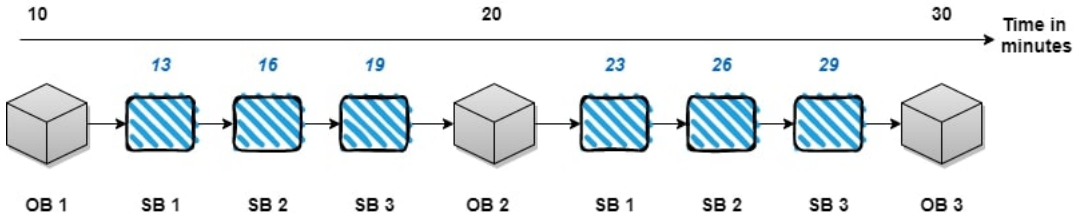

Figure 4.1: An example of integrating SB's

minute and the average block creation time is considered as $BT_{avg} = 10$ minutes. If we plug these values into equation (4.1) & (4.2), we will get 3 minutes or 180 seconds as the interval period of each SB creation. After the first OB is created (OB 1), three new SBs are created having a 3-minute interval. After the creation of SB 3, there is a 1 minute propagation time, and at the 20-minute mark, another OB is created. This process would continue unless the network decides the number of SBs would increase or decrease. Furthermore, due to the introduction of SBs, we have updated the mining fee process. Now only SBs would give the miners a block fee rather than OBs, ensuring that SBs are never skipped by miners, which would go against the whole concept of making it sustainable. A detailed description of transaction fees is discussed in section 4.1.4.

4.1.2 SB Architecture

As previously discussed, SBs would contain more user-focused transactions than focusing on the miners' benefits. However, integrating SBs into the blockchain requires a minor tweak in the header structure. The current Bitcoin header structure is shown in table 4.1 [5]. This header structure does not have any way to show what the next block is. This is obvious as the next block has not been created yet; thus, there is no hash value. We propose to change this structure to as shown in table 4.2.

Table 4.1: Current block header structure

Field	Size	Data
Version	4 bytes	Little endian
Previous block hash	32 bytes	Big endian
Merkle root	32 bytes	Big endian
Time	4 bytes	Little endian
Bits	4 bytes	Little endian
Nonce	4 Bytes	Little endian

In table 4.2, we can see that the block header has an extra section, 'Next SBs block hash' (this new attribute would also be added into OB's header structure). The change is implemented so that when the miners create their candidate OB, they would simultaneously create an SB that would be added after the OB. This enables the winning miner to add the following block hash to the OB header information as s/he has already created the block. The process above will have **two consequences**. Firstly, the miner has locked the SB candidate block, meaning it would not take any new transactions as that would change the hash. The second one is that it would

Table 4.2: SB block header structure

Field	Size	Data
Version	4 bytes	Little endian
Previous block hash	32 bytes	Big endian
Next SBs block hash	32 bytes	Big endian
Merkle root	32 bytes	Big endian
Time	4 bytes	Little endian
Bits	4 bytes	Little endian
Nonce	4 Bytes	Little endian

prevent any other miners from creating an SB as only the winning miner would have the candidate SB with the correct hash value. After a specific time, T_{SB} the miner would propagate the candidate SB which would be verified by the network and added to the blockchain. However, before propagating, the next SB would be created whose hash value would be added to the concerned SB (for our case SB 1). After the block is verified and added, the winning miner would receive his/her reward. If the miner after $2*T_{SB}$ has not created any SB, the OB block would be considered invalid, and the network would remove it and create a new OB. For our previously discussed example, we would have 3 SBs. After creating an OB, it has connected both ways to its previous and future next block. For the last SB block, in our case, the 3rd SB would have null for the 'Next SB's block hash' representing this is the last SB to be created in the cycle. Thus, not hampering the OB creation process. This structure ensures that the correct miner is always rewarded. Due to the network having full control over how many SBs will be created, they can choose not to create any SBs. In that scenario, the block structure of OB would be the same as the current one. SBs are constantly created, and we would have a fail-safe if they are not generated in time. Fig. 4.2 can assist in visualizing the proposal (the above explained ideas) where we can see a diagram of how the blocks are connected. Here OB2, OB3, SB1, SB2 have the new header, and SB3 has the old header structure.

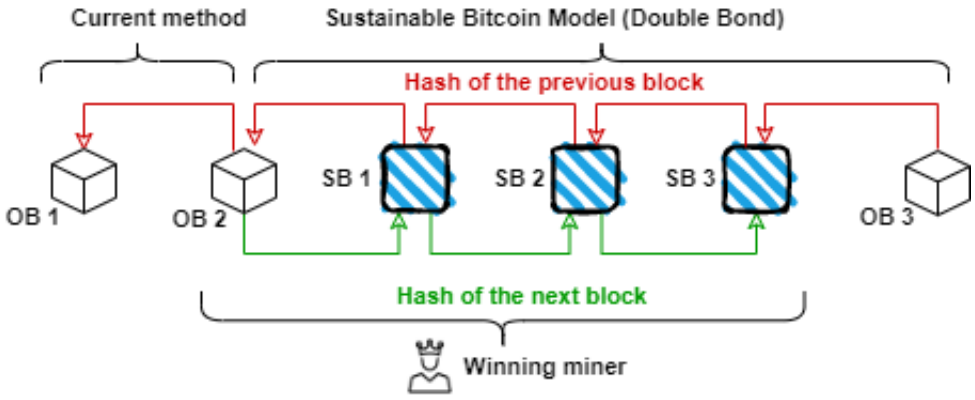

Figure 4.2: Connection between different blocks

4.1.3 SBM Workflow

If we look into the flow diagram in Fig. 4.3, we can see how the whole mining process works from start to finish. Every miner takes transactions from the mem-pool and

creates a candidate OB and SB. After the candidate block is created, miners start the mining process using PoW on OB. The miner who solves the puzzle first adds the OB to the blockchain. In the metadata of OB, s/he adds the hash of the SB candidate block, and in the SBs, metadata adds the OBs hash, which creates a **double bond** between them. Before propagating the SB, if there are more SBs to be made in the cycle, then the next SB is generated. Its hash is added to the SB to create the double bond. The SB block is added to the chain and propagated through the network after T_{SB} time has passed. The process goes on until the last SB is created. When another new OB is created, the cycle repeats.

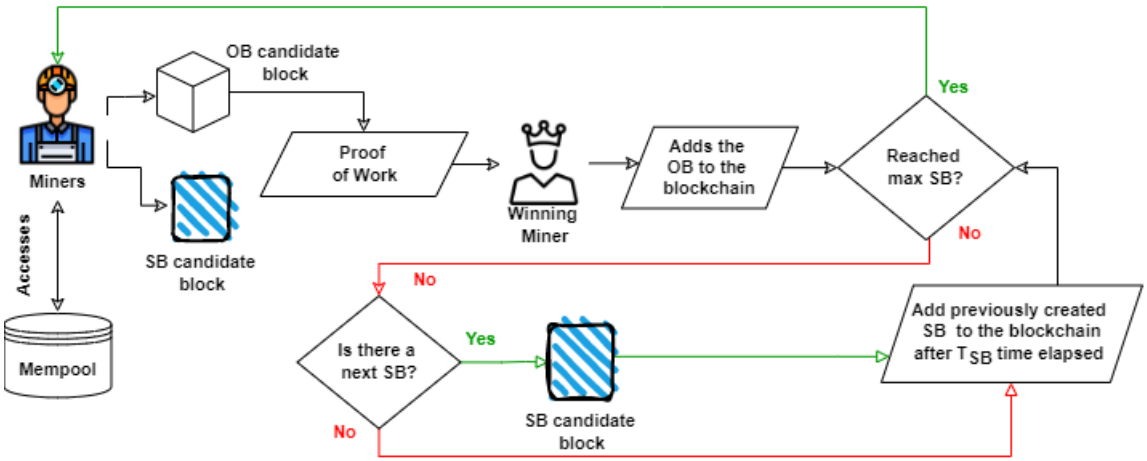

Figure 4.3: Work-flow diagram of the proposed SBM

4.1.4 Miners Earnings

In terms of transaction fees, the miners would earn much higher than the current approach due to the introduction of SB's. After an OB is created, the mem-pool (M_P) would adjust the transactions based on the equation (4.3). Here, all the transactions that are added in the newly created OB are individually removed from M_P. The same procedure is also followed for all the transactions located inside SB as they share the same pool, shown in equation (4.4). The total fee (T_f) accumulated for all the transactions the miner has added in both the OB and SB's is the sum of their amount, which can be seen in equation (4.5).

$$M_p = M_p - \sum_{i=1}^{n} T_{OB}[i] \tag{4.3}$$

$$M_p = M_p - \sum_{i=1}^{s} T_{SB}[i] \tag{4.4}$$

$$\sum_{i=1}^{n} T_f[i] + \sum_{j=1}^{C_{SB}} (\sum_{k=1}^{s} T_f[k]) \tag{4.5}$$

As mentioned before in section 4.1.2, the most significant difference from the current approach is that OB blocks would no longer give miners any block fees. Instead, the

fees would be divided among the SB's following equation (4.6).

$$R_{SB} = \frac{B_R}{C_{SB}} \tag{4.6}$$

Here each block reward is represented by R_{SB}, and the current block fee is represented by B_R. Looking into the current reward structure, we can see that the miner would be earning the following amount (based on the example mentioned in Fig. 4.1). R_{SB}=6.5/3=1.167 BTC for each block. If 3 SB's are created, when a miner generates an SB, he would be given 1.167 BTC as a reward. Failure to create SBs would mean that the miner would lose earnings for finding the nonce for the OB, as previously discussed. To summarize, the total final earning of the miner is the sum of all the transaction fees from both SB and OB and the block reward.

4.2 Transaction Selection

Transaction selection is the prerequisite for candidate block creation. Miners can select and add any transaction they see fit as long as the block has space for it. Even though the situation mimics the 0/1 knapsack problem, it can not be appropriately mapped due to the data format. In our testing, we have used the transactions of block 642120 and limited the block size to 500kb instead of the average one MB size limit for faster block creation. Due to the block being nearly 1.3 MB with its 1790 transactions, limiting it to 1 MB would not make sense. The results could not be generated even after running the algorithm for an hour (Ryzen 2600x, 16GB of ram, IDE: eclipse). Investigating the matter, we found out that the issue was with the data. Due to the dynamic programming nature of 0/1 knapsack, the problem is broken down into smaller portions to ensure faster results. In our case, the size is broken down into tiny portions such as 0.001, 0.002, and 0.003 MB, resulting in a vast number of columns. The result might be generated at some point; however, when miners try to create their candidate blocks, they want the results within seconds. This is due to the limited time they have to solve the nonce. So, the more time a miner is taking in selecting transactions, the fewer chances they are getting to solve the puzzle; as a result, the 0/1 knapsack approach is not suitable.

4.2.1 Proposed PBBK Model

We propose assigning values to transactions from 0 to 2 based on the fee and the fee to size ratio to solve the previously mentioned problem. These values are called fee-based priority P_{FB} and ratio-based priority P_{RB} , respectively. Firstly, we sort the transactions in ascending order based on the fees. The transactions belonging to the 25 percentile are assigned 0, ones in the 75 percentile get 2, and the rest are assigned 1. This is because we want to see the priority of the transactions. People who want their transactions to be added to the block quicker pay a good amount of fees. Even though the Bitcoin wallets suggest the minimum fee needed for the transaction to be added to the block, if the user pays more, it ensures a higher possibility of getting picked up by miners quicker. So lower the fee, the lower the priority value. The same logic goes for the fee to size ratio. The concept is simple; a person has limited space to rent. As a result, they will offer the space to the person who would give them the most benefits. After this value assigning process is

complete, these values are multiplied with one another to generate the final priority value P_T shown in equation (4.7). Thus, we would see 4 as the highest possible value and 0 as the lowest. Each transaction now has a non-fractional value which solves our initial problem.

$$P_T = P_{FB} * P_{RB} \tag{4.7}$$

The second issue is that we cannot use the block size as our sack size because the transactions have fractional size values. So when we subtract the transaction size from the block size, it would again generate a fraction that deviates from our goal. To solve this matter, we assign a priority limit value instead of a block size which has the same concept as a typical sack size. Originally we subtracted the weight of the picked item from the sack then calculated for the others. Likewise, the priority value P_T of each transaction would be subtracted from the total priority capacity mimicking the original algorithm, as more significant transactions would cut a large portion from the capacity.

Formulation For PBBK

PBBK's end goal is to ensure the miners earn the highest possible fees which is explained using equations (4.8), (4.9) and (4.10). Here n refers to the total amount of transactions inside the mem-pool. Each transaction size is referred to by TW_i, and its corresponding fee is indicated using TF_i. Adding transactions is a binary option represented by X_i. Individual transactions would have their priority value which is represented by TP_i. The maximum size of the block is defined by B_W, Finally, the priority value limit is refereed by B_p which would be traversed from $BP_{min} = 500$ to a network decided limit BP_{max}.

$$max \sum_{i=1}^{n} TF_i.X_i \tag{4.8}$$

s.t

$$\sum_{i=1}^{n} TW_i.X_i \leqq B_W \tag{4.9}$$

$$\sum_{i=1}^{n} TP_i.X_i \leqq B_P \tag{4.10}$$

The total fees earned in each B_p would be calculated based on Algorithm 2.
After going through Algorithm 2, it would generate transaction fees for each B_p. The system would then compare the results to select the ideal B_p, and based on that; the transaction would be chosen for the candidate block.

Numerical Analysis

In tables 4.3 and 4.4, we see different priority limit values generate different results (the transactions are sorted based on the priority value). The red and orange highlights represent the highest and second highest values, respectively. Due to the value of B_P being very irregular, no patterns could be recognized from this.

Algorithm 1: Transaction selection using PBBK

Data: Transactions in the mem-pool
Result: Total fees for each B_p
for *each* B_p *in* BP_{min} *to* BP_{max} **do**
  create an array PBBK[n, B_p]
  for *j from 0 to* B_p **do**
    for *i from 0 to n* **do**
      PBBK[i, j]=0
    end
  end
  for *i from 1 to n* **do**
    for *j from 1 to* B_p **do**
      PBBK[i,j]= max{PBBK[i-1,j], (TF_i + PBBK[i-1, j-TP_i)}
    end
  end
  B_p=B_p+100;
end

Table 4.3: Results for block 642120

Priority limit	Total earnings	Block size (KB)
500	7072.664327	980.457999
600	7470.319025	1016.982999
700	7706.710632	1025.650999
800	7557.184024	1024.645999
900	7557.184024	1024.645999
1000	7690.26343	1024.025999
1100	7683.222833	1024.930999
1200	7663.739728	1027.683999
1300	7579.118024	1024.090999
1400	7557.184024	1024.645999

On further inspection, we realize that our algorithm does not perform well compared to the greedy approach and the most beneficial technique ratio-based approach. In the greedy approach, the miners pick the most significant transaction, see if it fits, or not then move to the next one. The more significant fee to size ratio transactions are picked on the second approach. This approach generates the best earnings as the transactions are picked to provide the most benefit per byte. The results are shown in tables 4.5-4.10.
As mentioned before, using the greedy approach (table 4.6), we see a significant focus on large transactions. We see that transactions in the 75th percentile (based on fee) are all selected, whereas the low-fee transactions are not even touched. This situation is seen throughout all the blocks being tested. On the other hand, the most rewarding ratio-based strategy (table 4.7 and 4.8) has a somewhat balanced approach towards the selection. However, this still focuses on high fee transactions, which results in the most earnings among all the other approaches.
Even though our knapsack approach (table 4.9 and 4.10) has the worst performance among all the approaches, it does ensure that **100% of the low-fee transactions**

Table 4.4: Results for block 642265

Priority limit	Total earnings	Block size (KB)
500	7896.621228	1026.680999
600	7854.269625	1024.285999
700	4544.664505	693.164999
800	4999.517804	725.001999
900	8009.629214	1041.366998
1000	7902.958718	1024.712998
1100	7953.580615	1030.359998
1200	8041.904714	1040.834998
1300	7954.704714	1034.057998
1400	8009.629214	1041.366998

Table 4.5: Earnings and block size for the fee based greedy approach

Block	Total earnings	Total transactions	Size (KB)
642265	9815.777324	952	1024.025999
642120	8113.096298	837	1024.22
642267	11253.79042	982	1024.14
Average	9,727.55$	923.67	1024.128

are accepted. Moreover, it is the most balanced among all the methods, thus coming in second.
In another test scenario, we simulated a mem-pool containing nearly 72000 transactions collected from multiple blocks. P_{FB} and P_{RB} values were assigned to them, and the test data was run through the PBBK algorithm. For the result having the priority limit from 500 to 1700, we get nearly 113$, whereas other approaches generate more than 10,000$. After looking into the transactions, we found that only the small transactions were picked, and thus the whole block limit was occupied with them. The reason behind such occurrences is that we have assigned P_T =0 to low fee and low fee to size ratio transactions. During the transaction selection process, the system checks the final matrix value for each transaction, goes through the algorithm, and finds out if it is worth the space for benefits. The algorithm sees that transactions with a matrix value of 0 are not reducing the sack size (due to the priority value being 0) but generating benefits. So it considers these transactions and reduces the available block size. Sadly when there are large amounts of such small transactions with a P_T of 0, they start filling up the block. This is not a good approach for the miners as it brings them a negligible amount of earnings. On the positive side, this ensures that small transactions are picked. For our case, the total transaction for the block was above 3200.

Advantages Of PBBK

Even though PBBK is not the suitable path we are looking for, it still brings a few benefits. They are,

- It Ensures 100% selection of small fee and small ratio based transactions. For

Table 4.6: Transaction pick rate for the fee based greedy approach

Block	25th percentile	75th percentile	In between
642265	0/305 (0%)	52/52 (100%)	899/1305 (68.89%)
642120	0/447 (0%)	447/447 (100%)	390/896 (43.5%)
642267	0/645 (0%)	638/638 (100%)	344/1266 (27.17%)
Average	0%	100%	46.52%

Table 4.7: Earnings and block size for the ratio based approach

Block	Total earnings	Total transactions	Size (KB)
642265	10316.54842	1327	1033.9279
642120	9309.443505	1583	1082.28
642267	12790.04	2385	1024.892
Average	10,805.34$	1765	1047.03

example: paying for coffee should not require high fees, and the ratio would not be very high. With our approach, these transactions would always be added to the block.

- Making Bitcoin sustainable as smaller transactions are handled.
- A large number of transactions are put inside a block.
- Block size is close to the network decided limit.

If we look into the modified 0/1 knapsack approach from a social optimization aspect, it ensures smaller transactions get accepted into the network, which is not possible with the network's current approach. On the other hand, from a classic utilitarian perspective, it benefits the most people as multiple small transactions are being accepted, thus increasing usability and helping Bitcoin be sustainable in the long run.

4.2.2 Proposed APBBK Model

The drawback of PBBK is that it cannot generate benefits for the miners, thus making them less interested in mining. Furthermore, it cannot handle people cheating the system and paying less for more significant transactions. To solve this issue, we need a better approach to selecting transactions. For this, we have made a few changes to the process. This time we would look into the **fee**, **fee to size ratio** and **size** of the transaction for assigning values.

Formulation For APBBK

First of all, we would look into the fee-based priority value. Here, the transactions are sorted in ascending order then values are assigned. Fee priority values 1,2 and 3

Table 4.8: Transaction pick rate for the ratio based approach

Block	25th percentile	75th percentile	In between
642265	39/303 (12.871%)	50/52 (96.15%)	1238/ 1305 (94.866%)
642120	254/447 (56.82%)	441/447 (98.65%)	886/896 (98.89%)
642267	606/645 (93.95%)	610/638 (95.611 %)	1169/1266 (92.3%)
Average	54.55%	96.80%	95.35%

Table 4.9: Earnings and block size for the PBBK approach

Block	Priority	Total Earnings	Total Transaction	Size (KB)
642265	1100	9420.212903	1445	1038.121
642120	1700	7249.325088	1564	1024.99
642267	1600	10749.344	2029	1024.756
Average		9139.62733	1679.33	1029.289

are assigned to fees less or equal 25%, fees greater or equal 75%, and everything else, respectively. The same goes for the fee to size ratio and size-based priority values. After the assignment, we sum them together, generating values from 0 to 9. Table 4.11 can help better understand what these values represent.
After the process is complete, transactions having the final priority Value of 3 would be assigned to the SB mem-pool & the others to the OB mem-pool. Both OB & SB mem-pools are temporary and are created during candidate block creation, and this ensures that large transactions are always inside their appropriate blocks. These transactions would then be sorted based on the fee to size ratio concept discussed previously. The workflow diagram in Fig. 4.4 shows the path transactions take from the mempool to the final candidate block creation.

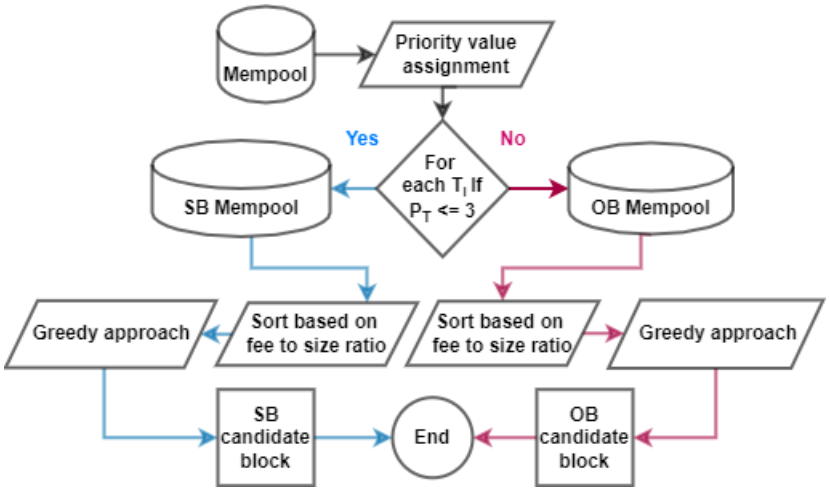

Figure 4.4: APBBK workflow

Table 4.10: Transaction pick rate for the PBBK approach

Block	**Priority**	**25th percentile**	**75th percentile**	**In between**
642265	1100	303/303 (100%)	50/52 (96.15%)	1092/1305 (83.68%)
642120	1700	447/447 (100%)	307/447 (68.68%)	809/896 (90.29%)
642267	1600	645/645 (100%)	388/638 (60.81%)	996/1266 (78.7%)
Average		100%	75.21%	84.22%

Table 4.11: Explanation on APBBK value assignment

Value (fee + size + free-size ratio)	Meaning	Examples
3 (1+1+1)	Low fee, small size, low fee to size ratio	Small payments. Example: grocery shopping.
9 (3+3+3)	High fee, large size, high fee to size ratio	Large transactions. Example: buying cars.
5 (2+1+2)	High Fee, small size, high fee to size ratio	Emergency transactions.

Numerical Analysis

To test our new model with APBBK, we have collected a total of 20,000 transactions from blocks 642120 to 642132. These transactions have gone through the sorting process, giving us the following division.

- **Low priority transactions (Low fee, small size, low fee to size ratio):** 1668
- **High priority transactions:** 18332

As previously discussed, OB only takes high-priority transactions (HPT). On the other hand, SB holds low priority transactions (LPT). The outcomes of APBBK are shown in table 4.12 and 4.13. Sadly, having the limitation of SB's only taking LPT causes the block to be more than 50% empty due to the lack of transactions. However, some transactions do not fall in the LPT criteria but should be eligible. An example is shown in table 4.14.

Table 4.12: Earnings from OB using APBBK

Total Earned	**Space left**	**Transaction count**
21132.98357	0.106000394	2620

So to make the process more in line with our goal, we would use the PBBK method in SB, and the OB remains the same because of its consistent performance. This will ensure a higher fee for the miners and assure that more transactions are processed.

Table 4.13: Earnings from one SB using APBBK

Total Earned	Space left	Transaction count
112.3758982	671.6429951	1668

Table 4.14: Example of a transaction from block 642122 that should be a part of LPT

Attributes	Values
Size (KB)	0.223
Fee ($)	0.4091
Fee/size ratio	1.8345
Priority fee	2
Priority size	1
fee/size ratio	2
Sum	5
LPT (3) HPT (>=4)	HPT

4.3 The Integrated Model

Finally, to achieve our goal of sustainability and benefiting miners and users alike, we have to take a different approach. The previous section shows that both PBBK and APBBK bring a welcome change; however, they have flaws that can be eradicated if merged. Below is a step-by-step block generation process using the combined concept.

1. When an OB cycle starts the miner takes all the transactions inside his mempool and sorts them based on fee to size ratio.
2. The sorted transactions go through the **PBBK** process.
3. The algorithm selects LPT to create the candidate SB.
4. The miner creates the candidate SB and removes these transactions from the mempool.
5. The remaining transactions go through the **APBBK** process.
6. APBBK algorithm differentiates HPT and LPT from the remaining transactions.
7. The miner uses HPT transactions and creates a candidate OB.
8. If all the HPTs are selected, and the OB still has space remaining, the miner would fill the rest of the space with LPTs.
9. The miner does the puzzle-solving process and, if s/he wins, they propagate the OB in the network with the hash value of the candidate SB (SB 1).
10. The miner removes the selected transaction in OB from the mempool.

11. If there are more SBs to be created, then the updated mempool (with the latest transactions coming in) goes through the PBBK process again. Otherwise, the cycle repeats.

4.4 Results

The earning through OB and SB for the combined model using the same 20,000 data is shown in table 4.15 and 4.16. Considering the system is based on one SB per cycle. The final tally would come to,

- **Miner earning from fees:** 22,307.87957$
- **Total transactions approved:** 7097

Table 4.15: Earnings from OB using SBM

Total earned	Space used	Transaction count
21132.98357$	1,023.894 KB	2620

Table 4.16: Earnings from one SB using SBM

Total earned	Space used	Transaction count
1174.896$	1024.009 KB	4477

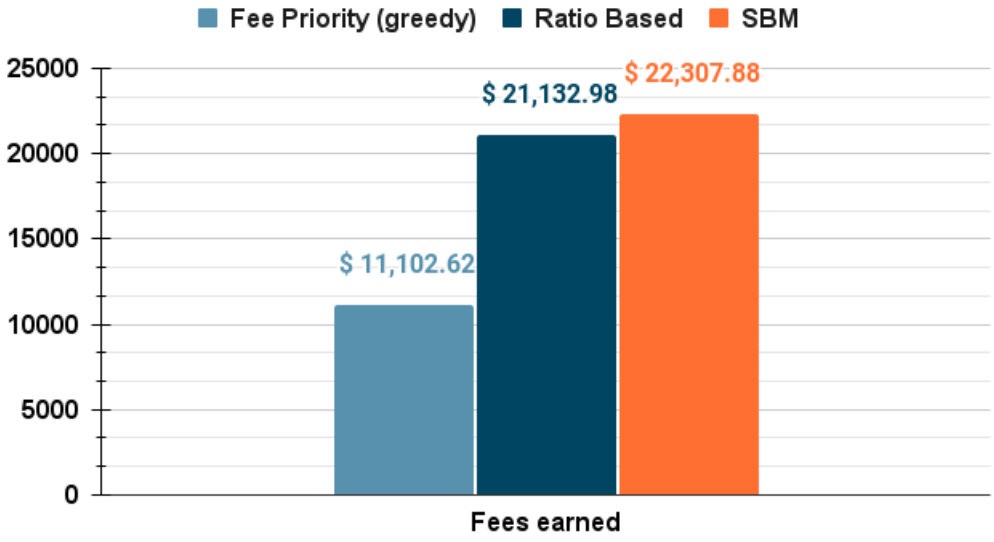

Figure 4.5: Fee earning comparison between the greedy approach, ratio based approach and SBM

We can see in Fig. 4.5 that in the greedy method, the max earnings would have been 11,102.62138$, whereas using SBM with PBBK, this bumps up to 22,307.87957$, and the number of transactions soared (Fig. 4.6). Even if we look at the average block transaction numbers, they are roughly between 1500- 2200 transactions. However, in our case, with nearly the same amount of energy, we are getting increased transactions and fees, thus reaching our goal.

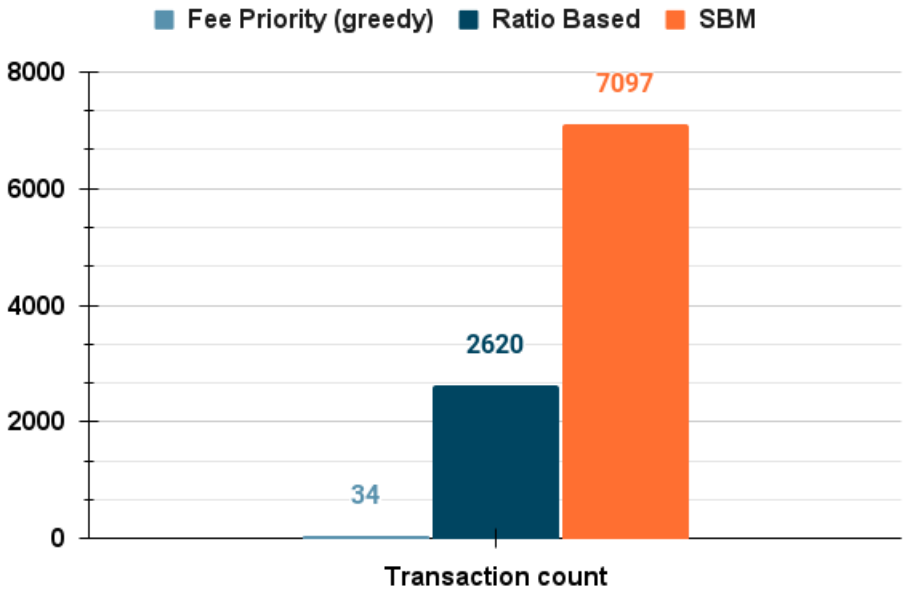

Figure 4.6: Transaction count comparison between the greedy approach, ratio based approach and SBM

4.5 Features & Accomplishments

The proposed SBM proves that a tweak in the Bitcoin architecture and optimal transaction selection positively impacts the cryptocurrency. To summarize, our work has the following contributions in making Bitcoin sustainable.

Increase transaction count and earnings for the miners: Due to the design of SBM, the network can decide on how many SB's are ideal. From Fig. 4.6 we can see that with the introduction of only one SB, the transaction count increased by **2.7 times** compared to the current ratio-based transaction selection. Furthermore, we can also see that with just one SB, TPS increased by **2.56 times** shown in Fig. 4.7. Similarly, with just 1 SB, the increased fees the miner would be getting is just a bonus with no increase in resource usage. As fees increase and block rewards decrease, this would lead to sustainability.

Faster confirmations: With the proposed model, blocks quickly get confirmations while maintaining blockchain-level security. Currently, transactions over 10,000$ dollars on an average require six confirmations, meaning 60 minutes of wait time. However, if one SB is considered, it can be reduced to 30 minutes, and if 3 SBs are considered, it can be done in 16 minutes which is a massive increase in performance. Fig. 4.8 shows the gradual decrease in wait times with different numbers of SB's.

Blockchain level security: SBM can maintain the same blockchain-level security with two types of blocks without increasing resource usage. Due to the architecture, no one except the winning miner can push the SB's and get their reward. The double bond concept ensures that SB's hashes are incorporated even before they are pushed. So it is almost impossible for others to match hash values and push their own SB's. Similarly, hackers can not hack the SB's and change the content as it would change the hash, thus breaking the bond and the OB creation cycle, alerting the whole network. Now, mischievous miners might want to put larger transactions

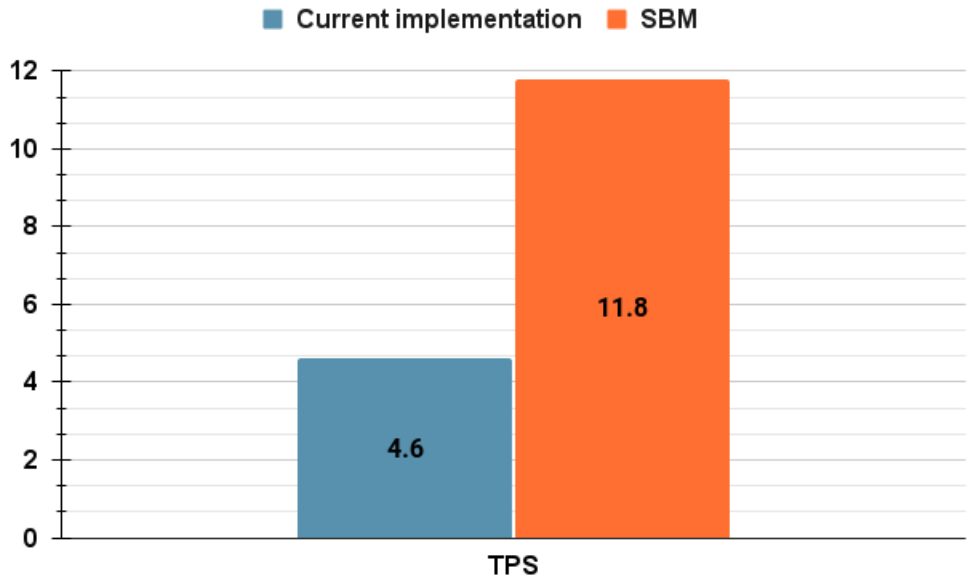

Figure 4.7: TPS comparison between current architecture and SBM

inside SB, thus defeating the purpose. To tackle such scenarios, the miners would validate each transaction (same as the current approach). If they see any outliers (calculated based on the transaction fees using the outlier formula) in SB, which holds more than 10% of the total SB's fee, they would mark the block as invalid and not propagate. This would ensure miners do not cheat the system as it would make all their created blocks (in the cycle) invalid, including the OB. Thus the miner will not be receiving any rewards for solving the puzzle. Due to this, the network would start a new OB creation cycle.

Reduced carbon emissions: As previously discussed in chapter 1 , every block on an average contains 2759 transactions and generates 191 tons of C02 [17], [40]. Therefore, we can conclude that each transaction produces 0.069 tons of carbon emissions on average. Whereas due to the dynamic nature of our model, this can be easily reduced. As previously discussed, we can safely assume SB's use insignificant resources compared to the amount of resources it takes to create an OB. This is due to not relying on any puzzle-solving process to generate SB's. As a result, we can ignore the cost/emissions for simplicity. Concerning the 10 minutes of average time to create an OB, we generated 7097 transactions with 191 tons of CO2 in one OB creation cycle for our previous example. Each transaction boils down to 0.026 tons of CO2, which is **62.318%** lower than the current implementation, as shown in Fig. 4.9. Furthermore, due to the nature of the model, it can increase or decrease but would always be equal or better than the current Bitcoin architecture.

4.6 Limitations & Future Work

In the previous section, we see how well SBM performs in reaching our goal; however, each proposed method has its own set of limitations. In terms of PBBK, we see the following,

- B_p is not a fixed value, so checking multiple ranges is required.

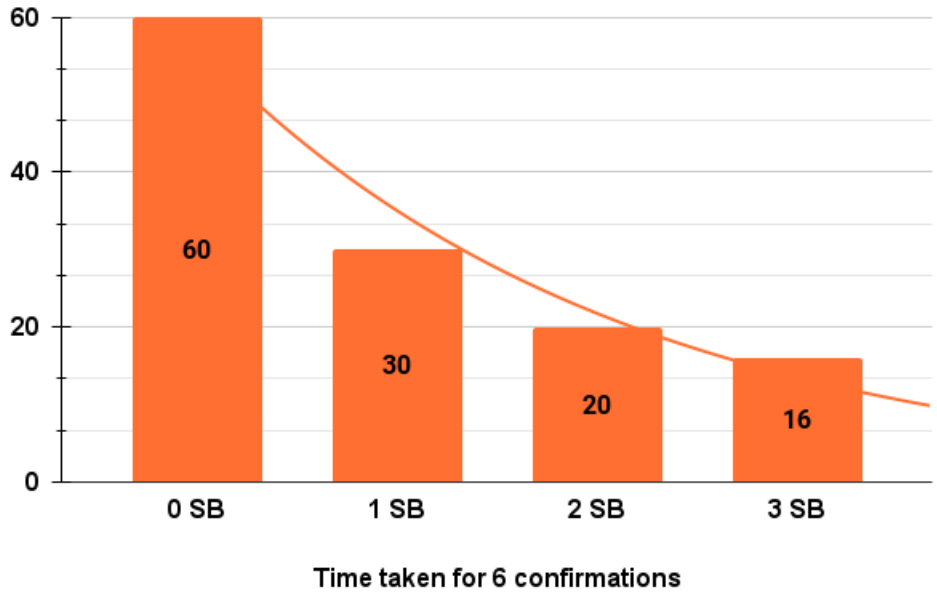

Figure 4.8: Gradual decrease in wait times for the 6th confirmation based on different number of SB's

- It is slower than the current methods.
- Does not provide enough transaction fee, especially when the mem-pool has a lot of small transactions.
- It can be misused by users to pay less for large transactions and get selected.

To solve a few of the limitations of PBBK, we introduced an advanced version that addresses some of the problems and brings in better transaction selection and organization. However, it has its shortcomings such as,

- Some transactions are not properly differentiated as shown in table 4.14.
- Due to the above case, there is a lack of LPT, which causes SB to be partially empty.

From above, we can see that both methods have their limitations and usefulness. Our final model combines these ideas and shows how the integrated model can help reach sustainability. However, the integrated model has its own challenges namely,
Increased processing: In the integrated model, we see the transactions going through 2 levels of processing. Initially, the transactions go through the PBBK algorithm, which identifies LPT to create SBs. Secondly, the remaining transactions go through the APBBK process to differentiate the HPT and LPT. This ensures the correct transactions are inside OB, and it is not empty if there are not enough HPT to fill the block. This two-level processing is undoubtedly time-consuming, taking away from the crucial puzzle-solving period.
Even though the time is negligible and every miner has to go through the same process, we believe this is a place for improvement. To further reduce processing time, both PBBK and APBBK selection algorithms should be combined so that the ideal differentiation can be done by passing through one process.

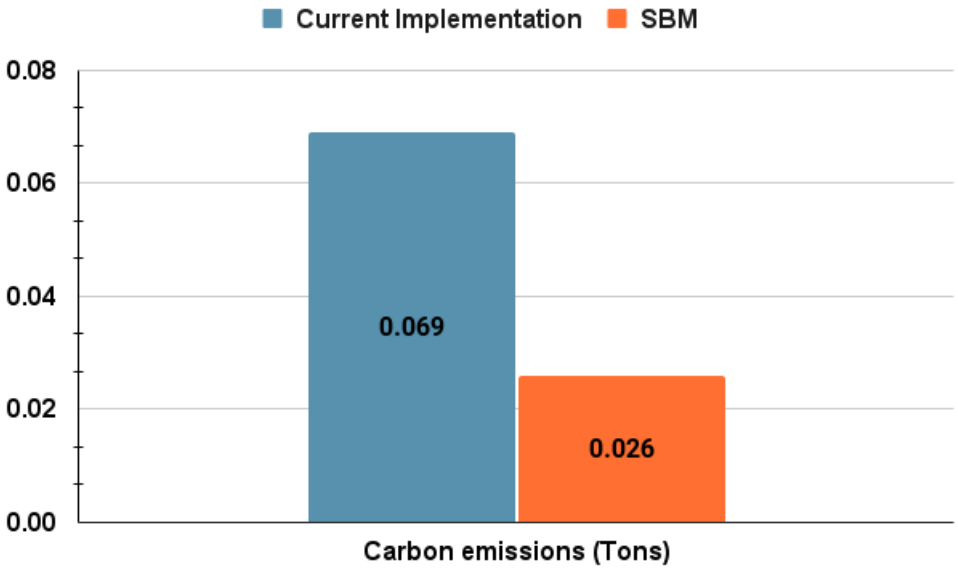

Figure 4.9: Comparison of carbon emission per transaction between the current architecture and SBM

Reliant on the moderators/miners: We mention that the number of SB can be changed by the network/moderators, enabling it to be dynamic. With more SBs, we can see a positive change as we can scale up or down based on the traffic/scenario. However, this relies on moderators and their judgement calls on the number of SBs to be created, which leaves areas of human error.
An AI-based model can be created to analyze the network traffic and other network-related attributes to select the ideal SB count and SB size, thus enabling it to break free of any dependencies.

Chapter 5

Predictive & Automated Dynamic Block Size Adjustment In The Bitcoin Network

5.1 Proposed Model & Data preparation

To keep Bitcoin alive in the following years, we need to ensure its sustainability. One of Bitcoin's limitations is, it can not adjust its rate dynamically based on the transaction throughput (transaction coming in). This has led to delays during high traffic, which is why Bitcoin is not as popular as other methods. An excellent example of this is during Christmas time when transactions increase. Financial institutions like VISA can adapt to this due to already having a very high throughput. However, Bitcoin can not adjust to this high traffic. As a result, someone buying Christmas presents might have to wait 30+ minutes to get their first confirmation. If they want to get the confirmation faster, they have to pay a large amount of fees. Which further pushes people away from Bitcoin. We propose an idea where the Bitcoin network can dynamically adjust itself to compete with others to tackle this problem. We introduce an AI-based model that would adjust the block size based on the network activity. In other words, if the network notices that the number of transactions has increased, the block size will increase as well, ensuring that more transactions are added to the chain. Figure 5.1 illustrates our proposal where during candidate block creation, our model would suggest the miner the optimal block size by analyzing multiple factors related to the Bitcoin network (factors are discussed in section 5.1.2).

5.1.1 Data Collection

One of the advantages of working with cryptocurrency, especially Bitcoin, is its transparency. Anyone on the internet can view blocks and the transactions associated with the block. However, that does not mean that the information related to the sender and recipient is visible. Sadly, for our case, as we would be analyzing the Bitcoin network, the process was not easy. As mentioned, getting the block information is easy; however, getting the network state at the time of block creation is troublesome to find. We required the transaction per second (TPS) of the Bitcoin network and the mempool stats for our case. To our surprise, a website

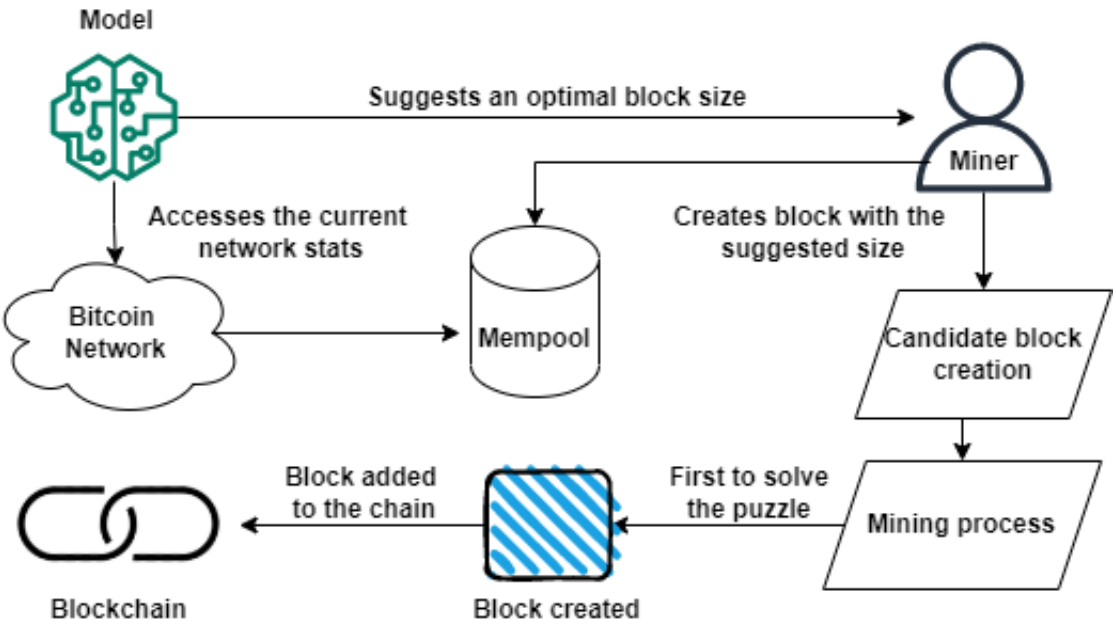

Figure 5.1: Workflow of the proposed model

contained these pieces of information. However, it contained the average value for the day, whereas we required hourly or 10-minute interval data. Additionally, a few dates were missing from different columns. Due to no other sources containing the required data, we had to work with what we had and map this data accordingly. To test our models, we have used the block data collected from June 14th, 2016, to August 20th, 2021. A total of **5 tables** were collected.

- **Transaction rate per second (TPS)**: Refers to the number of transactions added to the mempool per second [50]. The collected data showed how many transactions came into the Bitcoin network every second on an average for a day.
- **Mempool transaction count**: The total number of unconfirmed transactions in the mempool [49]. As people make transactions through Bitcoin, they pile up in the mempool. This table/data gives us the average count of transactions in the mempool for the day.
- **Mempool size:** The aggregate size in bytes of transactions waiting to be confirmed [47]. Similar to the table mentioned above, this data gives us the average size of the mempool for a day.
- **Mempool size growth**: The rate at which the mempool is growing in bytes per second [48]. Transaction rate differs based on time, exchange rate and other factors. This table gives us the average rate of change of the mempool size, giving us a deeper look into the network.
- **Block data**: Created and confirmed blocks from the mentioned time containing all the related information [46]. As we are dealing with block size prediction, we require previous block data. The block data table gives us all the information related to the generated block, such as the id, hash, Merkle root, size, weight etc.

To better understand the data, we first need to understand the mempool concept. Mempool is where all the unconfirmed transactions come in. Miners then take these transactions, validate them and add them to their candidate block. These candidate blocks go through the mining process, and if the miner can solve the puzzle first, their block is added to the chain. Then the network verifies the added block and all the transactions to ensure there was no wrongdoing. A large amount of transactions in the mempool means that the transactions are piling up, resulting in congestion in the network. This results in longer wait times and higher fees. To summarize, the first four tables help us understand the network's traffic. After merging the tables based on their timestamps, we get a total of **280563 rows of data with 42 columns**. Among them, the missing data count was

- **TPS**: 60132 rows
- **Mempool size**: 47735 rows
- **Mempool count**: 57735 rows
- **Mempool growth**: 65426 rows

To train our model, we need to handle these missing data issues, which is discussed in the next section.

5.1.2 Data Preprocessing

As there are missing data, this had to go through some preprocessing to make it perfect for our prediction models. The initial step was to run the data through an imputation process. So in our case, we have used a total of 6 approaches.

1. Remove null rows
2. Imputation using mean
3. Imputation using median
4. Imputation using most frequent
5. Imputation using deterministic regression
6. Imputation using stochastic regression

Through our research, we realized that among these six approaches, imputation using deterministic regression showed the best results. On the contrary, stochastic regression showed the worst result. Further investigation showed us that the randomness that stochastic regression brings was hurting the model's accuracy.
One of the challenges we faced while preprocessing the data was initially not approaching the problem from a time-series perspective. Time series data is handled differently compared to standard data. Keeping that in mind, we had divided our complete data into two parts: train data and test data with a 75/25 split. We have ensured that neither the test nor train data have past or future blocks, thus removing data leakage. Table 5.1 is a quick summary of the data.

Table 5.1: Test and train data stats

	Test Data	**Train Data**
Rows	70131	210432
Percentage	24.9965	75.0034
Date range	19/4/2020 - 20/8/2021	13/6/2016 - 18/4/2020

Then we ran the deterministic regression imputation method through each dataset to ensure there is no null data and no leakage from the train to test. We have not imputed the whole data as it would leak information to the test set, thus creating bias and inaccurate results. Table 5.2 shows a sample of the final data using deterministic regression imputation.
Now we need to set the target and feature for the models. As we are analyzing the network traffic so we would only be looking into a total of **9 features** given below.

1. **TPS** (*TPS*): Refers to the number of transactions coming into the network every second. The collection contains an average TPS for the day. An increase in TPS refers to more transactions coming into the network, thus increasing congestion. equation (5.1) shows the relationship between TPS (T_p) and network traffic (N_t).

$$T_p \propto N_t \tag{5.1}$$

2. **Mempool size** (*U_mempool_size*): Refers to the average mempool size of the miners. As previously discussed, mempool contains unconfirmed transactions. Unconfirmed transaction increase is proportional to the mempool size.

3. **Mempool count** (*U_mempool_count*): Refers to the average mempool count of the miners. Unconfirmed transaction increase is proportional to the mempool transaction count, which is also proportional to mempool size.

4. **Mempool growth** (*Growth*): This refers to how fast the mempool grows over time. Which, in turn, has an impact on network congestion. The higher the growth, the more increase in the size is expected. equation (5.2) shows a relationship between mempool size (MP_s), mempool count (MP_c) and mempool growth (MP_g).

$$MP_s \propto MP_c \propto MP_g \tag{5.2}$$

5. **Difficulty** (*difficulty*): To solve a block, miners have to solve a puzzle with X amount of difficulty. The difficulty changes every two weeks based on the network's hash rate and average block creation time. As the difficulty increases, the miners spend more time and resources to solve the puzzle; however, the reward is not increasing simultaneously. Thus the miners try to push more transactions to the block to get increased fees.

6. **Time** (*hour*): Through our research, we realized that time has a relationship with block size. We have seen that hours impact both network congestion and block size. So we have extracted the hour information from the block creation *median time* column.

Table 5.2: Sample data from the final test dataset

block_id	TPS	U_mempool_ count	U_mempool_ size	Growth	b_size_MB	difficulty	fee_per_kb	fee_per_kwu	cdd_total	hour
416203	3.1	25286.5	21.54333	1830.9	0.951815	1.96E+11	36151.7	9037.92	2130.987011	0
416710	2.316667	7712.5	23.45953	1014.517	0.00389	1.96E+11	234945	58736.3	0.179455634	12
417551	3.5	5815.5	7.950356	1247.15	0.951664	2.09E+11	47037.2	11759.3	12711.58452	22
418457	2.877145	8675	7.216364	737.4333	0.953673	2.09E+11	67735.6	16933.9	11875.9399	4
503534	4.216667	35853	127.131	1229.503	0.966603	1.93E+12	210951	53539.1	13302.66725	15
550628	3.316667	5125	3.104771	914.3333	1.210689	6.65E+12	23156.4	7356.07	57841.94316	21
573571	2.999703	8651.5	2.82867	1192.337	1.013703	6.35E+12	5168.41	1375.79	7777.038717	5

7. **Fee per kilobyte** (*fee_per_kb*): We came to realize that fees have a remarkable impact on the size of the block. When the network becomes congested with transactions during rush hours, people tend to give more fees than the wallet suggested amount. This gives those transactions a higher chance to get picked by the miners. This attribute is a good indication of the network's activity, which would help us make better predictions. equation (5.3) can further clarify the relationship between network traffic and transaction fees (T_f).

$$N_t \propto T_f \tag{5.3}$$

8. **CDD Total** (*cdd_total*): CDD, also known as "Coin Days Destroyed", helps us to measure the lifespan of coins that are transacted [43]. The CDD values show the number of coin days destroyed each day. The higher the number, the longer these coins accumulated prior to finally being spent. This would help us to understand the upcoming network activity and congestion.

9. **Fee per weight** (*fee_per_kwu*): After the SegWit soft fork, Bitcoin now has the concept of weight. Weight refers to the transaction base data along with witness data. Currently, the Bitcoin block weight limit is 4MB. This feature is similar to *fee_per_kb* as this shows us how much fee users are paying for each transaction in terms of its weight, giving us an indication of the network's activity.

Moreover, as we would be predicting the size of blocks, the target is the Size *(b_size_MB)* column. In Figure 5.2 we can see a heat map of the corresponding rows. We have also used a recursive feature elimination (RFE) method which gives us a ranking for the relevant features and eliminates the redundant ones, shown in Table 5.3.

Table 5.3: Recursive feature elimination (RFE) ranking for the selected features

Column Name	**RFE Rank**
U_mempool_count	1
fee_total	1
fee_per_kwu	1
TPS	5
fee_per_kb	6
U_mempool_size	9
hour	16
Growth	18
difficulty	30
cdd_total	34

The heatmap and RFE selected other features; however, they did not relate to the network or the block size; therefore, we had to discard them, thus finalizing these 9 features.

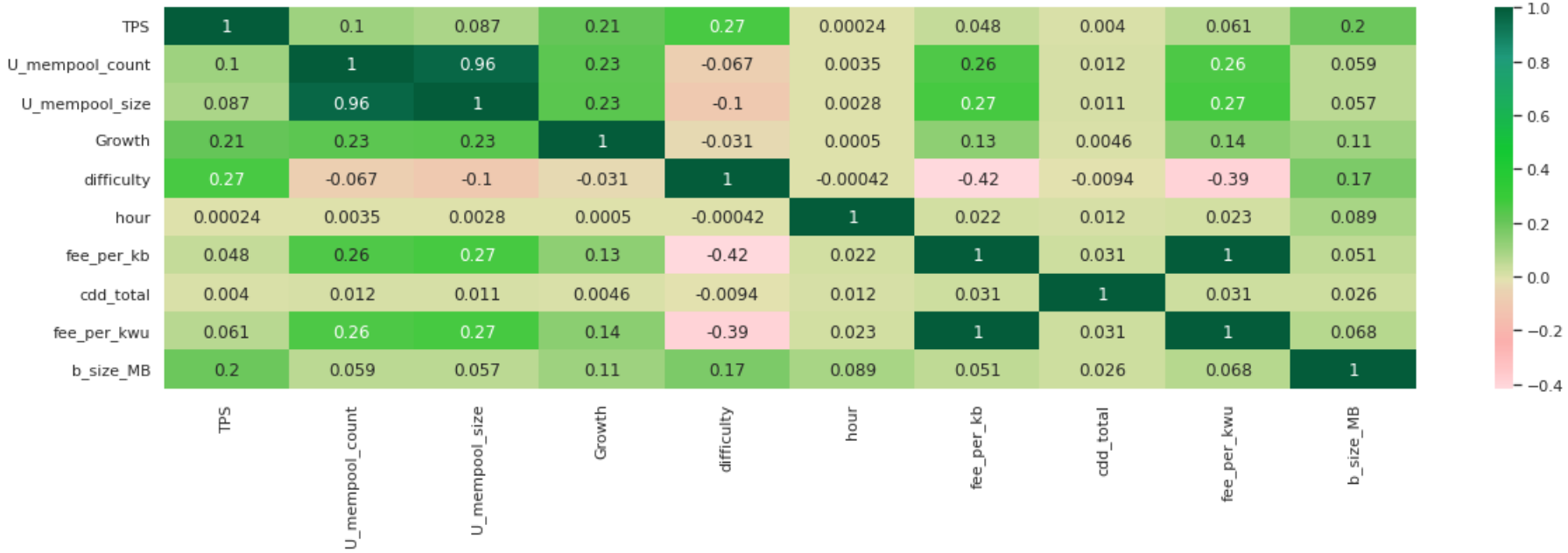

Figure 5.2: Heat map of the selected features

5.2 Models Used For Block Size Prediction

To reach our goal of finding the optimal block size based on the network activity, we need to find the model that would give us the best possible result. To do that, we have gone through multiple machine learning (ML) models. Due to the type of data we are dealing with, we can not use any classifications models, and we are limited to regression models only. We have gone through 12 algorithms, trained them, and tested them to find the best prediction through our research. For calculating errors, we have used metrics like mean absolute error (MAE), mean squared error (MSE), root mean squared error (RMSE) and coefficient of determination (R2), which helped us to pick the most efficient and accurate model. In the upcoming sections, we will go through each model and its results and finally compare to select the best one.

5.2.1 Performance Measures

Performance of machine learning models needs to be evaluated. A definition of the closeness between the desired value and the predicted value needs to be defined. This metric is known as ***cost/loss function*** and guides the neural network towards the desired outcome by estimating the closeness between the predicted and the desired value. The cost function is calculated at the end of every training iteration after the weights have been updated. The cost function that we used are known as ***mean absolute error***, which tries to minimize the average absolute error between the desired output and the predicted output, given by equation 5.4

$$\text{cost} = \frac{1}{n}\sum_{i=1}^{n}\left|Y_i - \hat{Y}_i\right| \tag{5.4}$$

$$\text{metric} = \frac{1}{n}\sum_{i=1}^{n}\left(Y_i - \hat{Y}_i\right)^2 \tag{5.5}$$

5.2.2 Decision Tree

Decision tree is one of the most popular machine learning models for prediction. It is a supervised model that uses a tree-like structure to reach the outcome. Each node on the decision tree is a condition or a decision that helps the model reach leaf nodes which are the results/predictions. Due to our data type, we have used a continuous variable decision tree. The model was initially performing slightly worse as decision trees are less appropriate for continuous attributes [32]. However, we have tuned the model to have the following: *max_leaf_nodes=2000, random_state=1, max_features=3, min_samples_leaf=7, min_samples_split=6*, which helped us to reach the best possible outcome from it. The results for the decision tree are given in Table 5.4 where we see the that best R2 score comes when the tree depth is set to 1000.

5.2.3 Random Forest Regressor

Random forest is basically multiple decision trees that can take the average to increase prediction accuracy. One of our reasons behind using random forest is that it generally has high accuracy, fits the model better and is efficient for large datasets such as ours [42]. To produce the best results we have tuned the model to have the

Table 5.4: Decision tree results

Tree Depth	MAE	MSE	RMSE	R2
5	0.205491	0.086452	0.294028	-0.01834
50	0.169	0.067183	0.259197	0.208633
500	0.159102	0.061648	0.248291	0.27383
1000	0.155736	0.061175	0.247335	0.279407
1500	0.157153	0.062368	0.249736	0.265352
2000	0.155593	0.062889	0.250776	0.259221
2500	0.155484	0.063537	0.252066	0.251578
3000	0.155236	0.063669	0.252328	0.250025
5000	0.156064	0.065828	0.256569	0.224601

following parameters: *random_state=1,min_samples_leaf= 5, min_samples_split= 21.* The results in Table 5.5 shows that at tree depth=500, we get the best R2 score.

Table 5.5: Random forest regressor results

Tree Depth	MAE	MSE	RMSE	R2
5	0.119769	0.049301	0.222039	0.419269
50	0.115237	0.046806	0.216346	0.448665
100	0.115221	0.046536	0.215723	0.451838
250	0.115124	0.046529	0.215705	0.451928
500	0.115006	0.046415	0.215442	0.453265

5.2.4 Multiple Linear Regression

Multiple linear regression, also known as MLR, uses multiple independent variables to predict the target variable. It is an extension of linear regression, which is limited to just one variable [36]. MLR is used in many financial predictions, such as oil price predictions, where multiple variables affect the forecast. Keeping it in mind, we have tested this model as we have nine independent variables driving the block's size. Table 5.6 shows us the results.

Table 5.6: Multiple linear regression results

MAE	MSE	RMSE	R2
0.202314	0.08359	0.28912	0.015368

5.2.5 Advanced Linear Regressions

To further test our data, we put it through some advanced linear regression models such as Ridge, Elastic Net, and Lasso. Ridge regression is used when there are independent variables that are correlated. In our case, we have TPS, mempool count, and size, so Ridge regression was one of our picks. Lasso regression stands

for Least Absolute Shrinkage and Selection Operator, which is a modified version of linear regression [19]. It uses the concept of shrinkage, where the data is shrunk towards the mean. Similar to Ridge, this is good for multicollinearity [15]. Ridge penalizes the model for the sum squared value of the weights, and Lasso penalizes for the sum of absolute values of the weights. Elastic net is a combination of both. Table 5.7 shows us a comparison between the best results from each model.

Table 5.7: Best results from Ridge, Lasso and Elastic Net regression

Models	**Best results from each model**				
	Alpha	**MAE**	**MSE**	**RMSE**	**R2**
Ridge	20000	0.20026	0.082795	0.287742	0.024734
Elastic Net	50	0.191484	0.080545	0.283804	0.051244
Lasso	50	0.193813	0.081377	0.285266	0.041442

5.2.6 Partial Least Squares Regression

PLS, also known as partial least squares regression, is used to find the relation between X and Y. As the name suggests, it does a least-squares regression on a smaller set of uncorrelated components. For our dataset, we have both correlated and un-correlated components/attributes. So PLS can give us some insights. The tuned results for PLS can be seen in Table 5.8

Table 5.8: Partial least squares regression results

N Components	**MAE**	**MSE**	**RMSE**	**R2**
7	0.198339	0.082059	0.286459	0.033412

5.2.7 Extreme Gradient Boosting

XGBoost is a decision-tree-based ensemble ML algorithm that uses a gradient boosting framework. In this method, the residual of all previous weak learners is continually corrected by adding new weak learners, and the final prediction result is the sum of multiple learners. When compared to other ML models, the XGBoost algorithm offers a faster computation speed and higher model performance. To avoid overfitting, tree model complexity is appended to the optimization objective as a regularization component.
Assume that we have a set of observations $(x_{11}, x_{12}, \ldots, x_{1k}, y_1), \quad (x_{n1}, x_{n2}, \ldots, x_{nk}, y_n)$, n is the number of observations, $x_{ik}(i = 1, \ldots, n)$ are the predictor variables, $y_i(i = 1, \ldots, n)$ is the response variable, and anticipates the outcome using M additive functions [2].

$$\hat{y}_i = \phi(\mathbf{x}_i) = \sum_{m=1}^{M} f_m(\mathbf{x}_i), \quad f_m \in \mathcal{F} \tag{5.6}$$

Table 5.9: XGBoost results

N Estimators	MAE	MSE	RMSE	R2
5	0.480129	0.261197	0.511074	-2.0767
50	0.158881	0.057815	0.240447	0.318986
500	0.112164	0.034562	0.185909	0.592882
1000	0.101416	0.032927	0.181459	0.612141
1500	0.099191	0.03313	0.182015	0.609758
2000	0.099269	0.033621	0.183361	0.603967

where $\mathcal{F} = \left\{f(\mathbf{x}) = w_{q(\mathbf{x})}\right\} \left(q : \mathbb{R}^m \to T, w \in \mathbb{R}^T\right)$ is the space of regression trees. Here q represents the structure of each tree that maps an example to the corresponding leaf index. T is the number of leaves in the tree. Each f_k corresponds to an independent tree structure q and leaf weights w. Unlike decision trees, each regression tree contains a continuous score on each of the leaf, w_i represents score on i-th leaf. For a given example, the decision rules are used in the trees (given by q) to classify it into the leaves and calculate the final prediction by summing up the score in the corresponding leaves (given by w). The iteration process for XGBoost algorithm is

$$1 \leq \mathrm{m} \leq \mathrm{M}; \quad F_{m+1}(x) = F_m(x) + f_m(i) \tag{5.7}$$

with an objective function of:

$$L(\phi) = \sum_i l\left(\widehat{y}_i, y_i\right) + \sum_m \Omega\left(f_m\right) \tag{5.8}$$

where,

$$\Omega(f) = \gamma T + \frac{1}{2}\lambda \|w\|^2 \tag{5.9}$$

Here l is a differentiable convex loss function that measures the difference between the prediction $\widehat{y}_i$ and the target y_i. The second term Ω penalizes the complexity of the model. From the previous equations we get:

$$\mathrm{L}(\phi) = \sum_i l\left(\hat{y}^{(m-1)} + f_m\left(x_i\right), y_i\right) + \sum_m \Omega\left(f_m\right) + \mathrm{C} \tag{5.10}$$

According to the second order of Taylor Expansion, the objective function could be transformed to:

$$\sum_{j=1}^{M}\left[\left(\sum_{i \in Ij} g_i\right)\omega_j + \frac{1}{2}\left(\sum_{i \in Ij} h_i + \lambda\right)\omega_j^2\right] + \lambda T \tag{5.11}$$

where g_i and h_i are the first order and second order derivatives respectively. And they become constant in the t-th iteration. XGBoost algorithm starts from one node and splits nodes continuously. In the process of splitting nodes, similarity score is used to select appropriate feature variables and splitting points, so as to find the appropriate tree structure. A **learning rate of 0.11** was used. Hyperparameter optimization was performed using grid search to obtain the best results which are shown in Table 5.9 for different number of estimators.

5.2.8 Other Boosting Algorithms

At its core, boosting algorithms attempt to improve predictions by converting weak learners to strong learners. Boosting algorithms are very popular in modern data science-based research works. Its vast popularity is due to outperforming typical ML models like RFR. We have used three different types of boosting algorithms other than XGBoost for our testing. To acquire the best result these models were tuned using grid search. Tuning for each algorithm is shown below:

- **Gradient Boosting Regressor (GBR)**:

  n_estimator=5, random_state=1, min_samples_leaf= 26, learning_rate= 0.7, min_samples_split= 6, max_depth=50

- **LightGBM (LGBM)**:

  n_estimator=5, learning_rate=0.25, random_state=39

- **ADABoost (ADAB)**:

  n_estimator=50, learning_rate=0.06

We have compared all the boosting algorithms and the results are shown in Table 5.10. Here, we can see that XGB performs the best with a **61.21%** score on R2, whereas ADAB performs the worst, most likely due to poor model fitting.

Table 5.10: Best results from GBR, LGBM, XGB and ADAB

Models	**Best results from each model**				
	N Estimator	**MAE**	**MSE**	**RMSE**	**R2**
GBR	5	0.126321	0.050778	0.225341	0.401868
LGBM	5	0.110082	0.035593	0.18866	0.580746
XGB	1000	0.101416	0.032927	0.181459	0.612141
ADAB	50	0.198656	0.076453	0.276502	0.099439

5.2.9 Long Short Term Memory

Long Short-Term Memory (LSTM) networks are a type of recurrent neural network capable of learning order dependence in sequence prediction problems. This is a behavior best-suited in complex problem domains like ours. In this work, LSTM cells are used as the nodes of recurrent neural networks (see Figure 5.3). In an LSTM cell there are extra gates, namely the input, forget and output gate that are used in order to decide which signals are going to be forwarded to another node. W is the recurrent connection between the previous hidden layer and current hidden layer. U is the weight matrix that connects the inputs to the hidden layer. $\widetilde{C}$ is a candidate hidden state that is computed based on the current input and the previous hidden state. C is the internal memory of the unit, which is a combination of the previous memory, multiplied by the forget gate, and the newly computed hidden state, multiplied by the input gate.
The equations that describe the behavior of all gates in the LSTM cell are described in the following equations (5.12), (5.13), (5.14), (5.15), (5.16), and (5.17).

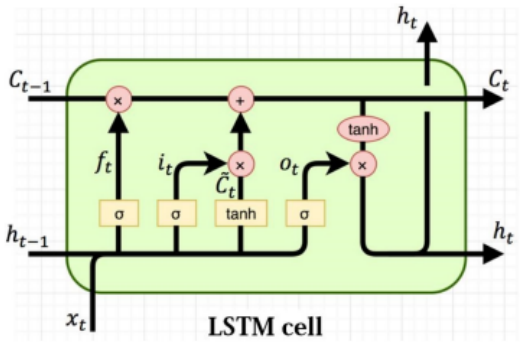

Figure 5.3: Structure of an LSTM cell

$$i_t = \sigma\left(x_t U^i + h_{t-1} W^i\right) \tag{5.12}$$

$$f_t = \sigma\left(x_t U^f + h_{t-1} W^f\right) \tag{5.13}$$

$$o_t = \sigma\left(x_t U^o + h_{t-1} W^o\right) \tag{5.14}$$

$$\tilde{C}_t = \tanh\left(x_t U^g + h_{t-1} W^g\right) \tag{5.15}$$

$$C_t = \sigma\left(f_t * C_{t-1} + i_t * \tilde{C}_t\right) \tag{5.16}$$

$$h_t = \tanh\left(C_t\right) * o_t \tag{5.17}$$

The way neural networks solve problems is not by explicit programming but rather by "learning" the solution based on given examples. In this research, we are focusing on supervised learning. At the end of the training, the neural network infers the right outputs through ***generalization***. A metric, cost/loss function, is calculated at the end of every training iteration after the weights have been updated. It guides the neural network towards the desired outcome by estimating the closeness between the predicted and the desired value.

$$\text{cost} = \frac{1}{n}\sum_{i=1}^{n}\left|Y_i - \hat{Y}_i\right| \tag{5.18}$$

$$\text{metric} = \frac{1}{n}\sum_{i=1}^{n}\left(Y_i - \hat{Y}_i\right)^2 \tag{5.19}$$

In this work, normalization was performed before initializing the model, to rescale continuous features that use different scales and ranges, for stable training. We have used **two LSTM layers** and **one dense output layer** for a sequential model that compiles with MAE and MSE as loss function and metric, respectively, using the Adam optimizer. As the Keras regression metrics do not comprise the R2 score, we opted for MSE, MAE, and RMSE (calculated from MSE) to compare with other models. The model fits with a **batch size of 128 and 50 epochs** seemed to deliver

the best result. Different rates for dropout layers have been tried out to avoid overfitting, and thus we settled on **0.2**. **KerasTuner** was used for hyper-parameter optimization. The results can be seen in Table 5.11 , where different dropout rates have been compared.

Table 5.11: LSTM loss and metrics for discrete dropout rates

Dropout	**Loss (MAE)**	**Metric (MSE)**	**Validation Loss (MAE)**	**Validation Metric (MSE)**
0.2	0.1490	0.0608	0.2919	0.1166
0.5	0.1660	0.0714	0.2991	0.1194

5.2.10 Model Comparison

After comparing the models' best results, we see that XGBoost has the highest accuracy with **61.21%**, whereas MLR has the lowest performance with 1.53%. In terms of mean absolute error (MAE), we see that the lower performing models such as MLR and LSTM have a very high error value than XGBoost, which has the lowest coming in around 0.101416 MB. Similarly, the mean squared error is also reasonable where XGB having 0.032927 MB and coming in a close second is LGBM with 0.035593 MB. A side by side result comparison can be seen in Tables 5.12 and figure 5.4 to 5.7. The predicted block size values generated by the models are fractional, as expected. However, when telling the miners the block size limit, we cannot give them a fractional number, so to keep it consistent, we resorted to ceiling the numbers to the closest multiple of 100. For example, we see predictions like 1.033 MB, 1.204 MB, 1.159 MB, which would be converted to 1 MB, 1.2 MB and 1.2 MB, respectively.

Table 5.12: Prediction model comparison

Model	**MAE**	**MSE**	**RMSE**	**R2**
DT	0.155736	0.061175	0.247335	0.279407
RFR	0.115006	0.046415	0.215442	0.453265
MLR	0.202314	0.08359	0.28912	0.015368
Ridge	0.20026	0.082795	0.287742	0.024734
Elastic Net	0.191484	0.080545	0.283804	0.051244
Lasso	0.193813	0.081377	0.285266	0.041442
PLS	0.198339	0.082059	0.286459	0.033412
GBR	0.126321	0.050778	0.225341	0.401868
LGBM	0.110082	0.035593	0.18866	0.580746
XGB	0.101416	0.032927	0.181459	0.612141
ADAB	0.198656	0.076453	0.276502	0.099439
LSTM	0.2991	0.1194	0.3455	N/A

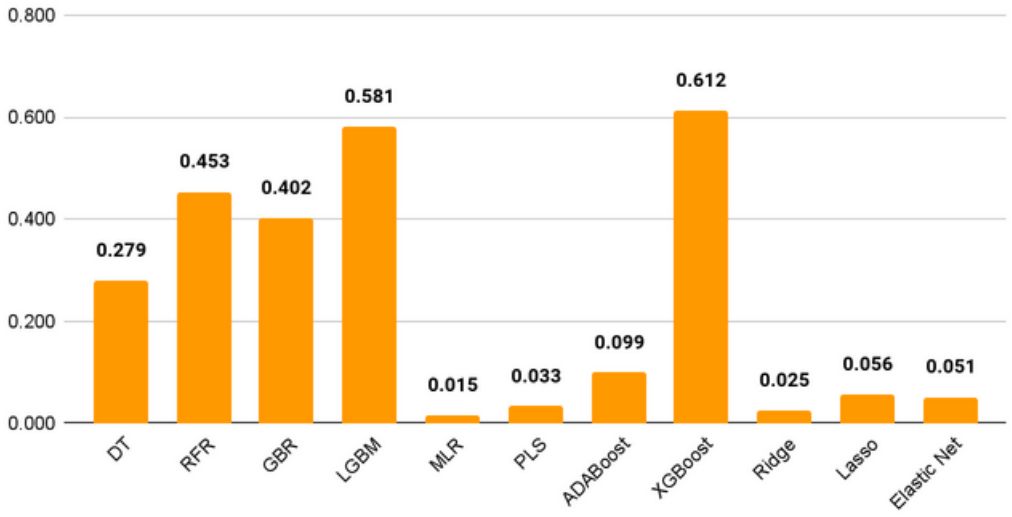

Figure 5.4: R2 score model comparison

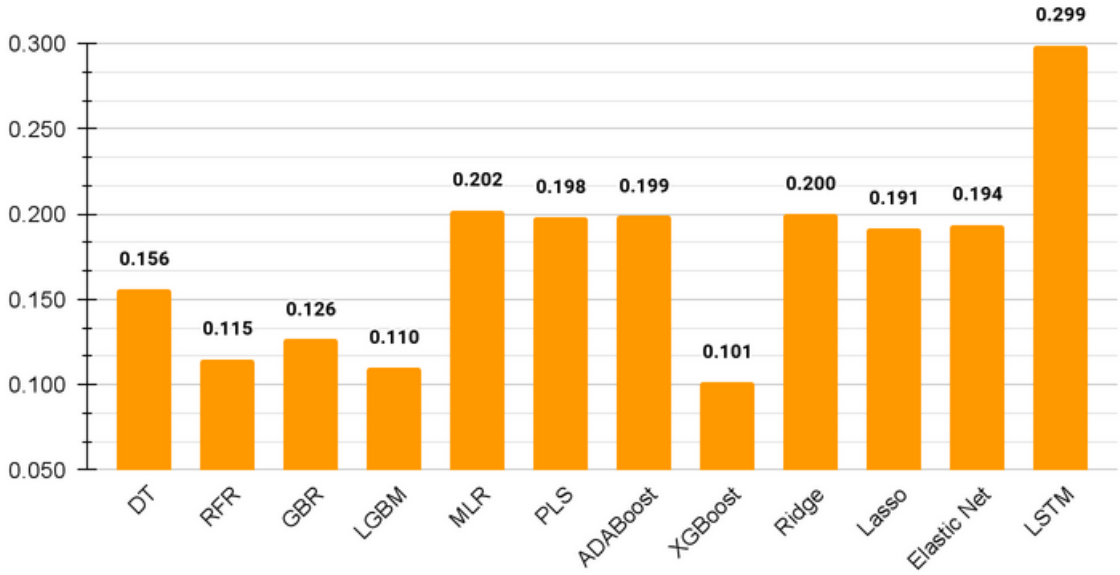

Figure 5.5: MAE score model comparison

5.3 Results

The goal is to make Bitcoin sustainable by tackling some of its flaws that have created a barrier for widespread adoption. Among them, we are trying to reduce wait times by analyzing the network traffic and the miners' situation to make the block sizes dynamic. Our prediction models can predict the next ideal block with **61.21%** accuracy. For example, during Christmas week if people use Bitcoin for their transactions, the flow in the network would be high, thus creating congestion and long wait times. With the help of our model, it would analyze the features/attributes such as the TPS, mempool growth and dynamically increase to create more room for transactions. This ensures more transactions are verified and get their first confirmation thus reducing wait times. We can think of this as a bottleneck effect where the bottle's neck expands based on the traffic for higher transaction throughput.

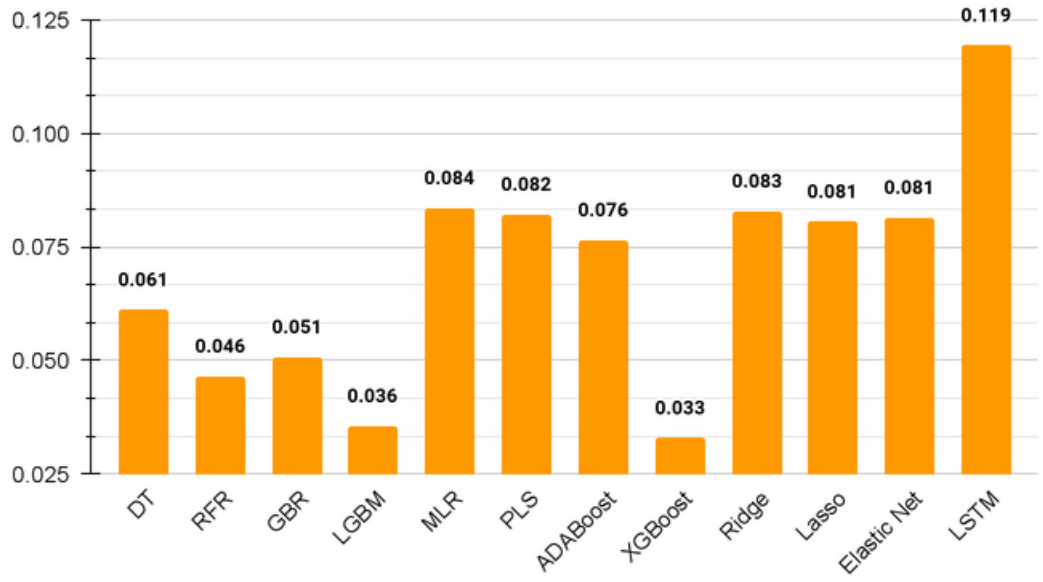

Figure 5.6: MSE score model comparison

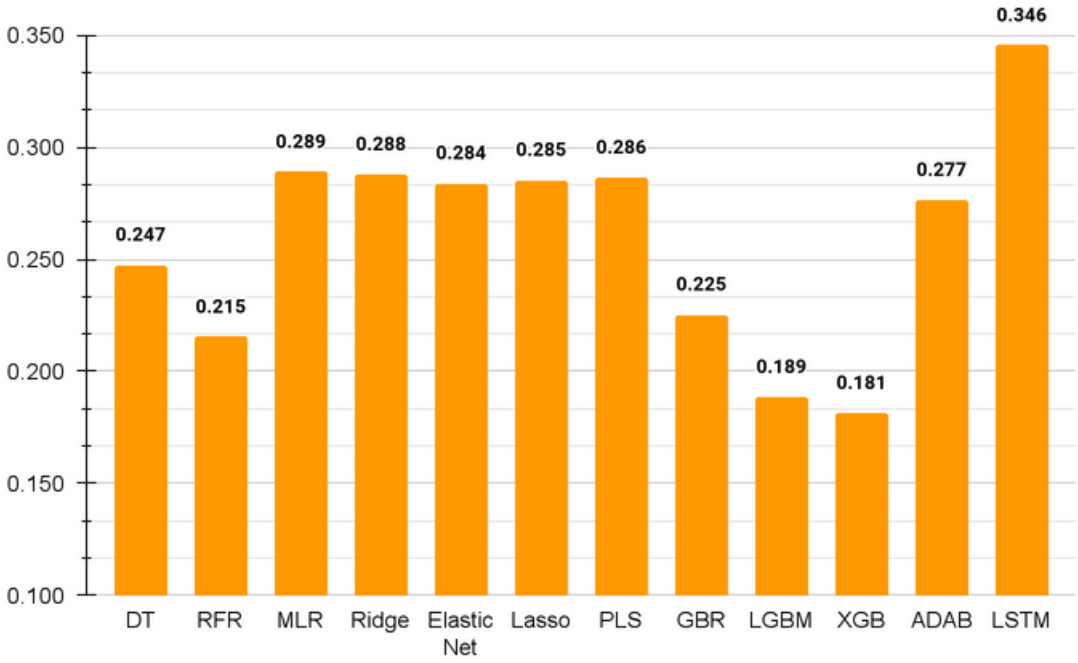

Figure 5.7: RMSE score model comparison

To simulate real-world scenarios, we collected the transaction data from all the blocks on 8th November 2021 and 13th November 2021 (10 am to 1 pm). These dates have been picked keeping in mind the fluctuation of transactions and block sizes during weekday and weekend start. We have also collected all the feature information from blocks generated on that day to better compare and understand the results. Blocks 708784 to 708786 were created on 8th November 2021 (Monday), which starts a weekday. On the other hand, blocks 709511 to 709517 are created on 13th November 2021 (Saturday), starting a weekend. In Table 5.13 and Figure 5.8,

we can see the comparison between the actual and predicted values.

Table 5.13: Comparison between actual size, predicted size and adjusted predicted size

Block Number	Actual	Predicted	Adjusted
708784	1.448054	1.159	1.2
708785	1.464258	1.143	1.1
708786	1.439841	1.027	1
709511	0.76545	1.05	1
709512	0.575538	0.898	0.9
709513	1.494906	1.033	1
709514	0.243604	1.204	1.2
709515	0.40711	0.242	0.2
709516	0.459791	0.947	0.9
709517	0.677498	1.036	1

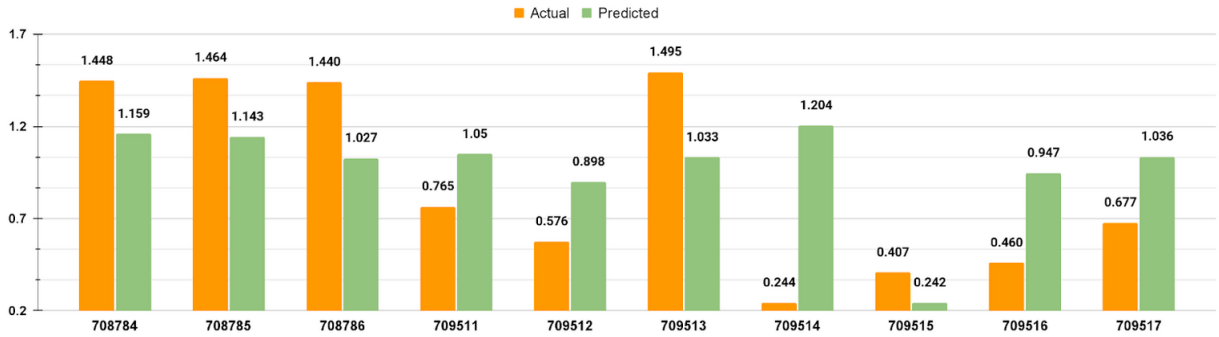

Figure 5.8: Block size prediction comparison

$$MP = \sum_{i}^{n} T_i \tag{5.20}$$

As mentioned, we have taken transactions from 10 AM to 1 PM from each relevant blocks creation day. Then we have passed the block sizes (both actual and predicted) through a greedy-based approach focusing on the fee to size ratio. As previously discussed, the mempool (MP) contains all the unconfirmed transactions (T_i) (shown in equation (5.20)). Initially, all the transactions inside the MP are sorted based on the *fee_per_KB* in descending order. Then it goes through the greedy approach where based on the systems specified block size CB_s, it checks if the size of transaction TS_i fits inside the candidate block (CB). If it does, the transaction is added to CB. Then its size is subtracted from the candidate block CB_s, and the transaction is removed from MP. This process goes on until all the transactions have been traversed. Algorithm 2 can further clarify the logic behind the transaction selection process for the candidate block.
Table 5.14 shows us that, as expected, sometimes our model generates higher fees and counts, and sometimes it doesn't. However, we do see an overall increase in the

Algorithm 2: Transaction selection

Data: Transactions in the miners mempool
Result: Candidate block
Descending sort transactions based on *fee_per_kb*
foreach T_i *in MP* **do**
  if $TS_i \leqslant CB_s$ **then**
    CB = CB + T_i
    $CB_s = CB_s$ - TS_i
    MP = MP - T_i
end
return CB

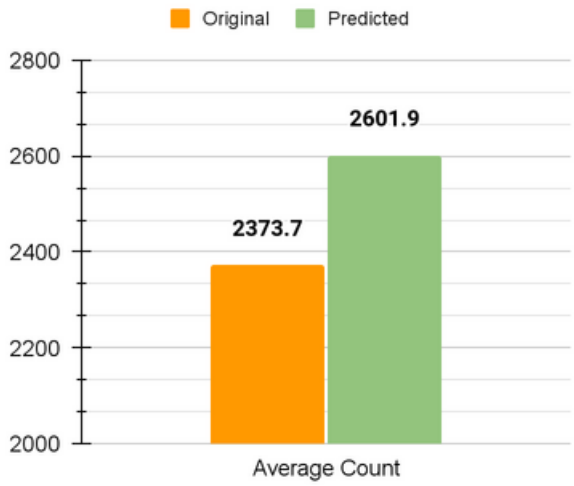

Figure 5.9: Transaction count comparison

number of fees accumulated by the miners, the increase in transactions thus having an impact on the TPS values (TPS is calculated assuming blocks are created every 10 minutes). We see a **9.3% and 66.75%, increase in fees and transaction count/ TPS**, respectively. Figure 5.9 & 5.10 can further help visualize the results. Again we need to keep in mind while going through the results that these are predicted by analyzing the network's activity and its related attributes.

5.4 Limitations & Future Work

One of the most significant limitations of our model is data. There are factors that we discovered through different Bitcoin price prediction models [9], [10], [27] such as unspent transaction count, average fee, and wallet users count. However, when incorporating these into our initial dataset, we realized that the amount of missing data for these columns was enormous. After merging the data, the total non-null rows were 2.2% of the whole data. So we had to discard such features. Furthermore, the data we used also had null values which was imputed to make it usable. The prediction would be more accurate and reliable if the model is incorporated into the Bitcoin network with more accurate data.
Finally, one of our realizations for not reaching a very high level of accuracy is

Table 5.14: Transaction count, total fees earned and TPS comparison

Block Number	Actual Size	Predicted Size	Actual Fees	Actual Count	Predicted Fees	Predicted Count	Fee Change(%)	Count Change(%)	Actual TPS	Original TPS
708784	1.448054	1.159	60,404.69	3896	56,642.75	2962	-6.23	-23.97	6.49	4.94
708785	1.464258	1.143	60,578.08	3915	56,148.18	2845	-7.31	-27.33	6.53	4.74
708786	1.439841	1.027	60,314.50	3862	54,121.53	2451	-10.27	-36.54	6.44	4.09
709511	0.76545	1.05	35,586.16	2148	38,516.24	2876	8.23	33.89	3.58	4.79
709512	0.575538	0.898	32,643.62	1476	37,082.74	2608	13.60	76.69	2.46	4.35
709513	1.494906	1.033	41,843.55	4052	38,359.07	2805	-8.33	-30.77	6.75	4.68
709514	0.243604	1.204	20,496.81	550	39,837.40	3331	94.36	505.64	0.92	5.55
709515	0.40711	0.242	28,203.08	916	20,395.95	545	-27.68	-40.50	1.53	0.91
709516	0.459791	0.947	29,983.72	1080	37,554.82	2782	25.25	157.59	1.80	4.64
709517	0.677498	1.036	34,347.16	1842	38,386.28	2814	11.76	52.77	3.07	4.69
Average	**0.897605**	**0.9739**	**40,440.14**	**2373.7**	**41,704.50**	**2601.9**	**9.34 %**	**66.75 %**	**3.96**	**4.34**

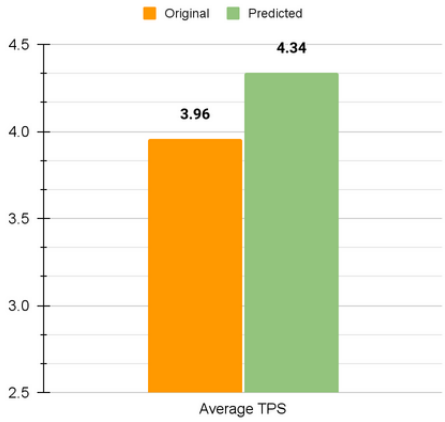

Figure 5.10: Transaction per second comparison

that the block sizes do not follow any pattern. In the current implementation, the mempool can be different for miners, their transaction picking can be very different, and most importantly, miners do not care about the network's activity. They do not care how many transactions are piling up. They only take those transactions that generate the most benefit and start the mining process. Due to this, the patterns are almost non-existent. Our goal is to understand the activity and flow of transactions, training the model using data that has no pattern and does not follow network activity, is complex; thus, the results are not in the high thresholds.
In the future, we would also like to make the difficulty dynamic based on the network traffic and the hash rate. So that when the traffic is high, miners can quickly generate blocks while not compromising security.

Chapter 6

Conclusion

For a few years, sustainability in Bitcoin has been a widely researched topic. Everyone realizes the potential of this cryptocurrency and wants to make it a widespread form of exchange. With our proposed SBM, we can ensure sustainability in Bitcoin by addressing Bitcoin's core problems. SBM ensures a good balance of transactions, thus removing the biases towards more significant transactions ensuring Bitcoin's wider usage. With its combined transaction selection and new architecture, our proposed model enables faster transactions verification while ensuring the miner is not facing losses or creating a larger carbon footprint. Due to the SB count being dynamic, it would always be beneficial no matter how many SB's are generated in the cycle. This ensures shorter wait times which is one of the biggest hurdles of Bitcoin, thus guaranteeing wider acceptability. As previously discussed, research works only focus on security or ensuring shorter wait times. However, our model can address both wait times and the imbalance of transactions, thus making our research novel.

On the other hand, in our second concept, we formed a path towards this sustainability with ML models by creating the ideal block size by examining network activity. We believe both these ideas combined can help Bitcoin reach its full potential. Even though Bitcoin is the largest cryptocurrency, it would gradually lose its value if the user count does not increase. Long wait time is the most significant disadvantage Bitcoin faces. In our research, we have shown how, tweaks with the architecture and with the help of ML, we can reduce the time, thus taking a small step towards making this notable cryptocurrency reach the valuation it deserves.

Bibliography

[1] S. Nakamoto, "Bitcoin: A peer-to-peer electronic cash system," 2009.

[2] T. Chen and C. Guestrin, "Xgboost: A scalable tree boosting system," Aug. 2016, pp. 785–794. DOI: 10.1145/2939672.2939785.

[3] J. Göbel and A. Krzesinski, "Increased block size and bitcoin blockchain dynamics," in *2017 27th International Telecommunication Networks and Applications Conference (ITNAC)*, 2017, pp. 1–6. DOI: 10.1109/ATNAC.2017.8215367.

[4] *Industry 4.0 explained*, https://home.kpmg/xx/en/home/insights/2017/11/industry-4-0-explained.html, Accessed: 03- Nov- 2021, 2017.

[5] X. Liang, S. Shetty, D. Tosh, C. Kamhoua, K. Kwiat, and L. Njilla, "Provchain: A blockchain-based data provenance architecture in cloud environment with enhanced privacy and availability," in *2017 17th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGRID)*, IEEE, 2017, pp. 468–477.

[6] T. Chomsiri and K. Kongsup, "P coin: High speed cryptocurrency based on random-checkers proof of stake," in *2018 Joint 10th International Conference on Soft Computing and Intelligent Systems (SCIS) and 19th International Symposium on Advanced Intelligent Systems (ISIS)*, 2018, pp. 524–529. DOI: 10.1109/SCIS-ISIS.2018.00092.

[7] S. Kim, "A novel bitcoin mining scheme based on the multi-leader multi-follower stackelberg game model," *IEEE Access*, vol. 6, pp. 48 902–48 912, 2018. DOI: 10.1109/ACCESS.2018.2867631.

[8] Y. Liu, X. Liu, C. Tang, J. Wang, and L. Zhang, "Unlinkable coin mixing scheme for transaction privacy enhancement of bitcoin," *IEEE Access*, vol. 6, pp. 23 261–23 270, 2018. DOI: 10.1109/ACCESS.2018.2827163.

[9] S. McNally, J. Roche, and S. Caton, "Predicting the price of bitcoin using machine learning," in *2018 26th Euromicro International Conference on Parallel, Distributed and Network-based Processing (PDP)*, 2018, pp. 339–343. DOI: 10.1109/PDP2018.2018.00060.

[10] T. Phaladisailoed and T. Numnonda, "Machine learning models comparison for bitcoin price prediction," in *2018 10th International Conference on Information Technology and Electrical Engineering (ICITEE)*, 2018, pp. 506–511. DOI: 10.1109/ICITEED.2018.8534911.

[11] Z. Ren and Z. Erkin, *Vapor: A value-centric blockchain that is scale-out, decentralized, and flexible by design*, 2018. arXiv: 1810.12596 [cs.DC].

[12] S. Velankar, S. Valecha, and S. Maji, "Bitcoin price prediction using machine learning," in *2018 20th International Conference on Advanced Communication Technology (ICACT)*, 2018, pp. 144–147. DOI: 10.23919/ICACT.2018.8323676.

[13] M. Zima, "(short paper) inputs reduction for more space in bitcoin blocks," in *2018 Crypto Valley Conference on Blockchain Technology (CVCBT)*, 2018, pp. 112–115. DOI: 10.1109/CVCBT.2018.00020.

[14] A. Aggarwal, I. Gupta, N. Garg, and A. Goel, "Deep learning approach to determine the impact of socio economic factors on bitcoin price prediction," in *2019 Twelfth International Conference on Contemporary Computing (IC3)*, 2019, pp. 1–5. DOI: 10.1109/IC3.2019.8844928.

[15] S. Glen, *Lasso regression: Simple definition*, https://www.statisticshowto.com/lasso-regression/, Accessed: 08-Dec- 2021, 2019.

[16] S. Jiang and J. Wu, "Bitcoin mining with transaction fees: A game on the block size," in *2019 IEEE International Conference on Blockchain (Blockchain)*, 2019, pp. 107–115. DOI: 10.1109/Blockchain.2019.00023.

[17] K. Muli, *The blockchain scalability problem the race for visa-like transaction speed*, https://towardsdatascience.com/the-blockchain-scalability-problem-the-race-for-visa-like-transaction\-speed-5cce48f9d44, Accessed: 17- Jul- 2021, 2019.

[18] D. O. M. Sanchez, "Sustainable development challenges and risks of industry 4.0: A literature review," in *2019 Global IoT Summit (GIoTS)*, 2019, pp. 1–6. DOI: 10.1109/GIOTS.2019.8766414.

[19] W. Xu, *What's the difference between linear regression, lasso, ridge, and elasticnet in sklearn?* https://towardsdatascience.com/whats-the-difference-between-linear-regression-lasso-ridge-and-elasticnet-8f997c60cf29, Accessed: 08-Dec- 2021, 2019.

[20] J. Yu, D. Kozhaya, J. Decouchant, and P. Esteves-Verissimo, "Repucoin: Your reputation is your power," *IEEE Transactions on Computers*, vol. 68, no. 8, pp. 1225–1237, 2019. DOI: 10.1109/TC.2019.2900648.

[21] L. B. Furstenau, M. K. Sott, L. M. Kipper, *et al.*, "Link between sustainability and industry 4.0: Trends, challenges and new perspectives," *IEEE Access*, vol. 8, pp. 140 079–140 096, 2020. DOI: 10.1109/ACCESS.2020.3012812.

[22] Y. Liu, X. Ma, L. Shu, G. P. Hancke, and A. M. Abu-Mahfouz, "From industry 4.0 to agriculture 4.0: Current status, enabling technologies, and research challenges," *IEEE Transactions on Industrial Informatics*, vol. 17, no. 6, pp. 4322–4334, 2020.

[23] Y. Liu, Y. Hei, T. Xu, and J. Liu, "An evaluation of uncle block mechanism effect on ethereum selfish and stubborn mining combined with an eclipse attack," *IEEE Access*, vol. 8, pp. 17 489–17 499, 2020. DOI: 10.1109/ACCESS.2020.2967861.

[24] S. Paavolainen and C. Carr, "Security properties of light clients on the ethereum blockchain," *IEEE Access*, vol. 8, pp. 124 339–124 358, 2020. DOI: 10.1109/ACCESS.2020.3006113.

[25] N. Papadis and L. Tassiulas, "Blockchain-based payment channel networks: Challenges and recent advances," *IEEE Access*, vol. 8, pp. 227 596–227 609, 2020. DOI: 10.1109/ACCESS.2020.3046020.

[26] G. Ramezan and C. Leung, "Analysis of proof-of-work-based blockchains under an adaptive double-spend attack," *IEEE Transactions on Industrial Informatics*, vol. 16, no. 11, pp. 7035–7045, 2020. DOI: 10.1109/TII.2020.2977689.

[27] M. Saad, J. Choi, D. Nyang, J. Kim, and A. Mohaisen, "Toward characterizing blockchain-based cryptocurrencies for highly accurate predictions," *IEEE Systems Journal*, vol. 14, no. 1, pp. 321–332, 2020. DOI: 10.1109/JSYST.2019.2927707.

[28] "Applications of blockchain technology in marketing systematic review of marketing technology companies," *Blockchain: Research and Applications*, p. 100 023, 2021, ISSN: 2096-7209. DOI: https://doi.org/10.1016/j.bcra.2021.100023. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S209672092100018X.

[29] CoinMarketCap, *All cryptocurrencies*, https://coinmarketcap.com/all/views/all/, Accessed: 17- Jul- 2021, 2021.

[30] ——, *Global cryptocurrency market charts*, https://coinmarketcap.com/charts/, Accessed: 09- Jun- 2021, 2021.

[31] "Consensus mechanism design based on structured directed acyclic graphs," *Blockchain: Research and Applications*, p. 100 011, 2021, ISSN: 2096-7209. DOI: https://doi.org/10.1016/j.bcra.2021.100011. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2096720921000063.

[32] *Decision tree*, https://www.geeksforgeeks.org/decision-tree/, Accessed: 21- Nov- 2021, 2021.

[33] W. Fan, J. Kim, I. Kim, X. Zhou, and S.-Y. Chang, "Performance analyses for applying machine learning on bitcoin miners," in *2021 International Conference on Electronics, Information, and Communication (ICEIC)*, 2021, pp. 1–4. DOI: 10.1109/ICEIC51217.2021.9369776.

[34] J. Frankenfield, *Ripple*, https://www.investopedia.com/terms/r/ripple-cryptocurrency.asp, Accessed: 09- Jun- 2021, 2021.

[35] B. Garrison, *How long does it take to send bitcoin?* https://www.cryptovantage.com/news/ask-cryptovantage-how-long-does-it-take-to-send-Bitcoin/, Accessed: 09- Jun- 2021, 2021.

[36] A. HAYES, *Multiple linear regression (mlr)*, https://www.investopedia.com/terms/m/mlr.asp, Accessed: 18- Oct- 2021, 2021.

[37] S. M. Hosseini Bamakan, A. B. Bondarti, P. B. Bondarti, and Q. Qu, "Blockchain technology forecasting by patent analytics and text mining," *Blockchain: Research and Applications*, p. 100 019, 2021, ISSN: 2096-7209. DOI: https://doi.org/10.1016/j.bcra.2021.100019. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2096720921000142.

[38] M. A. Imtiaz, D. Starobinski, and A. Trachtenberg, "Investigating orphan transactions in the bitcoin network," *IEEE Transactions on Network and Service Management*, vol. 18, no. 2, pp. 1718–1731, 2021. DOI: 10.1109/TNSM.2021.3056949.

[39] X. F. Liu, X.-J. Jiang, S.-H. Liu, and C. K. Tse, "Knowledge discovery in cryptocurrency transactions: A survey," *IEEE Access*, vol. 9, pp. 37 229–37 254, 2021. DOI: 10.1109/ACCESS.2021.3062652.

[40] P. Portal, *Comparing the carbon footprint of gold and bitcoiny*, https://www.visualcapitalist.com/comparing-the-carbon-footprint-of-gold-and-bitcoin/, Accessed: 03- Nov- 2021, 2021.

[41] S. Ramos, F. Pianese, T. Leach, and E. Oliveras, "A great disturbance in the crypto: Understanding cryptocurrency returns under attacks," *Blockchain: Research and Applications*, p. 100 021, 2021, ISSN: 2096-7209. DOI: https://doi.org/10.1016/j.bcra.2021.100021. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2096720921000166.

[42] *Random forest algorithm*, https://www.javatpoint.com/machine-learning-random-forest-algorithm, Accessed: 21- Nov- 2021, 2021.

[43] V. Tahiri, *Bitcoin on-chain analysis: Cdd indicator shows bullish trend conviction*, https://finance.yahoo.com/news/bitcoin-chain-analysis-cdd-indicator-103802500.html, Accessed: 07-Dec- 2021, 2021.

[44] S. Tanwar, N. P. Patel, S. N. Patel, J. R. Patel, G. Sharma, and I. E. Davidson, "Deep learning-based cryptocurrency price prediction scheme with interdependent relations," *IEEE Access*, vol. 9, pp. 138 633–138 646, 2021. DOI: 10.1109/ACCESS.2021.3117848.

[45] Z. Voell, *Bitcoin miners usually create 6 blocks per hour. they just banged out 16*, https://www.coindesk.com/markets/2020/05/01/bitcoin-miners-usually-create-6-blocks-per-hour-they-just-banged-out-16/, Accessed: 05- Sep- 2021, 2021.

[46] *Blockchair database dumps*, https://gz.blockchair.com/bitcoin/blocks/, Accessed: 15- Oct- 2021.

[47] *Mempool size (bytes)*, https://www.blockchain.com/charts/mempool-size, Accessed: 15- Oct- 2021.

[48] *Mempool size growth*, https://www.blockchain.com/charts/mempool-growth, Accessed: 15- Oct- 2021.

[49] *Mempool transaction count*, https://www.blockchain.com/charts/mempool-count, Accessed: 15- Oct- 2021.

[50] *Transaction rate per second*, https://www.blockchain.com/charts/transactions-per-second, Accessed: 15- Oct- 2021.