Detecting Patterns in the Ethereum Transactional Data using Unsupervised Learning

by

Eibhlín O'Kane.

Dissertation

Presented to the

University of Dublin, Trinity College

in fulfilment

of the requirements

for the Degree of

Master of Science in Data Science

University of Dublin, Trinity College

August 2018

Declaration

I, the undersigned, declare that this work has not previously been submitted as an exercise for a degree at this, or any other University, and that unless otherwise stated, is my own work.

____

Eibhlín O’Kane

August 29, 2018

Permission to Lend and/or Copy

I, the undersigned, agree that Trinity College Library may lend or copy this thesis upon request.

____

Eibhlín O'Kane

August 29, 2018

Acknowledgments

I would like to express my sincere gratitude to my supervisor Prof. Donal O'Mahony. Without his support, expertise, help and guidance, this dissertation would not have been possible. Through his in-depth knowledge in Blockchain technology Professor Donal O'Mahony was able to challenge my research and pose insightful questions that ultimately helped to strengthen my dissertation.

I also owe a great deal of gratitude to my parents and all those who encouraged and motivated me throughout this research.

Abstract

Ethereum is a development platform upon which smart contracts can be built and deployed. Smart contracts allow credible transactions to be executed without the intervention of third parties. They are also irreversible and completely trackable. The cryptocurrency of Ethereum is known as Ether which can be transferred between users by sending Ether from one user's address to another. All history of Ether owning and transferring, as well as smart contract deployment and interactions, are available on the public ledger known as the Ethereum Blockchain. While the transactional data available contains information such as the users' addresses, their true identity is hidden.

The decentralisation of the Ethereum blockchain, coupled with the pseudo-anonymity of its users, has paved the way for an unregulated technology in which users can deploy applications and transfer cryptocurrency from one user's address to another. It is hard to near impossible for law enforcement to detect suspicious patterns on the blockchain, thus making it an attractive environment for malicious activity.

Interest in detecting suspicious behaviour has gained a lot of interest from various parties over the last number of years with different means of identification being explored such as Ponzi schemes and using the Markov Logic Network and Markov Random Fields to exploit underlying links between potentially fraudulent users in the network.

In this thesis we explore the use of unsupervised learning as a means of detecting patterns in the Ethereum transactional data. We evaluate the accuracy of the patterns formed and investigate whether or not anomalies occur. This study provides a good starting point for future work to advance in the area of detecting suspicious behaviour on the Ethereum Network.

Contents

1 List of Figures

2 List of Tables

3 Introduction

Blockchains and Cryptocurrencies have grown in popularity since their creation. They have made headlines and have become somewhat of a hot topic for investors, developers, researchers and banks, to name a few. The Blockchain is a cutting-edge technology that was conceptualised in 2008 after the financial crash. One of its main goals is to give power back to the people so that users are in control of their own transactions and information.

Central banks began to investigate the use of the distributed ledger technology and look at ways in which they could capitalise on it. Many sweeping statements have been made about this new technology. In an article published by Forbes, they have demonstrated somewhat extreme views stating that the blockchain will end world poverty (Kuznetsov, 2017), while McKinsey have announced that it is set to revolutionise the world's economy (Tapscott, Tapscott, & Kirkland, 2016). In contradiction, an article published by MIT Technology stated that "The UN Says the Global Digital Divide Could Become a Yawning Chasm" (MIT, 2017).We see that more than 52% of the people in the world still don't have access to the internet. There is also a massive disparity in the connection speeds in different countries which begs the question as to how the blockchain is going to end world poverty when people in the developing world don't have access to it?

In addition, while the peaks and troughs of the price of cryptocurrencies made headlines across various webpages and newspapers, investors and traders have made huge profits from the bull market. The term 'hodling' was coined, originally a typo for hold, which came to stand for "hold on for dear life!". Investors use this term when they believe that the price of their cryptocurrency will rise one day. Success stories revealed how millionaires were born overnight while horror stories demonstrated how volatile the market was and investments in cryptocurrencies could plummet in a matter of moments.

Even with all the hype, very few people fully understand what the blockchain is and how cryptocurrencies work. Despite many researchers and developers delving into the unknown trying to discover its full potential and propose new applications of the Blockchain, an enigma still surrounds this novel technology. We wanted to become part of the unveiling and further aid in the understanding of this new technology by focusing on a particular Blockchain that was developed by Vitalik Buterin; The Ethereum Blockchain.

Bitcoin is also a popular Blockchain technology, however, much research has been carried out in order to comprehend the activities of the Bitcoin network whereas much less exists for the Ethereum Blockchain. Bitcoin is solely a cryptocurrency whereas Ethereum can be viewed as a development platform. To paraphrase the description of the technology by Vitalik Buterin, we can liken Ethereum to a smart phone upon which applications can be built and deployed.

The fact that this technology is decentralised means that there is no central organisation that is able to take control of and regulate the blockchain. This means that no government or regulatory body is held accountable for the actions of the blockchain. This feature, coupled with the fact that the identity of its users is anonymous, may give rise to fraudulent activity. Should fraud occur it will be very difficult to prosecute users for their wrongdoings. The blockchain has also been described as a medium for money laundering. Dealing in cryptocurrency has provided individuals with a means of evading tax in their home countries. Users can anonymously carry out transactions and exchange large or small sums of cryptocurrencies. Drug trafficking has also been linked to the Blockchain. The Silk Road, an online marketplace for illegal drugs, was where users worldwide bought and sold narcotics using only Bitcoin as a means of payment. In 2013 the leader of the Silk road was arrested and over one hundred and forty thousand Bitcoins (which estimated, at the time, to be around twenty-eight million dollars) was seized.

Being able to identify transactions associated with illegal activity is an area which we would like to explore. We want to know if it is possible to detect fraudulent activity on the blockchain. We will examine the transactional data on the Ethereum Blockchain and look for irregularities. These irregularities will be defined as transactions that deviate from the norm. Of course, these transactions may or may not be fraudulent but may be worthwhile flagging for inspection.

We decided that, in order to detect anomalies, we would first have to delve into the concept and structure of the Blockchain as well as try to discover the undertakings of the Ethereum transactional data by asking high level preliminary questions. Using machine learning techniques, we will try to make sense of the vastness of the transactional data as well as looking at ways in which we can form patterns in the Ethereum transactional data. Should clear patterns be detected, irregularities will be flagged by looking at transactions that do not appear to fit any one pattern.

3.1 Research Question

We wanted to create a piece of work that aids the understanding of the interactions on the Ethereum platform. In order to do this, we decided to investigate the Ethereum transactional data and as a result, the following research question has been proposed:

"Can we detect patterns in the Ethereum transactional data, using an unsupervised learning algorithm such as clustering?"

This question can be further subdivided into the following questions:

1 What unsupervised learning algorithm should we use?

2 How do we select features from the transactional data as inputs to our chosen algorithm?

3 Is there a series of experiments that we can carry out in order to show the effects of how varying the parameters of the algorithm can affect how distinguishable the patterns formed are compared to each other? That is, are there clear-cut patterns emerging?

4 How do we evaluate the results of these experiments and determine an optimal solution for the parameter selection?

5 After we have implemented the algorithm and clear-cut patterns have been detected, is it possible to spot irregular transactions that deviate from the normal pattern? That is, are there transactions that do not appear to fit any of the patterns formed?

3.2 Dissertation Structure

Section 3 provides an introduction to the dissertation. It includes a brief discussion on the background and motivation for the project as well as addressing the research question.

The aim of section 4, 5 and 6 is to provide the reader with a strong technical background which will be used to aid them in their understanding and answering of our research question.

Section 4 discusses the Blockchain technology. We provide the reader with some background knowledge as to what the blockchain is as well as introducing both Bitcoin and Ethereum; two renowned blockchain technologies.

Section 5 focuses on data management techniques that can be used when handling large volumes of data. We discuss how various file formats can help to control the overall size of the data as well as cloud computing and the approaches taken to handle 'Big Data'.

Section 6 discusses Machine Learning with a strong focus on clustering algorithms. We also discuss feature selection and how we select features from our data to use as inputs to these algorithms.

Section 7 presents the reader with the design of the dissertation. We discuss the techniques we adopted in order to manage the data as well as the algorithm that we selected. We also set up a series of experiments that are used to compare and contrast the performance of the algorithm.

Section 8 shows how we implemented our design and presents the results of the various experiments we carried out in this dissertation. We also discuss the limitations of the algorithm selected for the analysis.

Section 9 concludes the dissertation. Here, we reflect on the results of the experiments as well as discussing the challenges faced when carrying out the analysis. We also comment on the potential future work of this analysis.

4 What is the Blockchain?

4.1 A Distributed Ledger

The Blockchain is also known as a distributed ledger technology (World Bank, 2018). A ledger consists of a list of entries and often takes the form of a registry where a third party records the transactions between participants. Ledgers need not only consist of financial transactions. They can also contain public records such as births, deaths and marriages as well as semi-private records such as university degrees. Furthermore, ledgers can also hold private records such as medical accounts.

The transactions in a ledger are usually in chronological order with the majority of modern business ledgers being digitally stored on a central server. In this case a single party is responsible for validating entries, ensuring transactions are secure and certifying that historic data is preserved.

Alternatively, ledgers can be distributed. The distributed ledger differentiates itself by ensuring that its transactions are shared by a distributed network of computers. Furthermore, these transaction records are irreversible and cryptographically signed. Using cryptography allows users to communicate via encoded messages. It is important to note that every computer on the network has the same copy of the ledger with each computer receiving real live updates. This means that the distributed ledger offers total transparency to all members. Should a participant propose a change or a transaction be added to the ledger, all other participants must agree on this proposition before it can be added. That is, each transaction to be added to the ledger must obtain the majority mining consensus in order to be validated.

The idea of trust plays a very important role in the overall mechanism of a distributed ledger. All transactions must be authenticated and validated before they can be added. In most cases this involves the participant using a secure digital signature that can be generated by using Public-Key Cryptography. Before a transaction can be added to the ledger the network must agree on its validity. There must also be a consensus mechanism in place to ensure that all the computers on the network are holding the correct version of the ledger.

4.2 Blockchain Concept

Blockchain is a data structure that makes it possible to create a digital ledger of transactions and share it among a distributed network of computers (Stallings, 2017). It emulates a "trusted" computing service through a distributed protocol, run by nodes connected over the Internet (Cachin, 2016).

"It is a decentralized transparent ledger with transaction records—the database is shared by all network nodes, updated by miners, monitored by everyone, and owned and controlled by no one. It is like a giant interactive spreadsheet that everyone has access to and updates and confirms that the digital transactions transferring funds are unique." (Swan, 2015)

As in a ledger, the blockchain need not only focus on storing information based on monetary transactions. It can be used as a registry, inventory or exchange across a variety of industries such as finance or economic. The assets that are recorded, tracked and monitored can be either hard assets, for example physical property, or intangible assets such as votes or medical data. "The blockchain allows the disintermediation and decentralization of all transactions of any type between all parties on a global basis." (Swan, 2015)

This distributed ledger contains all the transactions that have ever been executed. Once a transaction is added it cannot be altered. This gives rise to a tamper proof ledger (Cachin, 2016) as it is computationally infeasible to change or remove a transaction. Adding a transaction to the ledger involves all of the nodes/computers on the network agreeing on its validity. In order to achieve this unanimous decision each node runs an algorithm and the only way a transaction can be approved is if the majority agree that the transaction is legitimate. In a blockchain, validation is carried out by miners. Miners maintain the blockchain by confirming, checking and recording new transactions (Swan, 2015). Miners must consent to a new entry before it can be added to the ledger. This process differs from the original ledger scheme where transactions are submitted to a central party who is solely responsible for updating, validating and distributing the ledger.

When people talk about the blockchain it is possible that they are making reference to one of two things. They could be talking about the blockchain network which is the network of computer or nodes. On the other hand, they could be using the term to describe the chain of blocks that make up the distributed ledger. One of the features of the blockchain is that it establishes a trustless environment. The blockchain is not centralised. That is, it is not stored in a central server. Instead, it is a shared data structure with each user having access to a copy of the blockchain where they are able to view the same copy of the ledger allowing them to read historic transactions and if they wish, write future ones. This means that if all users are holding the exact same copy of the blockchain and they query it, they will all receive the same answer.

Being a node on the network is not the only way to access the blockchain data. It is possible to view the transactions on the blockchain via lightweight client. This means that one doesn't download the entire blockchain. Instead they access in read-only mode by connecting to a node on the network. A full or core client downloads the entire blockchain.

4.3 Blockchain Structure

The exact structure of the blockchain will vary from platform to platform. However, there are a few fundamental features that are common across the design.

In a blockchain the transactions are governed by protocols and consensus mechanisms. As a transaction occurs it is added to a block where each block can contain one or multiple transactions. The number of transactions in a block is noted by the transaction counter. The blocks are added to the blockchain in a linear chronological order where each block has a unique successor and predecessor. Once a new block has been created it is broadcast to every node on the network which allows each user to update their local copy of the blockchain. If any user is not up to date with the most recent block they can ask the other participating nodes on the network for the updated information.

Each block has its own unique ID which can be viewed as the digital fingerprint of that block. In many cases this ID is a numeric value with the new blocks having a larger numeric representation than the older ones. The first block is known as the genesis block and contains the very first transactions ever executed on that chain.

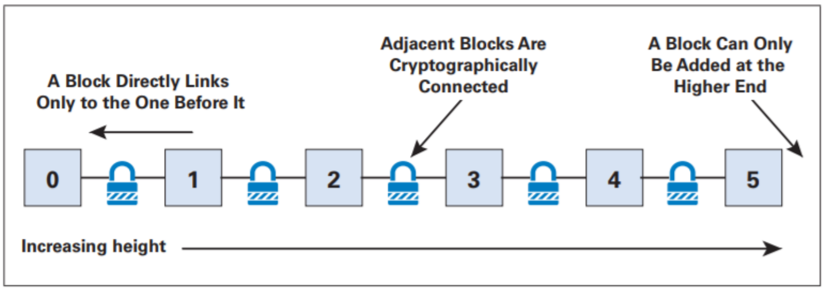
Adjacent Blocks Are Cryptographically Connected
A Block Can Only Be Added at the Higher End
A Block Directly Links Only to the One Before It
0
1
2
3
4
5
Increasing height

Figure 1: Building the Blocks in a Blockchain

Source: (Stallings, 2017)

Blocks are timestamped with each block representing a series of events or transactions that have occurred in a given time frame. When these multiple blocks link together we form what is known as a chain. Once a block is added to the chain it is in read-only mode which means that the information in that block cannot be over-written or edited.

The hash function is the cement that binds the blocks together and is created from the data that was in the previous block. Therefore, it can be seen as a digital fingerprint of the data in the previous block. A hash function is a cryptographic algorithm that takes any input and produces an output of specific size (Cormen, Leiserson, Rivest, & Stein, 2009). The reason hash functions are used in the blockchain is because of their one-way functionality. This means that given the output, it is highly mathematically and computationally improbable that the original input can be derived. This resistance feature is what adds to the blockchain's security.

Hash functions are not only used to bind blocks together but are also used when it comes to transactions. The transaction hash is computed from the set of data blocks that comprise the list of transactions (Stallings, 2017). Merkle trees enable the user to decide whether or not a transaction belongs in a block. Merkle trees are formed when pairs of nodes are repeatedly hashed until one hash is left. The hash that is left is known as the Merkle Root or Root Hash. If there is an uneven number of transactions, the last transaction is duplicated. One of the reasons why Merkle trees are used is because their proofs are computationally easy and fast with tiny pieces of data being carried along the network. This means that they are able to carry the transactions and require little memory or disk space.

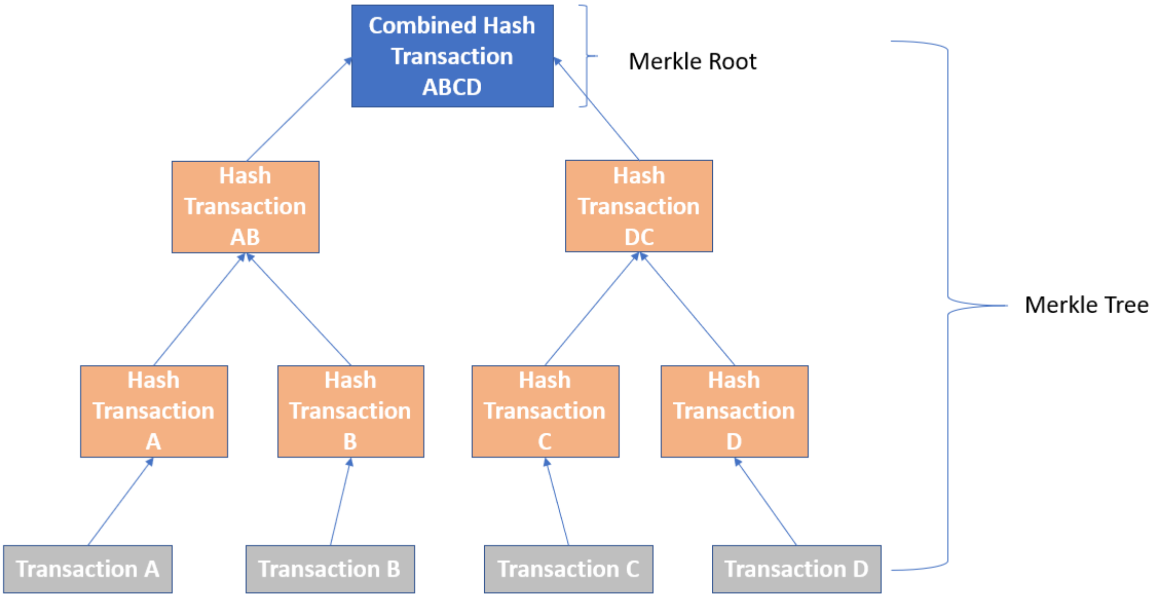

Figure 2: Hashing Transaactions

Another feature on the blockchain structure is a nonce. A nonce is one-time, random value that is generated to satisfy the proof of work. Finally, the 'mining difficultly' is used to indicate the degree of complexity in finding a new block. Both mining difficulty and nonce discussed in the next section.

4.4 Blockchain Mining

The act of linking blocks together is known as mining and those who carry out mining are known as miners. Not every blockchain participant is a miner. A miner can be a single user or can be a group of users that join together to pool their computing power. A miner can join the blockchain environment without asking for another miner's or user's permission. It is possible for the distributed system to function flawlessly with the coming and going of miners.

The role of the miner involves reviewing, registering and authorising transactions. The consensus protocol of the blockchain ensures that even though numerous conflicting transactions may take place, only one is permitted. This fundamental attribute ensures that double spending does not occur.

Each block is constructed by one miner where miners compete against each other to generate new blocks. Creating these new blocks requires a huge amount of computer processing power as the miners try to solve the hash function, by calculating a nonce, that meets the requirements set out by the mining difficulty. The mining difficulty is used to help regulate how many blocks are being created over a given time period. If finding a nonce is proving to be relatively easy the mining difficulty is increased. On the other hand, if it is too complicated the mining difficulty is reduced. By setting a difficulty it ensures that the process is more resource intensive. Once the nonce is calculated the miner can then add the block to the chain. (Swan, 2015)

Creating a block follows a certain process. Firstly, a list of transactions are broadcast by the user to the blockchain network. Each miner collects a varying number of these transactions and forms a cluster that they hope to turn into a new block. Inside this potential block the miner orders and authorises each transaction. The miner then begins to "solve the block" by trying different nonces until a nonce is found that satisfies the proof of work condition (Nimaosu, 2013). Of course, in general the miner is not successful on their first try at finding a nonce. Often this process is repeated multiple times with different nonces and different combinations of transactions. Once the block has been solved it is then broadcast to all the miners and added to the blockchain. Generally, there are no designated miners and the task of adding a block to the chain is carried out on a block to block basis.

Miners are selected on a Proof-of-Work mechanism with the miner producing the most Proof-of-Work being selected. Proof-of-Work is where every block in the blockchain is required to have evidence that a costly, non-reversible sacrifice of time and energy has been dedicated to that particular block and no others (Stallings, 2017). Producing a Proof-Of-Work is costly, however, its verification is straightforward. Only one miner can link a single block to the chain and using a proof of work mechanism ensures that this miner is randomly chosen. Miners receive an award that varies from application to application for adding a block which is usually of monetary value in the form of cryptocurrency (section 4.7).

4.5 Blockchain Forks

It is important to note that it is possible to have branching structures as the chain grows. They form when multiple miners create blocks around the same time. For example, Miner A can create a different block compared to Miner B by using a different set of transactions. The users and miners in the network are unable to decide which block to accept so they provisionally accept both and miners continue to work on the different branches. The branches compete against one another to be accepted as the true chain. Eventually a branch wins by exceeding in length compared to any other branch. All other branches become redundant and stop growing. These branches are known as forks and there are two types; hard forks and soft forks. Soft forks are temporary and rectify themselves when one of the branches surpasses the length of other competing ones. As a result, they are less prone to creating a chain split.

In comparison, hard forks are a much more permanent divergence in the blockchain and involve an opt-in consent from users in contrast to soft forks where consent is required by the miners. Hard forks do, however, grant developers larger flexibility when it comes to upgrading protocols. Developers do not have to consider whether the rules governing the protocol of the old chain satisfies the new one.

According to Vitalik Buterin (Buterin, Hard Forks, Soft Forks, Defaults and Coercion, 2017), hard forks can be further subdivided into two types: strictly expanding hard forks, which strictly expand the set of transactions that is valid, and so effectively the old rules are a soft fork with respect to the new rules, and bilateral hard forks, where the two rulesets are incompatible both ways.

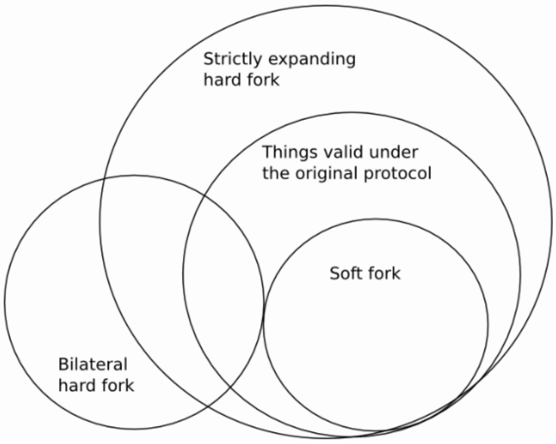

Figure 3:The Relationship between Hard and Soft Forks

Source: (Buterin, Hard Forks, Soft Forks, Defaults and Coercion, 2017)

Examples of hard forks can be found in the Bitcoin blockchain (section 4.8.2) and the Ethereum blockchain (section 4.9.3). Usually, when the hard fork is established it is given a new name to represent its distinct identity.

4.6 Blockchain Evolution

Since its conceptualisation, the blockchain technology has gone through a period of evolution. Similar to Melanie Swan and authors of the Blockchain Journal, we have decided to describe this evolution under three different categories; Blockchain 1.0, 2.0 and 3.0. Blockchain 1.0 allows for financial transactions to take place. It is essentially everything related to currency such as deploying cryptocurrencies and using them to carry out digital payments. The most prominent example is Bitcoin. Blockchain 2.0 differs in that it focuses on the decentralisation of markets in contrast to Blockchain 1.0 that focuses solely on the decentralisation of money and payments. Instead of transferring money Blockchain 2.0 looks at contracts (discussed later in section 4.9.1) and how one might transfer different types of assets both tangible and intangible such as stocks, bonds, mortgages and smart contracts. Blockchain 2.0 monitors these contracts from the moment they are created to each point where they are transferred and possibly divided. Blockchain 3.0 looks beyond currency and contracts instead focusing on blockchain application in areas such as government and health.

In the following section we will discuss the concept behind Blockchain 1.0 where we will focus on Bitcoin. In section 4.9 we will discuss Ethereum and look at how the concepts surrounding Ethereum relate to the ideas of Blockchain 2.0 and 3.0.

4.7 Cryptocurrency

A cryptocurrency is a digital asset that can be used as a medium of exchange. Cryptography and cryptographic techniques are used to create and secure the control of cryptocurrency coins (Judmayer, Stifter, Krombholz, & Weippl, 2017). The blockchain technology provides cryptocurrencies with the required level of security needed to withstand attacks on the state of the system and from preventing double spending. That is if user A sends user B two coins, user A is unable to send user C the exact same two coins. Unlike fiat currencies, the security rules of cryptocurrencies need to be enforced in a purely technological manner and without relying on a central authority (Narayanan, Bonneau, Felten, Miller, & Goldfeder, 2016).

Looking at a simple transaction from the sender’s perspective, an address and private key are used as well as a type of wallet software. The address is the location to where the cryptocurrency will be sent – the sender also has his/her own address which other users use to send cryptocurrency to them. The private key is the seal on the transmission, it is a cryptographic secret that is unique to the sender. Using a private key adds a level of security to the transaction operation. The private key, of a certain address, is used to unlock all the information surrounding the transactions of that address. If you lose your private key, then you cannot access your cryptocurrency. Wallets can be used to store a copy of the blockchain detailing all the transactions that have taken place or they can be used as a means of composing the transaction. It also stores the user's public and private keys.

When a user sends cryptocurrency, there is no physical exchange of coins. Instead what happens is a signing off of ownership in the sender's address to the receiver's address. In order to be able to access the coins being sent the receiver must hold the private key which unlocks the sender's public key. If the keys match, the transaction is recorded on the blockchain and simultaneously the balance is altered in the sender's and receiver's address (Ahamad, Nair, & Varghese, 2013).

Wallets usually fall under the following three categories – software, hardware and paper. Software wallets can be found on a desktop, as an app on a mobile or online in the cloud. Desktop wallets are stored locally on the PC in which the software was installed in contrast to online wallets which can be accessed from any PC in any location. This type of storage is known as hot storage as accessing these wallets can be done online. These types of wallets, unlike hard wallets, are at a higher risk of hackers being able to access their private keys as they are held on devices that are accessible over the internet.

Hardware wallets are physical devices such as USB sticks or external hard drives. Unlike software wallets, hardware wallets' superiority in security comes from the fact that they are stored offline. This type of storage is known as cold storage. Although a very secure way of storing keys it is susceptible to human error such as the owner losing the device or having it stolen. It is also vulnerable to damage which may leave the wallet unusable. Also, hard wallets are purchased from third parties and sometimes come with their own private key. There is a certain level of trust between the consumer and the company that these private keys have been logged securely and will not be leaked.

Paper wallets are similar to hard wallets in that they are stored offline. However, as public and private keys are stored on hard wallets they are printed on a physical piece of paper in paper wallets in the form of QR codes. When the user wishes to make a transaction they must scan this code. The biggest risk in paper wallets is losing or damaging the piece of paper containing the QR codes (Manning, 2018).

4.7.1 Coin washers/mixers

Not all transactions are as straight forward as sending a certain amount of cryptocurrency from one address to another. As discussed, the public blockchain broadcasts all its transactions to every node on the network. However, sometimes users may wish to conceal their identity and enhance the level of privacy surrounding their transactions. In order to achieve this they use coin mixers, also called coin washers and coin tumblers. Coin mixers provide the service of obscuring the links between the sender's and receiver's addresses. They are also capable of randomising the transaction amounts and adding time delays to the transactions (Coin Mixer, 2018). This means that crypto being sent will not arrive at the receiver's address for some time and the intermediary phase of the coin mixer obscures all ties. It is also possible in some coin mixers to even change the type of crypto being sent. For example, the sender could send crypto type A to the coin washer and specify that crypto type B is sent to the receiver. Coin mixers allow users to mask their transaction movements and prevent their transactions from being traced.

There are many reasons as to why a user would wish to use a coin washer. As the blockchain is a relatively new technology, some users are worried about hackers and the possibility that these hackers could possibly identify them in some way and potentially steal their cryptocurrency. Large financial institutions may also make use of coin washers to conceal their business dealings from their competitors. One of the biggest risks in the cryptocurrency world is that criminals will use and have used coin washers for illegal activity such as money laundering (World Crypto Index, 2018).

As an aura of mystery still surrounds coin mixers, it has been recommended that in order to add an extra layer of security, users should use a VPN or cycle their IP address when interacting with coin mixers to further mask their identity. In the future, it may become a legal requirement for coin mixers to disclose the originator's IP address to regulatory bodies or even the police.

4.7.2 Exchanges

Cryptocurrency, like any fiat currency, can be bought and sold through exchanges. Consumers can access the exchanges and convert their fiat currency to cryptocurrency or their cryptocurrency to fiat and even convert from one cryptocurrency to another.

As exchanges tend to deal with, in aggregate, very large volumes of cryptocurrencies, security is at the fore of their operation. Many exchanges use multiple cold wallets to store their wallets offline. Using hot wallets has led to serious problems for exchanges in the past. Mt Gox, one for the first Bitcoin exchanges in the world with an almost 80% trading volume became insolvent in 2014. Hackers siphoned off an alleged 850,000 bitcoins from the hot wallets. In the process the discovered a 'leak' in the hot wallet that allowed them to gain access to the cold wallets and drain all of the currency being held there (Norry, 2018).

Despite some organisations not recognising cryptocurrency as legal tender (Application of FinCEN's Regulations to Persons Administering,, 2013) many governments are calling for strict regulations to be in place surrounding cryptocurrency exchanges. There is a growing concern among governments that cryptocurrencies can be seen as a window into the realm of illegal activity. While the blockchain technology is decentralised and provides the user with a level of anonymity, once a user moves their money to a crypto exchange the mask is lifted and identity is revealed. Know Your Customer (KYC) is a verification procedure that many exchanges have been required to implement. Usually, the higher the withdrawal or deposit of cryptocurrency in the exchange, the more information is required under the KYC policy. Of course, there are still some exchanges operating a no KYC policy, however, withdrawals and deposits from such exchanges result in higher transaction fees (Lemieux, 2013).

However, some countries are still not content with these regulations. It is possible that this discontent may change over time and the feelings towards the regulation of cryptocurrencies may become more liberal or conservative. However, in February 2018, China announced that they are moving towards eliminating all cryptocurrency trading with an aim to ban all foreign exchanges (Perper, 2018).

4.8 Bitcoin

Bitcoin at its most fundamental level is a breakthrough in computer science – one that builds on 20 years of research into cryptographic currency, and 40 years of research in cryptography, by thousands of researchers around the world (Andreesen, 2014).

Bitcoin, a digital cash, is the first application of the blockchain technology. It was proposed by Satoshi Nakamoto in 2008 (Nakamoto, 2008). Bitcoin solves two challenges of digital cash; controlling its creation and avoiding its duplication all at the same time (Velde, 2013). That is, it is the first solution to the double-spending problem using a peer-to-peer network. This digital cash is not regulated, operating independently of any banks. Bitcoin uses the encryption techniques of the blockchain to create Bitcoins and verify and authorise their transactions.

4.8.1 Bitcoin Mining

One way in which a user can obtain Bitcoin is through mining on the Bitcoin network. Mining can be carried out individually or as part of a group in which users pool together their computing power. Having a large source of computing power allows a miner to process the hash function more quickly. Bitcoin mining works as follows. Firstly, transactions need to be verified to ensure that they are valid. These transactions are then bundled together to form a potential block. Next, the header of the most recent block that has been added to the chain is selected and inserted into the new potential block as a hash. The header on the block is an 80-byte header belonging to a single block (Bitcoin Glossary, 2009-2018). Miners then uses their computational power to try to find a solution to the hash function using the Proof-of-Work mechanism. Once a miner finds a solution, the new block is added to the existing chain and broadcast to all the nodes in the Bitcoin network. Miners receive a reward for finding a new block in the form of Bitcoin.

Adding blocks to the network is regulated so that a new block is created and appended at an average of every ten minutes. This regulation is controlled by adjusting the mining difficulty.

Every 2,016 blocks the network assess the performance of the previous 2,016 blocks and recalculates the mining difficulty so that the previous 2,016 blocks would have been generated in exactly two weeks. This leads to the desired result of producing one block every ten minutes xxx.

If the bitcoin mining is slow, that is if miners are taking a long time to produce a block, then the difficulty is lowered. However, if miners are found to produce blocks too quickly the mining difficulty is increased. The production of blocks may increase due to the fact that new miners have joined the network, thus increasing the hashing power (Taylor, 2013).

Mining is not the only way to acquire Bitcoin. Bitcoin can be sent from one account to another in the network. It is also possible for a person to purchase Bitcoin via an exchange or receive Bitcoin as a payment for goods and services. For example, in 2010 Bitcoin was used to purchase two pizzas. Laszlo, as they are known on bitcointalk.org, used 10,000 Bitcoins which was roughly equivalent to $25 US dollars (Bitcoin Talk, 2010). According to today's market 10,000 Bitcoins is now worth approximately $66million US dollars.

Bitcoin, like any cryptocurrency can be stored in a wallet which can be held in places such as a computer, over the web or on an external drive.

4.8.2 Bitcoin Hard Forks

Hard forks in the Bitcoin blockchain are defined as a permanent divergence in the block chain, that commonly occurs when non-upgraded nodes can't validate blocks created by upgraded nodes that follow newer consensus rules (Bitcoin Glossary, 2009-2018). This means that users who do not update their software continue to work on the 'old' chain whereas those who upgrade, begin to mine on the newly established hard fork.

Bitcoin has seen a few hard forks since its release, however, not all have been a success as mining on these blocks has decreased. Below is a chronological time line of some of the most prominent hard forks in the Bitcoin network:

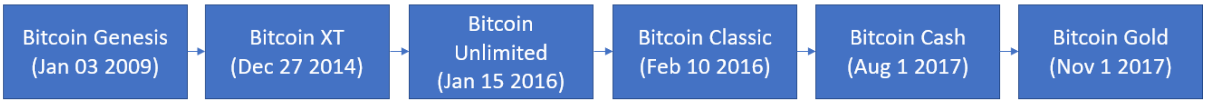
Bitcoin Genesis (Jan 03 2009)
Bitcoin XT (Dec 27 2014)
Bitcoin Unlimited (Jan 15 2016)
Bitcoin Classic (Feb 10 2016)
Bitcoin Cash (Aug 1 2017)
Bitcoin Gold (Nov 1 2017)

Figure 4: Bitcoin Hard Forks

Bitcoin XT (Bitcoin XT, 2014) was the first hard fork. It was launched by Mike Hearn with the aim of increasing the number of transactions per second from seven to twenty-four. This meant increasing the block size to 8MB. While it saw an initial success and even though it is still available, its popularity eventually declined.

Although Bitcoin XT declined there was still a desire to have bigger blocks and so Bitcoin Classic (Bitcoin Classic, 2016) was launched. Classic differed from XT in that instead of increasing the bloc size to 8MB it aimed to increase it to 12MB. Initially a success, Classic had around 2,000 nodes on the network however, today that number has dwindled to around 100.

Uncertainty still surrounds Bitcoin Unlimited (Bitcoin Unlimited, 2016) as although the code has been released, its developers have yet to specify what type of fork it will entail. The idea behind Unlimited again is block size related. Miners are able to set a limit of up to 16MB as the size of the block they wish to mine.

XT, Classic and Unlimited were hard forks caused by the launching of alternative software. Bitcoin Cash and Bitcoin Gold were hard forks that received support due to their cryptocurrency. Bitcoin Cash (Bitcoin Cash, 2017) was a response to Segregated Cash (Segwit, 2017), a soft fork that aimed to reduce the size of the bitcoin transaction. Some bitcoin users wished to avoid this protocol update and so opted for the hard fork Bitcoin Cash. According to Coinmarketcap.com, as of today, Bitcoin Cash is the fourth largest cryptocurrency in the market (Top 100 Cryptocurrencies By Market Capitalization, 2018).

Bitcoin Gold differs from the original Bitcoin in that it aims to reduce the importance of large miners on the network and make Bitcoin decentralised again (Bitcoin Gold, 2017). The creators of Cash believed that mining had become very centralised with miners who held superior equipment and hardware dominating the network. In order to achieve their goal, they focused on the proof-of-work algorithm, changing from SHA-256 hashing approach to memory hard equihash.

4.8.3 Accessing Bitcoin Data

The data in the Bitcoin blockchain is stored in a custom binary format. In order to access the blockchain data directly, one usually writes their own parsing function. The resulting information drawn from the blockchain can be checked via blockchain explorer platforms.

Platforms such as blockchain.info (Blockchain Info, 2017) and blockexplorer.com (Block Explorer, 2018) have been developed to provide information on the blocks, addresses, and transactions on the Bitcoin blockchain. They also provide both live and historic data in connection with the mining and network activity such as, the twenty-four-hour average block size, the aggregate number of confirmed Bitcoin transactions and the aggregate number of transactions pending. On some platforms statistics are shown in relation to the market price and the number of bitcoin transactions in the last twenty-four hours.

Blockchain.info have developed an API that allows the user to query the Bitcoin network and draw information in real time such as, the average time between blocks in seconds, the current difficulty target and the current block reward received by miners (Blockchain Info, 2017).

4.9 Ethereum

The focus of this dissertation will be on the transactional data of the Ethereum Blockchain. Vitalik Buterin, the founder of Ethereum, proposed the technology in 2013 via the release of his white paper (Buterin, Ethereum White Paper, 2013). Buterin wished to develop a technology superior to that of Bitcoin. He once wrote that "Bitcoin was designed to be a [Simple Mail Transfer Protocol] SMTP. It's a protocol that is very good at one particular task. It is good for transferring money, but it was not designed as a foundational layer for any kind of protocols to be built on top."

Vitalik wanted to create a more expansive technology that was capable of transferring not only cryptocurrency but also more complex assets such as smart contracts and smart property. Buterin once likened the Ethereum blockchain to a smart phone (Devcon3, 2017). Instead of the blockchain having a single functionality he wanted to implement a blockchain protocol that supported a programming language allowing the user to create applications. In order to achieve this, he decided to build and implement a robust scripting language within the blockchain structure. This led to the development of a Turing-complete programming language which can be executed on the blockchain.

A Turing-complete machine, called the “Universal Turing Machine”, was developed by Alan Turing. He wanted to create a machine that could take any program, in any language, and run it provided it had enough time and memory to do so (Agar, 2017). Ethereum is an example of a Turing-complete platform with its own distributed ecosystem. This ecosystem includes three main components; a file storing system, a messaging system and a consensus mechanism.

4.9.1 Ethereum Accounts

There are two types of accounts on the Ethereum Blockchain; externally owned accounts (EOAs) and contracts. Ethereum differentiates between these two types of accounts by examining the “state” of the ledger of these transactions (Wood, 2014). Both types of accounts contain a nonce, an ether balance and a storage field. The nonce is used to ensure that replay does not occur and the transaction is processed once. Ether is the cryptocurrency of the Ethereum blockchain and is used to process transactions. It can also be sent from one address to another. The contents of contract accounts differ from EOAs in that they have an additional field called the contract code. EOAs operate in a similar way to accounts on the Bitcoin network.

The private key is used to derive the public key which in turn is used to generate the address of the EOA. Using this address, it is possible to transfer of Ether from one address to another. These transactions are noted on the decentralised ledger.

Transactions are carried out in Ethereum using gas, which is an execution fee expressed in Ether. The STARTGAS and GASPRICE fields are used to prevent infinite loops and computationally excessive transaction processes. STARTGAS is a ceiling value on the number of computational steps allowed to be taken in order to complete the transaction. GASPRICE represents the fee that the sender must pay in order for the transaction to be processed. STARTGAS and GASPRICE help Ethereum combat denial of service attacks. Transactions that are seen to be computationally expensive will have a higher gas fee that those that are more feasible (Buterin, Ethereum White Paper, 2013). Gas is also used to prevent infinite loops in contracts by aborting a transaction once it exceeds the gas limit.

Contract accounts are different to EOAs and are usually referred to as smart contracts. “A smart contract is the simplest form of decentralized automation and is most easily and accurately defined as follows: a smart contract is a mechanism involving digital assets and two or more parties, where some or all of the parties put assets in and assets are automatically redistributed among those parties according to a formula based on certain data that is not known at the time the contract is initiated." (Buterin, DAOs, DACs, DAs and More: An Incomplete Terminology Guide, 2014).

Smart contracts can be viewed as computer programs, held in the blockchain that can be invoked to carry out a task. Buterin likens them to "autonomous agents" that live inside of the Ethereum execution environment, always executing a specific piece of code when "poked" by a message or transaction (Buterin, Ethereum White Paper, 2013). These smart contracts have their own balance of funds. Users can call the contract function through the API used by the contract program (Narayanan, Bonneau, Felten, Miller, & Goldfeder, 2016). It is also possible for the contracts to send or receive money.

The Ethereum platform and smart contracts have been likened to vending machines in the past. When a consumer wants to purchase a product from the vending machine they enter the coin and the code of that product and the product is returned without the use of an intermediary. Smart contracts are similar and can be thought of as the product held within the vending machine (Ethereum platform). It is possible for a user to interact with a smart contract without the need for a third party.

Smart contracts are written in the programming language 'solidity'. They can be used to formalise relationships between people, institutions and the assets they own (Kosba, Miller, Shi, Wen, & Papamanthou, 2016). The crucial attribute of a smart contract is that there exists only a fixed number of users. They can act as a legal instrument whereby they are executed should the terms of the contract be met. The rules of the contract are set out using machine readable code. Parties that interact with the contract must adhere to the rules should they wish the contract to execute. Of course, it is not obligatory that one should enter into agreement with a smart contract. If they are not satisfied with the conditions they have the option not to invoke it.

Ethereum, and its smart contract functionality, is being used to explore how one can manage various types of objects such as digital assets, financial instruments and recording ownership of intangible assets. The structure of these smart contracts ranges from simple to complex.

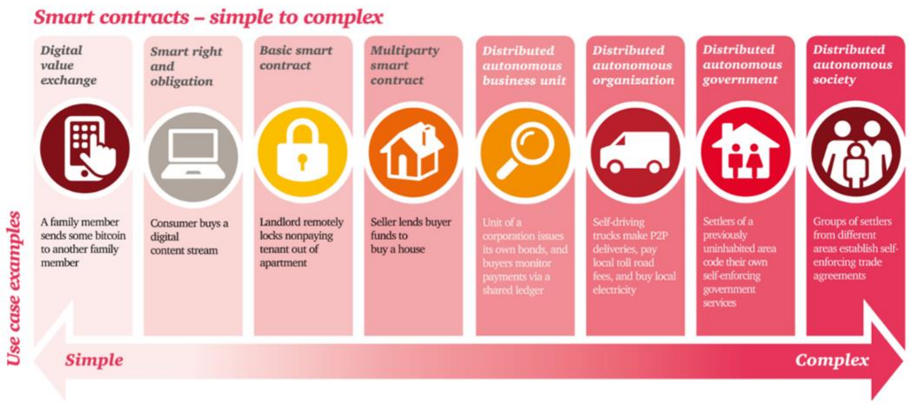
Smart contracts – simple to complex
Digital value exchange
Smart right and obligation
Basic smart contract
Multiparty smart contract
Distributed autonomous business unit
Distributed autonomous organization
Distributed autonomous government
Distributed autonomous society
A family member sends some bitcoin to another family member
Consumer buys a digital content stream
Landlord remotely locks nonpaying tenant out of apartment
Seller lends buyer funds to buy a house
Unit of a corporation issues its own bonds, and buyers monitor payments via a shared ledger
Self-driving trucks make P2P deliveries, pay local toll road fees, and buy local electricity
Settlers of a previously uninhabited area code their own self-enforcing government services
Groups of settlers from different areas establish self-enforcing trade agreements
Use case examples
Simple
Complex

Figure 5: Smart Contracts – Simple to Complex

Source: (PWC, 2018)

Contracts are able to send messages from one contract to another. Messages are similar to transactions produced by EOAs except that in a contract they are not produced by an external operator. Messages can be used to invoke code on the receiving contract. Similar to transactions, messages also have a STARTGAS value.

Smart contract functionality has given rise to the production of DAOs and DApps.

4.9.1.1 DAOs

DAOs can be considered as a democratic organisation built on the Ethereum platform. Buterin defines them as “an entity that lives on the internet and exists autonomously, but also heavily relies on hiring individuals to perform certain tasks that the automaton itself cannot do “ (Buterin, DAOs, DACs, DAs and More: An Incomplete Terminology Guide, 2014). In general, DAOs have a central manager that decides who the members of the organisation will be and what voting rules will be implemented. That is, DAOs require interaction from users in order to ensure that the mechanisms of the protocol are executed. DAOs also hold what is known as internal capital. This means that the DAO can use this capital to reward its users should they carry out certain activities.

4.9.1.2 DApps

DApps are decentralised applications which are similar to smart contracts but differ in two ways. There can be an infinite number of parties that interact with the application and the application itself does not need to be financial (Buterin, DAOs, DACs, DAs and More: An Incomplete Terminology

Guide, 2014). As they are decentralised they are not owned by one person but instead owned by many people. A DApp involves one or more contracts interacting with each other with the possible aim of delivering a service to the userbase. According to David Johnston's White Paper on the Theory of DApps (Johnston, 2015), "the application must use a cryptographic token (bitcoin or a token native to its system) which is necessary for access to the application and any contribution of value from (miners / farmers) should be rewarded in the application's tokens" is a necessary feature the application must attain in order to be considered a DApp.

Tokens are issued via a crowdsale which is called an Initial Coin Offering (ICO). ICOs raise capital to fund the application. Tokens can be purchased during the ICO by investors in exchange for ether.

In general, tokens are much easier to work with and can be thought of as tickets used to gain access and interact with a DApp, much like using tokens at arcades instead of using actual cash. There are various ways in which tokens can be used. For example, tokens can be used to identify the owner as shareholder of the DApp. This means that these tokens can give the owner the right to vote. Tokens can also be looked upon as a currency for that DApp. They do not grant the owner any rights on the development of the app (Dannen, 2017).

Of course, the development of these DApps and issue of tokens require some sort of regulation on Ethereum's behalf to aid in the protection of its users. Ethereum has issued a set of functions and rules that issuers of tokens must follow. This standard is known as ERC20. This standard is not obligatory, however, most DApp developers are encouraged to use it as being compliant with ERC20 makes it easier to manage and transfer tokens. Failure to use ERC20 can result in complications surrounding tokens' interaction with various smart contracts, wallets and exchanges.

A list of these token addresses can be found on block explorers, such as Etherscan.io. Kosala Hemachandra, founder of MyEtherWallet, a free, open-source, client-side interface for generating Ethereum wallets that also allows for interaction with the Ethereum blockchain (My Ethereum Wallet, 2018) has also published a list of token addresses to his GitHub page (Hemachandra, 2018).

Although the majority of DApps are legitimate, a lot of ICOs can be fraudulent. Etherscamdb.info is dedicated to identifying and tracking potential scam or fraudulent addresses that appear on the Ethereum Blockchain (Etherscamdb, 2017) which can be ICO related or they may fall under phishing.

Of course, the addresses flagged for inspection need not be related to DApps or ICOs. Scams are reported directly to the site or via google. These scams are then investigated, categorised and published to the site and GitHub account.

4.9.2 Mining in Ethereum

Mining in Ethereum differs from that of Bitcoin in that in Ethereum, as well as the transaction list, the blocks also contain a copy of the most recent state. In addition, the block difficulty and block number are stored in the block.

Before a miner can begin hashing, they must ensure that the previous block exists and is valid. They must also check the timestamp of the block and ensure that it is greater than the previous block but less than 15 minutes into the future. The proof-of-work must also be validated before hashing a new block can commence. The validity of the block number, difficulty, transaction root, uncle root and gas limit must also be considered.

The gas limit is used to ensure that efficient computation is being adhered to when trying to solve the proof of work algorithm, called the Ethash. The miners must find a nonce, that is below a certain level of difficulty, that satisfies the proof-of-work algorithm. The length of time spent locating the nonce is proportional to the difficulty level. This means that, like in mining bitcoin, the difficulty level can be used to control the number of blocks being created over a certain period of time.

In comparison to a block in the bitcoin network being created on average every ten minutes, the average time of producing a block in Ethereum is twelve seconds. This design in Ethereum provides a barrier to producing a fork and also to the 51% attack. The 51% attack is when a single miner dominates more than half of the mining power which could lead them to being able to rewrite historical blocks.

Miners receive an award of 3 ETH upon successful completion of a static block (Ethereum Mining Rewards, 2017). In some cases, individual miners will pool their resources together in order to benefit from 'strength in numbers'. Usually, this is achieved via cloud mining or cloud hashing[1].

Hashflare.io is an example of a cloud mining provider. They house hundreds of mining equipment in their data centres. When aspiring miners sign up to Hashflare they are connected to what is known as a pool. There are multiple pools to choose from with each pool containing various numbers of miners. Once mining commences and blocks are successfully hashed the customer receives a payment that is proportional to their share of the overall mining power (Hashflare, 2014-2018).

As with Bitcoin, mining is only one way of attaining the cryptocurrency Ether. The second way is to purchase Ether via an exchange or receive Ether as a payment for goods and services. CoinMarketCap (CoinMarketCap, 2018) and Etherchain provide information on the price of Ether compared to various fiat currencies. Coinmarketcap presents the price change in the form of graphs and while Etherchain also does the same they also provide the data in the form of a table. This allows the user to download the table and create their own plots (Ethereum Price in USD, 2015-2018).

4.9.3 Ethereum Hard Forks

As with Bitcoin, hard forks are also present on the Ethereum network, some of which were planned while others were unplanned. The following is a timeline of some of the hard forks in Ethereum:

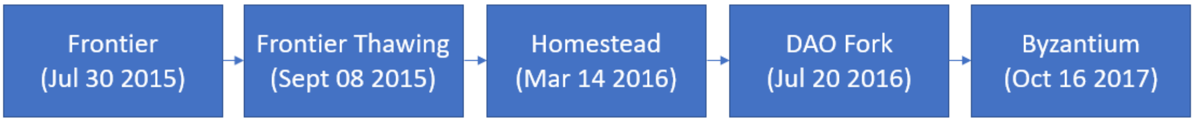

Figure 6: Ethereum Hard Forks

Frontier, Frontier Thawing, Homestead and Byzantium were planned with some of these hard forks bringing about updates to the Ethereum protocol. Undoubtedly, the most talked about hard fork is

[1] Users purchase mining capacity of hardware from datacentres. This means that the potential miner does not need to have the mining hardware locally. (Best Bitcoin Cloud Mining Contract Reviews and Comparisons, 2017)

the DAO fork. The DAO was a smart contract on the Ethereum network that served the purpose of providing a venture capital fund to be used in the creation of all future DApps on the network. However, upon creation, multiple bugs were present in the code, one of which was exploited by a hacker allowing huge amounts of funds to be drained. As a result, the DAO hard fork was introduced as an attempt to move funds tied to the DAO to an alternative smart contract whose sole purpose was to return the stolen funds to its investors. This hard fork split Ethereum into Ethereum and Ethereum Classic. Ethereum is the original Ethereum network that includes the correction of the DAO attack whereas Ethereum Classic contains the DAO error. Both forks have their own version of the Ether cryptocurrency, with Ethereum having Ether and Ethereum Classic supplying Ether Classic.

4.9.4 Accessing Ethereum Data

One way in which a user can access and download the Ethereum data is by using a command line interface called Geth. Geth can be downloaded locally on the user's PC allowing them to access and run a full Ethereum node which is implemented in Go, a programming language. The data can be accessed via three possible interfaces, namely, JSON-RPC, Javascript as well as command line options. Javascript uses web3.js which is an API that allows programs written in Javascript to interact with a node using JSON-RPC. Geth allows the user to mine ether, transfer funds between addresses, send transactions, explore the block's history and create smart contracts (Geth, 2017).

Etherscan is another way of accessing information and can be used to verify that the data downloaded using Geth is correct. Etherescan is a block explorer and analytics platform for Ethereum (Etherscan, 2018) that is operated independently of the Ethereum Foundation. Etherscan provides a search engine for the blockchain where users can look up transactions and check to see if they are pending or have been validated. Etherscan makes the blockchain information available by indexing the distributed ledger and publishing the information in real time through their site.

In January 2018, Andrew Collier launched the 'ether' package for R (Collier, Ether, 2018). The package uses the JSON-RPC API to connect to the Ethereum network. It allows the R user to interact with the Ethereum Blockchain by querying blocks and their transactions. For example, its inbuilt functions allow the user to retrieve information such as the current gas price, the most recent block number, the ether balance of any given account and the number of transactions initiated from a given address.

Collier has stated that, as part of future development, he wishes to build a function(s) that allows the user to interact with and interrogate smart contracts (Collier, An Ethereum Package for R, 2018).

5 Data Management

Since its launch in 2015, the size of the Ethereum Blockchain has grown exponentially. According to bitinfocharts.com there are almost six million blocks with roughly 300 blocks being created on average every hour. As of today, the size of the Ethereum database is 667.10GB. The data stored on the blockchain has the potential to reveal interesting trends and patterns which is something that we will explore later in section 8.2. However, it is clear that if we are to examine the blockchain data we first need to explore the various techniques available to handle data of such size. Over the last decade, many technologies have been launched to help users process, store and analyse 'Big Data'.

Apache Hadoop is an opensource software released in December 2011 designed to process and store large volumes of data. Hadoop achieves this by distributing the processing of data across multiple computers. By splitting large data files into smaller, more manageable clusters, it can then allocate these clusters to a single computer or thousands of computers. Each computer is able to store and process the data (Vavilapalli, 2013).

MapReduce is the programming paradigm in Apache Hadoop that focuses on the parallel processing of large scale data. It achieves this by splitting data into large various blocks, it allows analysis to be carried out in parallel (Dean & Ghemawat, 2008). MapReduce refers to two key distinct operations that occur in Hadoop. The first task is map. This involves taking the data set and converting it into different data sets where each data point is broken down into a tuple in the form of key/value pairs. The reduce task, performed after the map task, takes the map output and combines it into a smaller set of tuples depending on what the user's request is. For example, the user may wish to draw out the highest value for each key in the tuple set.

Python and R are both programming languages that are used to manage and carry out statistical analysis of data. They have made huge developments since their release leading to an increase in their capabilities when it comes to big data manipulation. With the launch of various libraries, it is possible to implement machine learning algorithms thus providing insight into the data. In Python, Pandas and NumPy are popular libraries used to process large multidimensional arrays and matrices. R also has numerous efficient data management libraries. The rmr2 is a library in the RHadoop package which allows the user to connect to Hadoop and create customised MapReduce functions.

As well as Hadoop, SQLite is another popular means of storing large volumes of data. SQLite, a type of SQL database engine, is a widely used database engine (SQLite, 2000-2018). Instead of the typical client-server database engine, SQLite is a serverless database system. 'SQLite3' is part of the Python Standard Library and provides an interface to the SQLite database. Using SQLite3 we can read and write files to SQLite. In R the package that allows the user to read and write files to the database's system is called 'RSQLite'. Storing files in .db format and accessing them through the DB browser for SQLite allows us to be selective in choosing data for our analysis. That is, all of the data does not have to be read directly into our chosen analytics tool. Instead, we can run an SQLite query and retrieve subsets of the data.

5.1 File formats

The format of files can have a huge impact on the amount of space being used by the data. .Rds and .RData are inbuilt file formats in R and are much quicker to process than the common .csv format. They are binary in nature and thus optimise speed and compression ratios (Gillespie & Lovelace, 2016).

Table 1 highlights the impact on disk usage for a given data set, containing 1,000 rows, saved in different formats.

Table 1: Comparing File Formats

Format	Size (MB)
co2.csv	16964.9
co2.feather	9360.3
co2.Rdata	62.6
co2.Rds	62.6

Source: (Gillespie & Lovelace, 2016)

Using R's native format can lead to a space saving of more than 100 times that of a .csv or .feather file.

Another popular file format is JSON (JavaScript Object Notation). Both the Ethereum tokens and scam files, sourced from GitHub, are in this format. JSON files are easy to create using the JavaScript Programming Language. JSON is built on two structures, an object, containing a collection of name/value pairs and an array, maintaining an ordered list of values (Bray, 2017). Its structured format is easy for both humans to read, write and parse. The 'jsonlite' package in R is used to generate and parse this data format (Ooms, Temple Lang, & Hilaiel, 2017).

5.2 Cloud Computing and Big Data

We believe that it is only when a user begins to access and understand the Ethereum blockchain data that they can truly appreciate its sheer volume and the rate at which it is produced. Without taking subsets of the data it most certainly falls under the category 'Big Data'. It is clear that scalability is a great hurdle when dealing with large, high dimensional data structures. Storing such data locally has become infeasible however the need to process large data volumes has not diminished. Instead, many enterprises have adopted cloud computing technology to help manage big data (Liu, 2013).

Cloud computing can be viewed as a way of outsourcing a task or a number of tasks (Armbrust, Fox, Griffith, Joseph, & Lee, 2010). Cloud computing is an umbrella term for many types of services including storage, software and infrastructure. Storage as a service is provided by Google Drive (Google Drive, n.d.) and Dropbox (Dropbox, n.d.). They allow the user to upload a certain amount of data for free and should they wish to increase the storage size they can do so for a fee.

Other companies provide infrastructure as a service for example Amazon Web Services (AWS) and Google Compute. They provide the user with the tools to set up and launch virtual machines which is essentially running a computer over the internet inside a local computer. The ability to rent such machines provides the user with the facility to integrate frameworks for parallel data processing. In doing so they can also access cloud resources and deploy their programs (Warneke & Kao, 2009). For example, a user can deploy a virtual machine on AWS and install statistical software such as R or Python. RosettaHub is another cloud-based service provider with a focus on creating an e-science and e-learning platform. It, too, provides access to multiple cloud based analytic and data management technologies including Rstudio. However, services like AWS and RosettaHub come at a cost.

Software as a service is another popular cloud computing facility. Instead of installing and maintain the software locally on your PC it is accessed via the internet. For example, Python (Python

Anywhere, n.d.) and R (RStudio Cloud, 2018) both have free web-based versions. It is possible to launch multiple scripts on these web-based versions and run computations in parallel allowing the user to perform large scale computing tasks. Of course, it is possible to open and run multiple scripts locally on one's PC, however, this is a much slower and limited process and less efficient than delegating tasks to a network of servers.

Essentially, cloud computing is using someone else's computational resources over the internet. Cloud computing is a very useful resource to avail of when it comes to processing, storing and analysing Big Data. It also allows for greater flexibility and security. The user is able to access the cloud from their laptop, phone or any device that is able to connect to the internet. The level of security of the cloud will depend on the cloud provider.

Cloud computing is a go to service when the demand for resources is high. For example, depending on numerous factors such as the volume of data to be processed or the type of computational task required, machine learning algorithms (section 6) and data analytics techniques can require a lot of processing power. Using the cloud enables the user to split the tasks in parallel across various servers. In general, if the data is considered to be 'Big data' and the computational task requires a large amount of power, it may be beneficial to avail of cloud computing. It may even be the case that the user's local PC or laptop does not have the resources or power necessary to carry out the computational task. In such cases, cloud computing is a must. Of course, when availing of this service it is important to be frugal when it comes to deciding how many servers one needs. Launching the power of ten servers when only five is needed is a waste of resources and can be detriment to one's budget.

6 Machine Learning

As well as being used for data manipulation and statistical analysis, R and Python also have built in libraries to implement machine learning techniques on data. Machine learning is a technique in which we, the users, programme a computer in a certain way, allowing the computer to learn from the data. To give a slightly more technical definition, Tom Mitchell describes it as:

"A computer program is said to learn from experience E with respect to some task T and some performance measure P, if its performance on T, as measured by P, improves with experience E."

(Mitchell, 1997)

6.1 Feature Selection

In general, machine learning involves taking a dataset with a given number of features where the user examines these features and if applicable, uses them as inputs to the machine learning algorithm. Of course, not every feature is seen as being relevant for the algorithm. For example, the user may wish to select features that give a meaningful representation to the data. In order to achieve this, the user may select features that have a certain degree of variability. In the case of the Ethereum transactional data, the block number may be less important that the number of transactions in a block. Of course, the choice of features for the machine learning algorithm will depend on the user's objective, the algorithm selected and the desired output.

Instead of relying solely on the user to select the important features in the data for the model, a random forest can be used. In order to explain what random forest is, we must first explain a decision tree. A decision tree model allows the user to develop a classification system that is able to predict or classify future observations based on a set of decision rules (Safavian & Landgrebe, 1991). In essence they can be used to make logical sense of the data. A decision tree is constructed of nodes, branches and leaves. Each node on the tree represents a feature of the data with the leaf representing the outcome. The branches represent a decision rule. The goal in generating a tree is to ensure that at each leaf the error is minimised. That is, we can enter each node with a very high level of accuracy.

When multiple decision trees join together they become known as a random forest. That is, a random forest is an ensemble of decision trees. In general, the predictive power of a random forest is superior to that of an individual tree. In a random forest, each decision tree builds upon a random subset of the data features.

From a random forest it is possible to determine how important an individual feature of the data is or in other words determine the variable importance within the dataset. This can be done by calculating the impurity at each node. Node impurity reflects how well the tree has split the data. A high node impurity results from a poor split whereas a low node impurity reflects a good split. A way of measuring node impurity is by using the Gini Index (Brieman, Friedman, Olsen, & Stone, 1984).

$$Gini = 1 - \sum_{j=1}^{C} p^2(j|t) \qquad where, p(j|t) = \frac{p(j,t)}{p(t)}$$

That is, $p(j|t)$ is the probability of a case falling into node j given that the case is in node t.

In order to decide if a variable is important or not we need to look at the mean decrease in Gini. This involves calculating the changes in Gini for each variable, summing them and then normalising them.

Running a random forest to aid decision making for feature selection is in effect a prelude to implementing an unsupervised learning algorithm.

6.2 Unsupervised Learning

There are many types of machine learning systems, however, when it comes to detecting patterns in data, unsupervised learning is the approach taken. Many researchers and companies have begun to delve into the art of machine learning in order to better understand the movements in the blockchain. ConsenSys, a company dedicated to harnessing the power of the Ethereum Blockchain, have developed an analytics platform called “Alethio” (ConsenSys, Alethio, 2017) which aims to help users visualise, interpret and react to the blockchain in real time (ConsenSys, Using Machine Learning to Understand the Ethereum Blockchain, 2018). Sarah Meiklejohn et al., used heuristic clustering to detect fraudulent behaviour in the Bitcoin network (Meiklejohn, et al., 2013). Similarly, Jason Hirshman et al, from Stanford, used unsupervised learning to detect abnormal behaviour in the Bitcoin transaction network (Hirshman, Huang, & Macke, 2013).

When we talk about unsupervised learning we are usually faced with data that is unlabelled and the system tries to learn without instruction. Unlabelled data is naturally occurring data such as tweets, photos or audio recordings. In the case of a data set, the data is said to be unlabelled if there is no indicator field in the data set explaining why the data has occurred in a certain way. That means there is no label on the data set that is descriptive or informative to the data set. For example, if we look at the Ethereum transactional data we could describe it as being unlabelled data. In order to transform this data into labelled data we could add a tag to each transactional indicating whether the amount transferred was small, medium of large. In the case of unsupervised learning of unlabelled data, we know little or nothing about what the results should look like. Unsupervised learning is not predictive in nature and as a result may be used to detect patterns in the data.

There are many types of unsupervised learning algorithms, however, clustering is the most popular. Clustering allows us to derive structures within the dataset that would otherwise be invisible to the naked eye. It is a method of creating groups of objects, or clusters, in such a way that objects in one cluster are very similar and objects in different clusters are quite distinct (Gan, Ma, & Wu, 2007). Clustering can be used to detect patterns within a dataset, however, at no point does the user tell the algorithm which group a particular point belongs to: the algorithm is able to decide this without the user's input. Clustering techniques can be broadly classified into two categories: hierarchal and partitional.

Hierarchical clustering algorithms deal with all values of k in the same run instead of constructing a single partition with k clusters. In this sense, hierarchical clustering can be viewed as a nested sequence of partitions. Hierarchical clustering can be split into two techniques: divisive and agglomerative. Agglomerative hierarchical clustering commences by placing each observation in its own cluster with pairs of clusters being merged until all observations are in a single cluster. It is known as the 'bottom up' approach. Divisive hierarchical reverses the procedure. All observations are placed in a cluster at the top of the hierarchy with recursive subdivision then taking place splitting the clusters into smaller pieces. This method is also called the 'top down' approach (Gan, Ma, & Wu, 2007).

Unlike hierarchical clustering, Partitioning methods split the data into k clusters. In partitioning algorithms two conditions must be met. The first being that each cluster must contain at least one observation with the second condition being that each observation must belong to no more than one cluster (Kaufman & Rousseeuw, Finding Groups in Data: An Introduction to Cluster Analysis, 1990).

6.2.1 Clustering and Evaluating Performance

There are two popular partitioning algorithms, namely, K-means and K-medoids (Partitioning Around Medoids (PAM)). These algorithms all require the user to select an initial parameter K, which is used in determining the number of clusters to be created. The way in which the user can determine the effectiveness of the number of clusters created is by examining the between sum of squares (BSS) divided by the total sum of squares (TSS). BSS is a measure of variability between clusters. This variability is calculated by summing the squared deviations from each cluster to another. The TSS is a summation of the variability within each cluster. The variability is calculated by taking the squared deviations from each observation in a cluster to that cluster's centre point and summing them.

When a cluster is created the aim is usually to have a high similarity within each cluster and a low similarity between each cluster. This means that if we have a high similarity within each cluster there will be low variability within each cluster. Also, if we have a low similarity between each cluster this would result in a high variance between the clusters.

6.2.2 K-Means

The term K-means clustering was coined by James MacQueen in 1967 and is the most popular partitioning clustering algorithm. The aim of K-means is to split a matrix of M datapoints and N dimensions into K clusters with the objective being that the within-cluster sum of squares is minimised (Hartigan & Wong, 1979). With a further objective that the between sum of squares (BSS) divided by the total sum of squares (TSS) in maximised. That is, the ratio BSS/TSS should be as close to 1 as possible.

Mathematically speaking that is, let $X = \{x_i\}, \text{where } i = 1, \ldots n$ be the set of n dimensional points to be clustered into a set of K clusters. The clusters are defined as $C = \{c_k, k = 1, \ldots K\}$. Let μ_k be the mean of cluster k with the square error between μ_k and c_k being defined as:

$$J(c_k) = \sum_{x_i \in c_k} ||x_i - \mu_k\ ||^2$$

and the goal is to minimise the sum of the square errors across all K:

$$J(c_k) = \sum_{k=1}^{K} \sum_{x_i \in c_k} ||x_i - \mu_k \,||^2$$

(Jain, 2009).

K-means can be applied to a wide range of data as it only requires a distance matrix for a pair of objects. In order to apply the K-means algorithm to a data set the user must first select k initial centroids, where k is the number of desired clusters. Each point in the data set is then assigned to the closest centroid with each group of points that are assigned to a certain centroid being referred to as a cluster. In order to establish which point belongs to which centroid a distance measure is used to evaluate the proximity of a point to a centroid. The Euclidean distance measure is often the preferred choice to quantify such distance and as a result, K-means finds spherical shaped clusters in data. It is important to standardise the data before applying the k-means algorithm as the Euclidean distance is sensitive to the differences in the magnitudes or scales of the input data (Milligan & Cooper, 1988).

Once the points have been assigned, the centroid of the cluster is updated based on the points in that cluster. This is repeated until the addition of new points renders the centroid unchanged, that is until all the centroids remain the same (Tan, Steinbach, & Kumar, 2013).

The following diagram depicts the process involved in implementing the K-means algorithm:

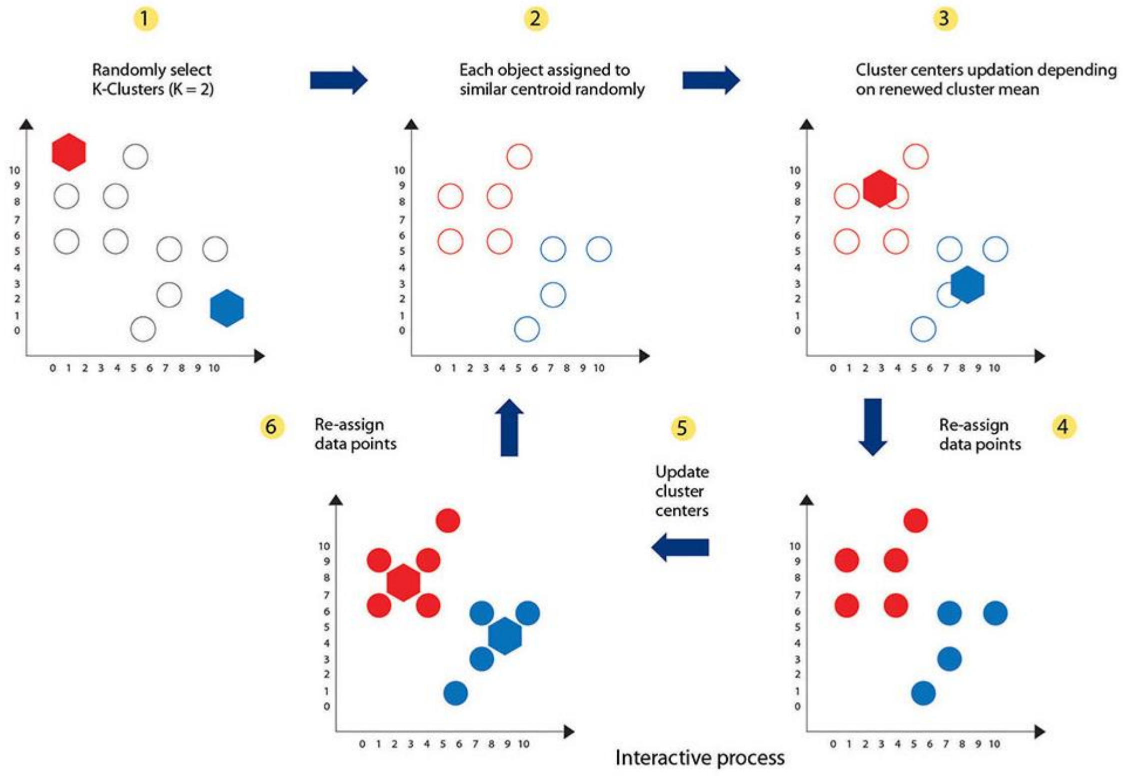
1
Randomly select K-Clusters (K = 2)
2
Each object assigned to similar centroid randomly
3
Cluster centers updation depending on renewed cluster mean
4
Re-assign data points
5
Update cluster centers
6
Re-assign data points
Interactive process
0 1 2 3 4 5 6 7 8 9 10

Figure 7: K-Means Clustering

Source: (K-Means Clustering, 2017)

6.2.3 K-medoids

K-medoids clustering or Partitioning Around Medoids (PAM) (Kaufman & Peter J. Rousseevw, Clustering by means of medoids, 1987) is where each cluster is represented by one of the objects in that cluster. Again, like K-means clustering, the objective of this algorithm is to ensure that the within-cluster sum of squares is minimised or that the between sum of squares divided by the total sum of squares is maximised. K-medoids aims to minimise the dissimilarities between points labelled to be in that cluster with the designated centre point of that cluster. It differs from K-means in that it selects datapoints as centres. These are known a medoids which are taken to be the most centrally located points in the set.

The algorithm operates by initially randomly selecting K of the n datapoints as the medoids. It then takes each data point and assigns it to a medoid based on closest proximity. For each medoid and data point the average dissimilarity of each data point to all the other data points associated to the medoid is calculated. This is repeated until there is no change on the allocation of data points. The goal of the algorithm is to minimise the average dissimilarity of the datapoints to their closest selected medoid (Huang, 1998).

6.2.4 Selecting K

Of course, selecting the parameter K, as the number of clusters can be somewhat subjective. There are various techniques that have been developed to aid the user in their choice such as, the elbow method, the average silhouette method, and the Davies Bouldin Index.

6.2.4.1 The Elbow Method

The Elbow Method looks at the total within cluster sum of squares as a function of the number of clusters. In selecting the optimal K, the user should choose the number of clusters so that when an additional cluster is added there is little to no improvement on the within sum of squares. In locating the optimal cluster, the user must first compute the clustering algorithm with various values for K, where each K gives a within sum of squares value. Each K is then plotted against its within sum of square value with the user hopefully producing an elbow shaped line graph. The location of the bend (elbow) in the graph signifies the most appropriate value for K.

6.2.4.2 The Average Silhouette Method

The average silhouette method measures how well each data point lies within its cluster. The higher the average silhouette width, the better the clustering. Different average silhouette measures are calculated for different values of K clusters. The optimal number of clusters K is the one that maximises the average silhouette over a range of possible values for K (Kaufman & Rousseeuw, Finding Groups in Data: An Introduction to Cluster Analysis, 1990).

6.2.4.3 The Davies-Bouldin Index

Finally, the Davies-Bouldin Index (Davies & Bouldin, 1979) is a cluster separation measure that can be used to determine the optimal value of K. Davies-Bouldin looks at the ratio between the within cluster distances and the between cluster and favours a low score.

6.3 Dealing with High Dimensional data

In many cases, as with the transactional data on the Ethereum Blockchain, the data can be described as high dimensional. That is, there exists multiple data points with multiple features. High dimensional data sets lead to several problems when applying clustering algorithms. In these types of data sets it is difficult to differentiate similar data points from dissimilar ones because the distance between any two points are highly similar (Beyer, Goldstein, Ramakrishnan, & Shaf, 1999). In addition, high dimensional data tends to have clusters embedded in subspaces with different clusters existing in different subspaces (Agrawal, Gehrke, Gunopulos, & Raghavan, 1998).

Due to these issues, clustering algorithms, such as K-means clustering, fail to perform well in high dimensional data. As a means of overcoming such a hurdle and allowing more coherent patterns to be detected, the data is reduced from a higher dimension to a lower dimension via a principal component analysis (PCA) (Jolliffe, 2002). PCA takes into consideration the interrelationship between variables and tries to retain the variability within the dataset. In a PCA the principal components are ordered in such a way that the first few represent most of the variability present in all of the original variables (Gan, Ma, & Wu, 2007). In order to project the data from a high dimension to a lower dimension, PCA adopts linear algebra techniques. Principal components are located on the basis that the new components are a linear combination of the original variables. The new components are also perpendicular to each other and they reflect the maximum amount of variation in the data (Tan, Steinbach, & Kumar, 2013). Finding this new dimensionality echoes the variability of the original data.

It is common for PCA to be applied to high dimensional data as a prelude to the application of the K-means algorithm (Hongyuan, He, Ding, Gu, & Simon, 2002). It has been proven that PCA automatically performs data clustering according to the K-means objective function and as a result is a highly suitable dimension reduction technique that aids the performance of the K-means algorithm (Ding & He, 2004).

In the past clustering algorithms such as K-means have been used to detect patterns in Bitcoin transactions (Hirshman, Huang, & Macke, 2013). In this dissertation we investigate the effects of clustering Ethereum transactions. We will also pay close attention to feature selection as well as how to determine the optimal value for K. In addition, we will also examine the effects of the algorithms performance by implementing the principal component analysis. The overall design of this analysis will be discussed further in section 7.

7 Design

In this section we will discuss the overall design of the analysis. Firstly, we will demonstrate how we will access and store the Ethereum Blockchain transactional data. In addition, we will also explore the other datasets that are available. These datasets will aid in describing the type of transactions taking place between addresses. Secondly, we will carry out a high level preliminary analysis of the data. This should aid us in understanding the environment surrounding the transactional data. It will also provide insight into the concept and structure of the blockchain. Finally, we will explore a clustering technique that will be used to detect patterns and possible anomalies within the transactional data.

7.1 Accessing and Storing Data

7.1.1 Ethereum Data

We will focus on analysing the Ethereum transactional data over a certain time frame. As a general approach, we will select a time series where there has been heightened activity on the blockchain.

Selecting such a time frame will ensure that there is a continuous flow of transactions with blocks being created frequently. This means that the data selected should be quite dense. Should we select a time frame that has a low level of activity it may be possible that we can determine clusters from the data manually by just looking at it. Running a clustering algorithm on such a data feed could possibly be seen as a waste of time. Ensuring that the data drawn is of a high velocity and of a large volume ensures that running the K-means clustering algorithm is worthwhile. We want to show that although the sheer size of the data selected may be somewhat overwhelming, it is possible to detect patterns within such a dataset.

Furthermore, aside from implementing a clustering algorithm to detect patterns we want to carry out a high-level preliminary analysis on the data to provide the reader with insight into the goings on of the transactional data. Choosing a period of high activity may present us with the opportunity to answer more questions in a more complete way. That is, we want to ensure that we are able to ask and answer questions more vigorously.

Etherscan (Etherscan, 2018) provides historical trends on the number of transactions over a given period of time so we will turn our attention there in order to locate a prime time frame.

In order to access this data, we will use the command line interface Geth and retrieve the data using the JSON-RPC interface. The data feed will be stored using the SQLite database. SQLite is also a free open source software and is compatible with 'DB Browser for SQLite' (DB Browser for SQLite, 2014). This is an open source, graphical user interface (GUI) front end that can be used as a cross platform, allowing us to search and edit the data. SQLite was selected for numerous reasons, one being that it is compatible with R which is our chosen programming language for carrying out the analysis. The package 'RSQLite' ('SQLite' Interface for R, 2018) in R allows us to establish a connection between R and the SQLite database. This then allows us to carry out our analysis[2]. SQLite is also compatible with Python which gives us that added level of flexibility should we wish to use Python to carry out our statistical analysis.

SQLite is also very simple to use as there is no configuration. There is no server process or the need for an administrator to create a new database instance or assign access permission to users. Should the system crash, for example, there are no actions required to recover the system. SQLite is compact. The SQLite library with all features enabled is less than 500KiB[3] in size. It is possible to disable some features and further reduce the size to 300KiB. The fact that it is so compact in comparison to many other databases, such as Berkeley DB library from Oracle or the CloudSpace database from IBM, means that SQLite is more favourable if computer RAM is a concern. In our case, we know that the magnitude of the data being drawn from the Ethereum Blockchain is quite large. Therefore, it is very important that the database engine itself is lightweight while still fit for purpose. This means that should we wish to store the data locally we want to choose a database that will not require a huge amount of memory (Distinctive Features Of SQLite, 2000).

[2] It will also be important to note which features of the Ethereum blockchain we wish to extract. We may wish to extract them all but some may be redundant as inputs to our analysis and extracting them may prove to be infeasible.

[3] 1 KiB equates to 1024 bytes

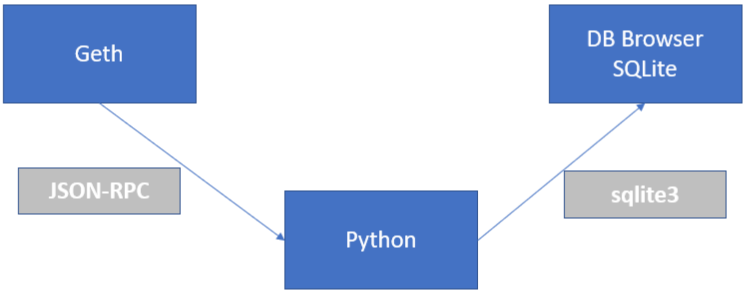

Figure 8: Accessing and Storing Raw Data

Should we wish to scale our analysis to incorporate more data it will be possible to use the cloud-based versions of both SQLite and R and launch workers on various nodes. DBHub.io is an open source data sharing, versioning and collaboration platform for SQLite databases (DBHub.io, 2016). DBHub.io was created by the developers responsible for DB Browser for SQLite. Their aim was to generate an optional cloud storage service for SQLite databases and they achieved this by developing and launching DBHub.io. Using a cloud storage service like DBHub.io would allow us to push our local database to the cloud. The cloud-based version of the data base can then be accessed using Rstudio cloud (RStudio Cloud, 2018). Using these cloud providers would allow us to deploy various sections of our analysis across various servers should we wish to do so. This in turn would aid us in overcoming the hurdle of scalability when it comes to processing and analysing large volumes of data.

7.1.2 External Data

Other datasets will also be used and where appropriate, merged onto the transactional feed thus providing us with a more granular description of the types of transactions being carried out. The datasets selected will provide information on scams, tokens, exchanges and coin mixers. The scams and tokens dataset will be sourced from various accounts on GitHub as discussed in section 4.9.1.2, whereas the information surrounding the exchange and coin mixers will be scraped manually from Etherscan. Information on the hourly price change will also be used which will allow us to translate the amount of Ether involved in a transaction into USD which will be sourced from etherchain.org (Ethereum Price in USD, 2015-2018) .

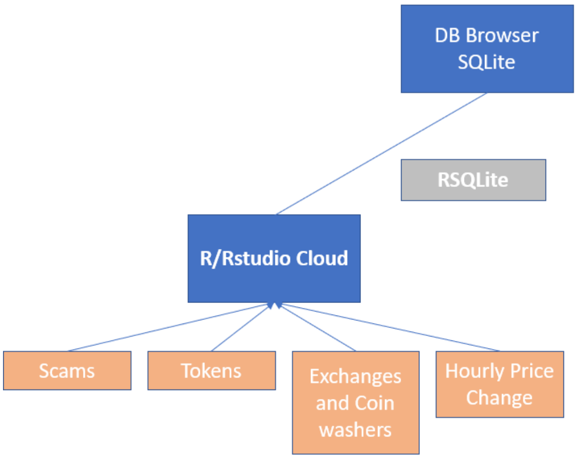

Figure 9: Adding External Data Feeds

7.2 High Level Preliminary Analysis

Once we have merged the external data files to the transactional data, we will carry out a high-level preliminary analysis in order to provide insight into the activities of the Ethereum transactions. We will look at the price trend in USD over the time frame selected as well as investigating how many transactions involve scams, tokens and exchanges or coin mixers. We will also try to answer the following questions:

1. What are the main types of scams that are occurring over this period?
2. How many transactions involve tokens and which tokens are the most popular?
3. Which exchanges are the most popular and how much ether in terms of USD is entering and exiting these exchanges?
4. Is there a correlation between the average number of transactions taking place in a day and the daily average price of Ether in USD?
5. Are there any miners that appear to dominate the mining community during this period of time?

When applicable, we will use ggplot (ggplot2, 2018) and other various packages in R in order to visualise the results.

7.3 Selecting Algorithm

In order to answer our research question, we need to ensure that an appropriate algorithm is selected. We want to be able to find groupings or hidden patterns within the Ethereum transactional data and as a result we decided that clustering the data would be most appropriate in achieving our goal. However, there are various types of clustering algorithms and selecting the most suitable involved a series of preliminary experiments. As discussed in section 6.2 we know that there are two types of clustering; partitioning and hierarchical. Taking a small subset of the data we decided to apply a hierarchical clustering technique to the data. In doing so we found the results to be somewhat meaningless as the algorithm detected the two obvious hierarchs; the transaction level data and the block level data. It was clear that this type of clustering was not suitable in finding the hidden patterns that we were looking for. Also, as we increased the size of the subset of data we were using to test the algorithm performance we found that the hierarchical clustering algorithm took longer to run as the size of the data increased. As the Ethereum transactional dataset is large, we concluded that hierarchical clustering would not be computationally feasible for such an analysis.

We then examined the effects of partitioning clustering and applied both the K-means algorithm and K-medoids algorithm to a subset of the data. Again, as we increased the size of the sample data being used to test the algorithm performance, we found that K-means was slightly faster to run than K-medoids. That is, we found that K-means had a lower computation cost than K-medoids. K-means is also more sensitive to noise/outliers than K-medoids. Although the main aim of our research is to see if we can detect patterns in the Ethereum transactional data using clustering, we do have a second objective that the K-means algorithm satisfies whereas K-medoids and hierarchical clustering do not and that is, if we wish to detect anomalies in the patterns formed, K-means is able to achieve this. Anomalies will be detected as being the outliers once the clusters have formed. That is, they will be the data points that are lying outside, rather than inside or close to the boundary of the formed clusters.

Another reason we are selecting K-means as our clustering technique is because, based on past research, it has proved to be a successful and robust method in detecting patterns. It is also highly suitable for transactional data (Jain, 2009). Many other researchers have used K-means as an unsupervised learning technique to detect anomalous behaviour in the bitcoin network (Hirshman, Huang, & Macke, 2013) (Lee & Pham, 2017) . Therefore, taking a similar approach when investigating potential anomalies in the Ethereum transactional data should prove to be suitable.

7.3.1 Feature Selection

When implementing the K-means algorithm, the main question we must ask ourselves is which features to select as inputs to our predictive model. Both feature selection and principal component analysis aim to reduce the number of features in a dataset, however, feature selection includes and excludes features present in the data without changing them. Principal component analysis differs in that it looks at the interrelationship between the features and tries to retain the variability within the dataset.

In order to determine which features are most relevant and most useful in clustering the Ethereum transactions we will first run the data through a random forest model. In order to determine the feature importance, we will calculate the node impurity at each split. Examining the mean decrease in Gini we will select the important features based on their lowest value. So as to not select every variable we will set a certain threshold that the mean decrease in Gini value must fall below. Features with corresponding values below this threshold will be selected as inputs to the model.

7.3.2 Examining Dimensionality

After feature selection, we will examine the dimensionality of the data. If we identify it as being high dimensional we will apply the principal component analysis (PCA) before running the data through the K-means algorithm (section 6.3). However, we will also evaluate the performance of the K-means algorithm with and without the application of the PCA. Of course, we will first standardise the data to have a mean of 0 and a variance of 1 as this ensures that an equal weighting is given to all the variables. If we have unequal variances, we run the risk of putting more weight on variables with smaller variances resulting in clusters being separated along variables that have a higher variance.

7.3.3 Selecting K

When applying the K-means algorithm we need to pre-select the number of clusters we wish to form. In order to achieve the optimal level of granularity, we will use three different measures for selecting the desired K, namely, the elbow method, the silhouette method and the Davies Bouldin index. We would expect that each method would give roughly the same value for K. As a means of comparison for the varying Ks and in order to understand the effects of implementing the PCA, we will create a table of results where we will compare the different between sum of squares divided by the total within sum of squares scores under the varying circumstances.

7.3.4 Evaluating Clusters and Detecting Anomalies

Firstly, we will carry out multiple experiments including applying K-means clustering with and without the implementation of the principal component analysis. We will also look at the performance of K-means given different values of K which will be determined using the three methods outlined in section 7.3.3. The clusters will be evaluated based on the ratio of between sum of squares divided by total sum of squares (section 6.2.1). Of course, we will seek to find a result that returns a value closest to 1.

In order to detect anomalies, we will look at the outliers across the clusters. First, we will identify the centres of each cluster. Taking an individual cluster, we will then calculate the distance of each point in that cluster to its centre. The points in the cluster that lie the furthest from their centre will be deemed as outliers. The Euclidean distance will be used as the metric for calculating the distance of a point to its centre.

$$Euclidean\ Distance = \sqrt{\sum (Cluster\ centre - point\ in\ cluster)^2}$$

8 Implementation

In this section we will discuss how we implemented our design. We will address the reason as to why we chose a certain time frame for selecting the Ethereum transactional data as well as how we accessed and stored the data. We will further try to understand the various aspects of the data by carrying out a high-level preliminary analysis. This should further aid our understanding of the data as well as identifying possible traits and features of the data. We will also discuss the various experiments carried out when testing the effectiveness of the K-means clustering algorithm. For each experiment we will discuss the implementation and the results. Both the high-level preliminary analysis and the subsequent implementation of the various clustering experiments will be carrying out using R. The raw Ethereum transactional data will be used for the K-means clustering, however, for the high-level preliminary analysis we will used the external data in conjunction with the raw transactional data. Using the external data will give us a better insight into the raw data and add an extra level of description to the transactions.

8.1 Accessing and Storing the Data

8.1.1 Ethereum Data

The Ethereum transactional data from the 01/01/2018 – 10/01/2018 inclusive was selected and stored in the DB Browser for SQLite. The reason for this selection was because during this time period, the peak in the number of transactions over a 24hour period occurred on the 4th January where a total of 1,349,890 transactions took place. During the first ten days we can see, from Figure 10, that there was heightened activity in the Ethereum transactional data. The fact that the transactions occurred at such a high velocity means that it would be impossible for the naked eye to determine clusters or patterns in this time frame or to answer high level preliminary questions about the data. Thus, this data selection proved to be a suitable candidate for our analysis as we want to show the positive effects K-means clustering can have on a large volume data set in determining clusters.

During the month of January, the price of Ether peaked. This was another reason for selecting a time frame during this specific month as we anticipated that there may be more variety in the transactions. That is, there may be a large amount of activity with exchanges where investors have decided to exchange their Ether for their desired fiat currency. We were interested to see what percentage of transactions involved interactions with exchanges (section 8.2.3).

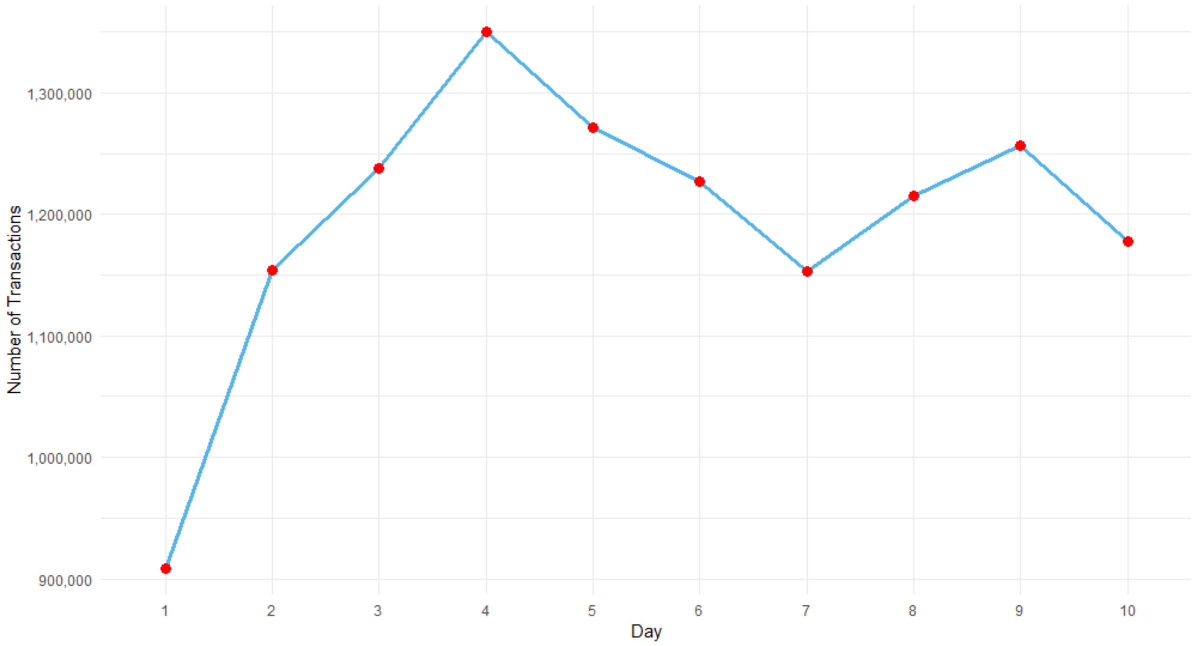

Figure 10: Number of Transactions for the first 10 days in January

Using Geth, we were able to connect to the Ethereum Blockchain using the JSON-RPC interface. Python was used to manipulate the data and certain features were selected at both the block level and transaction level. At the block level, the data feeds selected were the timestamp, the block number, the number of transactions present in the block, the gas used to generate the block as well as the gas limit and the miner who created the block. The timestamp on the Ethereum blockchain is of Unix format and was therefore translated to the standard day-month-year time format in Python which resulted in a more user-friendly format.

At the transaction level we were able to extract the sender's and receiver's addresses as well as the gas price and the gas used to carry out the transaction. The value transferred between addresses was also extracted. The reason we selected such a large number of features was because we did not want to use our personal judgement in determining the feature importance. Instead of being overly selective we decide to choose a large number of features with the intention of pruning them for the purpose of implementing the K-means algorithm at a later stage. It was also important not to prematurely drop any features as they may show some useful insights in our preliminary exploratory analysis.

8.1.2 External data

Using Etherscamdb.com we sourced over 4,000 scams (section 4.9.1.2) and tried to identify if any transactions from our Ethereum transactional data contained any scam addresses. The Etherscam database contains information such as the name of the scam, the URL connected with the scam, the address of the scam, the category into which the scam falls such as phishing or fake ICO and a short description detailing possible further information about the scam.

This scam database was merged onto the Ethereum Blockchain transactional data by examining if there was a match in address. If the sender or receiver of the transaction was a match to a scam address, then a flag was added to the Ethereum data indicating a scam with 1 indicating a presence and 0 indicating no presence.

Token addresses and their corresponding names were sourced from Kosala Hemachandra's GitHub account (section 4.9.1.2). Again, these addresses were merged onto the Ethereum Blockchain data by matching the addresses of the sender or receiver to the token addresses. If a transaction involves a token address a flag is added to the dataset where 1 indicates a presence and 0 indicates no presence.

Addresses relating to various exchanges such as Binance, Poloniex, Kraken and BitFinex were scraped manually from Etherscan. Addresses in connection to ShapeShift, a cryptocurrency exchange that also supports coin mixing, were also scraped.

In a similar fashion to the scam and token databases, the exchange dataset was also joined to the Ethereum Blockchain data. A new field was created that indicated whether or not the transferral of Ether from one address to the other involved an exchange address. If the sender's or receiver's address was marked as an exchange address, then a flag was added to the dataset with 1 indicating a presence and 0 indicating no presence.

The hourly price change of Ether in USD was sourced from Etherchain.org. This data set contains a timestamp and the corresponding USD price. In order to append this data to the Ethereum Blockchain we used the timestamp from both datasets. The USD price was added to the Ethereum data by rounding to the nearest hour. That is, if the timestamp on the Ethereum data contained 2.15pm and we have both the price change at 2pm and 3pm, then the price merged onto this data point is the price corresponding to 2pm. This information can be used as a proxy in estimating the amount of Ether in USD that is exchanged between addresses.

8.2 High Level Preliminary Analysis

Overall the purpose of this high-level preliminary analysis was to state the facts of the behaviour in the transactional data and use these discoveries as a reference to possibly identify and anticipate similar events occurring in the future. Using Rstudio cloud allowed us to run multiple analyses at once by deploying different scripts across various servers. That is, we did not have to wait for one server to finish running the code of a specific script to set another script running. This allowed us to save on computational time.

Firstly, taking the hourly price change from etherchain.org we plotted the price change in respect to the time.

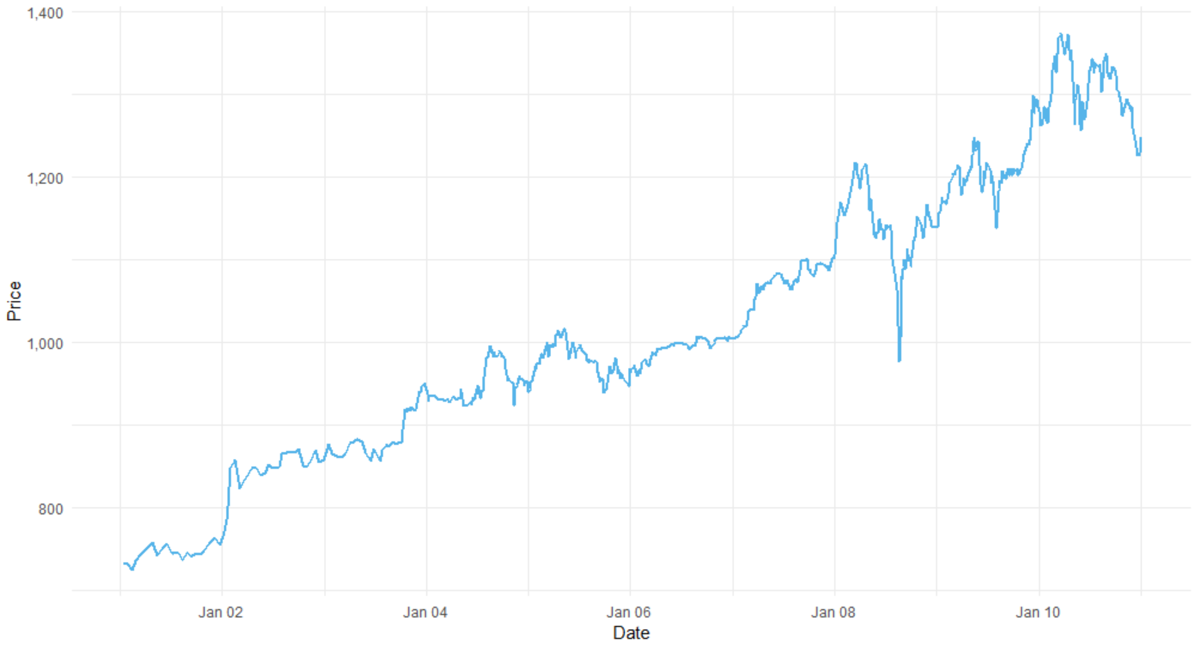

Figure 11: Price Fluctuations over 10 day period

During this time frame the price peaked on the 10th at 5.18am when ether was equivalent to 1,373.51 USD. The price of Ether in respect of USD was at its lowest on the 1st at 2.24am costing 725.42 USD. Overall, we can see that, despite the occasional wax and wane, there is an upward trend in the price movement.

The average amount of Ether transferred between accounts over this time period was approximately 12ETH. The five largest transactions to take place over this time period are as follows:

Table 2: Top 5 Largest Transactions

Transaction number	Timestamp	Sender	Receiver	Ether Value	Price of 1 Eth	USD Value	Name of Exchange
1	07/01/2018 06:50	0xfe9e8709d3215310075d67e3ed32a380ccf451c8	0x3f5ce5fbfe3e9af3971dd833d26ba9b5c936f0be	100,000	1067.65	106,765,000	BinanceWallet
2	07/01/2018 11:29	0xfe9e8709d3215310075d67e3ed32a380ccf451c8	0x564286362092d8e7936f0549571a803b203aaced	100,000	1083.02	108,302,000	BinanceWallet
3	04/01/2018 17:33	0xa44bddeccc14655838568cd1fcc05208e92f4e62	0x344d43a715bfa1b7c396e8fafcaef3fda88815a2	76,000	986.48	74,972,480	N/A
4	04/01/2018 17:55	0x344d43a715bfa1b7c396e8fafcaef3fda88815a2	0x6fc82a5fe25a5cdb58bc74600a40a69c065263f8	76,000	982.63	74,679,873	N/A
5	08/01/2018 14:30	0x3f5ce5fbfe3e9af3971dd833d26ba9b5c936f0be	0x564286362092d8e7936f0549571a803b203aaced	75,221	1063.47	79,995,277	BinanceWallet

Transactions number 1 and 2 were sent from the same address to two different Binance wallets. Both of these transactions occurred on the 7th of January equating to approximately $200million. Transactions 3 and 4 involved a middleman where the same amount was sent from A to B then from B to C with both of these transactions occurring within a thirty-minute interval. The last transaction involved the exchange of Ether between two different Binance wallets.

There were also a large number of transactions involving 0ETH. On further inspection it was discovered that the addresses involved in such a transaction were transferring tokens or interacting with contracts. In the case of contracts, users may interact with the contract by exercising voting rights or exchanging information, however, they do not transfer Ether, instead they just pay the gas fee.

8.2.1 Scams

What are the main types of scams that are occurring over this period?

In total, over the ten-day period, we identified 89 scam transactions. That is, either the sender or the receiver of the transaction was a scam address. These scams all fell under the category phishing. Fake wallets had been circulating over this time period with users depositing small amounts of Ether. However, the most common phishing site was a fake OmiseGo site that asked for private keys which affected 74 addresses. OmiseGo has branded itself as the next generation finance network and decentralised economy (OmiseGo, 2018). Should hackers be able to access users' private keys they would then be able to drain their accounts. The interactions involving the fake OmiseGo site took place between 5.50pm on the 9th and 11.30pm on the 10th of January.

8.2.2 Tokens

How many transactions involve tokens and which tokens are the most popular?

During this time period there were 1,434,293 (approximately 10.6%) transactions that involved token addresses. By calculating the frequency of each token, we discovered that TRX[4] was the most popular token with 12.20% of the token transactions involving this particular token. EOS[5] was also quite popular with a 6.90% frequency.

In the diagram below we display the frequency of each token, however, we have grouped the smaller frequencies together. That is, we have grouped frequencies ranging from close to 0% to less than 1.90% in the category 'Other'.

[4] TRX (Tronix) is the token for Tron. Tron is a decentralised entertainment content sharing platform that aims to place the ownership of content back in the creator's hand. For example, instead of a song being sold on iTunes and a fee being paid, the singer can use Tron to directly sell their song and receive all proceeds of the sale.

[5] EOS is a network and platform for applications that is built on top of the Ethereum platform. Its aim is to develop highly scalable applications that are able to interact with the Ethereum blockchain.

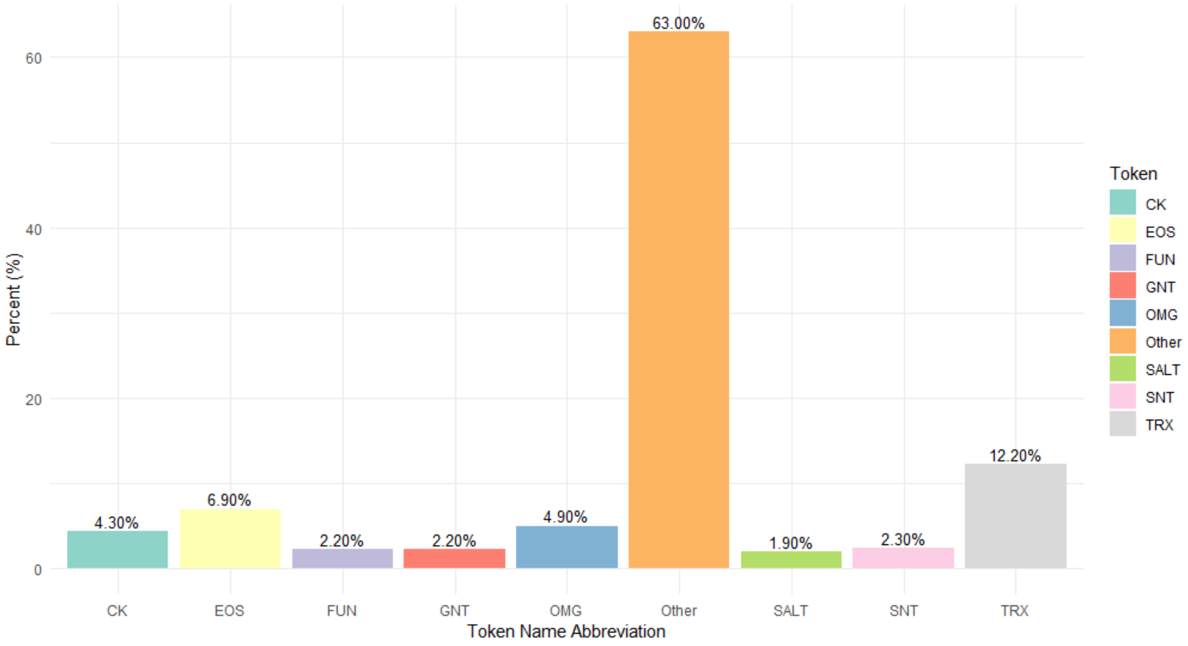

Figure 12: Token Frequency

8.2.3 Exchanges

Which exchanges are the most popular and how much ether in terms of USD is entering and exiting these exchanges?

We note that there were 1,736,279 (approximately 12.8%) transactions that interacted with addresses from one of the following exchanges:

- ShapeShift
- coinexchange.io
- BinanceWallet
- BitFinex
- Kraken
- Poloniex
- Bittrex
- YoBit

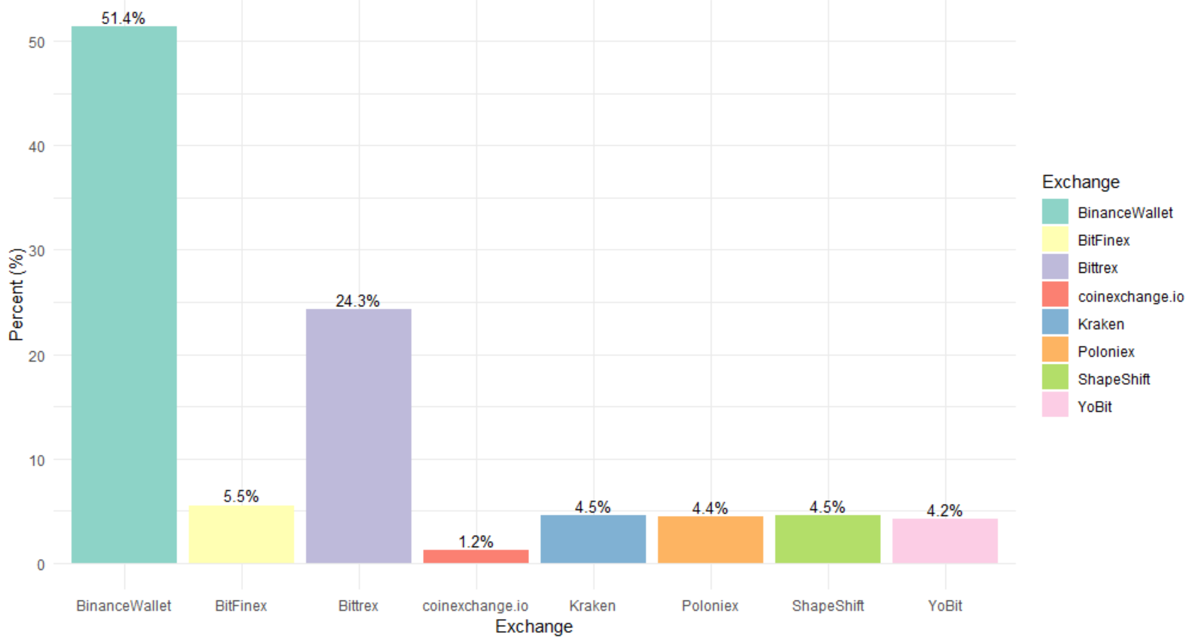

Figure 13: Exchange Frequency

The table below shows the inflow[6] and outflow[7] of Ether from the various exchanges. The difference is calculated as follows:

$$Difference = \frac{(Outflow - Inflow)}{Outflow}$$

Table 3: Exchange Outflow vs Inflow

Name	Outflow	Inflow	Difference (%)
BinanceWallet	3,061,318,183	3,555,940,163	-16%
BitFinex	1,679,257,987	1,690,602,069	-1%
Bittrex	1,208,410,390	782,311,191	35%
coinexchange.io	6,366,079	8,376,556	-32%
Kraken	823,199,437	46	100%
Poloniex	654,302,013	305,729,573	53%
Poloniex ColdWallet	193,821,500		100%
ShapeShift	113,172,050	110,448,519	2%
YoBit	33,486,244	36,509,370	-9%
Total	**7,773,333,883**	**6,489,917,487**	**17%**

[6] Inflow of Ether is defined as the amount of Ether entering the exchange; where users have sent Ether from their address to the address of an exchange.

[7] Outflow of Ether is defined as the amount of Ether exiting the exchange; where there has been a movement of Ether from an exchange address to a user address.

From the table we can see that Bitfinex, ShapeShift and YoBit appear to have quite an equal balance between Ether entering and leaving the exchanges as we see that the differences are small. Looking at coinexchange.io we see that there is roughly a 32% decrease in the amount of Ether exiting the exchange whereas in the case of Bittrex we see that there is roughly a 35% increase in Ether exiting the exchange.

The exact reason for such movements is unknown however we will keep in mind that the data selected is over a ten-day period and that these addresses were scraped manually from Etherscan and by no means produce an exhaustive list. In the case of the Poloniex cold wallet we see that all the funds from this cold wallet are released. That is, there is a 100% outflow of Ether from the exchange which can also be seen on the Kraken exchange.

Trying to find a granular reason as to why the movements of certain exchanges differ from others may result in a lot of speculation on our part, however, we will step back and look at the total movement. Calculating this movement, we see that there is an overall increase in the amount of Ether exiting the exchanges. We also know that during this time period, the value of Ether began to grow. As these exchanges deal not only specifically with Ether one possible reason for the movement is that cryptocurrency holders may have wished to exchange their current cryptocurrency for Ether or that they bought Ether from these exchanges using fiat currency. They may have done this under the pretence that the value of Ether would continue to rise and thus take advantage of the buy low sell high strategy.

8.2.4 Transactions vs Price

Is there a correlation between the average number of transactions taking place in a day and the daily average price of Ether in USD?

In calculating the correlation coefficient between the average daily price and the corresponding average daily transactions we are provided with a measure of similarity between the variables (Kaufman and Rousseeuw, 1990). In general, the correlation coefficient can range from -1 to 1 and examines the linear relationship between two variables with -1 indicating a negative linear relationship, 0 indicating no linear relationship and 1 indicating a strong positive linear relationship.

Looking at the average number of transactions taking place in a day and the average price of Ether expressed in terms of USD we see that there is a positive correlation present. Using the correlation function in R we calculated a correlation coefficient of approximately 0.5.

Table 4: Correlation Matrix

	Price	NumberTrans
Price	1	0.5053968
NumberTrans	0.5053968	1

This coefficient of 0.5 indicates that there is a moderately positive correlation which implies that as the average number of transactions increases or decreases so too does the daily average price. That is, they move in the same direction.

8.2.5 Miners

Are there any miners that appear to dominate the mining community?

During the first ten days in January we see that there are 3,629 miners who are active in generating blocks. In order to evaluate whether or not there are certain miners dominating the mining community we first calculated the frequency of each miner. That is, we calculated how many blocks each miner created in respect of their fellow miners' hashing successes. Taking the top five high frequencies we decided to segregate them from the lower frequencies. The top five frequencies ranged from 26.94% to 9.19% and any frequency under 9.19% was grouped in a category labelled 'Other'. The miners in the Ethereum Blockchain are denoted by their hex id, however, using Etherscan we were able to reveal these miners' identity.

From this data manipulation we were able to visualise the results below:

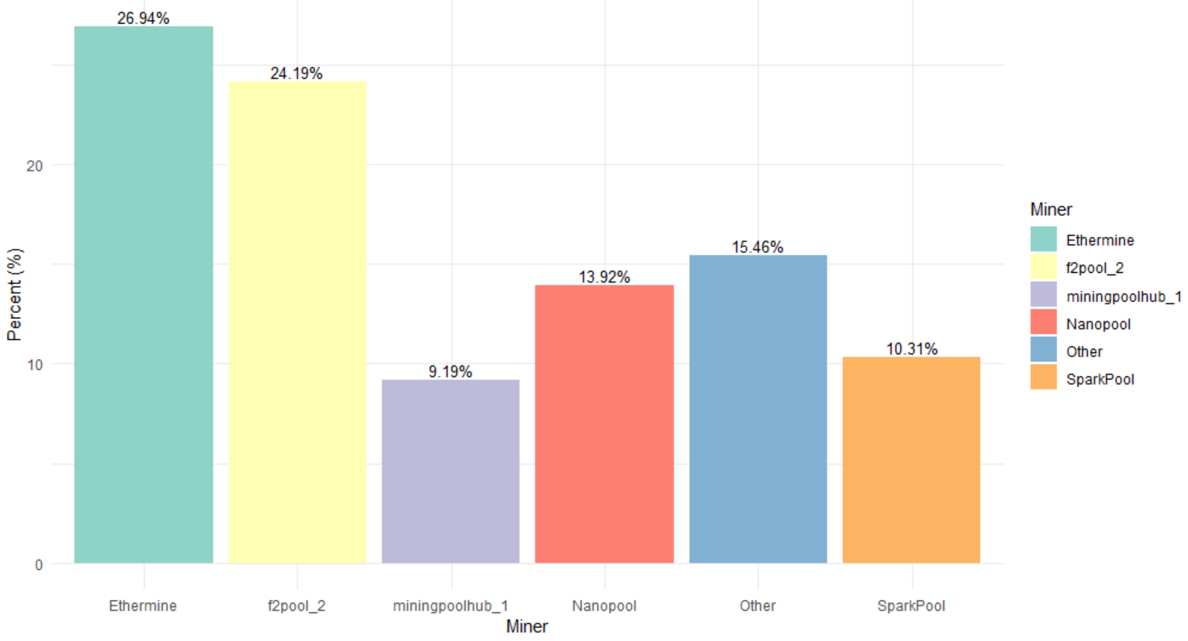

Figure 14: Miner Frequency

We can see that 'Ethermine' is the top miner over this period with an activity of nearly 27%. This means that over this period they mined 27% of the blocks. The miner 'f2pool_2' hashed just over 24% of the blocks. It can be seen that in aggregate almost 16% of the mining is generated by miners whose frequency ranges from close to 0% to just under 9%.

If we look at the top 25 miners by block number from Etherscan from the last seven days (24/07/2018 – 31/07/2018) we can see that despite the different time frame we notice a similar pattern. In both instances, we see that Ethermine is the top miner. However, we see that in comparison to the time frame in January SparkPool has increased their activity for the given time frame in July.

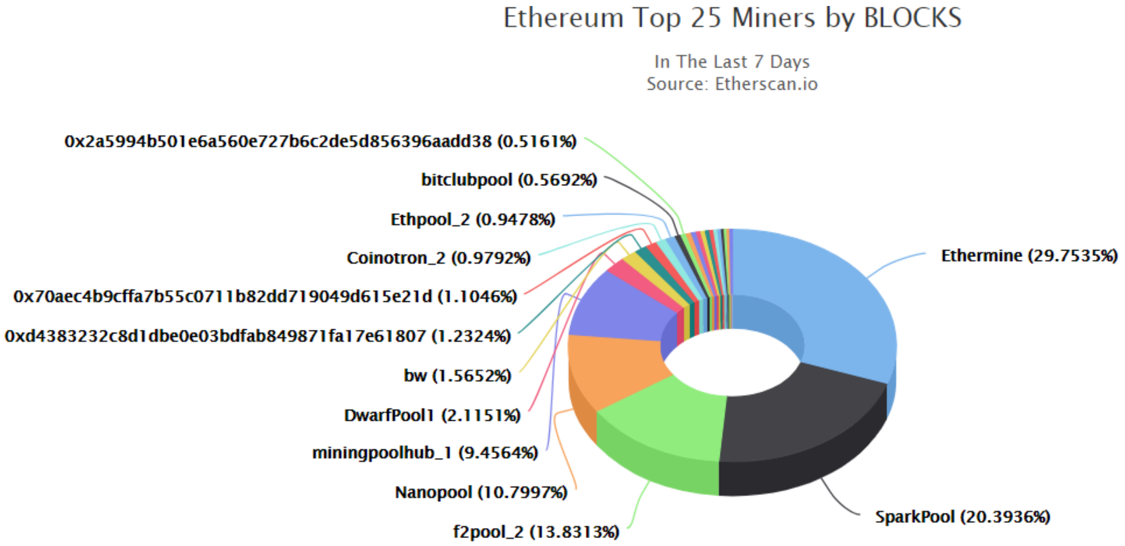

Figure 15: Top 25 Ethereum Miners (Source: Etherscan)

From our analysis we can see that no one miner appears to be dominating more than 50% of the mining successes. This proves to us that decentralisation is an active ingredient of the blockchain structure and concept.

8.3 Machine Learning

The algorithm selected to detect patterns in the Ethereum transactional data was K-means clustering. First, we determined which features would be selected for the algorithm and standardised the data. Second, we ran a series of experiments where we varied the parameter K. This parameter is used to predetermine the number of clusters to be formed by the algorithm. Across each experiment we tested the effects of the principal component analysis which was used to reduce the dimensionality of the data. As a means of comparison, we evaluated the between sum of squares divided by the total sum of squares ratio of the K-means clustering algorithm before and after the implementation of the principal component analysis.

This analysis was carried out using R studio cloud, where each experiment was carried out in parallel. That is, different experiments were launched across different servers.

8.3.1 Feature Selection

Using the Caret package in R (Caret, 2018) we first ran the data through the random forest algorithm. Upon examining the results of the random forest algorithm, we were able to calculate the mean decrease in Gini for each feature in the dataset. We the used the ‘varImpPlot’ function to plot the mean decrease in Gini values for each feature. Figure 16 below depicts the variable importance:

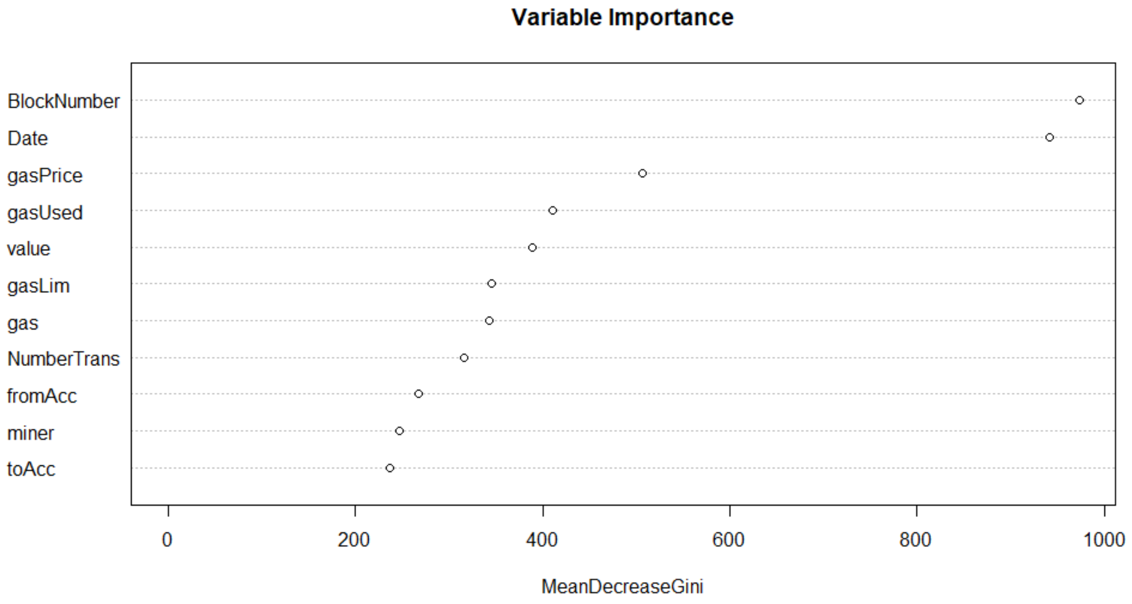

Figure 16: Variable Importance

From the above plot we decided to select the nine features that have a mean decrease in Gini less than 600. It is clear that both ‘Date’ and ‘BlockNumber’ are not as important when it comes to assessing the optimal features for the K-means algorithm.

8.3.2 Experiments

Before running each experiment, the features were selected and the data was standardised. We tested the effects of standardising the data before running it through the algorithm by taking a small subset. We noted that the algorithm returned more favourable results and formed more accurate clusters on the standardised data. All three experiments were conducted with and without the implementation of the principal component analysis and the results were noted. For each experiment, a different method was used in selecting the parameter K which denotes the number of clusters to be applied by the K-means clustering algorithm. The methods used were the silhouette method, the elbow method and the Davies-Bouldin Index.

8.3.3 Experiment 1

In this experiment we used the silhouette method to select the parameter K. Again, we standardised the data and implemented the K-means clustering algorithm using the value of K produced by the silhouette method. We also examined the effects of using the principal component analysis in conjunction with K-means.

8.3.3.1 Selecting K

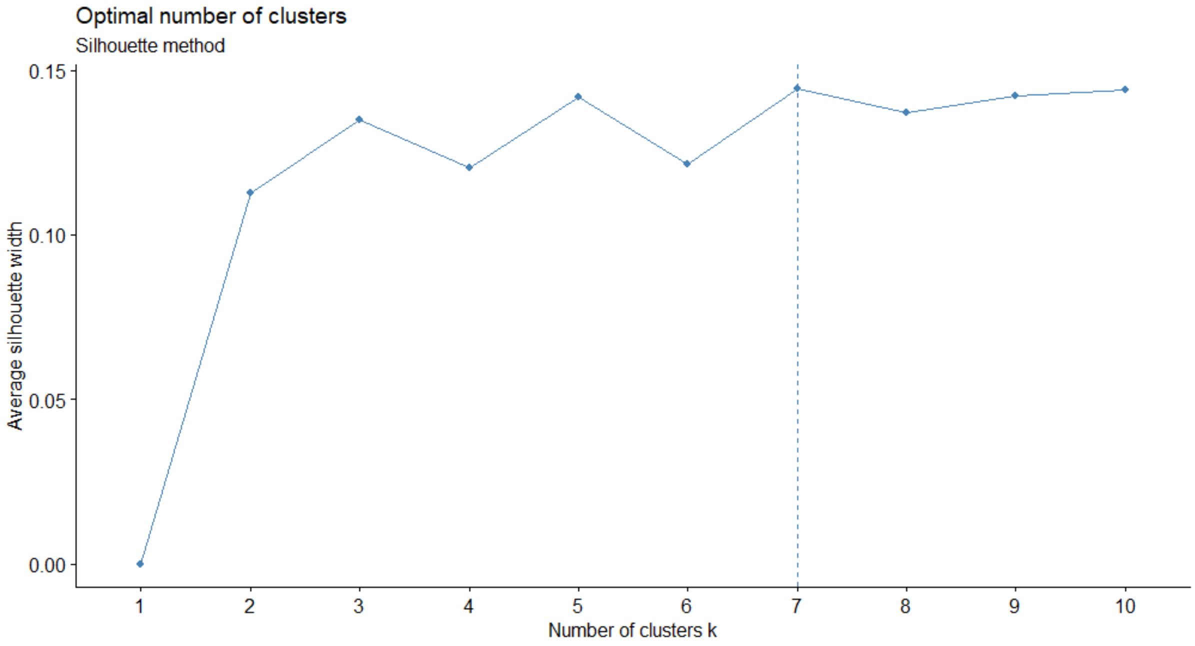

Figure 17: Silhouette Method to select K

The figure above (Figure 17), plots the number of clusters against the average silhouette width. As previously discussed in section 6.2.4.2, the silhouette value measures how well a point fits in its cluster. Here, we see that by selecting K to be 7 returns an average silhouette width of roughly 0.15. Average silhouette values close to 1 are considered to be optimal values. High values highlight the goodness of fit of a point to their cluster and how these points would fit poorly with neighbouring clusters. While 0.15 is certainly not a terrible result, it would have more favourable to return an average closer to 1.

8.3.3.2 Result

Table 5: Silhouette Method

Method	Silhouette
Number of Clusters (K)	7
K-means	41.20%
K-means and PCA	70.50%

From our analysis, selecting 7 to be the number of predefined clusters, we see that implementing the K-means algorithm returned a between sum of squares divided by the total within sum of squares ratio of 41.2%. We also see that using K-means together with the principal component analysis increased this ratio by about 27% to 70.5%. This means that the patterns form in that data using K-means and principal component analysis are more distinguishable than those formed by K-means only.

8.3.4 Experiment 2

In this experiment we used the elbow method to select the parameter K. Upon selecting the parameter K, we then standardised the data and implemented the K-means clustering algorithm. We also examined the effects of using the principal component analysis in conjunction with K-means.

8.3.4.1 Selecting K

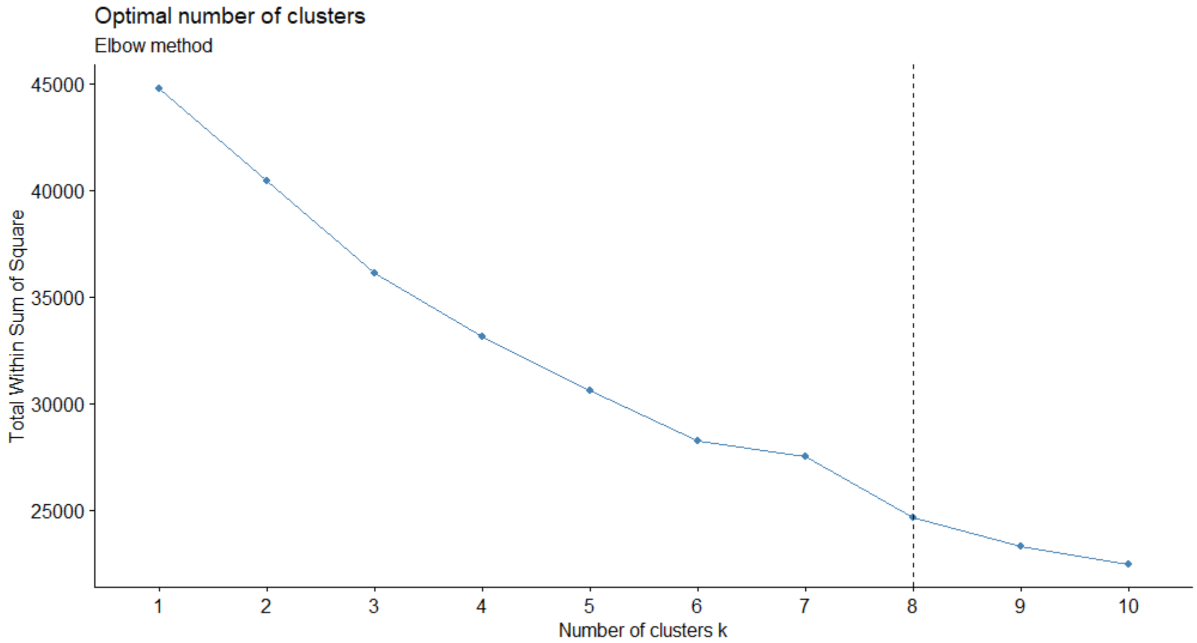

Figure 18: Elbow Method to select K

We have selected 8 to be the value for the parameter K. We note that the increasing the number of clusters to 9 does not improve the total within sum of squares by a drastic amount. Therefore, according to the principals of the elbow method (section 6.2.4.1), 8 should be selected as the value for K.

8.3.4.2 Result

The Elbow method shows that 8 should be the value chosen for parameter K. That is, 8 is the number of clusters we should select when running the K-means clustering algorithm. The table below (Table 6), shows the effects of implementing the K-means clustering algorithm on its own in comparison with using K-means together with the principal component analysis as a dimension reduction technique.

For K-means we see that the between sum of squares divided by the total within sum of squares ratio is 44.40%. However, if we implement the principal component analysis with the K-means clustering we see that there is an improvement in this ratio of almost 30%. Again, this means that the using K-means and principal component analysis together forms clearer patterns.

Table 6: Elbow Method

Method	Elbow
Number of Clusters (K)	8
K-means	44.40%
K-means and PCA	73.50%

8.3.5 Experiment 3

In this experiment we used the Davis-Bouldin index to select the parameter K. As with the other two experiments we compare the results of the K-means clustering algorithm both with and without using the principal component analysis as a means of dimension reduction.

8.3.5.1 Selecting K

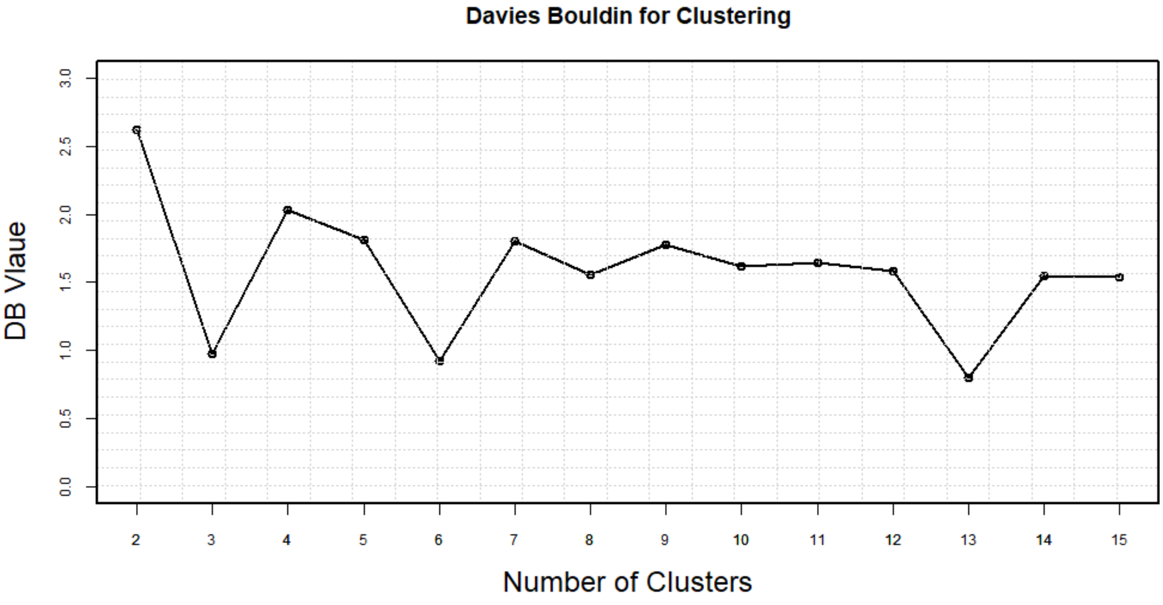

Figure 19: Davies-Bouldin to select K

On selecting the optimal value of K according to Davies-Bouldin we must look for the number of clusters that return the lowest index. From Figure 19, we see that the graph dips in three places; when the number of clusters is valued at 3, 6 and 13. The Davies-Bouldin indexes returned respectively are approximately 0.97, 0.92 and 0.8. Therefore, selecting the lowest index we set the number of clusters to be 13.

8.3.5.2 Result

Table 7: Davies-Bouldin Index

Method	Davies-Bouldin
Number of Clusters (K)	13
K-means	53.80%
K-means and PCA	81.20%

From the results we can see that using 13 as a value for K returns 53.8% as the ratio of between sum of squares divided by the total within sum of squares for K-means only. Using a combination of the principal component analysis and K-means algorithm returned a much higher ratio of 81.2%. This means that the patterns formed using both K-means and principal component analysis are clearer than using K-means only.

8.3.6 Experiment Observation

The table below compares the results of the three experiments:

Table 8: Comparison of Results

Method	Silhouette	Elbow	Davies-Bouldin
Number of Clusters (K)	7	8	13
K-means	41.20%	44.40%	53.80%
K-means and PCA	70.50%	73.50%	81.20%

Comparing the results of the analysis across all three experiments we see that by selecting 13 as the value for parameter K and implementing the principal component analysis in conjunction with the K-means algorithm returns the most favourable ratio. That is, we calculate a between sum of squares divided by total within sum of squares of 81.2%.

Across all three methods we can see that increasing the number of clusters improves the performance of the ratio. However, increasing the number of clusters is not proportional to the performance. That is, increasing the number of clusters from 7 to 13 results in an improvement of approximately 11% (81.2%-70.5%) and not an improvement of over 50% (7/13=53%).

The ability to create such clear-cut patterns is very important as it reduces the ambiguity surrounding the presence of outliers. This means that any outliers detected are very prominent. These anomalies will not fit any of the clearly defined patterns and therefore the ambiguity in their belonging to any specific cluster is dramatically reduced.

8.3.7 Detecting Possible Anomalies

Using the results of experiment 3 (section 8.3.5), we tested to see if we could detect outliers in the data. We identified the centres of the clusters and using the Euclidean distance, measured the distance of each point in a given cluster to its centre. We identified five points that lay the furthest from their centre. These data points came from the following Blocks:

Table 9: Block Level Data of Outliers

Date	BlockNumber	NumberTrans	gasUsed	gasLim	Miner
07/01/2018 22:13	4871219	239	0x79fce6	0x7a1200	0x829bd824b016326a401d083b33d092293333a830
07/01/2018 23:40	4871559	229	0x79f363	0x79f39e	0x00a86233f4d65f4018ca7f8626b96ce72af05a7c
04/01/2018 06:21	4851308	304	0x79ceba	0x7a11f8	0xea674fdde714fd979de3edf0f56aa9716b898ec8
02/01/2018 23:28	4844241	235	0x78b57d	0x7a02cb	0xf3b9d2c81f2b24b0fa0acaaa865b7d9ced5fc2fb
04/01/2018 21:27	4854802	198	0x7a0928	0x7a2142	0x071cc3d8ac987ae7beb9e422d50a2d6bbffa95fc

Where the transactions level data was as follows, respectively:

Table 10: Transactional Level Data of Outliers

fromAcc	toAcc	gas	gasPrice	Value
0xcad801117ca8dc9df4cc0c11e2751569cd4e62b5	0x8d12a197cb00d4747a1fe03395095ce2a5cc6819	0x3d090	0xe8d4a51000	1
0xec0c2163684d254abb47a29e66f0b9e7273ef6a6	0x5cbde4e07edfb73f38bec03ffea7641628b5f61f	0x5208	0x4a817c800	0.04967254
0x007174732705604bbbf77038332dc52fd5a5000c	0xe276e1b52db64ff190c31117197c4ff5a1665864	0xe57e0	0x4e3b29200	4.73477096
0x06edd22791726394cbe18ccf4e59dc3607a8c4a6	0xfd150fd971eaa74459c0d42ea447194930cca73e	0x5208	0x4a817c800	78.21297024
0x5e575279bf9f4acf0a130c186861454247394c06	0xf230b790e05390fc8295f4d3f60332c93bed42e2	0x249f0	0x824f87100	0

The fact that these transaction records lie outside the cluster indicates that they are possible anomalies in the transactional data. The reason as to why such anomalies occur is something that could be investigated further.

8.3.8 Limitations of K-means Clustering

Outliers is a common limitation cited when implementing K-means clustering. In many cases outliers are eliminated from the data before the algorithm is run (Singh, Malik, & Sharma, 2011). However, in some cases as with our case, we do not wish to eliminate these outliers. Instead we wish to highlight them as possible anomalies which is similar to the work of (Hirshman, Huang, & Macke, 2013).

Predicting the number of clusters is also a drawback in implementing the algorithm. Of course, as we demonstrated, there are numerous techniques that can be applied to the data in order to select the optimal K parameter. However, it can be seen that as you increase the parameter K, the resulting ratio of the between sum of squares divided by the total within sum of squares tends to improve. The key is to find the optimal number of clusters such that increasing the cluster by one does not greatly improve the ratio. This is why we used three different techniques in selecting K.

We also noticed that the algorithm was very sensitive to scaling. Standardising the data to have a mean 0 and variance of 1 before running the algorithm greatly improved the result (the ratio of between sum of squares divided by the total within sum of squares). This step could be overlooked but in doing so would have been detrimental to our resulting clusters. As mentioned in section 8.3.2, we tested the effects of standardising the data on a small subset and witnessed that more accurate clusters were formed once the data was standardised.

9 Conclusion

In this dissertation we can conclude that it is possible to detect patterns in the Ethereum transactional data using K-means clustering. We discovered that, despite its limitations, K-means clustering is more computationally feasible and leads to more meaningful results than hierarchical clustering or K-medoids clustering. Using principal component analysis as a prelude to implementing the K-means algorithm also dramatically improved the accuracy of the cluster formation allowing us to detect more accurately the outliers in the data. We would also suggest standardising the data before running the algorithm as doing so leads to more accurate clusters being formed.

One major drawback in using the K-means algorithm is in preselecting the parameter K to determine the number of clusters needed. It is a cumbersome process with many caveats. Selecting the optimal K has been a long-debated trial when it comes to implementing the K-means clustering algorithm and many techniques exist in order to help guide the user in the selection process. Selecting the parameter K is the most important aspect of implementing the K-means algorithm and we have explored three different techniques as a means of selecting K.

Going forward we would recommend that one makes use of cloud computing techniques when it comes to analysing and processing the transactional data. We also note the importance of the file format when it comes to the scalability of the data. Selecting the correct file format is key when it comes to storing the data. Without a doubt using the .rds file format produced files that were much more manageable than the common CSV file. In conjunction with the .rds file format, we would encourage users to carry out the analysis using the cloud-based version of Rstudio, Rstudio cloud. Using such a service allowed us to carry out multiple experiments at the same time. We did not have to wait until one server returned the results of an experiment before we set the next running. By launching three different servers we were able to run the three different experiments at the same time.

We also demonstrated the various insights we can draw from the Ethereum transactional data by carrying out our high-level preliminary analysis. We would always recommend carrying out such an analysis as in doing so may lead to a more robust formation of the research question.

Before future work can be considered it is key to note that patterns can be detected in the Ethereum transactional data and using K-means to do so is appropriate. It is possible to build upon this work and possibly change the research question to investigate supervised learning as opposed to unsupervised learning. However, this research we believe was a necessary prelude that needed to be undertaken before alternative supervised learning could take place.

9.1 Future Work

It would be interesting to look at the clusters in more depth and see if they represent known data structures. This would allow us to identify future transactions and possibly categorise them. That is, we could check to see if all the scam addresses form one cluster, all the token addresses form another and all the exchange addresses another.

As the Blockchain is an unregulated technology it is susceptible to fraud and illegal activity. Thus, we believe that it is important to try to identify such potential activities within the Blockchain. We know that Etherscam.db is flagging potential scam addresses, however, we believe that an interesting area in which to direct future study would be in isolating these scams in the Ethereum transactional data. It would then be interesting to apply the K-means clustering algorithm to this 'scams only' segment and identify features of the clusters formed. This could potentially allow future scams to be detected and intercepted before they get the chance to con innocent users. It may be worthwhile carrying out a supervised learning implementation of K-means on the 'scams only' segment. This would involve labelling the data before implementing the algorithm. One possible label would be classifying the scams based on the category field. That is, labelling the scams as, for example, phishing or fake ICO. Clustering the segment based on the type of scam would allow one to investigate whether or not scams of a certain category have similar features.

Similarly, it may also be interesting to look at the 'exchange only' segment of the data where we use the flag field to remove all the transactions involving exchange addresses. With this subset it may be worthwhile labelling the data with the label representing the name of the exchange involved. Labelling the data would allow the researcher(s) to carry out supervised learning and investigate whether or not transactions involving a given exchange form a specific cluster or not. If they do it is important that the researcher(s) evaluates the cluster performance and determines how well formed the clusters are which would further allow them to investigate whether or not transactions involving a given exchange have distinguishable features. Should this be possible it may be interesting for the exchanges to note this and perhaps from a marketing point of view they would be able to identify ways in which their exchange could possibly be more favourable over another competitor.

The future work suggested would involve building on the current research carried out and forming alternative research questions depending on which avenue the researcher wishes to travel down.

10 References

Agar, J. (2017). *Turing and the Universal Machine: The Making of the Modern Computer.* Icon Books.

Agrawal, R., Gehrke, J., Gunopulos, D., & Raghavan, P. (1998). Automatic Subspace Clustering of High Dimensional Data for Data Mining Applications. *ACM Vol. 27. No. 2.*, 94-105.

Ahamad, S., Nair, M., & Varghese, B. (2013, May). A Survey on Crypto Currencies. In *4th International Conference on Advances in Computer Science, AETACS* (pp. 42-48). Citseer.

Andreesen, M. (2014, January 21). *Why Bitcoin Matters.* Retrieved from New York TImes: https://dealbook.nytimes.com/2014/01/21/why-bitcoin-matters/

Application of FinCEN's Regulations to Persons Administering,. (2013, March 18). Retrieved from The Financial Crimes Enforcement Network: https://www.fincen.gov/sites/default/files/shared/FIN-2013-G001.pdf

Armbrust, M., Fox, A., Griffith, R., Joseph, A. D., & Lee, G. (2010). A view of cloud computing . *Communications of the ACM 53(4)*, 50-58.

Best Bitcoin Cloud Mining Contract Reviews and Comparisons. (2017). Retrieved from Bitcoin Mining : https://www.bitcoinmining.com/best-bitcoin-cloud-mining-contract-reviews/

Beyer, K. S., Goldstein, J., Ramakrishnan, R., & Shaf, U. (1999). When is "nearest neighbor" meaningful? *Proceedings of the 7th International Conference on Database Theory*, 217-235.

Bitcoin Cash. (2017). Retrieved from https://www.bitcoincash.org/

Bitcoin Classic. (2016). Retrieved from https://bitcoinclassic.com/

Bitcoin Glossary. (2009-2018). Retrieved from Bitcoin.org: https://bitcoin.org/en/glossary/

Bitcoin Gold. (2017). Retrieved from https://bitcoingold.org/

Bitcoin Talk. (2010, May). Retrieved from https://bitcointalk.org/

Bitcoin Unlimited. (2016). Retrieved from https://www.bitcoinunlimited.info/

Bitcoin XT. (2014). Retrieved from https://bitcoinxt.software/

Block Explorer. (2018). Retrieved from https://blockexplorer.com/

Blockchain Info. (2017). Retrieved from https://www.blockchain.com/explorer

Bray, T. (2017). *The javascript object notation (json) data interchange format (No. RFC 8259).*

Brieman, L., Friedman, J. H., Olsen, R. A., & Stone, C. J. (1984). *Classsification and Regression Trees.* Boca Raton: Chapman and Hall.

Buterin, V. (2013). *Ethereum White Paper*. Retrieved from Ethereum Wikipedia: https://github.com/ethereum/wiki/wiki/White-Paper

Buterin, V. (2014, May 6). *DAOs, DACs, DAs and More: An Incomplete Terminology Guide.* Retrieved from Ethereum Blog: https://blog.ethereum.org/2014/05/06/daos-dacs-das-and-more-an-incomplete-terminology-guide/

Buterin, V. (2017, March 14). *Hard Forks, Soft Forks, Defaults and Coercion.* Retrieved from Vitalik Buterin's Website: https://vitalik.ca/general/2017/03/14/forks_and_markets.html

Cachin, C. (2016). Architecture of the Hyperledger Blockchain Fabric. In *Workshop on Distributed Cryptocurrencies and Consensus Ledgers (Vol 310).* Zurich.

Caret. (2018, May 27). Retrieved from https://cran.r-project.org/web/packages/caret/caret.pdf

Coin Mixer. (2018). Retrieved from https://www.coinmixer.io/

CoinMarketCap. (2018). Retrieved from https://coinmarketcap.com/currencies/ethereum/

Collier, A. (2018, January 7). *An Ethereum Package for R.* Retrieved from Exegetic: http://www.exegetic.biz/blog/2018/01/ethereum-r-package/

Collier, A. (2018). *Ether.* Retrieved from Cran: https://cran.r-project.org/web/packages/ether/ether.pdf

ConsenSys. (2017). *Alethio*. Retrieved from https://aleth.io/

ConsenSys. (2018, April 3). *Using Machine Learning to Understand the Ethereum Blockchain.* Retrieved from ConsenSys: https://media.consensys.net/using-machine-learning-to-understand-the-ethereum-blockchain-1778485d603a

Cormen, T. H., Leiserson, C. E., Rivest, R. L., & Stein, C. (2009). *Introduction to algorithms.* Massachusetts: MIT press.

Dannen, C. (2017). *Introducing Ethereum and Solidity.* Berkeley: Apress.

Davies, D. L., & Bouldin, D. (1979). A Cluster Separation Measure. Pattern Analysis and Machine Intelligence. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 224-227.

DB Browser for SQLite. (2014, September). Retrieved from http://sqlitebrowser.org/

DBHub.io. (2016). Retrieved from https://dbhub.io/

Dean, J., & Ghemawat, S. (2008). MapReduce: simplified data processing on large clusters. *Communications of the ACM 51(1)*, 107-113.

Devcon3. (2017). Retrieved from https://ethereumfoundation.org/devcon3/

Ding, C., & He, X. (2004). K-means Clustering via Principal Component Analysis. *Proceedings of the twenty-first international conference on Machine learning*, 29.

Distinctive Features Of SQLite. (2000). Retrieved from SQLite: https://www.sqlite.org/different.html

Dropbox. (n.d.). Retrieved from https://www.dropbox.com/

Ethereum Mining Rewards. (2017). Retrieved from Ethereum Wikipedia: https://github.com/ethereum/wiki/wiki/Mining#mining-rewards

Ethereum Price in USD. (2015-2018). Retrieved from Etherchain: https://www.etherchain.org/charts/priceUSD

Etherscamdb. (2017). Retrieved from https://etherscamdb.info/

Etherscan. (2018). Retrieved from https://etherscan.io/aboutus

Gan, G., Ma, C., & Wu, J. (2007). *Data Clustering: Theory, Algorithms, and Applications (ASA-SIAM Series on Statistics and Applied Probability).* Siam.

Geth. (2017, December 21). Retrieved from Ethereum Wikipedia: https://github.com/ethereum/go-ethereum/wiki/Geth

ggplot2. (2018, July 3). Retrieved from https://cran.r-project.org/web/packages/ggplot2/ggplot2.pdf

Gillespie, C., & Lovelace, R. (2016). *Efficient R Programming.* O'Reilly Media Inc.

Google Drive. (n.d.). Retrieved from https://www.google.com/drive/

Hartigan, J. A., & Wong, M. A. (1979). A K-Means Clustering Algorithm. *Royal Statistical Society. Series C (Applied Statistics)*, 100-108.

Hashflare. (2014-2018). Retrieved from https://hashflare.io/

Hemachandra, K. (2018). *MyEtherWallet.* Retrieved from GitHub: https://github.com/kvhnuke

Hirshman, J., Huang, Y., & Macke, S. (2013). *Unsupervised Approaches to Detecting Anomalous Behavior in the Bitcoin.* Stanford University.

Hongyuan, Z., He, X., Ding, C., Gu, M., & Simon, H. D. (2002). Spectral relaxation for k-means clustering. *Advances in neural information processing systems*, 1057-1064.

Huang, Z. (1998). Extensions to the k-means algorithm for clustering large data sets with categorical values. *Data mining and knowledge discovery 2(3)*, 283-304.

Jain, A. K. (2009). Data clustering: 50 years beyond K-means. *Pattern Recognition Letters 31(8)*, 651-666.

Johnston, D. (2015). *Decentralized Applications White Paper and Spec.* Retrieved from GitHub: https://github.com/DavidJohnstonCEO/DecentralizedApplications

Jolliffe, I. T. (2002). Principal Component Analysis. *International encyclopedia of statistical science*, 1094-1096.

Judmayer, A., Stifter, N., Krombholz, K., & Weippl, E. (2017). *Blocks and Chains: Introduction to Bitcoin, Cryptocurrencies, and Their Consensus Mechanisms.*

Kaufman, L., & Peter J. Rousseevw. (1987). Clustering by means of medoids. *Statistical Data Analysis based on the L1 Norm*, 405-416.

Kaufman, L., & Rousseeuw, P. (1990). *Finding Groups in Data: An Introduction to Cluster Analysis.* John Wiley and Sons.

K-Means Clustering. (2017). Retrieved from BrandIdea: http://brandidea.com/kmeans.html

Kosba, A., Miller, A., Shi, E., Wen, Z., & Papamanthou, C. (2016). Hawk: The blockchain model of cryptography and privacy-preserving smart contracts. *IEEE symposium on security and privacy (SP)*, 839-858.

Kuznetsov, N. (2017, June 24). *How Emerging Markets And Blockchain Can Bring An End To Poverty.* Retrieved from Forbes: https://www.forbes.com/sites/nikolaikuznetsov/2017/07/24/how-emerging-markets-and-blockchain-can-bring-an-end-to-poverty/

Lee, S., & Pham, T. T. (2017). *Anomaly Detection in Bitcoin Network Using Unsupervised Learning Methods.* Stanford: arXiv preprint arXiv:1611.03941.

Lemieux, P. (2013). Who Is Satoshi Nakamoto? *Regulation 36(3)*, 14. Retrieved from Steemit.com.

Liu, H. (2013). Big Data Drives Cloud Adoption in Enterprise no. 4. *IEEE internet computing*, 68-71.

Manning, L. (2018, July 5). *How Peter Kroll's Paper Wallet Protects Cryptocurrency.* Retrieved from Bitcoin Magazine: https://bitcoinmagazine.com/articles/how-peter-krolls-paper-wallet-protects-cryptocurrency/

Meiklejohn, S., Pomarole, M., Jordan, G., McCoy, D., Levchenko, K., Voelker, G. M., & Savage, S. (2013). A Fistful of Bitcoins: Characterizing Payments Among. *ACM*, 127-140.

Milligan, G. W., & Cooper, M. C. (1988). A study of standardization of variables in cluster analysis. *Journal Of Classification*, 181-204.

MIT. (2017, September 15). *The UN Says the Global Digital Divide Could Become a Yawning Chasm.* Retrieved from MIT Technology Review: https://www.technologyreview.com/the-download/608887/the-un-says-the-global-digital-divide-could-become-a-yawning-chasm/

Mitchell, T. M. (1997). *Machine Learning WCB.* Boston, MA: McGraw-Hill .

My Ethereum Wallet. (2018). Retrieved from https://www.myetherwallet.com/

Nakamoto, S. (2008). *Bitcoin: A Peer-to-Peer Electronic Cash System.* Retrieved from Bitcoin.org: https://bitcoin.org/bitcoin.pdf

Narayanan, A., Bonneau, J., Felten, E., Miller, A., & Goldfeder, S. (2016). *Bitcoin and Cryptocurrency Technologies : A Comprehensive Introduction .* Princeton University Press.

Nimaosu. (2013, June). *Ethereum Glossary.* Retrieved from Ethereum Wikipedia: https://github.com/ethereum/wiki/wiki/Glossary#ethereum-blockchain

Norry, A. (2018, July 2). *The History of the Mt Gox Hack: Bitcoin's Biggest Heist.* Retrieved from Blockonomi.com: https://blockonomi.com/mt-gox-hack/

OmiseGo. (2018). Retrieved from https://omisego.network/

Ooms, J., Temple Lang, D., & Hilaiel, L. (2017, June 1). *Jsonlite.* Retrieved from Cran: https://cran.r-project.org/web/packages/jsonlite/jsonlite.pdf

Perper, R. (2018, February 6). *China is moving to eliminate all cryptocurrency trading with a ban on foreign exchanges.* Retrieved from uk.businessinsider.com: http://uk.businessinsider.com/china-eliminates-all-cryptocurrency-trading-2018-2

PWC. (2018, March 22). *How Smart Contracts Automate Digita Business*. Retrieved from http://usblogs.pwc.com/emerging-technology/how-smart-contracts-automate-digital-business/

Python Anywhere. (n.d.). Retrieved from https://www.pythonanywhere.com/

RStudio Cloud. (2018). Retrieved from https://rstudio.cloud/

Safavian, S. R., & Landgrebe, D. (1991). A survey of decision tree classifier methodology. *IEEE transactions on systems, man, and cybernetics 21.3*, 600-674.

Segwit. (2017). Retrieved from https://segwit.org/

Singh, K., Malik, D., & Sharma, N. (2011). Evolving limitations in K-means algorithm in data mining and their Removal. *International Journal of Computational Engineering & Management*.

SQLite. (2000-2018). Retrieved from https://www.sqlite.org/index.html

'SQLite' Interface for R. (2018, May 6). Retrieved from RSQLite: https://cran.r-project.org/web/packages/RSQLite/RSQLite.pdf

Stallings, W. (2017). *A Blockchain Tutorial.* Internet Protocol Journal.

Swan, M. (2015). *Blockchain: Blueprint for a New Economy.* O'Reilly Inc.

Tan, P.-N., Steinbach, M., & Kumar, V. (2013). *Introduction to Data Mining.* Boston: Pearson.

Tapscott, D., Tapscott, A., & Kirkland, R. (2016, May). *How blockchains could change the world.* Retrieved from McKinsey&Company: https://www.mckinsey.com/industries/high-tech/our-insights/how-blockchains-could-change-the-world

Taylor, M. B. (2013, November 28). Bitcoin and the age of bespoke silicon. *Proceedings of the 2013 International Conference on Compilers, Architectures and Synthesis for Embedded Systems, IEEE press*, 16. Retrieved from Blockchain Informer.

Top 100 Cryptocurrencies By Market Capitalization. (2018). Retrieved from CoinMarketCap: https://coinmarketcap.com

Vavilapalli, V. K. (2013, June 12). Apache hadoop yarn: Yet another resource negotiator. *Proceedings of the 4th annual Symposium on Cloud Computing*, 5.

Velde, F. R. (2013). *Bitcoin: A primer.* Chicago: The Federal Bank Chicago.

Warneke, D., & Kao, O. (2009). Nephele: Efficient Parallel Data Processing in the Cloud. *Proceedings of the 2nd workshop on many-task computing on grids and supercomputers*, 8.

Wood, G. (2014). Ethereum: A secure decentralised generalised transaction ledger. *Ethereum project yellow paper 151* , 1-32.

World Bank. (2018, April 12). Retrieved from https://www.worldbank.org/en/topic/financialsector/brief/blockchain-dlt

World Crypto Index. (2018). Retrieved from https://www.worldcryptoindex.com/what-is-a-coin-mixer/