MSc in Corporate Finance

Department of Business and Management

Chair of Asset Pricing

Asset Allocation and Portfolio Optimisation:

Focus on Cryptocurrencies

SUPERVISOR

Prof. Paolo Porchia

CO-SUPERVISOR

Prof. Marco Pirra

CANDIDATE

Luca Fiorino

ID: 722591

ACADEMIC YEAR

2020/2021

0. Abstract

The first chapter introduces the crypto market, its trends, and the instrument utilised during the whole analysis. Then, a first, short analysis is conducted on cryptocurrencies.

The second chapter will actualise the research conducted by Holoviatuk, O. "Cryptocurrencies as an asset class in portfolio optimisation", based on 2014-2019 data. Since the latter investigation, the market changed with the sentiment on cryptocurrencies. The period under analysis range from June 2018 to June 2021, and it takes into consideration the market highly stressed by COVID-19. Since both individual and institutional investors are considering much more to invest - directly and indirectly - in the crypto world, we study Bitcoin, Ethereum, and more in general cryptos. Should they be considered an alternative asset class as of today? The answer will be provided at the end of the chapter.

The third chapter develops the Modern and Post-Modern Portfolio Theories by combining Crypto, Stocks, Bonds, Commodities, FX and Real Estate. A total of 20'000 combinations are run to find the best tangency portfolio.

Immediately after the above analysis, an obvious question springs to mind: How to invest in Crypto-assets? Moreover, what should be the time horizon? Chapter 4 presents the GRU: an algorithm that can predict the Bitcoin and Ethereum prices with extreme precision for the following twenty days. However, it could be applied to a wide range of cryptocurrencies and could be helpful to consciously invest in an asset class still not yet studied in deep, which could surprise even the more sceptical investors.

1. Introduction

Since their introduction in 2008, cryptocurrencies have piqued the interest of researchers from various fields. Not only are these cryptocurrencies backed by a set of carefully reasoned and integrated scientific computer theories, but their digital nature perfectly aligns with the Internet era, allowing them to be used in situations where traditional fiat currencies have failed. Tax-free transactions and a reasonable level of data protection guaranteed by untracked payments are just a few of the significant advantages.

This section will discuss the trend of cryptocurrencies and the technology and characteristics of two of the most important ones: Bitcoin and Ethereum.
After, we will present the instruments used during our research: CRIX – the most well-known proxy for cryptocurrencies – and five proxies that represent the five par excellence traditional asset classes – Stocks, Bonds, Commodities, FX, and Real Estate –. Finally, the chapter ends with a concise reasoning about cryptocurrencies: are they really a currency, a financial bubble, or none of them?

1.1 The technology behind cryptocurrencies

As for every cutting-edge technology, blockchain, which underpins cryptocurrencies, encounters both enthusiasm and resistance. While some believe that blockchain heralds the dawn of a new digital era, others contend it is a rising financial bubble or just a scheme for money laundering. On the other side, it should also be considered that ten years of expanding adoption of blockchain technology, its integration into public spheres, and its use in everyday transactions demonstrate its practical application.
Cryptocurrencies emerged as the first wave of blockchain-based applications. Satoshi Nakamoto pioneered this technology's first implementation in his 2008 article "Bitcoin: A Peer-to-Peer Electronic Cash System," in which he stated: "What is required is an electronic payment system based on cryptographic proof rather than trust, allowing any two willing parties to transact directly between each other without the need for a trusted

third party"[1]. In other terms, blockchain is a decentralized, tamper-resistant transaction and data management system in which records are held across a network of multiple nodes. Another way of thinking about blockchain is as a distributed ledger that spans a network of numerous holders, places, or devices.

As shortly mentioned above, blockchain comprises a series of ordered back-linked blocks that include information about transactions. Each block's transactions are combined and hashed into a binary tree, or Merkle tree, with the top (root) of the tree being preserved in each record[2]. Since they are part of a chain, blocks retain the hashes of all preceding blocks and replay them from the chain's origin. When the original data is modified, the hash is also modified and does not match the original fingerprint anymore. It leads to the necessity of rehashing all of the following blocks. This protects the system's integrity by making it nearly impossible to rewrite all the hashes and modify the data contained within the chain.

What distinguishes blockchain technology is a set of three components that enables the creation, updating, verification, and auditing of data across the system without the interference of third parties.

The first component is the peer-to-peer (P2P) network[3], a collection of identically privileged nodes connected via a shared infrastructure. The blockchain database is then dispersed among various nodes, granting access to data to all network members. As a result, there is no need to rely on an intermediary, as blockchain technology can confirm and maintain a permanent record-keeping procedure that ensures personal data privacy.
The second component that provides secure, non-modifiable communication is cryptography. The blockchain protects against retrospective alterations to records using a cryptographic hashing technique that acts as a fingerprint for authenticating the record. Once an initiator signs a transaction, it is validated and disseminated via the network of nodes until it is included in all nodes' blocks.
The third component is the consensus algorithm, which ensures the database's consistency when a new transaction requires approval.

[1] Bitcoin: A Peer-to-Peer Electronic Cash System, S. Nakamoto, 2008
[2] Bitcoin: A Peer-to-Peer Electronic Cash System, S. Nakamoto, 2008
[3] Peer-to-Peer Networks: Architectures, Applications and Challenges, J. Sen, 2013

As Xingxiong Zhu states, “there are many common consensus algorithms, such as proof-of-work (PoW), proof-of-stake (PoS), practical byzantine fault-tolerance (PBFT), delegated proof-of-stake (DPOS), Ripple, and Tendermint. They differ in computational complexity, fault-tolerance, scalability, performance, and effectiveness”[4]. Proof-of-Work is the most often used consensus algorithm at the heart of Bitcoin and Ethereum. It requires miners to solve a mathematical problem, typically a hash function, to obtain consensus, which needs a lot of computer power and thus much energy.[5]

1.2 Bitcoin

1.2.1 Timeline

Bitcoin was suggested in 2008 by Satoshi Nakamoto in the paper "Bitcoin: A Peer-to-Peer Electronic Cash System." Nakamoto views a trusted third party as an unreliable middleman and advocates for an alternative system in which any party may easily confirm transactions, obviating the requirement for a trust-based intermediary.

Bitcoin's source code was released in early 2009, but initially received little attention. The first Bitcoin transaction occurred a few days after the launch, when Nakamoto sent 10 BTC to Hal Finney, one of the project's early supporters and collaborators, for testing purposes. Following that, Bitcoin continued to operate as a distinct currency system until May 2010, when the global economy collapsed.

In the following years, more institutions began to accept Bitcoin, and its price began to rise, but not without any problem. In August 2010, a significant weakness in Bitcoin was exploited, allowing transactions to be improperly confirmed, essentially removing the 21 million BTC upper limit on Bitcoin issuance[6]. The vulnerability was immediately identified and patched, and even though this was the sole significant assault as of today, cryptographic attacks continued. Indeed, most of the remaining prominent Bitcoin exchanges have all been hacked.

Bitcoin split into different structures in August 2017 due to a protracted argument over how to handle scalability. Without getting into technical details, the total number of

[4] Research on blockchain consensus mechanism and implementation, X. Zhu 2019
[5] Bitcoin and Cryptocurrency Technologies, A. Narayanan, J. Bonneau, E. Felten, 2016
[6] Bitcoin History: The Complete History of Bitcoin [Timeline]

bitcoins will converge to 12 million BTC in 2140, and regardless of whether the Bitcoin community agrees to remove the upper bound in the future, this event will immediately influence Bitcoin's price and transactions.

1.2.2 Technical Review

Bitcoin has no formal sense of ownership; all transactions are anonymous. The owner controls the cryptographic private key necessary to sign the subsequent hashed transactions.

It utilises a distributed global public ledger in which each block represents a transaction. Since every transaction is dependent on the status of the prior transaction, the blocks create a chain referred to as the blockchain. At each point in time, the longest blockchain is viewed as the consensus blockchain. Each transaction creates an integer value denoting the transaction's amount and a script snippet indicating the transaction's status. The subsequent transaction's input must match the prior transaction's output. In addition, the previous and current transactions' scripts must terminate correctly, and the total amount transacted must not exceed the total input amount. The entire transaction is then hashed using SHA-2562, with the hash value serving as the transaction's globally unique identifier[7].

Now that a set of stringent transaction rules have been established, it must be ensured that they operate in a network environment. What is needed is a protocol that addresses one of the most perplexing aspects of building a digital currency: the double spending dilemma. In the context of Bitcoin, the double-spending issue arises when the same bitcoins are spent several times concurrently. Bitcoin defends itself against the attack by utilising the global public ledger, which admits transactions only if published to the global public ledger. To be successfully posted to the global public ledger, the transaction has to be accepted and verified by the network of miners.

Arguably the most creative aspect of Bitcoin, the consensus mechanism, also known as the Nakamoto consensus, utilises computer power to validate each transaction. As additional users trade with one another, the hashing difficulty constantly increases.

[7] Mining Process in Cryptocurrency Using Blockchain Technology: Bitcoin as a Case Study , A. A. Aljabr, A. Sharma, K. Kumar, 2019

Furthermore, since the cryptographic quiz does not entirely remove the potential of contradictory concurrent publications, each transaction block awaits the addition of six further "confirmation" blocks before announcing its publication to the global public ledger. With the growing difficulty of hashing, this practice results in a longer verification time for each transaction[8].

1.3 Ethereum

1.3.1 Timeline

In 2014, Buterin and other Ethereum co-founders raised almost $18 million through a crowdsourcing effort in which they sold Ether (Ethereum tokens) to participants. In 2015 Frontier, Ethereum's first live release, was announced. Since then, the platform has evolved dramatically, to the point of having hundreds of developers.

Ethereum shares many of the same difficulties as Bitcoin at the beginning, most notably in terms of scalability. In 2016, an unnamed hacker stole $50 million in ETH, raising concerns about the platform's security. That resulted in the Ethereum community splintering into two blockchains: Ethereum and Ethereum Classic (ETHC).

Even though the price of Ethereum has fluctuated dramatically, the cryptocurrency increased by more than 13,000 per cent in 2017. While this phenomenal growth is appealing to many investors, the volatility causes others to be cautious.

Ethereum's structure has been improved in the last years in reaction to protection issues, and since it is not as monopolistic as Bitcoin, it results in more open to reforms that could eventually make it a superior solution to Bitcoin.

Ethereum supporters believe that its key benefit over Bitcoin is that it allows individuals and businesses to do much more than merely move money between entities, and Bloomberg called it "the hottest platform in the world of cryptocurrencies and blockchains".

To solve some of the classical problems of Ethereum, such as high gas fees, unlikely scalability, and network’s congestion, Ethereum 2.0 was launched in 2020. The

[8] Mining Process in Cryptocurrency Using Blockchain Technology: Bitcoin as a Case Study , A. A. Aljabr, A. Sharma, K. Kumar, 2019

implementation process should terminate in late 2022.

1.3.2 Technical Review – Ethereum 2.0

Ethereum early blockchain implementations struggled with performance because they relied on a processing power-intensive mechanism called Proof of Work to validate and record transactions. The issue with proof of work is that it is inefficient by design. To begin resolving that issue, Ethereum 2.0 will migrate to a more efficient Proof-of-Stake system. In such a system, an algorithm selects the node that records each transaction, with the likelihood of selection rising with the quantity of currency controlled by the node's owner. That enables a significant reduction in the complexity of cryptographic work, resulting in substantial throughput benefits for the entire network. Since each node must stake its currency to participate, attacking the network would remain prohibitively expensive.

Additionally, to boost Ethereum's efficiency and scalability, the upcoming updates will add a processing approach known as Sharding[9]. Currently, all participating nodes must verify all data added to the chain. This indicates that the slowest participant restricts the overall system's processing, raising transaction costs and lowering throughput.
By incorporating Sharding, Ethereum 2.0 can significantly improve resource utilisation efficiency. That will be accomplished by delegating data verification tasks to distinct nodes, each responsible for checking only the data it receives. This enables parallel processing over the entire blockchain, potentially increasing overall capacity severalfold. With this new technique combined with the move to Proof-of-Stake, the new Ethereum blockchain should be far swifter and more efficient than its precursor.

One of the characteristics contributing to Ethereum's success as a platform and as a legitimate challenger to Bitcoin's dominance is its implementation of the Ethereum Virtual Machine (EVM). The EVM is a multi-node execution environment that enables smart contracts. These smart contracts differentiate Ethereum's blockchain from a purely financial system. On the EVM, smart contracts may be used to run games and carry out complicated financial transactions.

[9] Ethereum 2.0 Includes Major Changes That Could End Bitcoin's Blockchain Dominance, A. Kovačević, 2019

The introduction of eWASM[10] will enable the execution of Ethereum application code directly in modern web browsers, a significant advance above the EVM. Additionally, it will allow programmers to build code for the blockchain using a variety of languages, including Rust, C, and C++. eWASM will significantly boost the number of prospective programmers for the ecosystem in one fell swoop by removing the requirement for users to learn a native Ethereum-only language.

1.4 CRIX

Professor Wolfgang Härdle and his team of academics at Humboldt University in Berlin developed the CRIX Index for crypto markets, bought in 2021 by Royalton Partners.

Although it originated as an academic endeavour and is not traded, CRIX effectively represents the market and is used as a benchmark by academics and traders. Additionally, it is tailored to the crypto market's peculiarities, including a highly dynamic internal structure, the possibility of often disappearing and reappearing coins, and significant volatility. As explained below, the CRIX can face these characteristics by constantly applying weights reallocation and evaluation.

Mathematically, the CRIX is a market index constructed using the Laspèyres method. Laspèyres' index is defined as:

$$INDEX_t^{Laspèyres} = \frac{\sum_i P_{it} Q_{i0}}{\sum_i P_{i0} Q_{i0}} \qquad 1.1$$

With P_{it} denoting the price of crypto i at time t and Q_{i0} denoting the amount of crypto i at time 0. P_{i0} denotes the price at time 0. The CRIX is a minor variation:

[10] Breaking Down ETH 2.0 – eWASM and EVM Explained, Moralis Academy, 2020

$$CRIX_t = \frac{\sum_i P_{it} Q_{i0}}{\sum_i P_{i0} Q_{i0}} \qquad 1.2$$

Where MV_{it} constitutes the market capitalisation of cryptocurrency *i* at time *t*. The *Divisor* secures the stability of the changes. The *Divisor* for CRIX is:

$$Divisor = \frac{\sum_i MV_{i0}}{1000} \qquad 1.3$$

As a result, the CRIX's initial value is *1000*. The *Divisor* is revised whenever the total number of coins in a cryptocurrency changes. This ensures that only changes in prices impact CRIX.

$$\frac{MV_{i,t-1}}{Divisor_{t-1}} = INDEX_{T-1}^{CRIX} = INDEX_t^{CRIX} = \frac{MV_{i,t}}{Divisor_t} \qquad 1.4$$

$Divisor_{t-1}$ is the $Divisor$ prior to changing the number of coins. $Divisor_t$ is the one immediately after that.

A cryptocurrency can have a large market capitalisation but be seldom traded. Therefore, the following rule is used:

$$ADTV_i \geq ADTV_{0.25} \qquad 1.5$$

Where $ADTV_{0.25}$ is the 0.25 percentile of the Average Daily Trading Volume distribution of all cryptos over the last period, and $ADTV_i$ represents the Average Daily Trading Volume of the single crypto.[11]

[11] www.royalton-crix.com

If a cryptocurrency satisfies the rule, it is eligible to be included in the CRIX list of constituents.

For reasonably stable markets, a settled number of elements may be the right approach.

In any case, CRIX weights each cryptocurrency according to its market capitalization, and its constituents are re-evaluated quarterly and rebalanced monthly.[12]

We will use it as a proxy for cryptocurrencies in the following sections.

1.5 Asset classes and their proxies

We will often talk about stocks, bonds, and other asset classes in the following chapters.

Table 1 shows the five asset classes we will analyse and the proxies that have been chosen.

Stocks	Vanguard FTSE Developed World UCITS ETF	VDEV
Bonds	Xtrackers II Global Government Bond UCITS ETF	XSGI
Commodities	Bloomberg Commodity CMCI Composite SF UCITS ETF	CCUSAS
FX	Euro Currency Index	EUR_I
Real Estate	iShares Developed Markets Property Yield UCITS ETF	DPYA

Table 1: Asset Classes and their proxies

These ETFs track the developed countries' economies through different perspectives. The chosen currency is the USD.

The *Vanguard FTSE Developed World UCITS ETF* is an exchange-traded fund launched by Vanguard Group (Ireland) Limited. The fund is co-managed by Vanguard Asset

[12] CRIX an Index for cryptocurrencies, S. Trimborn, 2018

Management, Limited and The Vanguard Group, Inc. It invests in developed countries' public equity markets across the globe. The fund aims at investing in large-cap and mid-cap companies operating across diversified sectors. In addition, the fund seeks to replicate the performance of the FTSE Developed Index by employing a representative sampling methodology.

The *Xtrackers II Global Government Bond UCITS ETF* is an exchange-traded fund launched by Deutsche Asset Management S.A. It is co-managed by Deutsche Asset Management Investment GmbH and Deutsche Asset Management (U.K.) Limited. The fund invests in developed countries' fixed income markets across the globe. It invests in local currency-denominated, fixed-rate sovereign bonds with a maturity of at least one year. The fund invests in the investment-grade securities rated by S&P, Moody's, and Fitch. It replicates the performance of the Citi World Government Bond Index - Developed Markets by employing a representative sampling methodology.

The *Bloomberg Commodity CMCI Composite SF UCITS ETF* is an exchange-traded fund launched by LSAM Investments. It is co-managed by UBS Asset Management (U.K.) Ltd and UBS Global Asset Management, London. The fund utilises derivatives such as futures and swaps to invest in commodity markets - energy, precious metals, industrial metals, agriculture, and livestock commodities. In addition, it seeks to replicate the performance of the UBS Bloomberg Constant Maturity Commodity Index Excess Return by employing synthetic replication methodology.

The *Euro Currency Index* represents the arithmetic ratio of four major currencies against the Euro: US Dollar, British Pound, Japanese Yen and Swiss Franc.

The *iShares Developed Markets Property Yield UCITS ETF* is an exchange-traded fund launched by BlackRock Asset Management Ireland Limited. It is managed by BlackRock Advisors (U.K.) Limited. The fund invests in the developed countries' public equity markets globally except for Greece. It invests in real estate sector companies, including real estate investment trusts (REITs) and real estate holding & development companies. The fund invests in dividend-paying companies across all market capitalisations. It replicates the performance of the FTSE EPRA/NAREIT Developed Dividend + Index by employing a representative sampling methodology.

1.6 Crypto Trends

The year 2021 was a great one for the cryptocurrency industry. Indeed, at the end of 2021, the Cryptocurrencies Market Capitalisation has reached 2.368,53 billion US Dollars.[13]

In 2021, Ethereum outperformed its only larger peer, Bitcoin. The EIP-1559 upgrade, which lowered ETH's inflation rate, was one of the causes that contributed to such a dramatic outperformance.

In 2021, Ethereum increased by 425 per cent, while Bitcoin increased by 66%. The S&P 500, the benchmark index of the United States, rose 31%, while gold fell slightly.

However, the foundations of the largest crypto coin have begun to strengthen. According to Glassnode, just 18.34% of the entire circulating supply is now at a loss.[14]

Despite the current price movements, long-term investors' holdings climbed by 1.84 million Bitcoins in 2021, while short-term holders' supply has decreased by 1.42 million Bitcoins.

In 2021, the exchange lost a total of 67,800 Bitcoins. Based on the preceding two facts, the number of holders increases while the supply decreases.

The Bitcoin hash rate has been drastically reduced due to China's ban. However, it has not only recuperated but has increased by 27% during the same year. The latter demonstrates that it endured the prohibition and outperformed its previous record.

Stable coins are gaining traction in the cryptocurrency industry. Being connected to physical assets, such as commodities and government-issued currencies, they can reduce cryptocurrency fluctuation. As a result, stable coin market capitalisation has climbed from USD 5 billion in 2020 to USD 120 billion at the end of 2021. Aside from acting as a bridge between fiat currencies and crypto-assets, they also serve as a reasonably safe "parking area" for crypto volatility, being used as collateral in crypto-asset derivative transactions or decentralised financing ("DeFi")[15].

[13] www.coinmarketcap.com
[14] The Week Onchain (Week 52, 2021), Insights Glassnode
[15] Crypto year at glance: Ethereum ouperformed in 2021, Metaverse in new DeFi, The Economic Times, 2022

1.6.1 COVID-19 Impact Assessment

The COVID-19 pandemic harmed the cryptocurrency market since its level of stability has significantly decreased, and cryptocurrencies have become more volatile. Furthermore, bitcoin has a low level of regularity compared to international equities markets, which reduced the demand for cryptocurrency during the pandemic. On the other hand, the demand for traditional asset classes also decreased due to the crisis.

1.6.2 Untapped Potential in Emerging Markets

Developing economies provide enormous opportunities for cryptocurrency businesses to expand their operations by facilitating access to capital and financial services. Bitcoin has already allowed many companies and individuals to expand and thrive as a source of revenue. The economy is gradually moving to meet these demands, and cryptocurrencies have huge potential.

Evolving demographics, increased consumerism, and openness to new technologies such as IoT, Blockchain, and others create an attractive potential for cryptocurrency in emerging countries. For instance, according to Oxford Business Group, Nigeria is the leading country in cryptocurrency adoption, owing to its use for sending remittances.[16]

In addition, the Philippines' central bank allowed 16 bitcoin exchanges. As a result, the country is quickly becoming one of the world's most significant adopters of cryptocurrencies. Furthermore, as smartphone adoption increases in Latin America and Africa, mobile payment service providers will offer increasingly complex services on mobile phones. This may represent a significant opportunity for market expansion.

1.7 Classification of Cryptocurrencies

[16] Can cryptocurrencies drive a Covid-19 recovery in emerging markets?, Oxford Business Group, 2021

Since cryptocurrencies are uncommon for financial markets, researchers and investors have yet to classify them fully. While some experts identify them as a financial bubble, others believe they are a new type of currency; others still consider them a distinct asset class.

As Central Banks define it, a traditional currency must officially fulfil three tasks to be designated currency: unit of account, store of value, and medium of exchange. Generally, only large-cap cryptocurrencies can meet all conditions mentioned above, while the rest struggle to meet even one.

The unit of account is a currency's primary function; it enables measuring value in a specified unit and allows comparisons with other currencies. Digital currencies are made up of precise, individual, and quantifiable account units. This function is satisfied by high-cap cryptocurrencies and stablecoins since their value is determined and comparable. Conversely, cryptos with low daily volume do not meet this criterion. [17]

The term "store of value" refers to the ability of an item to retain purchasing power in the future so that it can be more, less, or equally valuable and exchanged later. It requires a degree of certainty regarding the asset's future value, which may be challenging with crypto assets due to their high volatility. For example, gold and digital currencies can store value, decoupled from fiat money, and act as a haven during a crisis; nevertheless, only gold retains these characteristics over time. According to some researchers, daily exchanges of Bitcoin, Ethereum, and Litecoin have exceeded even the annual inflation rates of recession-stricken countries such as Mexico and South Africa, implying that it is safer to hold the Mexican Peso than top crypto cryptocurrencies if we value them under the volatility perspective[18]. Due to the high degree of volatility, the crypto assets' suitability as a safe store of value is debatable as long as the market will not stabilise. [19]

For an instrument to serve as a medium of exchange, it must be universally accepted and exchangeable for all accessible products and services. In addition, it must function as an intermediary and avoid the constraints inherent in barter transactions. According to a recent Hartford Steam Boiler poll[20], at least one-third, or roughly 36%, of small to medium-sized enterprises in the United States currently accept cryptocurrencies as payment for goods and services.

[17] Can cryptocurrencies fulfil the functions of money?, S. Ammous, 2018
[18] Crypto-Assets Unencrypted, S. Kim, A. Sarin, D. Virdi, 2018
[19] [18] Can cryptocurrencies fulfil the functions of money?, S. Ammous, 2018
[20] Survey released by Hartford Steam Boiler and Insurance Co. (HSB)

In March 2021, Paypal Holdings, Inc. acquired Curv, a cryptocurrency security firm established in Israel. With this acquisition, PayPal Holdings enhances and expands its support to cryptocurrencies and digital assets. Curv is a cryptocurrency security firm established in Israel. In addition, Paypal customers in the United States can utilise the digital currency at retailers who accept PayPal payments. Accepted cryptocurrencies include Bitcoin, Ethereum, and Litecoin.

Other major market players accept cryptocurrencies as payment in direct and indirect ways, such as Microsoft Corporation, AT&T, Twitch, Amazon.com, Starbucks, Etc.

However, most cryptocurrencies partially fit this requirement, as they are not readily usable for every recurring payment. For example, ETH, USDT, and USDC enable access to other crypto assets and mediate between fiat money and cryptocurrency. Consequently, cryptocurrencies should be fully considered a medium of exchange just for crypto assets.[21]

Following the Central Banks criteria, cryptos can not be considered a currency today.

2. Cryptocurrencies as an asset class

Given the new and emerging status of the cryptocurrency market, it is critical for investors and regulators to understand the properties of cryptocurrencies and their relationships with other asset classes. Indeed, despite their rapid growth, the literature on cryptocurrency markets is still significantly less developed than that on traditional asset classes.

This chapter will explicate the literature's requirements to be considered an asset class, and we will challenge this definition by analysing cryptocurrencies.

2.1 The seven requirements of an asset class

By definition, cryptocurrencies must meet seven basic requirements to be designated a distinct asset class.[22] [23] [24] [25]

1. *Stable aggregation*

An asset class's composition should be reasonably stable. Otherwise, determining its optimal composition would require continuous monitoring and analysis, and maintaining it would require periodic rebalancing. Both initiatives may be too expensive.

Asset classes whose constituents are weighted according to their relative capitalisations are stable, as their relative capitalisations fluctuate proportionately as their prices vary. By contrast, a proposed asset class whose members are weighted according to time-varying criteria such as momentum, value, or size may lack the necessary composition stability to qualify as an asset class. It is sufficient, of course, a matter of empirical judgment. For instance, momentum is inherently less stable than value, which is less stable than size. As a result, a collection of momentum stocks is unlikely to qualify as an asset class, whereas stocks within a specified capitalisation range may qualify.

The above conditions may be verified through qualitative analysis of cryptocurrencies.

2. *Investable*

An asset class's essential component should be easily investable. If not, the investor could choose a replicating securities batch that tracks the economic variable. However, replication may present two difficulties. To begin, in addition to the uncertainty associated with the economic variable's out-of-sample behaviour, the investor is exposed to the uncertainty associated with the mapping coefficients that define the relationship between the economic variable and the replicating securities. Second, the investor faces increased rebalancing expenses because the optimal composition of replicating securities varies with time.

To test the capacity for the investment of cryptocurrencies, we need to prove easy access to channels of direct investing for this class.

3. *Internally Homogeneous*

[22] Cryptocurrencies As an Asset Class? An Empirical Assessment, D. Bianchi, 2020
[23] Cryptocurrencies as an asset class in portfolio optimisation, O. Holovatiuk, 2019
[24] Asset Allocation: From Theory to Practice and Beyond, W. Kinlaw, M. P. Kritzman, D. Turkington, 2021
[25] Portfolio Management: Theory & Practice, Schultz Collins, 2008

The components of an asset class should be comparable. If they are not, the investor implicitly constrains the efficiency of two or more separate groupings within a larger group. Conversely, if the suggested asset class is segmented into distinct groups, the investor can weigh them optimally.

For instance, domestic equities may perform significantly differently than foreign equities, and foreign equities may behave differently than equities from emerging economies. As a result, investors may be able to construct a more efficient portfolio by disentangling various equity markets and weighting them according to their relative contributions to a portfolio's expected utility, rather than relying on their weights in a broad global index. Not only may the best weights for these components change concerning to one another, but the optimal allocation to equities could change concerning the allocation that would occur if they were treated as a unified class.

The internal homogeneity of the asset class can be proved when assets are positively correlated. Thus, we expect correlation coefficients to be positive from 0 to 1.

4. *Externally Heterogeneous*

Each asset class should be sufficiently distinct from the other asset classes in a portfolio and linear combinations of the other asset classes. If asset classes are excessively similar, the investor will be forced to waste superfluous efforts analysing their projected return and risk characteristics and determining the most efficient strategy to invest in them.

Our investigation will be based on the statistical analysis of the asset classes' properties and the comparison of their profiles. We will compute the daily mean, standard deviation, median, median absolute deviation, minimum, maximum, skewness, kurtosis, and maximum rangerange. To satisfy the External Hetereogenity requirement, the statistical features of each class must be distinct from those of other asset classes.

5. *Expected Utility*

The addition of an asset class to a portfolio should increase the expected utility of the portfolio. That could happen in one of two ways. First, including the asset class may boost the portfolio's expected return. Second, its inclusion may help reduce portfolio risk, either

because it carries little risk on its own or because it has minimal correlations with the portfolio's other asset types.

The expected return and risk characteristics of an asset class should not be determined solely based on averages over various market conditions. For example, a particular asset class, such as commodities, may have a low expected return and a high-risk profile on average throughout changing market regimes but may provide extraordinary diversification against financial assets during moments of severe financial upheaval. Given an extreme aversion to significant losses, which generally occur during moments of financial turmoil, commodities can indeed increase a portfolio's expected utility, despite their average expected return and risk characteristics.

To increase the expected utility of a portfolio, an asset class must be externally heterogeneous. However, not all externally heterogeneous asset classes increase expected utility. While an asset class may be different, its expected return or risk may be insufficient to increase a portfolio's expected utility. As a result, we could avoid considering the external heterogeneity requirement because it is included in the expected utility.

Modern and Post-Modern Portfolio theories are used to check expected utility.

6. *Selection Skill*

The distinction between random and skilful selection is modest but critical. A portfolio's expected utility should not require the asset allocator's ability to discover superior investments to raise the portfolio's expected utility. Even if the asset allocator randomly selects investment managers within the same asset class or by investing passively, the expected utility should rise. Since not all investors possess selection ability, this should not preclude them from engaging in asset allocation.

Analysis of existing Exchanged Traded Funds will serve as a test for this standard.

7. *Access at a reasonable price*

Investors should be able to allocate a significant portion of their portfolios to an asset class without incurring high transaction costs or materially hurting the liquidity of their portfolios. If investing in an asset class is extremely expensive, the after-cost improvement in projected utility may be insufficient to justify inclusion. Additionally, if the inclusion of the asset class

significantly restricts the portfolio's liquidity, it may become prohibitively expensive to maintain the portfolio's optimal weights or fulfil cash demands, reducing expected returns. An analysis of bid-ask spread and liquidity is conducted to prove this characteristic of cryptocurrencies.

2.2 Results

1. *Stable Aggregation*

The stability of digital assets helps to their asset class conformance. Cryptocurrencies resilience is aided by blockchain technology. For instance, Bitcoin has been able to retain stability due to halving. The blockchain can thus control demand and supply. That ensures the asset's stability. Cryptocurrencies are expected to become more stable over time due to various causes. These include greater liquidity, institutional involvement, regulatory reforms, and worldwide participation. In addition, the failure to add additional assets to this class has contributed to its stability. Under these circumstances, asset aggregation can be considered stable. Additionally, two unique characteristics distinguish cryptocurrencies: it is a peer-to-peer network exchange and is entirely electronic. As a result, the asset class's first requirement is satisfied.

2. *Investable*

Cryptocurrencies must allow easy investment access to qualify as an asset class.

While directly purchasing cryptocurrency is the most typical way to add crypto exposure to a portfolio, there are several other ways to invest in cryptocurrency.

It is possible to buy and hold the most established digital currencies, such as Bitcoin and Ethereum, or unknown cryptocurrencies, on more than 300 exchanges. For cryptos that are new on the market, there is the possibility of participating in the Initial Coin Offering (ICO), the equivalent of an Initial Public Offering (IPO) that raises money to create a new coin or service related to cryptocurrencies. However, to participate, it is often needed to acquire a more conventional digital currency and have prior knowledge of wallets and exchanges.

It is possible to invest in businesses entirely or partially focused on cryptocurrencies, such as mining firms and hardware manufacturers, and companies promoting cryptocurrencies,

such as PayPal Holdings, Inc. and Robinhood Markets, Inc. . Investing in companies with significant crypto holdings on their balance sheet could constitute another option.

Cryptocurrency-focused funds are an excellent alternative to avoid picking specific cryptocurrency companies. There are many exchange-traded funds (ETFs), including index and futures funds. Certain crypto-focused funds invest directly in cryptocurrencies, while others invest in cryptocurrency-related enterprises or derivatives, such as futures contracts.

Becoming a cryptocurrency miner or validator is perhaps the simplest way to invest in bitcoin is to mine it or function as a validator in a cryptocurrency network. Cryptocurrency miners and validators earn cryptocurrency rewards, which they can keep or exchange for another currency.

In conclusion, cryptocurrencies offer many ways to invest in the market. The typical one is also the easiest, and it allows investors to enter the market with little to no difficulties. Of course, in some cases, an understanding of the methodology is required, but it can not be considered a barrier to entry. Thus, cryptocurrencies meet the criteria of being investable.

3. *Internally Homogeneous*

This criterion can be verified by looking at Figure 1. It shows the Pearson's correlation matrix, also referred to as the "product-moment correlation coefficient" (PMCC).

Correlation coefficients range between 1 and -1. The regression slope determines the sign of the correlation: a value of +1 indicates that all data points lie on a line where Y rises as X increases, and vice versa for a value of -1. A value of 0 indicates that the variables are not linearly dependent on one another.

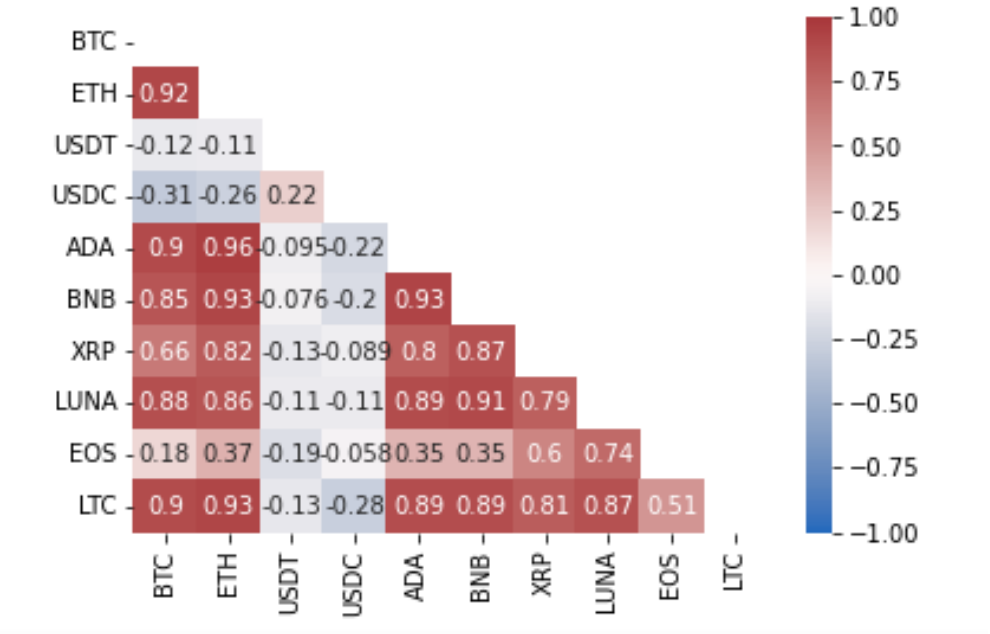

Figure 1: Pearson's correlation matrix. Own work computed in Python.

The cryptocurrencies with the highest market capitalisation are analysed.

As expected, while the coefficients vary, the majority indicates a significant positive connection among the titles within the class. Some correlation values exceed 0.9. Two exceptions are present: USDT and USDC, two of the most popular and versatile cryptocurrency coins that emerged in the last few years. Also known as stablecoins, they aim to maintain their value pegged to the USD. Consequently, they show no linear relationship with other cryptos and a slight correlation between themselves. Being stablecoins principally used as a medium of exchange to enter and exit the positions in other cryptos by deceiving transaction fees, the above study suggests that cryptocurrencies exhibit internal uniformity, which is a critical characteristic of an asset class. The third condition is therefore satisfied.

4. *Externally Heterogeneous*

Table 2 summarizes the descriptive data for all asset classes. The data spans the first of June 2018 to June 2021.

The table shows that the CRIX generates the highest values for each parameter analysed, except for Kurtosis. For example, the mean, or daily return, is 0.28 per cent, while the highest parameter for the traditional asset classes is given by equities, with a daily return of 0.051%.

Asset Class	Mean	Standard Deviation	Median	MAD	Maximum	Minimum	Range	Skewness	Kurtosis
CRIX	0,00279	0,04598	0,00225	0,02111	0,20851	-0,23857	0,44708	-0,16021	3,37573
Stocks	0,00051	0,01223	0,00115	0,00519	0,09149	-0,10084	0,19233	-0,70557	12,44665
Bonds	0,00017	0,00621	0,00036	0,00353	0,03426	-0,03753	0,07180	-0,09354	4,75255
Commodities	0,00009	0,00872	0,00063	0,00628	0,03432	-0,04181	0,07613	-0,60549	3,22508
FX	-0,00003	0,00246	-0,00009	0,00184	0,01236	-0,01041	0,01041	0,38727	2,79924
Real Estate	0,00039	0,01413	0,00183	0,00568	0,08046	-0,10630	0,18676	-0,40850	11,55284

Table 2: Descriptive data for all asset classes. Own work computed in Python.

On the other side, risk measures such as Standard deviation and Mean Absolute Deviation are significantly higher than the other asset classes. Compared to the stocks, they are respectively 3.8 and 4.1 times bigger. Again, the CRIX's range is the highest. However, Stocks and Real Estate have a very high range compared to their historical data.

Except for the FX, all the asset classes have their bell curve negatively skewed. Stocks, Commodities, and Real Estate are the asset classes more impacted by the COVID-19 crisis. This skewness shock is one of the most damaging characteristics of the recession. The sharp drop in the skewness of the distribution of cumulative returns, and the subsequent left shift in the tails of the cumulative stock returns' distribution, was slightly less significant during the Great Recession, as shown in Figure 4.

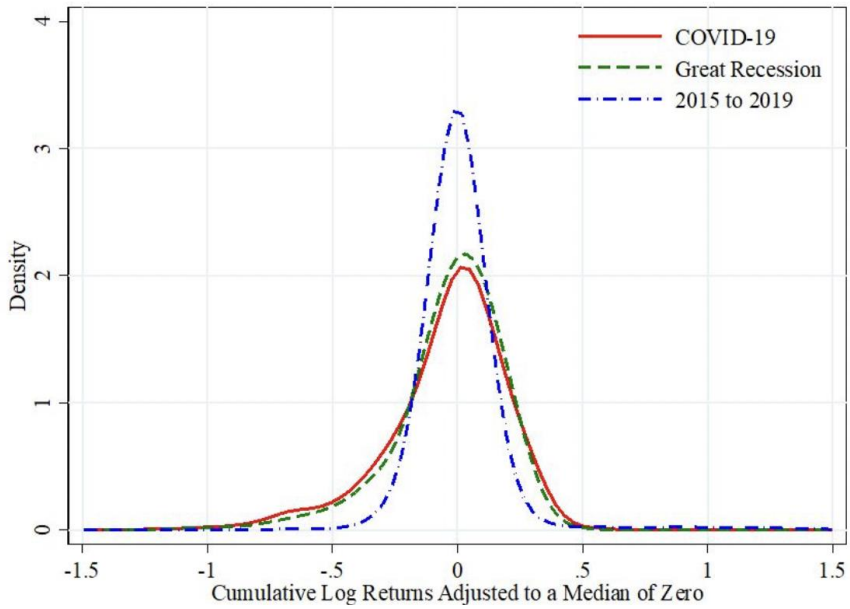

Figure 2: Cross-sectional distribution of cumulative returns, "Skewed Business Cycles", Bloom, Guvenen, Salgado

On the other hand, Cryptocurrencies demonstrated resiliency since they manifest a skewness equal to just -0.16.

The kurtosis of 3.4 shows a slight leptokurtic distribution for CRIX, with large tails and extreme values. The same reasoning made for Skewness can be applied for Kurtosis. Indeed, Stocks and Real Estate are the asset classes that carry the highest level of risk, having a Kurtosis of 12.4 and 11.5, respectively. If compared to the other asset classes, Cryptocurrencies show a high level of strength again to challenging moments.

5. *Expected Utility*

Through the Modern Portfolio Theory and the Post-Modern Portfolio Theory, we will have a clear view of the expected utility of cryptocurrencies and how they can optimise a portfolio using different strategies. It will be shown in the next chapter.[26]

6. *Selection skill*

As explained in the Methodology section, this criterion implies that an investor should not be required to possess any unique abilities to select an asset. Employing a cryptocurrency index enables investors to sidestep the selection of specific currencies. If not, even buying a basket of cryptocurrencies would allow buyers to invest in the asset class without having particular picking abilities (i.e. purchasing the ten coins with the highest Market Capitalisation).

7. *Access at a reasonable price*

The final criterion examines the transaction costs and liquidity of cryptocurrencies. Table 3 summarises the results for the Spread Percentage and the Turnover Ratio.

The market Spread Percentage is the difference between the order book's highest bid and the lowest ask price. The formula can be represented as below:

[26] The expected utility of cryptocurrencies will analysed in details in the 3rd chapter: Portfolio Optimisation

$$Spread\ Percentage = \frac{(A_{L_t} - B_{H_t})}{P_t} \qquad 2.1$$

Where A_t is the lowest ask price, B_{H_t}consists in the highest bid price, and P_t is the close price at time t.

Market spreads can have a significant impact on the strategy's performance. Cryptocurrencies are often traded weekly or monthly, so a low Spread Percentage is desirable.

We can immediately notice that the Spread Percentage can have very different values, depending on the Crypto itself. For instance, USDT and USDC - the Stablecoins - show a Spread Percentage of 1,3% and 1,2%, respectively. These values are as low even when compared to the stock market. On the other side, the bid-ask spread percentage can arrive at 9,8%, such as in the case of LUNA. The market can be considered pretty variable under this perspective, but it generally shows higher costs than traditional asset classes.

Another critical factor to be taken into consideration are the trading commissions. Cryptocurrencies exchange fees range from 0.1% to 1%. They are modest compared to traditional assets, which can arrive at up to 5% of the investment.

Criptocurrency	Spread Percentage	Turnover Ratio
BTC	0.045704	0.122677
ETH	0.062202	0.327462
USDT	0.013385	0.430209
LTC	0.069112	0.649066
USDC	0.012129	0.419209
ADA	0.078677	0.102380
BNB	0.068968	0.082632
XRP	0.069856	0.168712
LUNA	0.097669	0.066653
EOS	0.073212	0.680860

The turnover ratio is the best measure to understand how liquid an asset or an entire market is. The formula can be represented as:

Table 3: Spread Percentage and Turnover Ratio for a basket of 10 cryptos. Own work, computed in Python

$$Turnover\ Ratio_t = \frac{Sh_t}{NSh_t} \qquad 2.1$$

Where h_t denotes the coins traded at time t, and NSh_t denotes the coins in circulation on day t.

Table 3 indicates that the market is highly liquid, with positive peaks for the single cryptos above 60%, and with a mean of 24,8% for the whole basket.

Notwithstanding the high Spread Percentage, the Crypto assets show a high Turnover Ratio, a synonym of high liquidity, and low fees compared to the traditional asset classes. Because of that, we can consider the 7th criterion as fully satisfied.

3. Portfolio Optimisation

3.1 Theory behind Portfolio Optimisation

3.1.1 Modern Portfolio Theory

Harry Markowitz presented the modern portfolio theory under the title "Portfolio Selection" in the *Journal of Finance* in 1952. A portfolio is considered a weighted linear combination of the assets that make it up.

The Modern Portfolio Theory is based on the following assumptions:

• financial assets are correlated;

• the market is perfect: there are no entry barriers, taxes, transaction costs, entry barriers, information is freely available to all, and there is perfect competition;
• investors are risk-averse. They aim at avoiding dangerous investments;
• investors are rational. They attempt to maximise their returns for a given degree of risk;
• all investors have the same time horizon for making investment decisions.

According to Markowitz, the investor's objective is to maximise the portfolio's return while minimising its risk. The outcome is what he calls: "the efficient portfolio".[27]

The portfolio enables either maximisation of return or minimisation of risk for a given return. All the efficient portfolios form what Markowitz refers to as "the efficient frontier". These are the portfolios that prudent investors seek. By investing in additional securities, an investor can reap the benefits of diversification, a reduction in the riskiness of the portfolio. Diversification refers to the concept that a portfolio should contain uncorrelated assets (or just weakly correlated).

The portfolio return is determined as the sum of the proportionally weighted assets' returns, as below:

$$E(R_p) = \sum_i w_i \, E(R_i) \qquad 3.1$$

Where R_p denotes the portfolio return, R_i is the return on the asset i, w_i denotes the weight of an individual asset, and i denotes the number of assets in the portfolio.

Portfolio variance is defined as a function of the correlation coefficients, volatilities, and weights of each asset pair in the portfolio (Markowitz, 1952), as shown in the below equation:

[27] Portfolio Selection, H. Markowitz, 1952

$$\sigma_p^2 = \sum_i w_i^2 \sigma_i^2 + \sum_i \sum_{j \neq i} w_i w_j \sigma_i \sigma_i p_{ij} \qquad 3.2$$

Where σ denotes the standard deviation of an individual asset, and p is the correlation coefficient between the returns on a pair of assets i and j.

The volatility, or risk of a portfolio, is measured as follows:

$$\sigma_p = \sqrt{\sigma_p^2} \qquad 3.3$$

The covariance of the constituent assets determines the variance of the entire portfolio. The greater the covariance among assets, the higher the portfolio's volatility. This relationship enables the use of uncorrelated assets to achieve diversification benefits.

As mentioned before, the efficient frontier is a batch of optimal portfolios that provide the highest expected return for a specified risk grade or the lowest return for a specified level of risk. Portfolios that drop below the efficient frontier are suboptimal because they do not generate a reasonable return rate relative to the risk grade. Likewise, portfolios clustering to the right of the efficient frontier is suboptimal because they carry a higher risk relative to the defined rate of return.

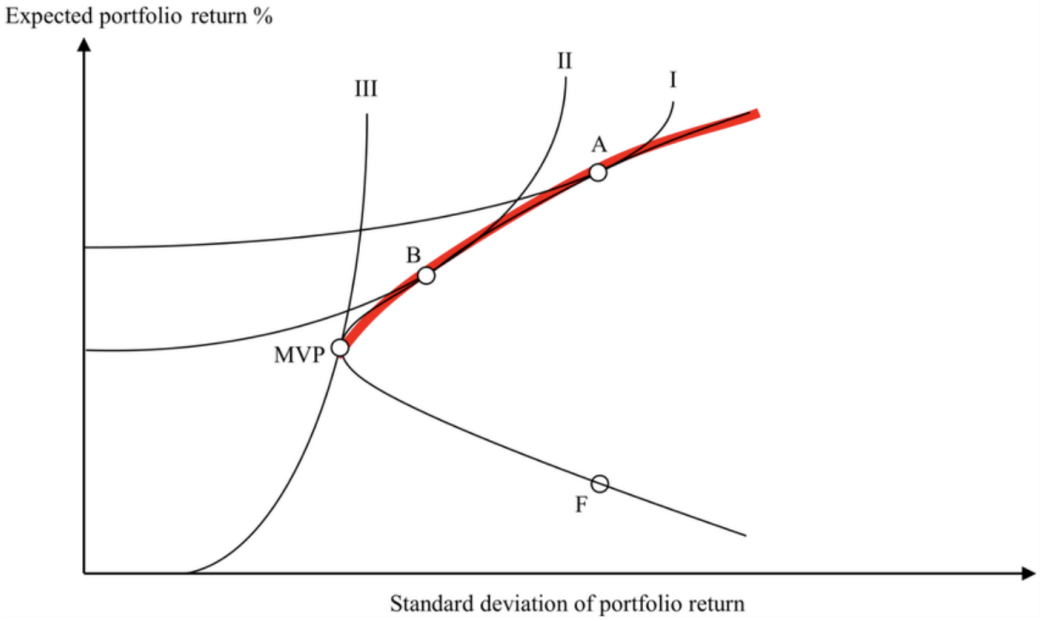

Figure 3: Portfolio selections on Markowitz's efficient frontier; Indifference Curves (Copeland et al. 2005)[28]

By introducing the risk-free tangent line from the y-axis point of this rate to the efficient frontier's upper bound, the capital allocation line is determined, forming a new efficient frontier. Tangency portfolios are a collection of risky assets with the highest Sharpe ratio, which can be calculated using the following formula:

$$S_a = \frac{E[R_a - R_b]}{\sigma_a} = \frac{E[R_a - R_b]}{\sqrt{var[R_a - R_b]}} \qquad 3.4$$

Where R_a denotes the portfolio return, R_b denotes the risk-free or benchmark return, and S_a denotes the asset's excess return's volatility. A greater Sharpe ratio signifies a higher rate of return on the risk unit.[29]

3.1.2 Post-Modern Portfolio Theory

[28] Portfolio selections on Markowitz's efficient frontier; Indifference Curves (Copeland et al. 2005)
[29] Asset allocation: Management style and performance measurement, W. F. Sharpe, 1992

Due to the significant volatility of cryptocurrencies, the Post-Modern Portfolio Theory is used to determine the reliability of portfolio improvements due to crypto-asset inclusion. For each portfolio formed using PMPT optimisation, the downside risk and the Sortino ratio are assessed. This enables one to determine whether the addition of cryptocurrencies, despite their high volatility, provides diversification benefits and improves portfolio performance.

The Post-Modern Portfolio Theory was devised in 1991 by software designers Brian M. Rom and Kathleen Ferguson in response to perceived problems and limits in Markowitz's Modern Portfolio Theory, although the latter is still largely used in portfolio management and asset allocation.

According to Brian M. Rom and Kathleen W. Ferguson, two of the most significant improvements made by the Post-Modern Portfolio Theory formulation are the downside risk and asymmetrical return distributions[30].

While the MPT assumes that the risk is symmetrical, the PMPT's risk is asymmetrical in nature, being the weights assigned to a loss greater than those assigned to gain. The Downside risk is quantified with a target semi-deviation, referred to as the Downside Deviation, which captures negative returns. Therefore, the actual risk occurs when returns fall below a certain level, although positive moves above this level are desirable for an investor and do not constitute a risk.

Purpose	***Risk Measure***	***Outperformance vs Benchmark***	***Risk Compared to Benchmarks' risk***	***Excess Return per Unit of Risk***
MPT Version	Standard Deviation	Alpha	Beta	Sharpe Ratio
PMPT Version	Downside Risk (DR)	Omega Excess; Also Excess Return (Above MAR)	DR vs Benchmark DR (Though various Betas could be calculated using DR Components)	Sortino Ratio (Excess Return DR)

Table 4: MPT versus PMPT Measures. P. Swisher & G. W. Kasten (2005), Contributions to Post-Modern Portfolio Theory[31]

[30] Post-Modern Portfolio Theory Comes of Age, B. M. Rom, K. W. Ferguson, 1994
[31] Contributions to Post-Modern Portfolio Theory ,P. Swisher & G. W. Kasten,2005

The standard deviation influenced by downside risk is now defined as the annualised standard deviation of asset returns that fall below the investor-specified minimum acceptable threshold. In other words, it is a semi-deviation from the aim. The downside risk is expressed as a percentage, comparable to standard deviation (Sortino and Van Der Meer, 1991)[32].

$$d = \sqrt{\int_{-\infty}^{t} (t-r)^2 f(r) dr} \qquad 3.5$$

When *d* denotes downside risk or deviation, *t* is the minimum acceptable rate of return (*MAR*) – assumed as equal to the risk-free rate –, *r* denotes a random return, and *f(r)* denotes the annual return distribution function. Assume that MAR equals the risk-free rate, which in our instance is zero.

So, Rom and Ferguson replaced MPT's Sharpe ratio with the Sortino Ratio[33].
Volatility skewness, which quantifies the percentage of the overall variation in distribution due to returns above and below the mean, was the second portfolio-analysis statistic added to the PMPT rubric.

$$Sortino\ ratio = \frac{r-t}{d} \qquad 3.6$$

Where *r* denotes yearly return, *t* denotes the target rate of return, and *d* denotes downside risk.

3.2 Results

[32] Downside Risk, F. A. Sortino, R. Van Der Meer, 1991
[33] Performance Measurement in a Downside Risk Framework, F.A. Sortino, L. N. Price

This section will see in detail the tangency portfolios, assess portfolio weights, and generate performance measures such as the Sharpe ratio. The model we run follows the below assumptions:

1. The indices are reflective of the asset class as a whole. Having chosen ETFs, it includes recomposition of the weights, and it comprehends eventual dividends.
2. The risk-free rate is equal to zero.
3. No transaction costs apply.
4. The maximum weight of a single asset in a portfolio does not exceed 50% to avoid a single asset class gaining dominance and maintaining a well-diversified portfolio.

3.2.1 Modern Portfolio Theory: Application

As mentioned in the above paragraph, the Tangency portfolio is analysed for the Modern Portfolio Theory. It leads to the optimisation of the Sharpe Ratio. To find the best combination overall for the proxies mentioned in Section 1.3, we run 20'000 combinations on Python over the period spanning from June 2018 to June 2021.

The weights optimisation for the Tangency Portfolio is shown in Table 5.[34] [35] [36]

Tangency Portfolio	CRIX	Stocks	Bonds	Commodities	FX	Real Estate
Without Crypto, Long positions only	0	0,35555	0,34883	0,12727	0	0,16835
With Crypto, Long positions only	0,09364	0,3185	0,34557	0,11635	0	0,12594
Without Crypto, Lond and Short positions	0	0,5	0,5	0,20839	-0,46105	0,25266
With Crypto, Long and Short positions	0,11849	0,40915	0,44446	0,15106	-0,28768	0,16452

Table 4: Tangency Portfolio: weights optimisation (MPT). Own work computed in Python.

We immediately notice that, when just long positions are allowed, the system recognises the FX proxy as inconvenient for maximising the Sharpe Ratio. Consequently, it is not allocated

[34] Portfolio Optimization using MPT in Python, P. Tyagi, 2021
[35] PyPortfolioOpt, pypi.org
[36] Portfolio Optimization with Python using Efficient Frontier, S. Dash, 2020

into the portfolio. The same logic applies to the long and short positions allowed scenario. The script prefers to sell the FX in variable measures, depending on the presence of the CRIX. Indeed, in the fourth scenario, the system finds it more convenient to buy more Cryptocurrencies and short sell the FX proxy in a lower measure. We find that the two asset classes with the highest allocation, for the Tangency Portfolio case, are Stocks and Bonds. Indeed, the two proxies result as the most used in the portfolio composition, spanning from the 31.85% and 34,56% - respectively - of the portfolio with Crypto and only long positions allowed, to the 50% each in the portfolio without Crypto, Long and Short positions allowed. Quick reminder: we set the maximum threshold equal to 50% to allow a good level of diversification. Finally, Crypto is found as convenient to insert in a diversified portfolio. Their optimal share range from 9 to 12%.

To analyse the four portfolios mentioned above in depth, risk-reward measures for each are shown in Table 6.

Tangency Portfolio	Sharpe Ratio	Expected Annual Return	Annual Volatility
Without Crypto, Long positions only	0,763	0,060	0,079
With Crypto, Long positions only	1,082	0,104	0,096
Without Crypto, Lond and Short positions	0,784	0,090	0,115
With Crypto, Long and Short positions	1,086	0,135	0,124

Table 5: Tangency portfolio's descriptive statistics (MPT). Own work computed in Python

The Expected Annual Return is significantly higher in the portfolios which include Cryptocurrencies: 10.4% and 13.5% versus the 6% and 9% of the portfolios not comprising Cryptos. This change – that leads to an increase in Annual Expected Returns of 1.7 and 1.5 times – is due to the inclusion of the CRIX in a measure equal just to 9% and 12% of the whole portfolio and does not twist the portfolio composition.

Of course, also the Annual Volatility augments, from 7.9% and 11.5% of the portfolios without Cryptos to the 9.6% and 12.4% of the portfolios including them. However, this increase is equal to 1.2 and 1.1 times. It means that the benefit of adding a basket of cryptocurrencies to a portfolio, both with long-only and long and short positions, leads to benefit in terms of risk-return tradeoff.

That is well explained by the Sharpe Ratio, which shows the average return earned in excess of the risk-free rate per unit of volatility. In our case, however, having set the risk Free equal to zero, the formula could be represented as below:

$$Sharpe\ Ratio = \frac{R_p}{\sigma_p} \qquad 3.7$$

The computed Sharpe Ratio demonstrate that crypto-assets certainly provide an investor with diversification benefits due to their distinct risk/reward profile and lack of association with other asset classes. Consequently, allocating ≅ 10% of the capital to include a basket of cryptocurrencies in a portfolio results in risk-adjusted outperformance.

3.2.2 Post-Modern Portfolio Theory: Application

The Post-Modern Portfolio Theory's best allocation are analysed in Table 7.[37] [38]

Tangency Portfolio	CRIX	Stocks	Bonds	Commodities	FX	Real Estate
Without Crypto, Long positions only	0	0,41098	0,37502	0,018809	0,00823	0,186963
With Crypto, Long positions only	0,12754	0,2675	0,22557	0,131646	0,10911	0,138648
Without Crypto, Long and Short positions	0	0,24601	0,10471	0,06636	-0,46054	0,122392
With Crypto, Long and Short positions	0,15382	0,28776	0,13951	0,075466	-0,21765	0,125807

Table 6: Tangency Portfolio: weights optimisation (PMPT). Own work computed in Python.

The Tangency Portfolio, in this case, maximises the Sortino Ratio, which takes the portfolio's return, subtracts the risk-free rate, and then divides that amount by the asset's downside deviation.

In our case, however, the risk-free is equal to zero. Hence, the formula occurs as below:

[37] Pymcef, pypi.org
[38] Investment Portfolio Analysis: Sortino Ratio, B. Mullen, 2021

$$Sortino\ Ratio = \frac{R_p}{\sigma_d} \qquad 3.8$$

With R_p being the Expected Portfolio Return, and σ_d the Downside Standard Deviation.

Comparing the MPT Portfolio maximising the Sharpe Ratio and the PMPT Portfolio maximising the Sortino Ratio, we immediately notice that the trend is the same. Shorting the FX proxy whenever short positions are allowed and allocating the highest weight among traditional asset classes to Stocks.

However, under the Post-Modern Portfolio Theory, CRIX is found more convenient than under the MPT, having an asset allocation equal to 12,75% and 15,38% for both the portfolios in which it is allowed.

Tangency Portfolio	Sortino Ratio	Annual Return	Downside SD	Upside SD
Without Crypto, Long positions only	0,956718	0,076104	0,079547	0,061092
With Crypto, Long positions only	1,683431	0,132237	0,078552	0,071841
Without Crypto, Long and Short positions	1,010896	0,048956	0,048428	0,037204
With Crypto, Long and Short positions	1,677691	0,153023	0,091211	0,084816

Table 7: Tangency portfolio's descriptive statistics (PMPT). Own work computed in Python.

The conducted risk analysis suggests that the annual return is much higher including Cryptocurrencies, being 1,7 and 3,1 times higher than the alternative without the latter asset class.

On the other hand, Downside Standard Deviation remains stable for the portfolios with long positions only, and sharply increase in the one allowing short positions.

Summarising, Sharpe and Sortino ratios of the tangency portfolios lead to the same result: including cryptocurrencies increases the portfolio profitability leading to a less than proportional augment in risk. Hence, it permits to optimise the portfolios constituted just by traditional asset classes.

Nonetheless, price prediction techniques instruments may significantly increase the performance of portfolios including cryptocurrencies, and augment the investment awareness. This will be discussed in the next chapter.

4. Deep learning and Crypto prices prediction

4.1 Introduction to Machine Learning

Deep learning and artificial intelligence have recently established the foundation for bitcoin portfolio optimisation. Researchers examined a variety of state-of-the-art machine learning algorithms for forecasting Bitcoin prices, notwithstanding the difficulty associated with price volatility and dynamism. In cryptocurrency price prediction, machine learning techniques such as recurrent neural networks (RNN) and long short-term memory (LSTM) have been demonstrated to outperform standard time series models[39].

[39] Fundamentals of Recurrent Neural Network (RNN) and Long Short-Term Memory (LSTM) Network, A. Sherstinsky, 2020

Numerous studies have been conducted on Bitcoin price prediction utilising machine learning and time series analysis[40].

With restricted data, neural networks such as the LSTM and gated recurrent unit (GRU) can effectively analyse prior data to learn from non-linear patterns. Compared to standard time-series techniques, deep models require specific training and hyperparameter adjustment to get results that may be computationally intensive for large datasets. However, because market data for stock and cryptocurrency price prediction is usually restricted and computing complexity is irrelevant, light learning models can be employed effectively. Thanks to that, these models in the future may have a significant impact on quantitative finance.

Traditionally, in research on deep learning, LSTM has been primarily employed to analyse time series. On the other hand, some analyses indicate that the GRU method outperforms the LSTM model. The simplicity of the GRU model, in which forgetting and updating occur concurrently, is effective in predicting Bitcoin prices. Furthermore, the Gated Recurrent Unit model outperformed the LSTM and bi-LSTM models, exhibiting much lower Mean Absolute Percentage Error (MAPE) and Root Mean Square Error (RMSE) (RMSE).

Research findings indicate that when machine learning models are implemented with sufficient understanding and backtesting, they can help manage portfolio risk and minimise financial losses. As a plan for the future, it would be interesting to investigate additional elements affecting the cryptocurrency market's prices, such as social media, tweets, and trading volume.

This chapter focuses on the Gated Recurrent Unit (GRU) model to forecast future cryptocurrency prices using machine learning algorithms and artificial intelligence approaches by achieving accurate predictions to assist investors.

4.2 Review of Literature

[40] Dueling Network Architectures for Deep Reinforcement Learning, Z. Wang, M. Hessel, H. van Hasselt, M. Lanctot, N. de Freitas, 2016

Machine learning is a subset of artificial intelligence capable of forecasting the future using historical data. Machine Learning based models have several advantages over other forecasting models; as earlier researches have demonstrated, they produce a result close to or identical to the actual result, demonstrating a high result's accuracy. Neural networks (NN), support vector machines (SVM), and deep learning are examples of machine learning. Some researchers, in 2018, focused their work on forecasting time series data in particular and used two machine learning algorithms: random forests (RF) and stochastic gradient boosting machine (SGBM). The results demonstrate that the machine learning ensemble technique can forecast Bitcoin prices[41].

The decision-making process must make the appropriate choice at the appropriate moment, thereby mitigating the investing process's inherent risks. Others aggregated Realised Volatility data using minute-sampled Bitcoin returns over three-hour periods. Many machine learning techniques, including ANN (MLP, GRU, and LSTM), SVM, and ridge regression, were employed to forecast future values based on historical data. The results were compared to the heterogeneous auto-regressive realised volatility (HARRV) model with optimum lag settings. The data indicate that the proposed approach accurately forecasts prices, implying that the technology might estimate prices for a range of cryptocurrencies[42]. Finally, researchers in "A Novel Cryptocurrency Price Prediction Model Using GRU, LSTM and bi-LSTM Machine Learning Algorithms" compared three models for predicting BTC, ETH, and LTC's prices (LSTM, GRU, and bi-LSTM). According to the experimental results, GRU had the again the best performance.[43]

4.3 Gated Recurrent Unit (GRU) Model

The Gated Recurrent Unit (GRU) is a Recurrent Neural Network that uses gating methods to govern and manage the flow of information between neural network cells. GRUs was introduced in 2014[44] by Cho et al. and can be viewed as a new design compared to the widely adopted LSTM proposed in 1997 by Sepp Hochreiter and Jürgen Schmidhuber[45].

[41] Comparative Performance of Machine Learning Algorithms for Cryptocurrency Forecasting, N. A. Hitam, A. R. Ismail, 2018
[42] Artificial Neural Networks for Realized Volatility Prediction in Cryptocurrency Time Series, R. Miura, L. Pichl, T. Kaizoji, 2019
[43] A Novel Cryptocurrency Price Prediction Model Using GRU, LSTM and bi-LSTM Machine Learning Algorithms, M. J. Hamayel, A. Y. Owda, 2021
[44] Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling, J. Chung, C. Gulcehre, H. Cho, Y. Bengio, 2014
[45] Long Short-term Memory, S. Hochreiter, J. Schmidhuber, 1997

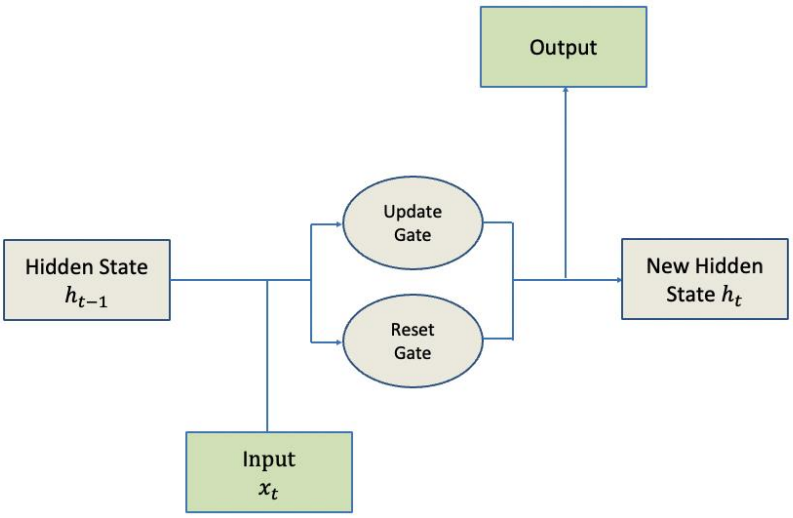

Figure 4: Representation of the GRU's functioning

The GRU's structure enables it to capture dependencies from big data sequences adaptively and without deleting information from earlier segments of the sequence. That is accomplished by using gating units similar to those used in LSTMs, which address the traditional RNN's vanishing/exploding gradient problem. These gates are in charge of determining how much information should be retained or discarded at each step.
Apart from its internal gating method, the GRU acts similarly to an RNN in which the GRU cell consumes sequential incoming data together with the memory, also known as the hidden state. The hidden state is subsequently re-fed into the RNN cell together with the sequence's subsequent input data. This procedure, similar to a relay system, continues until the desired output is obtained.

The GRU's capacity to retain long-term dependence is due to the computations performed within the GRU cell to generate the hidden state. While LSTMs transmit two distinct states between cells — the cell state and the hidden state — which contain the long and short-term memories, respectively, GRUs transmit only one hidden state between time steps. Due to the gating mechanisms and computations that the hidden state and input data undergo, this hidden state is capable of holding both long-term and short-term dependence concurrently[46].

[46] Gated Recurrent Units (GRU), Dive into Deep Learning, 2020

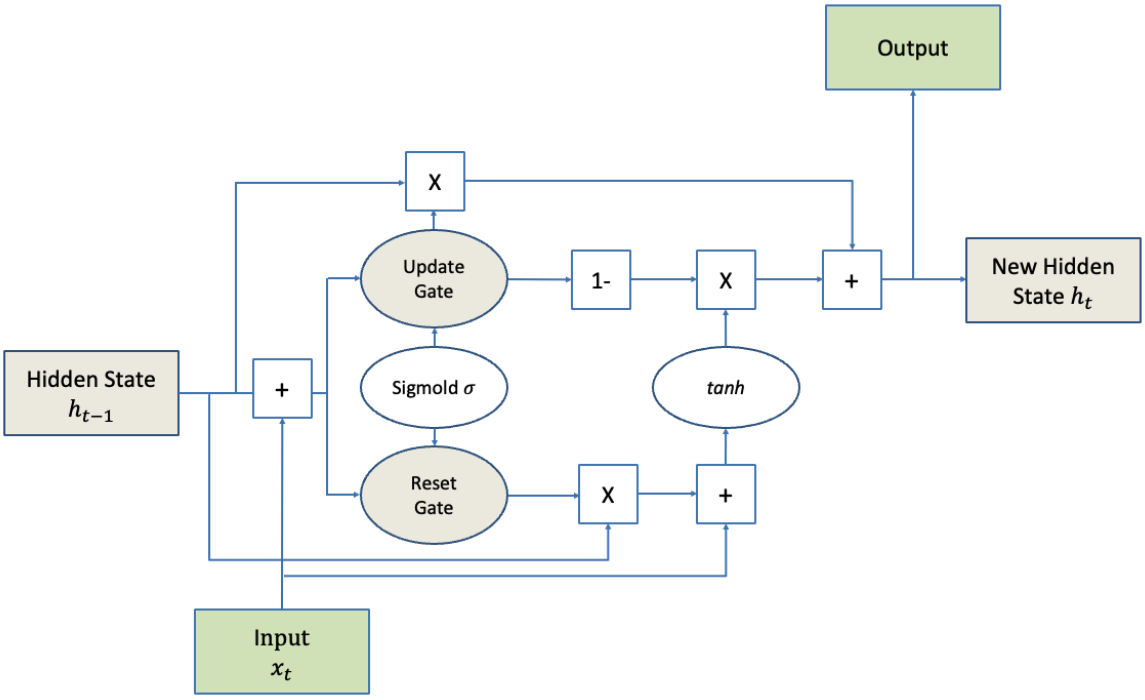

Figure 5: GRU's Structure in detail

The GRU cell comprises only two gates: the Update and Reset gates. As with the gates in LSTMs, these gates in the GRU are taught to filter out irrelevant information while retaining helpful information. These gates are vectors with values ranging from 0 to 1 multiplied by the input data and hidden state. A 0 in the gate vectors signifies that the matching data in the input or hidden state is irrelevant and will thus be returned as a zero. A 1 value in the gate vector, on the other hand, indicates that the relevant data is significant and will be used.

While the structure may appear complex because of the vast number of connections, the underlying mechanism can be broken down into three distinct steps.

Reset Gate

The Reset gate is derived and calculated using both the previous time step's hidden state and the current step's input data.

The above is accomplished mathematically by multiplying the previous hidden state and the current input by their respective weights and summing them before applying a sigmoid function to the sum. The sigmoid function transforms the values between 0 and 1, enabling the gate to filter out less-important information in the following phases.

$$Gate_{reset} = \sigma(W_{input_{reset}} \cdot x_t + W_{hidden_{reset}} \cdot h_{t-1}) \quad 4.1$$

When the entire network is trained by back-propagation, the weights in the equation are changed to ensure that the vector retains only meaningful information.

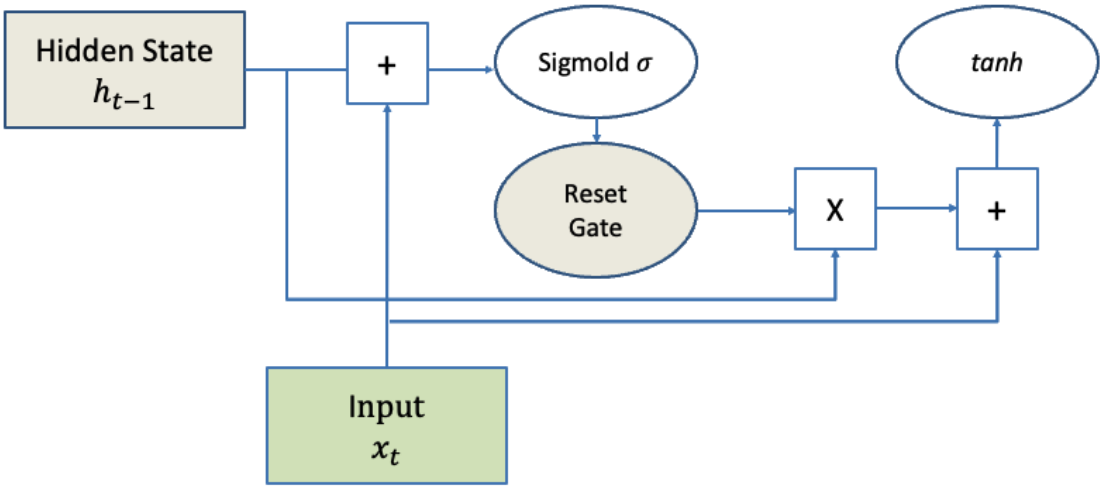

Figure 5: Reset Gate

The previous hidden state will be multiplied by a trainable weight and the reset vector element-wise (Hadamard product). This procedure will determine which information from earlier time steps should be retained together with the current inputs. Simultaneously, the current input will be multiplied by a trainable weight, and the product of the reset vector and the previous hidden state will be added. Finally, the final result will be applied to a non-linear activation *tanh* function to yield *r* in the equation 4.2.

$$r = \tanh(gate_{reset} \odot (W_{h1} \cdot h_{t-1}) + W_{x1} \cdot x_t) \quad 4.2$$

The Update gate is computed using the previous hidden state and the current input data, like the Reset gate.

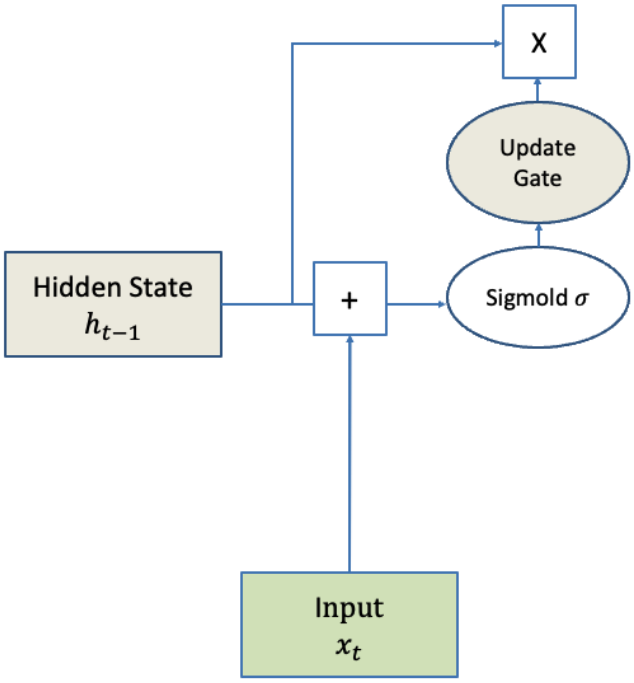

Figure 5: Update Gate

The Update and Reset gate vectors are constructed using the same algorithm, but the weights multiplied by the input and hidden state are unique to each gate, resulting in unique final vectors. That enables the gates to perform their intended function.

$$Gate_{update} = \sigma(W_{input_{update}} \cdot x_t + W_{hidden_{update}} \cdot h_{t-1}) \quad 4.3$$

The Update vector will then be multiplied by the previously concealed state element-by-element to yield u in the equation below, which will be utilised to compute our final output later.

$$u = gate_{update} \odot h_{t-1} \quad 4.4$$

Additionally, the Update vector will be employed in a subsequent operation to obtain the final output. In this case, the Update gate's goal is to assist the model in determining how much prior knowledge stored in the previous concealed state should be kept for the future.

Finally, the Update gate is reused to access the updated hidden state.
This time, the element-wise inverse of the same Update vector will be used (1 - Update gate) and multiplied by the output of the Reset gate, r. This action instructs the Update gate on which portion of the new data should be placed in the concealed state.
Finally, the result of the preceding operations will be added to the output of the previous step's Update gate, u. That will update the hidden state.

$$h_t = r \odot \left(1 - gate_{update}\right) + u \qquad 4.5$$

By feeding this new hidden state through a linear activation layer, it can also be used as output for that time step.

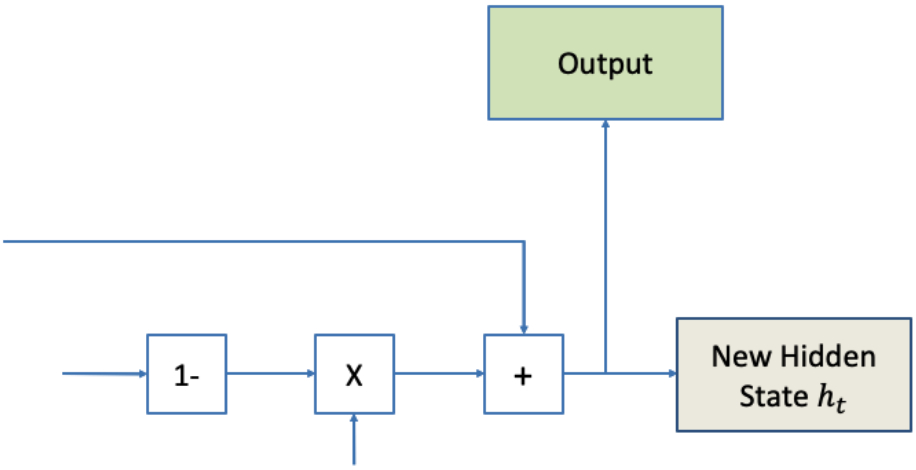

Figure 6: Final Output

As demonstrated in the processes above, the Reset gate is in charge of determining which portions of the previous hidden state should be combined with the current input to propose a new hidden state.

Additionally, the Update gate determines how much of the previous hidden state should be preserved and how much of the new proposed hidden state (derived from the Reset gate) should be added to the final hidden state. When the Update gate is multiplied by the previous hidden state, the network decides which parts to retain and reject. Subsequently, it patches up the missing pieces of information by filtering the suggested new hidden state from the Reset gate using the inverse of the Update gate.

That enables the network to maintain long-term ties. For example, if the Update vector values are near to 1, the Update gate can choose to preserve the majority of earlier memories in the hidden state without recalculating or updating the complete hidden state.

The vanishing and exploding gradient problem happens during back-propagation, particularly when the RNN processes long sequences or has numerous layers. During training, the error gradient is utilised to update the network's weight in the correct direction and magnitude. However, this gradient is determined using the chain rule, beginning at the network's end. As a result, the gradients undergo continuous matrix multiplications during back-propagation and vanish or explode exponentially over long sequences. A too slight gradient prevents the model from updating its weights properly, whereas a too large gradient makes it unstable.

The gates in GRU assist in resolving this problem. While conventional RNNs always replace the entire hidden state's content at each step, GRU retains most of the existing hidden state while adding new content. That enables back-propagation of error gradients without them vanishing or inflating too quickly due to addition operations.[47] [48] [49]

4.4 Evaluation Matrix

[47] Gated Recurrent Units (GRU), Dive into Deep Learning, 2020
[48] Introduction to Gated Recurrent Unit (GRU), S. Saxena, 2021
[49] Understanding GRU Networks, S. Kostadinov, 2017

The most frequently used statistic for regression tasks is the root mean square error (RMSE) (root-mean-square error). It is the square root of the average squared difference between the actual and anticipated prices:

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2} \qquad 4.6$$

The actual score for the *ith* data point is denoted by y_i, whereas the predicted value is denoted by $\hat{y}_i$. This formula can be understood intuitively as the Euclidean distance between actual and predicted prices vectors, averaged by *n*, whit *n* denoting the number of data points.

Even though it is less robust to outliers than other indicators, RMSE is frequently used as an evaluation statistic, mainly when working with deep learning techniques.

The advantage of using RMSE is that the output value is in the same unit as the desired output variable, which simplifies interpreting loss.

The mean absolute percentage error (MAPE), alternatively known as the mean absolute percentage deviation (MAPD), is a statistical measure of a forecasting method's prediction accuracy. Typically, accuracy is expressed as a ratio specified by the formula:

$$MAPE = \frac{100\%}{n}\sum_{t=1}^{n}\left|\frac{A_t - F_t}{A_t}\right| \qquad 4.7$$

Where A_t denotes the observed value, and F_t denotes the forecasted value. Their difference is calculated by dividing it by the actual value of At. The absolute value of this ratio is calculated by adding the absolute values of all projected points in time and dividing them by the number of fitted points, *n*.

Due to its natural interpretation in terms of relative error, mean absolute percentage error is frequently used as a loss function for regression concerns and model evaluation.

4.5 Dataset

The dataset used in the analysis was obtained from www.marketwatch.com and www.coinmarketcap.com. Prices were collected daily from 1st January 2013 to 1st January 2022.

4.6 Results

This section summarises the findings for Bitcoin and Ethereum.

Figure 7 compares the actual and predicted prices of BTC using the GRU model. The graph demonstrates that the difference between predicted and actual prices is tiny throughout the testing set, with some time-frame variations.

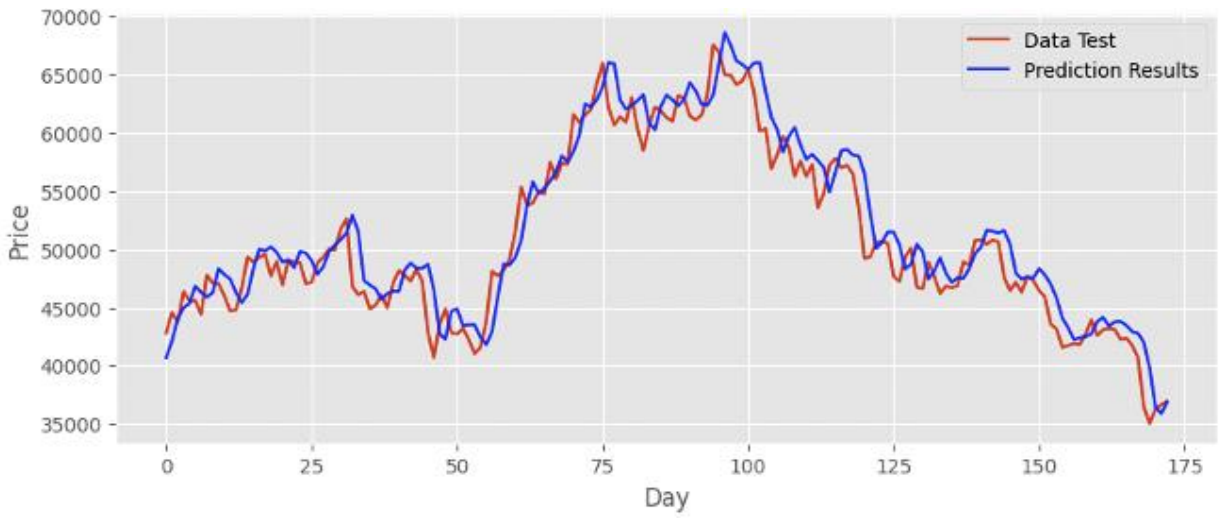

Figure 7: Comparison between Bitcoin Actual Data and Prediction Results. Own work, computed in Python.

The GRU's Mean Absolute Percentage Error (MAPE) for BTC is equal to 0,17, while the Root Mean Square Error (RMSE) is 10671, which can be considered excellent since nine years of data have been analysed.

Bitcoin	Mean	Standard Deviation	Minimum	1st Quartile	Median	3rd Quartile	Maximum
Actual Results	50789,14	7495,168637	35030,25	45879,5742	48896,7227	57248,457	67566,82813
Prediction Results	51897,289	7767,080078	35786,043	46410,2578	49804,2539	58457,3164	69313,58594

Table 8: Descriptive statistics for Bitcoin's Actual and Prediction results. Own work, computed in Python.

According to statistical analysis, the predicted price has a mean of 51.897,29 USD, a maximum of 69.313,59 USD, and a minimum of 35.786,04 USD, whereas the actual price has a mean of 50.789,14 USD, a maximum of 67.566,83 USD, and a minimum of 35.030,25 USD. So, the results can be considered satisfactory.

Figure 8 compares the actual and predicted prices of ETH using the GRU model.

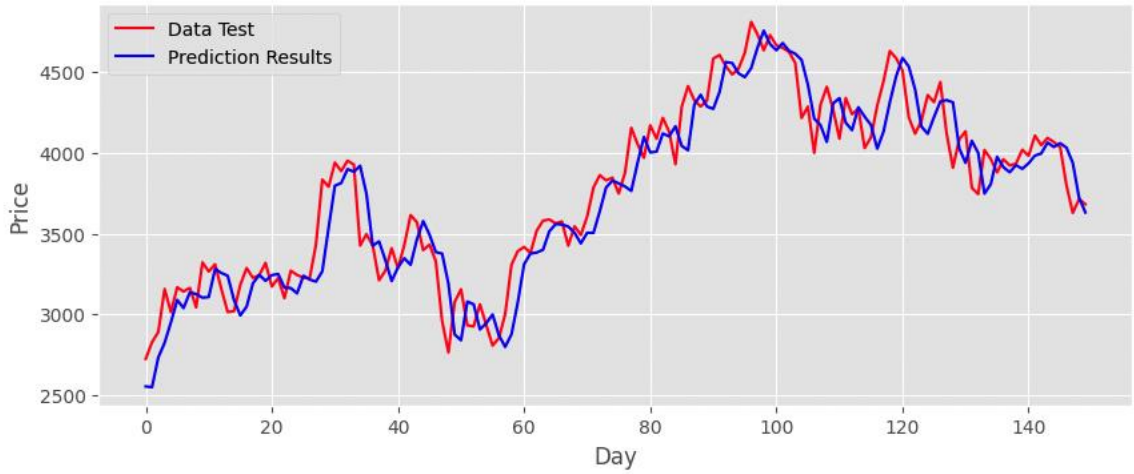

Figure 8: Comparison between Ethereum Actual Data and Prediction Results. Own work, computed in Python.

With a Mean Absolute Percentage Error of 0,168 and a Root Mean Square Error of 761,54, the model can be considered accurate also for predicting Ethereum.

Ethereum	Mean	Standard Deviation	Minimum	1st Quartile	Median	3rd Quartile	Maximum
Actual Results	3768,5323	535,103651	2724,61987	3292,08051	3832,6051	4191,65796	4812,087402
Prediction Results	3666,7988	512,575378	2539,31543	3207,28613	3737,73694	4068,76001	4623,424316

Table 9: Descriptive statistics for Ethereum's Actual and Prediction results. Own work, computed in Python.

According to statistical analysis, the predicted price has a mean of 3666,80 USD, a maximum of 4623,42 USD, and a minimum of 2539,31 USD, whereas the actual price has a mean of 3768,53 USD, a maximum of 4812,09 USD, and a minimum of 2724,62 USD. Therefore, the mean difference between actual and forecasted prices is 101,73 USD.

Both for Bitcoin and Ethereum, the above results can be considered satisfactory. They are valuable for investment purposes and can definitely help investors include cryptocurrencies in their portfolios.

5. Conclusion

Notwithstanding their young age, cryptocurrencies confirmed a certain financial and composition strength. They are developing fast. The technology is continuously progressing, i.e. Ethereum 2.0, and the public diffusion continues to augment.

By stressing cryptocurrencies with the seven requirements provided by several researchers, we found that cryptocurrencies are ready to be considered an asset class. They are probably unconventional, high volatile, but the analysis conducted on the portfolio optimization, by running more than 20.000 combinations, states that buying a percentage ranging from 9 to 15% of the portfolio as a whole leads to the optimization of the risk-return tradeoff. On the other side, the following must be taken into consideration. The above is valid for a basket of cryptocurrencies. In order to invest in single cryptos or to compose an own portfolio of chosen crypto-assets, the model presented in Section 4 would be helpful, being able to predict the price with an accuracy of 83% for a range of 20 days.

Data Source

Vanguard FTSE Developed World UCITS ETF	VDEV	www.bloomberg.com
Xtrackers II Global Government Bond UCITS ETF	XSGI	www.investing.com
Bloomberg Commodity CMCI Composite UCITS ETF	CCUSAS	
Euro Currency Index	EUR_I	
iShares Developed Markets Property Yield UCITS ETF	DPYA	
Cryptocurrency Index	CRIX	www.bloomberg.com
Bitcoin	BTC	www.coinmarketcap.com
Ethereum	ETH	www. marketwatch.com
Tether	USDT	
Litecoin	LTC	
USD Coin	USDC	
Cardano	ADA	
Binance Coin	BNB	
Ripple	XRP	
Terra	LUNA	
EOS.IO	EOS	

Bibliography

Papers:

Portfolio Selection, H. Markowitz, 1952
Bitcoin: A Peer-to-Peer Electronic Cash System, S. Nakamoto, 2008
The valuation of the crypto-assets, EY, 2019
The Cross-Section of Cryptocurrency returns, N. Borri, K. Shakhnov, 2018
Portfolio selections on Markowitz's efficient frontier; Indifference Curves (Copeland et al. 2005)
Asset allocation: Management style and performance measurement, W. F. Sharpe, 1992
Aggregation, Capital Heterogeneity, and the Investment CAPM, A. S. Goncalves, C. Xue, L. Zhang, 2017
Peer-to-Peer Networks: Architectures, Applications and Challenges, J. Sen, 2013
A New Look At Minimum Variance Investing, B. Scherer, 2010
Post-modern portfolio theory supports diversification in an investment portfolio to measure investment's performance, D. Rasiah, 2012
Comparative Performance of Machine Learning Algorithms for Cryptocurrency Forecasting, N. A. Hitam, A. R. Ismail, 2018
Long Short-term Memory, S. Hochreiter, J. Schmidhuber, 1997
Artificial Neural Networks for Realized Volatility Prediction in Cryptocurrency Time Series, R. Miura, L. Pichl, T. Kaizoji, 2019
A Novel Cryptocurrency Price Prediction Model Using GRU, LSTM and bi-LSTM Machine Learning Algorithms, M. J. Hamayel, A. Y. Owda, 2021
Is Modern Portfolio Theory still modern?, A.B. Davidow, 2020
Recurrent Neural Networks for Empirical Asset Pricing, F. M. Cruz, 2020
Python Deep Learning: Create A GRU (RNN) In TensorFlow, K. Jacobs, 2017
Research on blockchain consensus mechanism and implementation, X. Zhu, 2019
CRIX an Index for blockchain based Currencies, S. Trimborn, W. K. Härdle, 2016
Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling, J. Chung, C. Gulcehre, H. Cho, Y. Bengio, 2014
Bitcoin and Cryptocurrency Technologies, A. Narayanan, J. Bonneau, E. Felten, 2016
Deep Sequence Modeling: Development And Applications In Asset Pricing, L. W. Cong, K. Tang, J. Wang, and Y. Zhang, 2020
F5: optimised crypto-currency investment strategies, H. Elendner, Humboldt-Universität zu Berlin, 2018

A Gated Recurrent Unit Approach to Bitcoin Price Prediction, A. Dutta, S. Kumar, M. Basu, 2020
Investment Portfolio Analysis: Sortino Ratio, B. Mullen, 2021
Breaking Down ETH 2.0 – eWASM and EVM Explained, Moralis Academy, 2020
Making Markowitz's Portfolio Optimization Theory Practically Useful, B. Zhidong, L. Huixia, W. Wing-Keung, 2016
Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling, J. Chung, C. Gulcehre, K. H. Cho, Y. Bengio, 2014
Post-Modern Portfolio Theory Comes of Age, B. M. Rom, K. W. Ferguson, 1994
Contributions to Post-Modern Portfolio Theory ,P. Swisher & G. W. Kasten,2005
Research on blockchain consensus mechanism and implementation, X. Zhu 2019
Know The Best Evaluation Metrics for Your Regression Model, R. Agrawal, 2021
Performance Measurement in a Downside Risk Framework, F.A. Sortino, L. N. Price
Performance Metrics (Error Measures) in Machine Learning Regression, Forecasting and Prognostics: Properties and Typology, A. Botchkarev, 2019
Bitcoin and Cryptocurrency Technologies, A. Narayanan, J. Bonneau, E. Felten, 2016
Crypto Currencies. An analysis of Market Liquidity, M. Zobeli, C. Kley, B. E. Stumpp, 2020
Mining Process in Cryptocurrency Using Blockchain Technology: Bitcoin as a Case Study , A. A. Aljabr, A. Sharma, K. Kumar, 2019
CRIX an Index for cryptocurrencies, S. Trimborn, 2018
Can cryptocurrencies fulfil the functions of money?, S. Ammous, 2018
Crypto-Assets Unencrypted, S. Kim, A. Sarin, D. Virdi, 2018
Cryptocurrencies As an Asset Class? An Empirical Assessment, D. Bianchi, 2020
Portfolio Optimization using MPT in Python, P. Tyagi, 2021
Fundamentals of Recurrent Neural Network (RNN) and Long Short-Term Memory (LSTM) Network, A. Sherstinsky, 2020
Dueling Network Architectures for Deep Reinforcement Learning, Z. Wang, M. Hessel, H. van Hasselt, M. Lanctot, N. de Freitas, 2016

Reports:

Long-Term Capital: Market Assumptions. Time-tested projections to build stronger portfolios, J.P. Morgan Asset Management, 2022
Global Cryptocurrency Market Report 2021: COVID-19 Implications and Growth, The Business Research Company, 2021
Global Cryptocurrency Market by offering, by type, by end user, by company, by region, Forecast & Opportunity, 2026, TechSci Research, 2021

Articles:

The World's Cryptocurrency Is Now Worth More Than $3 Trillion, J. Ossinger, Bloomberg, 2021
Can cryptocurrencies drive a Covid-19 recovery in emerging markets?, Oxford Business Group, 2021
Public vs. Private Equity Returns: Is PE Losing Its Advantage?, Bain & Company
Ethereum 2.0: Ethereum's new dawn, Edith M., 2019
Know The Four Types of Cryptocurrencies Based On Their Utility, N. Das, 2022
Ethereum 2.0 Includes Major Changes That Could End Bitcoin's Blockchain Dominance, A. Kovačević, 2020
Gated Recurrent Units (GRU), Dive into Deep Learning, 2020
Introduction to Gated Recurrent Unit (GRU), S. Saxena, 2021
Understanding GRU Networks, S. Kostadinov, 2017
When will Ethereum 2.0 fully launch? Roadmap promises speed, but history says otherwise, J. Magas, 2020
Crypto is fully banned in China and 8 other countries, M. Q. Gutierrez, 2022
Crypto year at a glance: Ethreum outperformed in 2021, Metaverse is new DeFi, WazirX Trade Desk, 2022
The expanding functions and uses of stablecoins, Mitsutoshi Adachi, Alexandra Born, Isabella Gschossmann and Anton van der Kraaij, 2021
Who Accepts Bitcoin as Payment?, Joshua Sophy, 2022
Blockchain: A Very Short History Of Ethereum Everyone Should Read, Bernard Marr, 2020
The skewness of the shock, N. Bloom, F. Guvenen, S. Salgado, 2020
Crypto Primer: Valuing cryptocurrencies, Sygnum, 2021

Gated Recurrent Unit (GRU) With PyTorch, Gabriel Loye, 2019
Ethereum 2.0 Includes Major Changes That Could End Bitcoin's Blockchain Dominance, A. Kovačević, 2019
Survey released by Hartford Steam Boiler and Insurance Co. (HSB)
The Week Onchain (Week 52, 2021), Insights Glassnode
Crypto year at glance: Ethereum ouperformed in 2021, Metaverse in new DeFi, The Economic Times, 2022
Cryptocurrencies as an asset class in portfolio optimisation, O. Holovatiuk, 2019
Portfolio Optimization with Python using Efficient Frontier, S. Dash, 2020

Academic Books:

Portfolio Management: Theory & Practice, Schultz Collins Lawson Chambers, 2008
Asset Allocation: From Theory to Practice and Beyond, W. Kinlaw, M. P. Kritzman, D. Turkington, 2021

Web References:

www.royalton-crix.com
Bitcoin History: The Complete History of Bitcoin [Timeline]
www.coinmarketcap.com
PyPortfolioOpt, www.pypi.org

APPENDIX 1: Asset Allocation and Portfolio Optimisation Scripts

```
!pip install yfinance
!pip install yahoofinancials

!pip install cryptocmd

import time
import pandas_datareader.data as pdr
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import io
from cryptocmd import CmcScraper
from math import *

LUNA=CmcScraper("LUNA", "01-06-2016", "30-06-2021")
# Pandas dataFrame for the same data
LUNA = LUNA.get_dataframe()
DOT=CmcScraper("DOT", "01-06-2016", "30-06-2021")
# Pandas dataFrame for the same data
DOT= DOT.get_dataframe()
DOT['Turnover Ratio']=DOT['Volume']/DOT['Market Cap']
DOT['Spread Percentage']=(DOT['High']-DOT['Low']/DOT['Close'])
LUNA['Turnover Ratio']=LUNA['Volume']/LUNA['Market Cap']
LUNA['Spread Percentage']=(LUNA['High']-LUNA['Low']/LUNA['Close'])
BTC=CmcScraper("BTC", "01-06-2016", "01-06-2021")
BTC=BTC.get_dataframe()
BTC['Spread Percentage']=((BTC['High']-BTC['Low'])/BTC['Close'])
BTC['Turnover Ratio']=BTC['Volume']/BTC['Market Cap']
ETH=CmcScraper("ETH", "01-06-2016", "01-06-2021")
```

```
ETH=ETH.get_dataframe()
ETH['Spread Percentage']=((ETH['High']-ETH['Low'])/ETH['Close'])
ETH['Turnover Ratio']=ETH['Volume']/ETH['Market Cap']
USDC=CmcScraper("USDC", "01-06-2016", "01-06-2021")
USDC=USDC.get_dataframe()
USDC['Spread Percentage']=((USDC['High']-USDC['Low'])/USDC['Close'])
USDC['Turnover Ratio']=USDC['Volume']/USDC['Market Cap']
SOL=CmcScraper("SOL", "01-06-2016", "01-06-2021")
SOL=SOL.get_dataframe()
SOL['Spread Percentage']=(SOL['High']-SOL['Low']/SOL['Close'])
SOL['Turnover Ratio']=SOL['Volume']/SOL['Market Cap']
ADA=CmcScraper("ADA", "01-06-2016", "01-06-2021")
ADA=ADA.get_dataframe()
ADA['Spread Percentage']=(ADA['High']-ADA['Low']/ADA['Close'])
ADA['Turnover Ratio']=ADA['Volume']/ADA['Market Cap']
BNB=CmcScraper("BNB", "01-06-2016", "01-06-2021")
BNB=BNB.get_dataframe()
BNB['Spread Percentage']=(BNB['High']-BNB['Low']/BNB['Close'])
BNB['Turnover Ratio']=BNB['Volume']/BNB['Market Cap']
USDT=CmcScraper("USDT", "01-06-2016", "01-06-2021")
USDT=USDT.get_dataframe()
USDT['Spread Percentage']=(USDT['High']-USDT['Low']/USDT['Close'])
USDT['Turnover Ratio']=USDT['Volume']/USDT['Market Cap']
XRP=CmcScraper("XRP", "01-06-2016", "01-06-2021")
XRP=XRP.get_dataframe()
XRP['Spread Percentage']=(XRP['High']-XRP['Low']/XRP['Close'])
XRP['Turnover Ratio']=XRP['Volume']/XRP['Market Cap']

Spread_Percentage_list=[BTC['Spread Percentage'].mean(),ETH['Spread
Percentage'].mean(),USDT['Spread Percentage'].mean(),SOL['Spread
Percentage'].mean(),USDC['Spread Percentage'].mean(),ADA['Spread
Percentage'].mean(),BNB['Spread Percentage'].mean(),XRP['Spread
```

```
Percentage'].mean(),LUNA['Spread Percentage'].mean(),DOT['Spread
Percentage'].mean()]

turn_overratio_list=[BTC['Turnover Ratio'].mean(),ETH['Turnover
Ratio'].mean(),USDT['Turnover Ratio'].mean(),SOL['Turnover
Ratio'].mean(),USDC['Turnover Ratio'].mean(),ADA['Turnover
Ratio'].mean(),BNB['Turnover Ratio'].mean(),XRP['Turnover
Ratio'].mean(),LUNA['Turnover Ratio'].mean(),DOT['Turnover Ratio'].mean()]

data_correlation = {'Cryptocurrency':crypto_list,'Spread Percentage':
Spread_Percentage_list,'Turnover Ratio':turn_overratio_list}
coin_rate = pd.DataFrame(data_correlation,columns=['Cryptocurrency','Spread
Percentage','Turnover Ratio'])
coin_rate

import seaborn as sns

#Correlation Matrix
data_correlation = {'BTC':BTC['Close'],'ETH': ETH['Close'],'USDT':
USDT['Close'],'USDC' : USDC['Close'],'ADA'
:ADA['Close'],'BNB':BNB['Close'],'XRP':XRP['Close'],'LUNA':LUNA['Close'],'DOT':DO
T['Close'],'SOL':SOL['Close']}

df =
pd.DataFrame(data_correlation,columns=['BTC','ETH','USDT','USDC','ADA','BNB','XRP'
,'LUNA','DOT','SOL'])

corrMatrix = df.corr()

mask = np.triu(np.ones_like(corrMatrix, dtype=bool))
sns.heatmap(corrMatrix, annot=True, vmax=1, vmin=-1, center=0, cmap='vlag',
mask=mask)
plt.show()
```

```
stocks = yf.download('^GSPC', start='2016-06-01', end='2021-06-01')
bonds=yf.download('VUN.TO',  start='2016-06-01', end='2021-06-01')

import yfinance as yf

cumulative_return_bonds=(bonds['Adj Close'].iloc[-1]-bonds['Adj
Close'].iloc[0])/bonds['Close'].iloc[0]
annualized_return_bonds=pow((1+cumulative_return_bonds),365/(bonds.index[-1]-
bonds.index[0]).days)-1
cumulative_return_stocks=(stocks['Close'].iloc[-1]-
stocks['Close'].iloc[0])/stocks['Close'].iloc[0]
annualized_return_stocks=pow((1+cumulative_return_stocks),365/(stocks.index[-1]-
stocks.index[0]).days)-1
annualized_stdev_stocks=(stocks['Close'].pct_change().iloc[1:].std())*sqrt((stocks.index[-
1]-stocks.index[0]).days)
annualized_stdev_bonds=(bonds['Close'].pct_change().iloc[1:].std())*sqrt((bonds.index[-
1]-bonds.index[0]).days)
annualized_sharpe_ratio_bonds=annualized_return_bonds/annualized_stdev_bonds
annualized_sharpe_ratio_stocks=annualized_return_stocks/annualized_stdev_stocks
annualized_sharpe_ratio_stocks
i = np.argmax(np.maximum.accumulate(bonds['Close']) - bonds['Close']) # end of the
period
j = np.argmax(bonds['Close'][:i]) # start of period
drawdown_bonds=abs((bonds['Close'][i]-bonds['Close'][j])/bonds['Close'][j])
i = np.argmax(np.maximum.accumulate(stocks['Close']) - stocks['Close']) # end of the
period
j = np.argmax(stocks['Close'][:i]) # start of period
drawdown_stocks=abs((stocks['Close'][i]-stocks['Close'][j])/stocks['Close'][j])


index_list=['Risk Return Measurements','Stocks','Bonds']
```

```
data = {'Risk Return Measurements':['Annualized Return','Annualized Standard
Deviation','Annualized Sharpe Ratio','Maximum DD'],'Stocks':
[annualized_return_stocks,annualized_stdev_stocks,annualized_sharpe_ratio_stocks,drawd
own_stocks],'Bonds':
[annualized_return_bonds,annualized_stdev_bonds,annualized_sharpe_ratio_bonds,drawd
own_bonds]}

risk_return_df = pd.DataFrame(data,columns=index_list)

risk_return_df



daily_std_stocks=stocks['Close'].pct_change().iloc[1:].std()
daily_std_bonds=bonds['Close'].pct_change().iloc[1:].std()
daily_return_stocks=annualized_return_stocks/250
daily_return_bonds=annualized_return_bonds/250
y=[daily_return_stocks,daily_return_bonds]
x=[daily_std_stocks,daily_std_bonds]

plt.scatter(x,y,c='coral')

# naming the x axis
plt.xlabel('Daily Standard Deviation - axis')
# naming the y axis
plt.ylabel('Daily Return - axis')
plt.style.use('seaborn')



from google.colab import files
upload=files.upload()
(bonds.index[-1]-bonds.index[0]).days/365
```

```
dowjones_realestate=pd.read_csv(io.BytesIO(upload['Dow Jones Real Estate Historical Data.csv']))
dowjones_realestate


upload=files.upload()
fx=pd.read_csv(io.BytesIO(upload['Dow Jones FXCM Dollar Historical Data.csv']))


import matplotlib.pyplot as plt
import seaborn as sns
from matplotlib import style
from matplotlib.pyplot import plot

# use higher resolution for plots
%matplotlib inline
%config InlineBackend.figure_format = "retina"

# scipy for computing skewness and kurtosis
import scipy.stats as stats


mean_list=[stocks['Close'].pct_change().iloc[1:].mean(),bonds['Close'].pct_change().iloc[1:].mean()]
sd_list=[stocks['Close'].pct_change().iloc[1:].std(),bonds['Close'].pct_change().iloc[1:].std()]
median_list=[stocks['Close'].pct_change().iloc[1:].median(),bonds['Close'].pct_change().iloc[1:].median()]
```

```
mad_list=[stocks['Close'].pct_change().iloc[1:].mad(),bonds['Close'].pct_change().iloc[1:].
mad()]
maximum_list=[stocks['Close'].pct_change().iloc[1:].max(),bonds['Close'].pct_change().ilo
c[1:].max()]
minimum_list=[stocks['Close'].pct_change().iloc[1:].min(),bonds['Close'].pct_change().iloc
[1:].min()]
range_list=[maximum_list[0]-minimum_list[0],maximum_list[1]-minimum_list[1]]
skew_list=[stats.skew(stocks['Close'].pct_change().iloc[1:]),stats.skew(bonds['Close'].pct_
change().iloc[1:])]
Kurtosis_list=[stats.kurtosis(stocks['Close'].pct_change().iloc[1:]),stats.kurtosis(bonds['Clo
se'].pct_change().iloc[1:])]
data = {'Asset Class':['Stocks','Bonds'],'Mean':mean_list,'SD' : sd_list,
'Median':median_list,'MAD':mad_list,'Maximum':maximum_list,'Minimum':minimum_list,
'Range':range_list,'skew':skew_list,'Kurtosis':Kurtosis_list}
asset_class_descriptive = pd.DataFrame(data,columns=['Asset
Class','Mean','SD','Median','MAD','Maximum','Minimum','Range','skew','Kurtosis'])


asset_return_df=pd.DataFrame()
asset_return_df['Return_Stocks']=stocks['Close'].pct_change().iloc[1:]
asset_return_df['Return_Bonds']=bonds['Close'].pct_change().iloc[1:]
asset_return_corrMatrix = asset_return_df.corr()
print (asset_return_corrMatrix)
mask = np.triu(np.ones_like(asset_return_corrMatrix, dtype=bool))
sns.heatmap(asset_return_corrMatrix, annot=True, vmax=1, vmin=-1, center=0,
cmap='vlag', mask=mask)
plt.show()


# use the function regplot to make a scatterplot
sns.regplot(x=asset_return_df['Return_Stocks'], y=asset_return_df['Return_Bonds'])
```

```
def random_portfolios(num_portfolios, mean_returns, cov_matrix, risk_free_rate):
    results = np.zeros((3,num_portfolios))
    weights_record = []
    for i in xrange(num_portfolios):
        weights = np.random.random(4)
        weights /= np.sum(weights)
        weights_record.append(weights)
        portfolio_std_dev, portfolio_return = portfolio_annualised_performance(weights,
mean_returns, cov_matrix)
        results[0,i] = portfolio_std_dev
        results[1,i] = portfolio_return
        results[2,i] = (portfolio_return - risk_free_rate) / portfolio_std_dev
    return results, weights_record


#create function for easy plotting
def multiPlot(asset):
    # distribution plot using simple returns
    fig, ax = plt.subplots(nrows = 1, ncols = 2, figsize = (16, 6))
    sns.histplot(asset.pct_change().iloc[1:], ax = ax[0], bins = 30, color = "lightcyan", stat =
"density")
    sns.kdeplot(asset.pct_change().iloc[1:], ax = ax[0], color = "blue", linestyle = "--", label
= "Kernel Density")
    ax[0].set_xlim(-0.3, 0.3)
    ax[0].set_xlabel("Simple Returns")
    ax[0].set_title("Histogram of simple returns")

    # compute normal distribution for comparison with asset distributions
    mean, std = stats.norm.fit(asset.pct_change().iloc[1:])
    x = np.linspace(-0.3, 0.3, 300)
    p = stats.norm.pdf(x, mean, std)
```

```
    ax[0].plot(x, p, "r", linewidth = 2, label = "Normal")
    ax[0].legend()

    # probability plot
    stats.probplot(asset.pct_change().iloc[1:], dist = "norm", rvalue = True, fit = True, plot =
plt)
    plt.xlabel("Normal Distribution")
    plt.ylabel("Sample Distribution")


bonds['Daily Return'] = bonds['Adj Close'].pct_change(1)
stocks['Daily Return'] = stocks['Adj Close'].pct_change(1)


from scipy import stats


port_weights=[]
port_returns=[]
port_volatility=[]


def sortino(returns, rf=0, periods=252, annualize=True, smart=False):

    if rf != 0 and periods is None:
        raise Exception('Must provide periods if rf != 0')

    returns = _utils._prepare_returns(returns, rf, periods)

    downside = _np.sqrt((returns[returns < 0] ** 2).sum() / len(returns))

    if smart:
```

```
        # penalize sortino with auto correlation
        downside = downside * autocorr_penalty(returns)

    res = returns.mean() / downside

    if annualize:
        return res * _np.sqrt(
            1 if periods is None else periods)

    return res


index_returns_annual = [annualized_return_stocks,annualized_return_bonds]


# get daily and covariance of returns of the stock
cov_index_returns_annual  = np.cov(index_returns_annual)

# empty lists to store returns, volatility and weights of portfolios
port_returns = []
port_volatility = []
stock_weights = []

# set the number of combinations for imaginary portfolios
num_assets = len(index_returns_annual)
num_portfolios = 50000

cov_index_returns_annual


def create_results_dataframe(portfolio, number_portfolios,asset_yearly_returns,
cov_matrix):
```

```
    results_temp = np.zeros((4 + len(portfolio) - 1, number_portfolios))

    for i in range(number_portfolios):
        # select random weights for portfolio holdings
        weights = np.array(np.random.random(2))

        # rebalance weights to sum to 1
        weights /= np.sum(weights)

        # calculate portfolio return and volatility
        portfolio_return = np.sum(asset_yearly_returns * weights)
        portfolio_std_dev = np.sqrt(np.dot(weights.T, np.dot(cov_matrix, weights)))

        risk_free_return = 0

        vol=(weights.T @ cov_matrix @ weights)

        # store results in results array
        results_temp[0, i] = portfolio_return
        results_temp[1, i] = portfolio_std_dev

        # store Sharpe Ratio (return / volatility) - risk free rate element excluded for
simplicity
        results_temp[2, i] = (results_temp[0, i] - risk_free_return) / results_temp[1, i]



        # iterate through the weight vector and add data to results array
        for j in range(len(weights)):
            results_temp[j + 3, i] = weights[j]

       # convert results array to Pandas DataFrame
    results_df = pd.DataFrame(results_temp.T, columns=['ret', 'stdev', 'sharpe', portfolio[0],
```

```
                                        portfolio[1]])

    return results_df


portfolio=['Stocks','Bonds']
number_portfolios=1000
asset_yearly_returns=[annualized_return_stocks,annualized_return_bonds]


asset_yearly_returns=[annualized_return_stocks,annualized_return_bonds]
asset_yearly_returns


portfolio_index=create_results_dataframe(['Stocks','Bonds'], 10000,asset_yearly_returns,
asset_return_corrMatrix )


def max_sharpe_ratio(results_df):
    """locate portfolio with highest Sharpe Ratio"""
    return results_df.iloc[results_df['sharpe'].idxmax()]

def min_volatility(results_df):
    """locate portfolio with lowest volatility"""
    return results_df.iloc[results_df['stdev'].idxmin()]


crypto_min_vol_portfolio = min_volatility(portfolio_index)
print(crypto_min_vol_portfolio)
crypto_max_sharpe_portfolio = max_sharpe_ratio(portfolio_index)
print(crypto_max_sharpe_portfolio)
```

```
def plot_graph(results_df, max_sharpe_port, min_vol_port):
    ax = results_df.plot(kind= 'scatter', x = 'stdev', y='ret', s = 30,
                         c=results_df.sharpe, cmap='RdYlBu',edgecolors='.1', figsize=(20,10))
    ax.grid(False, color='w', linestyle='-', linewidth=1)
    ax.set_facecolor('1')
    ax.set_xlabel('Volatility')
    ax.set_ylabel('Returns')
    ax.tick_params(labelsize = 14)

    # # plot red star to highlight position of portfolio with highest Sharpe Ratio
    ax.scatter(max_sharpe_port[1], max_sharpe_port[0], marker=(5, 1, 0), color='r', s=1000)
    # # plot green star to highlight position of minimum variance portfolio
    ax.scatter(min_vol_port[1], min_vol_port[0], marker=(5, 1, 0), color='g', s=1000)


plot_graph(portfolio_index, crypto_max_sharpe_portfolio, crypto_min_vol_portfolio)

plt.show()


#Portfolio with inclusion of CryptoCurrency


def create_sortino_ratio(returns, periods=252):
    """
    Create the Sortino ratio for the strategy, based on a
    benchmark of zero (i.e. no risk-free rate information).
    Parameters:
    returns - A pandas Series representing period percentage returns.
    periods - Daily (252), Hourly (252*6.5), Minutely(252*6.5*60) etc.
    """
```

```
    return np.sqrt(periods) * (np.mean(returns)) / np.std(returns[returns < 0])


def create_results_dataframe_sortino(portfolio, number_portfolios,asset_yearly_returns,
cov_matrix):
    results_temp = np.zeros((4 + len(portfolio) - 1, number_portfolios))

    for i in range(number_portfolios):
        # select random weights for portfolio holdings
        weights = np.array(np.random.random(2))

        # rebalance weights to sum to 1
        weights /= np.sum(weights)

        # calculate portfolio return and volatility
        portfolio_return = np.sum(asset_yearly_returns * weights)
        portfolio_std_dev = np.sqrt(np.dot(weights.T, np.dot(cov_matrix, weights)))

        risk_free_return = 0

        vol=(weights.T @ cov_matrix @ weights)

        # store results in results array
        results_temp[0, i] = portfolio_return
        results_temp[1, i] = portfolio_std_dev

        # store Sharpe Ratio (return / volatility) - risk free rate element excluded for
simplicity

        results_temp[2,i]= create_sortino_ratio(portfolio_return)

        # iterate through the weight vector and add data to results array
```

```
        for j in range(len(weights)):
            results_temp[j + 3, i] = weights[j]

        # convert results array to Pandas DataFrame
        results_df = pd.DataFrame(results_temp.T, columns=['ret', 'stdev', 'sharpe', portfolio[0],
                                                           portfolio[1]])

        return results_df


weights = np.array(np.random.random(2))


portfolio_index=create_results_dataframe_sortino(['Stocks','Bonds'],
10000,asset_yearly_returns, asset_return_corrMatrix )


/usr/local/lib/python3.7/dist-packages/numpy/core/_methods.py:234: RuntimeWarning:
Degrees of freedom <= 0 for slice
  keepdims=keepdims)
/usr/local/lib/python3.7/dist-packages/numpy/core/_methods.py:195: RuntimeWarning:
invalid value encountered in true_divide
  arrmean, rcount, out=arrmean, casting='unsafe', subok=False)
/usr/local/lib/python3.7/dist-packages/numpy/core/_methods.py:226: RuntimeWarning:
invalid value encountered in double_scalars
  ret = ret.dtype.type(ret / rcount)


asset_yearly_returns
portfolio_return = np.sum(asset_yearly_returns * weights)
portfolio_return
```

APPENDIX 2: Crypto Prediction

Bitcoin Script

```
!pip install cryptocmd

import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import matplotlib.dates as mdates
from keras.models import Sequential
from keras.layers import Dense, Dropout,GRU
from keras import optimizers
import io
from sklearn.preprocessing import MinMaxScaler
from cryptocmd import CmcScraper
from google.colab import files
seed = 1234
np.random.seed(seed)
plt.style.use('ggplot')

start_date="01-01-2013"
end_date="01-01-2022"

pip install yfinance

import warnings
warnings.simplefilter(action='ignore', category=FutureWarning)

uploaded = files.upload()
dataraw=pd.read_csv(io.BytesIO(uploaded'BTC_USDSAMPLE1.csv')),index_col='Date',
parse_dates=['Date'])

uploaded = files.upload()
```

```
ETH=pd.read_csv(io.BytesIO(uploaded.get('ETH-USD.csv')),index_col='Date',
parse_dates=['Date'])

# use feature 'Date' & 'Close'
dataset = pd.DataFrame(dataraw['Close'])
print(' Count row of data: ',len(dataset))
fig = plt.figure(figsize=(20, 10))
plt.plot(dataset,color="blue")
plt.xlabel('Date')
plt.ylabel('Bitcoin Price')
plt.gca().xaxis.set_major_formatter(mdates.DateFormatter("%Y-%m-%d"))
plt.title('Bitcoin Price')
plt.show()

# use feature 'Date' & 'Close'
ETH = pd.DataFrame(ETH['Close'])
print(' Count row of data: ',len(ETH))
fig = plt.figure(figsize=(20, 10))
plt.plot(ETH,color="blue")
plt.xlabel('Date')
plt.ylabel('ETHERUM Price')
plt.gca().xaxis.set_major_formatter(mdates.DateFormatter("%Y-%m-%d"))
plt.title('ETHERUM Price')
plt.show()

#Min-Max Normalization
ETH_norm = ETH.copy()
ETH['Close']
scaler = MinMaxScaler()
ETH_norm['Close'] = scaler.fit_transform(ETH[['Close']])
ETH_norm = ETH.copy()
ETH['Close']
```

```
scaler = MinMaxScaler()
ETH_norm['Close'] = scaler.fit_transform(ETH[['Close']])
ETH_norm

fig = plt.figure(figsize=(10, 4))
plt.plot(dataset_norm,color='green')
plt.xlabel('Date')
plt.gca().xaxis.set_major_formatter(mdates.DateFormatter("%Y-%m"))
plt.title('Bitcoin Data Normalized')
plt.show()
fig = plt.figure(figsize=(10, 4))
plt.plot(ETH_norm,color='green')
plt.xlabel('Date')
plt.gca().xaxis.set_major_formatter(mdates.DateFormatter("%Y-%m"))
plt.title('ETH DATA Normalized')
plt.show()

# Partition data into data train, val & test
totaldata = dataset.values
totaldatatrain = int(len(totaldata)*0.8)
totaldataval = int(len(totaldata)*0.1)
totaldatatest = int(len(totaldata)*0.10)

# Store data into each partition
training_set = dataset_norm[0:totaldatatrain]
val_set=dataset_norm[totaldatatrain:totaldatatrain+totaldataval]
test_set = dataset_norm[totaldatatrain+totaldataval:]

# use feature 'Date' & 'Close'
dataset = pd.DataFrame(dataraw['Close'])
print(' Count row of data: ',len(dataset))
```

```
fig = plt.figure(figsize=(14, 6))
plt.plot(dataset)
plt.xlabel('Date')
plt.ylabel('Bitcoin Price')
plt.gca().xaxis.set_major_formatter(mdates.DateFormatter("%Y-%m"))
plt.title('Bitcoin Price')
plt.show()

# Initiaton value of lag
lag = 2
# sliding windows function
def create_sliding_windows(data,len_data,lag):
    x=[]
    y=[]
    for i in range(lag,len_data):
        x.append(data[i-lag:i,0])
        y.append(data[i,0])
    return np.array(x),np.array(y)

# Formating data into array for create sliding windows
array_training_set = np.array(training_set)
array_val_set = np.array(val_set)
array_test_set = np.array(test_set)

# Create sliding windows into training data
x_train, y_train = create_sliding_windows(array_training_set,len(array_training_set), lag)
x_train = np.reshape(x_train, (x_train.shape[0],x_train.shape[1],1))
# Create sliding windows into validation data
x_val,y_val = create_sliding_windows(array_val_set,len(array_val_set),lag)
x_val = np.reshape(x_val, (x_val.shape[0],x_val.shape[1],1))
# Create sliding windows into test data
x_test,y_test = create_sliding_windows(array_test_set,len(array_test_set),lag)
```

```
x_test = np.reshape(x_test, (x_test.shape[0],x_test.shape[1],1))

# Hyperparameters
learning_rate = 0.0001
hidden_unit = 64
batch_size=500
epoch = 1000

# Architecture Gated Recurrent Unit
regressorGRU = Sequential()

# First GRU layer with dropout
regressorGRU.add(GRU(units=hidden_unit, return_sequences=True,
input_shape=(x_train.shape[1],1), activation = 'tanh'))
regressorGRU.add(Dropout(0.1))
# Second GRU layer with dropout
regressorGRU.add(GRU(units=hidden_unit, return_sequences=True, activation = 'tanh'))
regressorGRU.add(Dropout(0.2))
# Third GRU layer with dropout
regressorGRU.add(GRU(units=hidden_unit, return_sequences=False, activation = 'tanh'))
regressorGRU.add(Dropout(0.3))

# Output layer
regressorGRU.add(Dense(units=1))

# Compiling the Gated Recurrent Unit
regressorGRU.compile(optimizer=tensorflow.keras.optimizers.Adam(lr=learning_rate),los
s='mean_squared_error')

# Fitting ke data training dan data validation
pred = regressorGRU.fit(x_train, y_train, validation_data=(x_val,y_val),
batch_size=batch_size, epochs=epoch)
```

```
# Graph model loss (train loss & val loss)
fig = plt.figure(figsize=(10, 4))
plt.plot(pred.history['loss'], label='train loss')
plt.plot(pred.history['val_loss'], label='val loss')
plt.title('model loss')
plt.ylabel('loss')
plt.xlabel('epoch')
plt.legend(loc='upper right')
plt.show()

# Tabel value of training loss & validation loss
learningrate_parameter = learning_rate
train_loss=pred.history['loss'][-1]
validation_loss=pred.history['val_loss'][-1]
learningrate_parameter=pd.DataFrame(data=[[learningrate_parameter, train_loss,
validation_loss]],
                                    columns=['Learning Rate', 'Training Loss', 'Validation Loss'])
learningrate_parameter.set_index('Learning Rate')

import tensorflow

from tensorflow import keras

# Partition data into data train, val & test
totaldata = dataset.values
totaldatatrain = int(len(totaldata)*0.8)
totaldataval = int(len(totaldata)*0.1)
totaldatatest = int(len(totaldata)*0.1)

# Store data into each partition
training_set = dataset_norm[0:totaldatatrain]
```

```
val_set=dataset_norm[totaldatatrain:totaldatatrain+totaldataval]
test_set = dataset_norm[totaldatatrain+totaldataval:]

# graph of data validation
fig = plt.figure(figsize=(10, 4))
plt.plot(val_set)
plt.xlabel('Date')
plt.gca().xaxis.set_major_formatter(mdates.DateFormatter("%Y-%m"))
plt.title('Data Validation')
val_set

fig = plt.figure(figsize=(10, 4))
plt.plot(dataset_norm)
plt.xlabel('Date')
plt.gca().xaxis.set_major_formatter(mdates.DateFormatter("%Y-%m"))
plt.title('Data Normalized')
plt.show()

# Tabel value of training loss & validation loss
learningrate_parameter = learning_rate
train_loss=pred.history['loss'][-1]
validation_loss=pred.history['val_loss'][-1]
learningrate_parameter=pd.DataFrame(data=[[learningrate_parameter, train_loss,
validation_loss]],
                                    columns=['Learning Rate', 'Training Loss', 'Validation Loss'])
learningrate_parameter.set_index('Learning Rate')

# Implementation model into data test
y_pred_test = regressorGRU.predict(x_test)

# Invert normalization min-max
y_pred_invert_norm = scaler.inverse_transform(y_pred_test)
```

```
# Comparison data test with data prediction
datacompare = pd.DataFrame()
datatest=np.array(dataset['Close'][totaldatatrain+totaldataval+lag:])
datapred= y_pred_invert_norm

datacompare['Data Test'] = datatest
#datacompare['Prediction Results'] = datapred

datacompare
datapred.shape

# Calculatre value of Root Mean Square Error
def rmse(datatest, datapred):
    return np.round(np.sqrt(np.mean((datapred - datatest) ** 2)), 4)
print('Result Root Mean Square Error Prediction Model :',rmse(datatest, datapred))

def mape(datatest, datapred):
    return np.round(np.mean(np.abs((datatest - datapred) / datatest) * 100), 4)

print('Result Mean Absolute Percentage Error Prediction Model : ', mape(datatest,
datapred), '%')

# Comparison data test with data prediction
datacompare = pd.DataFrame()
datatest=np.array(dataset['Close'][totaldatatrain+totaldataval+lag:])
datapred= y_pred_invert_norm

datacompare['Data Test'] = datatest
datacompare['Prediction Results'] = datapred
datacompare
```

```
# Create graph data test and prediction result
plt.figure(num=None, figsize=(10, 4), dpi=100,facecolor='w', edgecolor='k')
plt.title('Graph Comparison Data Actual and Data Prediction')
plt.plot(datacompare['Data Test'], color='red',label='Data Test')
plt.plot(datacompare['Prediction Results'], color='blue',label='Prediction Results')
plt.xlabel('Day')
plt.ylabel('Price')
plt.legend()
plt.show
```

Ethereum Script

```
!pip install cryptocmd

import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import matplotlib.dates as mdates
from keras.models import Sequential
from keras.layers import Dense, Dropout,GRU
from keras import optimizers
import io
from sklearn.preprocessing import MinMaxScaler
```

```
from cryptocmd import CmcScraper
from google.colab import files
seed = 1234
np.random.seed(seed)
plt.style.use('ggplot')

import tensorflow
from tensorflow import keras

import warnings
warnings.simplefilter(action='ignore', category=FutureWarning)

uploaded = files.upload()
ETH=pd.read_csv(io.BytesIO(uploaded.get('ETH-USD.csv')),index_col='Date',
parse_dates=['Date'])

# use feature 'Date' & 'Close'// Plotting Close Price
ETH = pd.DataFrame(ETH['Close'])
fig = plt.figure(figsize=(30, 20))
plt.plot(ETH,color="red")
plt.xlabel('Date')
plt.ylabel('ETHERUM Price')
plt.gca().xaxis.set_major_formatter(mdates.DateFormatter("%Y-%m-%d"))
plt.title('ETHERUM Price')
plt.show()

#Min-Max Normalization
ETH_norm = ETH.copy()
ETH['Close']
scaler = MinMaxScaler()
ETH_norm['Close'] = scaler.fit_transform(ETH[['Close']])
ETH_norm = ETH.copy()
```

```
ETH['Close']
scaler = MinMaxScaler()
ETH_norm['Close'] = scaler.fit_transform(ETH[['Close']])
fig = plt.figure(figsize=(10, 4))
plt.plot(ETH_norm,color='green')
plt.xlabel('Date')
plt.gca().xaxis.set_major_formatter(mdates.DateFormatter("%Y-%m"))
plt.title('ETH DATA Normalized')
plt.show()

# Partition data into data train, val & test
totaldata = ETH.values
totaldatatrain = int(len(totaldata)*0.80)
totaldataval = int(len(totaldata)*0.10)
totaldatatest = int(len(totaldata)*0.10)

# Store data into each partition
training_set = ETH_norm[0:totaldatatrain]
val_set=ETH_norm[totaldatatrain:totaldatatrain+totaldataval]
test_set = ETH_norm[totaldatatrain+totaldataval:]

# Initiaton value of lag
lag = 2
# sliding windows function
def create_sliding_windows(data,len_data,lag):
    x=[]
    y=[]
    for i in range(lag,len_data):
        x.append(data[i-lag:i,0])
        y.append(data[i,0])
    return np.array(x),np.array(y)
```

```
# Formating data into array for create sliding windows
array_training_set = np.array(training_set)
array_val_set = np.array(val_set)
array_test_set = np.array(test_set)

# Create sliding windows into training data
x_train, y_train = create_sliding_windows(array_training_set,len(array_training_set), lag)
x_train = np.reshape(x_train, (x_train.shape[0],x_train.shape[1],1))
# Create sliding windows into validation data
x_val,y_val = create_sliding_windows(array_val_set,len(array_val_set),lag)
x_val = np.reshape(x_val, (x_val.shape[0],x_val.shape[1],1))
# Create sliding windows into test data
x_test,y_test = create_sliding_windows(array_test_set,len(array_test_set),lag)
x_test = np.reshape(x_test, (x_test.shape[0],x_test.shape[1],1))

# Hyperparameters
learning_rate = 0.0001
hidden_unit = 64
batch_size=500
epoch = 1000

# Architecture Gated Recurrent Unit
regressorGRU = Sequential()

# First GRU layer with dropout
regressorGRU.add(GRU(units=hidden_unit, return_sequences=True,
input_shape=(x_train.shape[1],1), activation = 'tanh'))
regressorGRU.add(Dropout(0.1))
# Second GRU layer with dropout
regressorGRU.add(GRU(units=hidden_unit, return_sequences=True, activation = 'tanh'))
regressorGRU.add(Dropout(0.2))
# Third GRU layer with dropout
```

```
regressorGRU.add(GRU(units=hidden_unit, return_sequences=False, activation = 'tanh'))
regressorGRU.add(Dropout(0.3))

# Output layer
regressorGRU.add(Dense(units=1))

# Compiling the Gated Recurrent Unit
regressorGRU.compile(optimizer=tensorflow.keras.optimizers.Adam(lr=learning_rate),los
s='mean_squared_error')

# Fitting ke data training dan data validation
pred = regressorGRU.fit(x_train, y_train, validation_data=(x_val,y_val),
batch_size=batch_size, epochs=epoch)

# Graph model loss (train loss & val loss)
fig = plt.figure(figsize=(10, 4))
plt.plot(pred.history['loss'], label='train loss')
plt.plot(pred.history['val_loss'], label='val loss')
plt.title('model loss')
plt.ylabel('loss')
plt.xlabel('epoch')
plt.legend(loc='upper right')
plt.show()

# Implementation model into data test
y_pred_test = regressorGRU.predict(x_test)

# Invert normalization min-max
y_pred_invert_norm = scaler.inverse_transform(y_pred_test)

# Comparison data test with data prediction
datacompare = pd.DataFrame()
```

```
datatest=np.array(ETH['Close'][totaldatatrain+totaldataval+lag:])
datapred= y_pred_invert_norm

datacompare['Data Test'] = datatest
datacompare['Prediction Results'] = datapred
# Calculatre value of Root Mean Square Error
def rmse(datatest, datapred):
    return np.round(np.sqrt(np.mean((datapred - datatest) ** 2)), 4)
print('Result Root Mean Square Error Prediction Model :',rmse(datatest, datapred))

def mape(datatest, datapred):
    return np.round(np.mean(np.abs((datatest - datapred) / datatest) * 100), 4)

print('Result Mean Absolute Percentage Error Prediction Model : ', mape(datatest,
datapred), '%')

# Create graph data test and prediction result
plt.figure(num=None, figsize=(10, 4), dpi=100,facecolor='w', edgecolor='k')
plt.title('Graph Comparison Data Actual and Data Prediction')
plt.plot(datacompare['Data Test'], color='red',label='Data Test')
plt.plot(datacompare['Prediction Results'], color='blue',label='Prediction Results')
plt.xlabel('Day')
plt.ylabel('Price')
plt.legend()
plt.show
```