A Quantamental approach to Bitcoin trading: Are we swinging for the Fences?

Student: Agustin Alba Chicar ag.albachicar@gmail.com

Director: Pablo Roccatagliata proccatagliata@gmail.com

A thesis presented for the degree of
Master in Management and Analytics

Universidad Torcuato Di Tella
Business school
Argentina
May 2021

Contents

1 Abstract **3**

2 Resumen **4**

3 Introduction **5**
3.1 Problem description 5
3.2 Domain 7
3.2.1 Fundamental trading strategies 7
3.2.2 Cryptocurrencies 8
3.2.3 Financial machine learning 11
3.3 Scope 14

4 Resources **15**
4.1 Data 15
4.1.1 Bitcoin price data 15
4.1.2 Bitcoin features 16
4.1.3 Social features 19
4.2 Software and tooling 19

5 Methods **21**
5.1 Features 21
5.1.1 Fundamental features 24
5.1.2 Domain specific features 30
5.1.3 Supremum Augmented Dickey Fuller 43
5.1.4 Social features 47
5.2 Pipeline 49
5.2.1 Primary model 50
5.2.2 Secondary model 52
5.2.3 Training 54
5.2.4 Feature importance 57
5.2.5 Bet sizing 59
5.2.6 Back testing 59

6 Results **66**
6.1 Model results 66
6.2 Strategy results 70

7 Conclusion **74**

8 Future work **76**

9 Appendix **78**
9.1 Code 78

Bibliography **80**

1 Abstract

Financial companies from all around the world have started to focus their investments in quantitative and algorithmic funds. Those methods run in server applications that execute automatic trades. It is important to distinguish high frequency trading from machine learning trading. The latter is used and analyzed in detail in the present work.

This project explains the development of a trading strategy on Bitcoins based on machine learning techniques. A pipeline proposal is shown which is based on Lopez de Prado's book ([Pra18]). Some modifications are introduced in the book's pipeline to adjust a momentum primary model on Bitcoins, and to incorporate and study features that would let estimate the size of the primary model bets (secondary model to be trained on top of the first model). The range of features to analyze goes from financial metrics derived from Bitcoin prices and volumes, to Bitcoin and blockchain related features and finally social indexes which incorporate interest and animosity towards Bitcoin itself.

The pipeline proposed in [Pra18] and implemented in this thesis rigorously handles the dataset, the involved models and finally the posterior backtesting strategies. Details about statistical foundation of the involved methods, algorithm complexity and implementation and domain explanations (such as those related to cryptocurrencies) can be found. The pipeline allows to gather enough information to compare and decide whether a propose strategy is good enough to be implemented. We will use this to compare models that introduce microstructure indexes such as SADF (Supremum Augmented Dickey Fuller) in comparison and conjunction with social indexes.

2 Resumen

Empresas financieras en el mundo comenzaron hace ya algunos años a focalizar sus esfuerzos e inversiones en fondos basados en métodos cuantitativos implementados con algoritmos y corriendo en servidores que ejecutan compras y ventas automáticamente. Distinguimos el trading de alta frecuencia del trading basado en aprendizaje de máquina, el cual será objeto de estudio en este trabajo.

En particular, el presente trabajo muestra el desarrollo de una estrategia de compraventa de Bitcoins basada en técnicas de aprendizaje automático. Se propone una implementación del proceso presentado en el libro de Lopez de Prado [Pra18]. Al proceso anterior se lo modifica para poder incorporar un modelo de momentum sobre Bitcoins y se realiza un estudio sobre los features que permitirán mejorar un modelo secundario (a aprender) para dimensionar los tamaños de las posiciones que la estrategia fundamental (de momentum) proponga. Los features a analizar y comparar van desde métricas financieras del activo subyacente, métricas propias de la tecnología de blockchain y Bitcoin como criptoactivo para terminar con métricas sociales que den información sobre interés y animosidad.

El proceso planteado en [Pra18] e implementado en este trabajo busca hacer un cuidado riguroso del dataset y los modelos empleados así como posterior backtesting de la estrategia. Este informe detalla los detalles de la implementación tanto a nivel estadístico, algorítmico y de dominio (criptoactivos). A partir de los resultados obtenidos en cada etapa del proceso y finalmente en backtesting podremos analizar el proceso de generación de estrategias sino que también comparar algunas para entender el proceso de selección. En particular, es de relevancia en este trabajo utilizar en contraste índices de microestructura como SADF (Supremum Augmented Dickey Fuller) en comparación y conjunto con los índices sociales.

3 Introduction

3.1 Problem description

In the last decades, the notorious technology improvements enabled several industries to grow exponentially. Specifically, in the finance industry we can identify a more recent trend that involves considerable investment efforts in the quantitative and algorithmic trading (see [Comb] to learn more about the BlackRock case). It is interesting for this research to focus initially on two types of solutions from the above group: high frequency trading (see [BW12]), known as HFT, and machine learning for trading [DHB20]. The first group exploits the computational power to arbitrate between securities at high frequencies to execute transactions faster than others whereas the other exploits the vast amount of available data through data mining techniques and statistical models.

Different technological improvements enabled the spread of these techniques in the financial industry. First, having more and more data to process required more and more storage which started to cost less and less per storage unit, i.e. $/GB ([Kom] and the updated version [Kom]). Secondly, the widespread use of mobile devices supported by a more and more connected world ([ITU]) enabled faster data transfers reducing distance, information lead time, or just broadly speaking the connectivity costs. Another vertical that supported this growth was the raise in computational power which followed Moore's Law [Moo] allowed more and more complex algorithms to be run in acceptable time (with respect to the needs at hand) enabling old theoretical solutions to have a practical one. For example, Roosenblatt's perceptron dates from 1958 [Ros58] but it was not after some decades that commercial applications of those neural networks were made available.

Moreover, the digitalization process and internet access growth [MO15] allow more people to participate in markets, access information and make new decisions. The behavior of each economic agent, i.e. individual, in the market became of tremendous importance. It is not anymore a handful of people making decisions but entire societies. Behavioral economic analysis is required to better understand market fluctuations and its evolution. For example, herds and bubbles have been registered for centuries [Mac69] to nowadays with two of the most recent and bigger bubbles: the Dotcom bubble [Hay] and the US financial crisis in 2008 [Sin].

In parallel, another phenomenon occurred in 2008 when Satoshi Nakamoto released the Bitcoin paper [Nak08] and then in 2009 when the Bitcoin software release was published [Nak]. Pushed by Bitcoin, a new technology ecosystem appeared in the last decade backed up by the blockchain tech-

nology. As of February 2021, it is said that over 4,000 cryptocurrencies are available but only twenty of them concentrate the 90% of the market [Bes]. A document of the World Bank [Chi18] showed that Bitcoin transfers were used for gambling and dark web transactions but it gained attraction in 2016 to ease the process international transactions and then to finance private endeavors. Governments started to experiment with blockchain technologies and what started a decade ago as a niche experiment, it reached a market capitalization of $1,022,439,972,862 in March 9th 2021 [Coi].

Bitcoin and other cryptocurrencies are part of a broader collection named cryptoassets [BT17]. The term currency is subject to discussion but omitting the subjective appreciations in the academy in favor or against, we should also consider cryptocommodities and cryptotokens. In [BT17] there is a good introduction about these terms and thorough examples for each of them based on their used and recent attraction.

One of the main characteristics about Bitcoin is that its total emission and its emission rate has been defined by design. Every four years the incentive to miners is reduced to a half (it started in 50BTC). This event is called halving and introduces a structural change in the market because incentives suffer dramatic changes (they are cut down to a half). As it will be later explained, these events involve a high volatility in Bitcoin price as well as a change in market's regime. Strategies around this type of asset should either stay away of halvings or consider them somehow to avoid important losses. In [Pra18], structural breaks are considered to inform models with statistical tests about structural breaks in market that would yield to the "best risk / reward ratios".

This research focuses on a building a strategy development pipeline to build, train and evaluate financial trading strategies and will be exercised with Bitcoin. A primary model based on momentum will provide the main trading signals and a secondary machine learning model will provide the bet size. Financial indexes (such as price, volume and volatility), structural break indexes (such as Supremum Augmented Dickey-Fuller), Bitcoin related indexes (such as stock to flow and number of new addresses) and social animosity indexes (such as fear and greed index) will be evaluated to improve the secondary model performance. Finally, from backtesting procedures metrics will be determined to assess the strategy performance.

This document has the following outline:

- Section 3 presents the problem to solve, provides context about each involved discipline, comments about the state of the art and defines the scope of this research.
- Section 4 presents the used data sources.

- Section 5 analyzes in detail the features and describes the pipeline and successive iterations over it.
- Section 6 presents the results obtained. Pure machine learning model results are separated from financial results.
- Section 7 discusses the results and wraps the document.
- Section 8 presents some unresolved questions and potential lines of work to continue with this research effort.

3.2 Domain

This section outlines the theoretical background of each related knowledge domain involved in this research. The following list introduces each subsection:

- Subsection 3.2.1 describes the primary model which is a momentum strategy and why it was chosen.
- Subsection 3.2.2 provides background about cryptoassets, cryptocurrencies and in particular Bitcoin and its ecosystem.
- Subsection 3.2.3 introduces the methodology Lopez de Prado explains in [Pra18], answers why machine learning is applied and introduces the structure break indexes with strong focus on SADF.

As detailed in 3.1, the research problem involves many knowledge domains and data from different sources. When working in finance, models may be dynamic because markets *evolve*. Among all the available strategies to derive the primary model, momentum was chosen.

3.2.1 Fundamental trading strategies

There are many algorithmic fundamental trading strategies. We can use the classification in [GZ18]:

- Impact driven: orders placed in a market affect the price of the stocks based on their volume and the liquidity. Strategies like volume-weighted average price (VWAP) or time-weighted average price (TWAP) can be found in this group.

- Cost driven: it not only considers implicit costs as the above but also the explicit costs of the market (e.g. commissions and access fees). We can find implementation shortfall in this group.
- Newsreader: based on the semi-strong form efficiency, these algorithms exploit news feeds to derive trading signals out of non-structured data.
- Market making: this group exploits the spread in the bid-ask prices.
- Statistical arbitrage: strategies in this group focus on two main premises: an asset tends to a *mean* value in the long run (one can profit from deviations) or an asset's price is nonstationary. Mean reversion and momentum strategies belong to this group.

Momentum based strategies focus on deriving when a price starts to rise and drop to derive signals and profit by placing long and short positions on the asset. To derive these events, two different speed (fast and slow) moving average signals are used. The specific timestamps at which the fast and slow averaged signals cross determine an event. A fast signal crossing above the slow signal generates a buy event and the opposite generates a sell signal. A simple variation involves using exponential moving averages instead of simple moving averages and it increase complexity to even entire portfolios behind ETFs like MTUM ([Bla]).

In [CP13] the authors explored the performance of momentum with market information of more than 200 years and confirm the strategy generally outperforms the market. In [IM14] the strategy is demystified in favor of a better comprehension of its strong and weak points. Based on the extensive literature around this strategy in particular, implementation simplicity, compatibility with the available data (no book order available, just stamped prices and volumes) this strategy was chosen to work as primary model.

3.2.2 Cryptocurrencies

Satoshi Nakamoto published [Nak08] with the intention to kick start a decentralized peer-to-peer cash system. However, it did not only do that but also gave birth to a new technology hype which is in continuous evolution and now is heading to mature the stack of services on top of the blockchain technology ([20]).

Let me present a few useful definitions:

- Bitcoin: is the software that facilitates the transfer and custody of the bitcoin currency.

- bitcoin: a cryptocurrency.
- Blockchain: is a distributed and digital ledger, in this case of the Bitcoin software. It keeps track of all debits and credits of bitcoin applying a very clever and sophisticated fashion (see [coma] for a comprehensive description).

Bitcoin's blockchain technology provides several features:

- It is publicly distributed. There are no secrete transactions.
- Serves as a historical record of *all* transactions.
- It is immutable. Transactions will only be appended in the shape of new information blocks, but the *verified* past cannot be changed.
- It is secure, i.e. each transaction is cryptographically verified to ensure valid funds.

Transactions are verified via a *Proof-of-Work* (PoW) algorithm which requires nowadays specialized hardware to process the cryptographic algorithm in a timely and competitive manner. Timely and competitive go hand in hand because the first *miner* in claiming the block hash (result of the PoW) obtains a bitcoin reward provided by the Bitcoin software. The block hash is expensive to obtain but easy to verify what makes the acceptance of a new block a fast process for the entire network.

So far, we have introduced some of the technical characteristics of Bitcoin and supporting technologies. It is important to mention what this payment network provides to users. On one hand we have the privacy. Transactions happen from one wallet [comc] to another (cryptography comes in to properly describe how they work which is out of the scope of this document) and there is no direct nor easy way to know how is the owner of the wallet. Simply, there is information record about the owner of the wallet operation other than the transaction ledger (the blockchain itself). On the other hand, Bitcoin runs on the internet (using TCP [comb]) even on countries with network traffic control which sets the basis to trade worldwide without any regulation.

Another important aspect about bitcoin is the issuance model. Miners are paid every time they append a new block and receive a reward in bitcoins. The reward was initially set to 50 bitcoins and Bitcoin would adjust the mining complexity to get a new block every ten minutes on average. On a 210,000 blocks basis, the reward gets reduced to a half although the complexity keeps on adjusting to have the same throughput of one block every

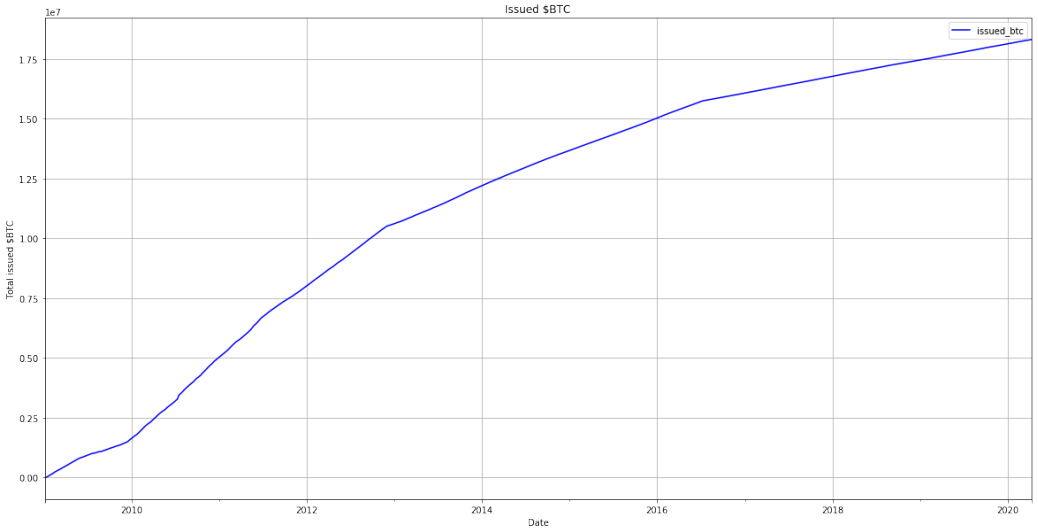

Figure 1: Issued bitcoins through time. Vertical scale is in tenths of millions of bitcoins. Glassnode data.

ten minutes. The total amount of bitcoins to be issued is 21 million and we can see in figure 1 the amount of bitcoins issued by April 2020.

And in 2 we can see the evolution of bitcoin issuance with time. By April 2021, the total amount of issued bitcoins is 18.66 millions roughly a 88.9% of the 21 million.

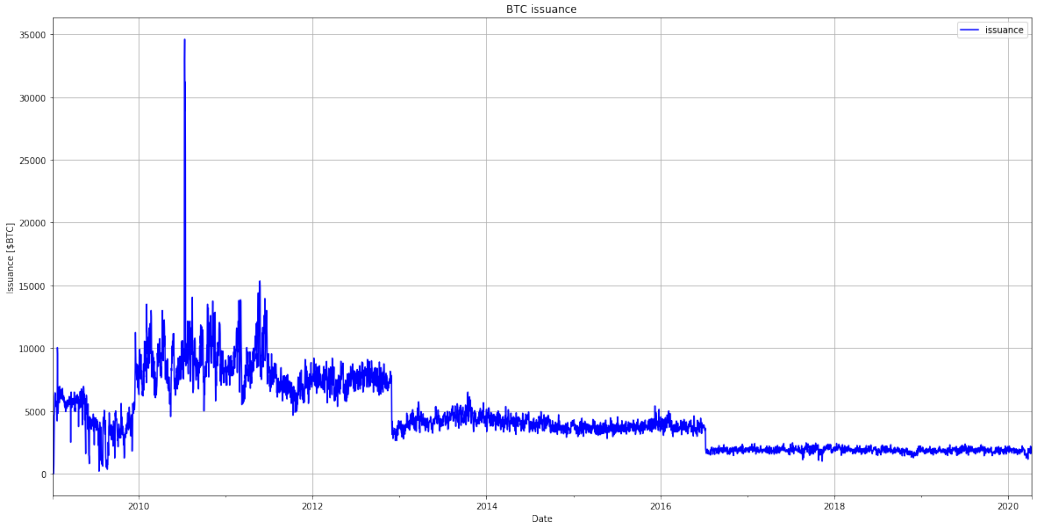

Figure 2: Bitcoin issuance through time. Glassnode data.

As it could be seen in 2 and in 1 the total number of bitcoins is decel-

erating. By design, bitcoin is a scarce asset. In [Pla], PlanB suggests to use a stock to flow (S/F) model to estimate the value of bitcoin scarcity. In figure 3 the close price series and the S/F ratio are shown to better explain its correlation. In blue we can see the how S/F ratio varies with time and in green we see the close price of bitcoin (note the vertical axis is plotted using a logarithmic scale). In red dotted vertical lines we see the days of the halvings (on 2012/11/28, 2016/07/09, and 2020/05/11). Along this feature (S/F ratio) others will be used that are very specific of the Bitcoin infrastructure and technology. Refer to 4.1.2 for a comprehensive description of the features.

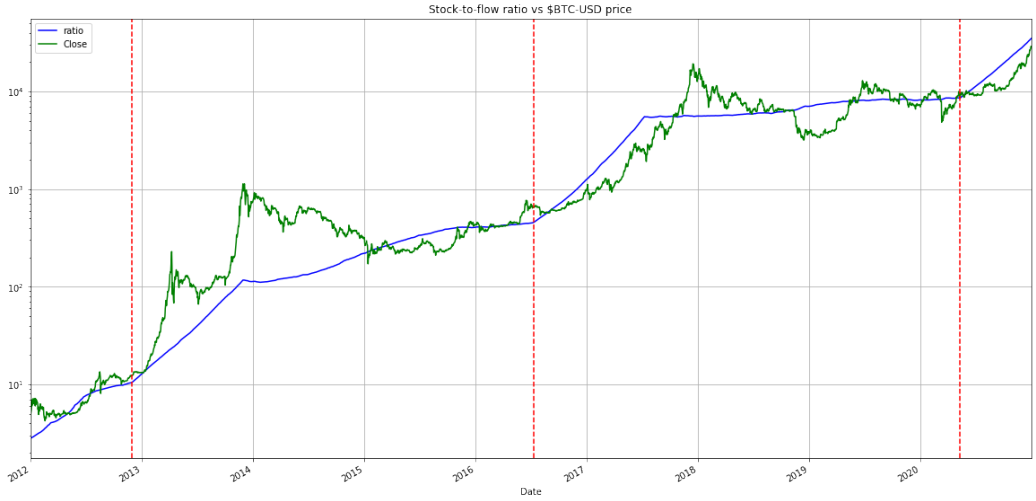

Figure 3: S/F ratio versus the close daily price in USD. Plotted in logarithmic vertical scale.

After Bitcoin software was released and gained attraction from different communities, other projects were created *from* it (e.g. the so-called *forks*). Bitcoin is an open source software project ([Bita]) with a compatible license for both private and public usage which enabled people all around the world to use this technology for multiple purposes such as promotion of new bitcoin-like coins, projects running on top of the blockchain technology, etc.

In this research project we will use bitcoin: the top cryptocurrency by market capitalization as of April 2021 and the first cryptocurrency which started a revolution.

3.2.3 Financial machine learning

This research project involves applied machine learning to finance where it is specifically applied to a trading strategy to learn the size of the position

at each bet. In [Pra18] it is described the procedure to properly develop a machine learning pipeline for a financial application like this one. Lopez de Prado described in [Lop15] problems in financial research publications. The author relates those problems to bad practices and practitioners' ethics sometimes but he also recommended ways to mitigate those issues and their consequences. We could find a similar approach in [Pra18] with a more in deep explanation of all the aspects which attain the machine learning pipeline to develop.

As stated before, in this research project we will develop a strategy based on momentum to determine when and how (long vs. short) place our positions. That would define our primary model which follows a secondary model (the machine learning model) to derive the size of the position. Sizing the bet is a two step process: first we obtain the probability that the buy or sell signal is accurate and second we compute the size of the the bet out of the probability. The model used to derive the probability could be a logistic regression, a tree based model, a neural network, a SVM or any model the researcher considers and *finds* that yields better *results*. This research project will focus on comparing some tree based models and evaluating based on [Pra18] recommendations to mitigate common flaws that will certainly derive into runtime strategy biases and, consequently, losses.

In preparation to train the model, the feature engineering stage will be performed. In this case, univariate, bivariate, scaling and other techniques ([Ali18]) are relevant but not enough. At least two chapters of [Pra18] workout in detail two aspect of the time series analysis from a feature point of view: sample uniqueness (explained in section 5.2.2) and sample stationarity (see chapter 15 [Ham94]) vs. loss of memory dilemma (explained in detail in section 5.1). One can find thorough explanations of the effects of bootstrap sampling and their effects on the model error (see section 7.11 of [Tre17]) but it does not usually consider the *duration* of a sample because those in the dataset are often considered both independent and identically distributed (IID) as well as without duration. This is not the case of financial events like this one. Two or more *signals* might overlap due to high volatility in the market or to correlated information. If the model does not take into account that those *signals* refer to the same unique *event* we would incur in unbalanced datasets when training a model due to variations in *event* representativeness. On the other hand, to perform inference we require signals to be stationary but that comes at the expense of removing memory which is essential to obtain the predictive power. Lopez de Prado proposes to use fractional differentiation (see [Hos81] for the possibly first known occurrence of the method) to determine the minimum differentiation factor d such that the non stationarity null hypothesis of the series is rejected. Hence, some

memory will remain in the series and the model can benefit from it.

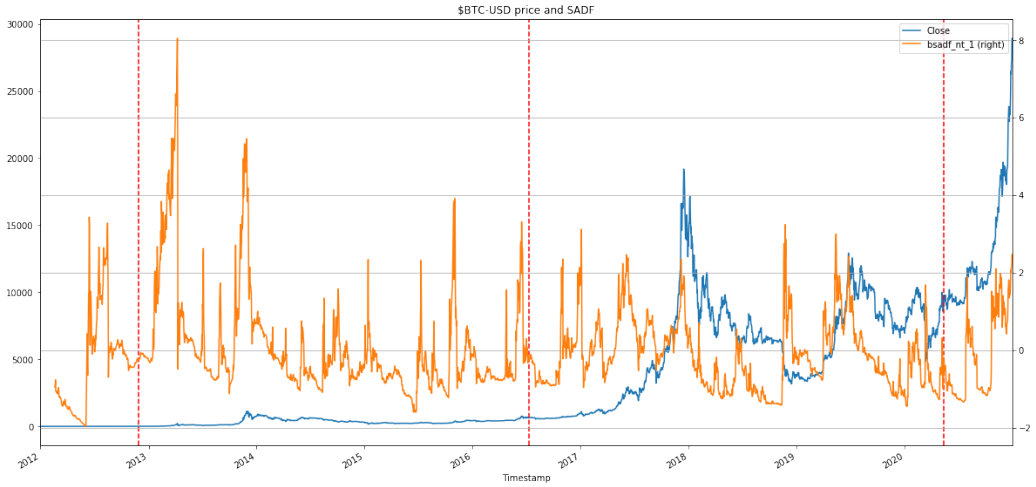

Figure 4: Bitcoin price evolution and SADF index.

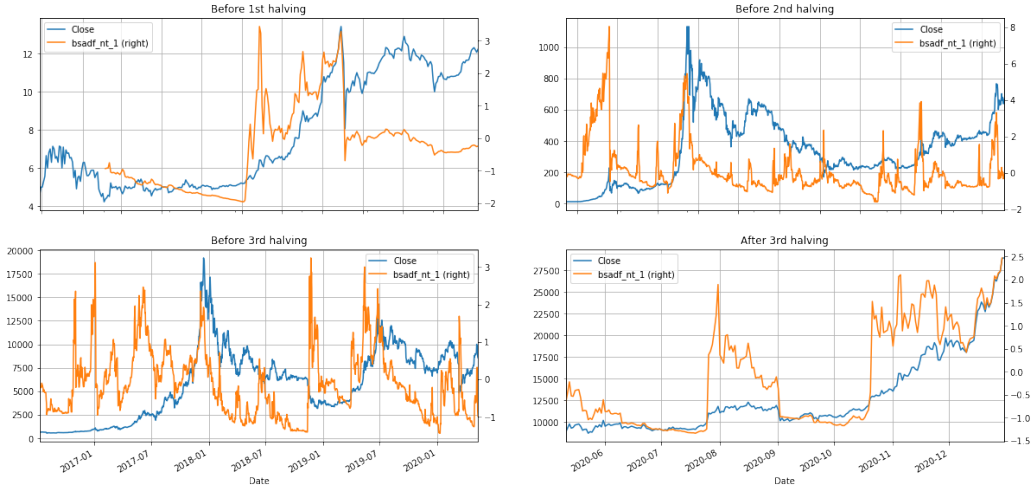

Figure 5: Bitcoin price evolution and SADF index by issuance period.

In 3.2.2, the Stock to Flow model and halvings were introduced. The scarcity model and the issuance scheme imply regime breaks in the price series. Those events are called *structural breaks* and specific features can be built to provide prediction power to our models, see 5.1.3. In chapter 17 of [Pra18], the author classifies into four the types of structural break tests: CUSUM tests, explosiveness tests, right-tail unit-root tests and sub/super-martingale tests. One can find within the explosiveness tests the Supremum

Augmented Dickey-Fuller test (SADF, [Pet11]) which evaluates successive bubble-like behaviors. The core idea of the index is that the price follows a random walk series and at some point it becomes an explosiveness test which ends up in a bubble burst. This process could repeat. The SADF index would raise when the bubble is growing and drastically decrease when the bubble explodes. Figure 4 shows the evolution of the index over the prices. It is not trivial to see the relevance of this index unless we partition the price series by halvings and see it in figure 5.

Moreover, cross validation for training will be discussed and some tweaks we can use to improve the training technique as well as which model metrics we should use to compare machine learning models. Once all the features and the pipeline to train the model is in place, dimensionality reduction by feature selection will be applied to reduce the amount of required data while preserving model performance. Here, different methods will be compared in favor of increased decision robustness. Finally, backtesting of the strategy will be performed with strong emphasis on the the methodology.

3.3 Scope

This thesis aims to thoroughly describe a financial machine learning pipeline for strategy training and validation. It will be executed over bitcoin price with a fundamental momentum strategy. Multiple source features will be examined and used: financial, bitcoin and Bitcoin features, social media and structural break features. Feature engineering for time series will be applied and discussed in favor of determining the implications of sample uniqueness and series stationarity. Ensemble tree models will be used, trained and verified via cross validation with sample adjustments in favor of reduced leakage. Strategy and hyperparameter optimization, and feature selection will be conducted prior to back testing. The final stage will assign bet sizes and run back tests with budget metrics to quantitatively determine whether staking, momentum alone or this full strategy is the best one. Figure 41, which is in section 5.2, shows the aforementioned pipeline.

4 Resources

This section provides reference to the data inputs for the models to be developed and the software tools to implement them.

In a research project like this one the use of open source and permissively provides a lot of tools to work with a very low entry cost.

4.1 Data

The reader will find a handful of data sources listed below. They belong to different providers and are licensed differently but with enough freedom to perform a research like this one. These datasets also have different formats, i.e. a small *impedance mismatch* needs to be solved. Some datasets are CSV files ([RFC]) and others are JSON ([Int]) files. Making these data sources compatible is a common software engineering problem and it is solved either in the data ingestion layer or when the data is first obtained. Given that this datasets where downloaded as a whole (the entire data series), they were kept untouched and then before they are used, only those JSON files were transformed into CSV files. To do so, the pandas python library has been used.

4.1.1 Bitcoin price data

Kaggle [Inc] is a web page that hosts data science competitions. They allow people and organizations to host datasets, determine the competition terms and conditions so people can compete against each other for either the honor or of beating the others or other type of awards. In particular, there is one competition ([Zie]) which holds a by-minute dataset of different bitcoin market values since the inception. Data is updated every quarter and it is provided by *bitcoincharts.com* [Bitb]. The dataset has the following series:

- Timestamp: Start time of time window (60s window), in Unix time[1].
- Open: Open price at start time window.
- High: High price within time window.
- Low: Low price within time window.
- Close: Close price at end of time window.
- Volume_(BTC): Volume of BTC transacted in this window.

- Volume_(Currency): Volume of corresponding currency transacted in this window.
- Weighted_Price: VWAP- Volume Weighted Average Price.

Dataset is licensed under the Creative Commons Attribution-ShareAlike 4.0 International License [Coma].

Given the massive amount of information that this dataset provides, we have created derived datasets with different sample periods: by hours and by days. To do so, we have grouped the data for the same period and computed each feature.

4.1.2 Bitcoin features

In favor of incorporating meaningful features, Bitcoin-related data will be added. Glassnode ([Glaa]) is "a blockchain data and intelligence provider that generates innovative on-chain metrics and tools for digital asset stakeholders". It offers data series under different *tiers*. The free tier lets data consumers use the series in non-commercial applications (see [Glab] for the terms and conditions).

- Timestamp: date and time of the sample.
- Addresses:
  - new-addresses: The number of unique addresses that appeared for the first time in a transaction of the native coin in the network.
  - total-addresses: The total number of unique addresses that ever appeared in a transaction of the native coin in the network.
  - active-addresses: The number of unique addresses that were active in the network either as a sender or receiver. Only addresses that were active in successful transactions are counted.
  - sending-addresses: The number of unique addresses that were active as a sender of funds. Only addresses that were active as a sender in successful non-zero transfers are counted.
  - receiving-addresses: The number of unique addresses that were active as a receiver of funds. Only addresses that were active as a receiver in successful non-zero transfers are counted.

[1]Unix time is a way to express timestamps in seconds that uses the integer count of seconds since epoch in January 1st, 1970. These timestamps are always expressed in UTC as the reference is.

- Blocks:
  - blocks-mined: The number of blocks created and included in the main blockchain in that time period.
  - block-interval-mean: The mean time (in seconds) between mined blocks.
  - block-interval-median: The median time (in seconds) between mined blocks.
  - block-size-mean: The mean size of all blocks created within the time period (in bytes).
  - block-size-total: The total size of all blocks created within the time period (in bytes).
- Fees:
  - fees-total: The total amount of fees paid to miners. Issued (minted) coins are not included.
  - fees-mean: The mean fee per transaction. Issued (minted) coins are not included.
- General indicators:
  - sopr:The Spent Output Profit Ratio (SOPR) is computed by dividing the realized value (in USD) divided by the value at creation (USD) of a spent output. Or simply: price sold / price paid. This metric was created by Renato Shirakashi. For a detailed commentary see this post.
  - ratio & daysTillHalving: The Stock to Flow (S/F) Ratio is a popular model that assumes that scarcity drives value. Stock to Flow is defined as the ratio of the current stock of a commodity (i.e. circulating Bitcoin supply) and the flow of new production (i.e. newly mined bitcoins). Bitcoin's price has historically followed the S/F Ratio and therefore it is a model that can be used to predict future Bitcoin valuations, see [Pla].
  - price-drawdown-from-ath: The percent drawdown of the asset's price from the previous all-time high.
  - market-cap: The market capitalization (or network value) is defined as the product of the current supply by the current USD price.

  - circulating-supply: The total amount of all coins ever created/issued, i.e. the circulating supply.

- Transactions:
  - transaction-size-total: The total size of all transactions within the time period (in bytes).
  - transaction-rate: The total amount of transactions per second. Only successful transactions are counted.
  - transaction-size-mean: The mean size of a transaction within the time period (in bytes).
  - transfer-volume-median: The median value of a transfer. Only successful transfers are counted.
  - transfer-volume-total: The total amount of coins transferred on-chain. Only successful transfers are counted.
  - transfer-volume-mean: The mean value of a transfer. Only successful transfers are counted.
  - transaction-count: The total amount of transactions. Only successful transactions are counted.

- Unspent / spent transactions:
  - utx-os-created: The number of created unspent transaction outputs.
  - utxo-value-spent-mean: The mean amount of coins in spent transaction outputs.
  - utxo-value-spent-median: The median amount of coins in spent transaction outputs.
  - utxo-value-spent-total: The total amount of coins in spent transaction outputs.
  - utxo-value-created-total: The total amount of coins in newly created UTXOs.
  - utx-os-spent: The number of spent transaction outputs.
  - utxo-value-created-mean: The mean amount of coins in newly created UTXOs.

As the reader might have already realized, all these features introduce a lot of network information. However, its use in the model will be evaluated later on based on performance metrics.

All these features have been downloaded in one JSON file each and using the pandas library they have been merged into one single CSV file that has the same format as the bitcoin price and volume in section 4.1.1.

4.1.3 Social features

One of the objectives of this research effort was to evaluate the performance of a trading strategy when it incorporates social media information. Building a mood index such as CNN's fear and greed [CNN] is a research project on its own. In the past year, i.e. 2020, a lot of services like that one which specialize in cryptoassets were released. They incorporate data from social networks (most of them use Twitter and Reddit which offer HTTP APIs and SDKs to easily integrate in different programming languages), search data (e.g. Google) and market data (e.g. recent operated volume). Mixing these data sources, they provide an indicator that tells whether the market is eager to take long or short positions against the asset they measure.

Because of the complexity of making an accurate indicator out of social media and the maturity of the provided services, instead of building a new social index other two data sources were used:

- Google Trends ([LLC]): we have collected the trend of the keyword *bitcoin* throughout time. The index goes from 0 (no interest at all) to 100 (high interest) and has a weekly sampling frequency.
- Alternative.me ([Alt]): provides a combined index whose is range goes from 0 (extreme fear) to 100 (extreme greed) that indicates the mood of the audience that follows bitcoin. It also provides a categorical classification with five levels of the mood. This dataset contains daily data.

These two indexes are introduced and adjusted by time stamp to pair the other data entries.

4.2 Software and tooling

This research project makes extensive use of open source software tools. To name a few:

- Python: programming language with extensive adoption in the data science community as well as *R*.
- numpy: one of the main Math libraries available in Python. It has extensive support for arrays and matrices.

- scikit-learn: provides support for most of the machine learning models and tooling.
- pandas: library for data management in Python.
- statsmodels: provides support for statistical models and hypothesis tests in Python.

5 Methods

5.1 Features

In this section, all the features of the model are introduced. We will present univariate and bivariate analysis of the features as well as different techniques to transform them, in particular fractional differentiation.

When working with inference models, features should be stationary. Common procedures to features like prices involve integer differentiation to end up working with price returns instead. The latter would remove entirely the price series memory which is required by the model to effectively predict the output. Other methods involve applying power transformations such as logarithms, square roots or box-cox transformations. We are not interested in those for price series. They will drastically affect scales and might collapse movements around the trend while preserving the trend.

In [Hos81] the fractional differentiation method was introduced, and Lopez de Prado takes the method and explains the model in chapter 5 of [Pra18]. I will present the mathematical model and explain the algorithm in what follows.

Let B be a backshift operator, i.e. delay operator, to be applied to a matrix of real valued features X_t such that $B^k X_t = X_{t-k}$. Also, we can express the positive integer powers of a binomial as $(x+y)^n = \sum_{k=0}^{n} \binom{n}{k} x^k y^{n-k}$. When considering real valued exponents, combinatorial number $\binom{n}{k}$ becomes (after substitution of n an integer by d a real number) $\binom{d}{k} = \frac{d(d-1)...(d-k+1)}{k!}$ which coincides with the integer formula (for a comprehensive and detailed explanation on the topic, chapter of [GKP94] is recommended). Thus,

$$(x+y)^d = \sum_{k=0}^{\infty} \binom{d}{k} x^k y^{d-k} \tag{1}$$

Equation 1 presents an infinite series, a key difference with respect to the integer counterpart. If we replace x by 1 and y by $-B$, the backshift operator, one can write from 1:

$$(1-B)^d = \sum_{k=0}^{\infty} \binom{d}{k} (-B)^k$$

$$(1-B)^d = \sum_{k=0}^{\infty} \frac{\prod_{i=0}^{k-1}(d-i)}{k!} (-B)^k$$

$$(1-B)^d = 1 - dB + \frac{d(d-1)}{2!}B^2 - \frac{d(d-1)(d-2)}{3!}B^3 + ... \tag{2}$$

Equation 2 presents the foundation of the fractional differentiation method. If one could find the value of d such that a series X gets differentiated and becomes stationary while preserving as much memory as possible a later model would be able to exploit that memory to predict the output. In particular, when the value of d ends up being less than 1. Equation 2 also presents a problem, it is a infinite series when d is noninteger which makes the operation to be non exact for the general case due to the impossibility of applying infinite multiplications and sums. Lopez de Prado proposes a solution for both issues.

A fractionally differentiated series X can be expressed for a given d as:

$$X_t^d = \sum_{k=0}^{\infty} w_k X_{t-k}$$

The vector w of weights in the above equation follows:

$$w = 1, -d, \frac{d(d-1)}{2!}, -\frac{d(d-1)(d-2)}{3!}, ..., (-1)^k \prod_{i=0}^{k-1} \frac{(d-i)}{k!} \tag{3}$$

One can derive by inspection of equation 3 a generative and recursive expression for each item in the series:

$$w_k = -w_{k-1} \frac{d-k+1}{k} \tag{4}$$

Equation 4 is easy to implement as a programming function which is ideal for this application. It is also interesting to evaluate the tendency of w_k as k tends to infinite, see figure 6.

As it can be seen in figure 6, coefficients tend to zero as k increases. It can also be proved the convergence of coefficients w_k. Let's analyze the following when $k > d$ and $w_{k-1} \neq 0$:

$$|\frac{w_k}{w_{k-1}}| = |\frac{d-k+1}{k}| < 1$$

what makes $|w_k| < |w_{k-1}|$ leading to $\lim_{k\to\infty} w_k = 0$.

When implementing fractional differentiation on a real time series, one has two options:

1. Adjust the length of w_k vector by weight loss with a certain threshold.
2. Work with a fixed number of coefficients.

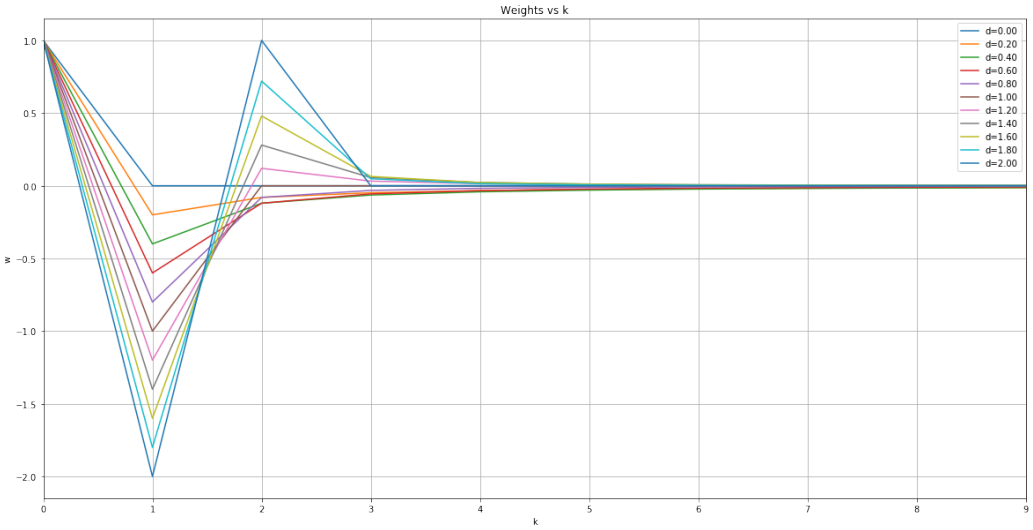

Figure 6: Weight values vs. k for different d values.

Option 1 requires the operation to compute the size of w_k vector to account for weight loss given a certain threshold. Weight loss cam be computed:

$$\lambda_l = \frac{\sum_{j=T-l}^{T} |w_j|}{\sum_{i=0}^{T-1} |w_i|} \tag{5}$$

where:

- T is the length of the time series,
- l is the index of the sample from the end where the fractional differentiation occurs.
- λ_l the weight loss at l index

One should discard all samples whose $\lambda_l < \tau$ and τ is the threshold. As $d \to 0$, the energy of the weights decreases leading to more weight loss and more discarded samples.

Option 2 comes with the simplicity of having always the same vector of coefficients w_k such that $|w_k| > \tau$ and τ is a user defined threshold. It comes with the advantage of having no drift as option 1 and just needs to drop l samples at the beginning, being l the value of l that makes w_k less or equal to τ. In this thesis, option 2 is used.

So far, how to compute the weights vector was explained. Now, we just need to address the value of d. X_t might be stationary already which leads

to $d^* = 0$ with d^* the value of d that preserves most memory making the time series stationary. When X_t has a *unit root* (see chapter 15 of [Ham94]), $0 \leq d^* \leq 1$. And when X_t exhibits an explosive (bubble) behavior, $d^* > 1$. Unit roots can be tested with Dickey Fuller hypothesis test (see chapter 17 of [Ham94]). The null hypothesis of the test claims the series has a unit root. After determining a certain confidence level one can derive an *optimum* d^* by:

1. Define a vector of d values in range of 0 to 1.
2. For each value of d:
   (a) Fractionally differentiate X_t with d given a certain amount of weights. Obtain X_t^d.
   (b) Compute the Dickey Fuller statistic, ADF, for X_t^d.
   (c) Compute the p-value of the test.
3. Choose d^* that yields the maximum p-value between all p-values that are less or equal to the confidence level.

This process has two flaws:

- It is computationally time complex. Each time we apply the fractional differentiation, we are processing a $O(n^2)$ algorithm. Computing the ADF of the differentiated series requires a differentiation and model estimation which yields at least another $O(n^2)$ process. Finally, we iterate through a vector of d adding another dimension.
- *Optimum* d is subject to the granularity of the d vector of samples. The smaller the step, the more information one could preserve in the final time series, but the more iterations are required which impacts directly in the aforementioned item.

Regardless, all features that expose non-stationary characteristics could be transformed and stabilized while preserving memory.

5.1.1 Fundamental features

The dataset of bitcoin prices and volume comes with open, close, highest and lowest daily prices, and volume expressed in transacted coins and their market value in USD. Figures 7 and 8 show the bitcoin daily prices. Figure 9 shows the volume series.

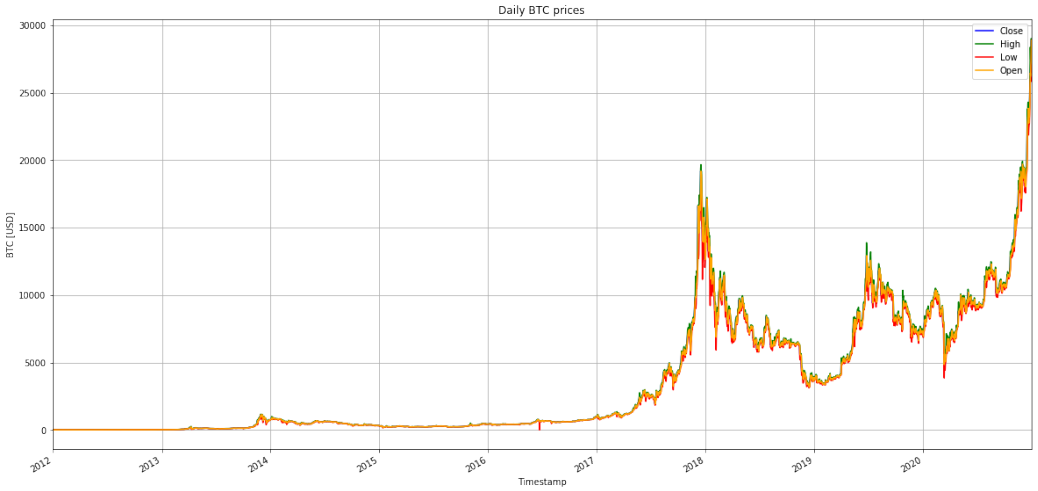

Figure 7: Overlapped open, close, high and low bitcoin daily prices in USD.

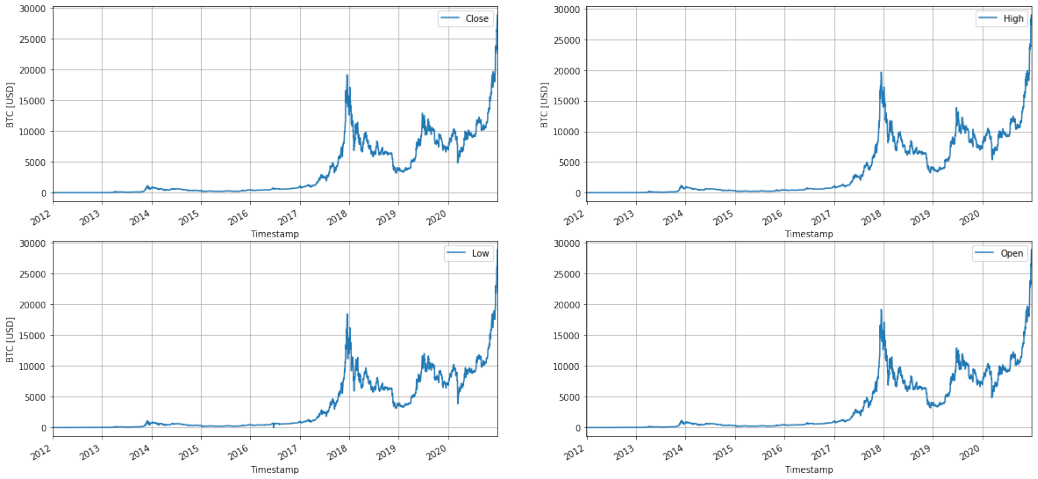

Figure 8: Split of open, close, high and low bitcoin daily prices in USD.

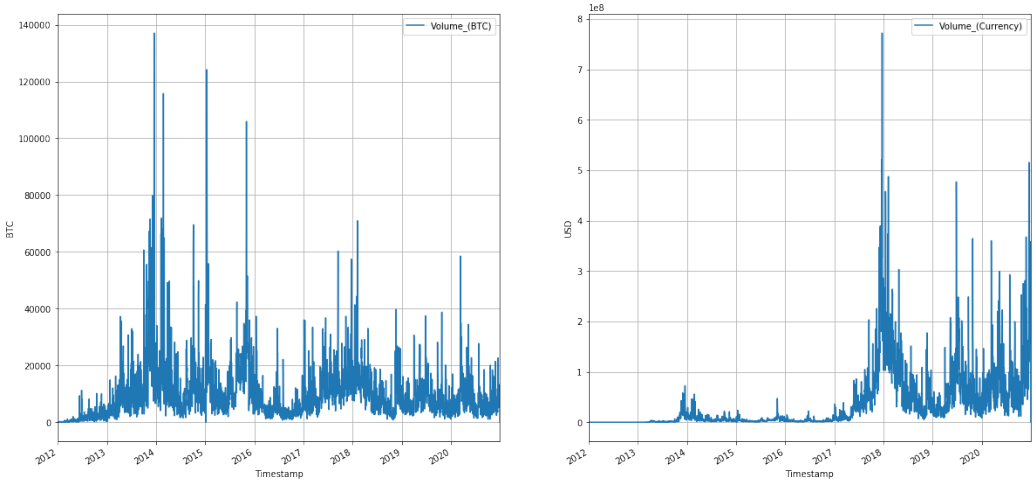

Figure 9: The graph on the left shows the bitcoin daily volume. The graph on the right shows the market volume valuation in USD.

For volume series, the fractional differentiation was not enough to make the series stationary. An extra log transformation was proposed to stabilize it. Results can be seen in figure 10.

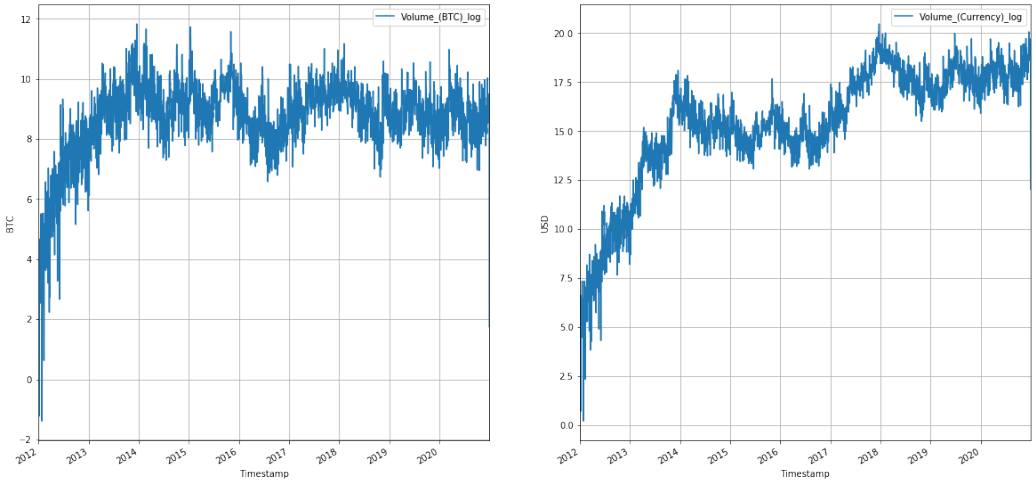

Figure 10: Same features as in 9 but taking the logarithm.

Price series require fractional differentiation as explained in 5.1. We could obtain the best d^* for each price time series applying the method previously described. Ten samples in the range $[0, 1]$ are taken to look for the best d^* which shields the results in table 1.

Just for illustration purposes, see figure 11 that shows the Close price series and the same series with an overlap of the fractionally differentiated

Feature	d^*
Close	0.4
Open	0.4
High	0.4
Low	0.2

Table 1: Fractional differentiation order for price series.

counterpart. Note the y axis on the right which tracks the scale of the fractionally differentiated Close price series. It can be seen perfectly the effect of differentiating the series.

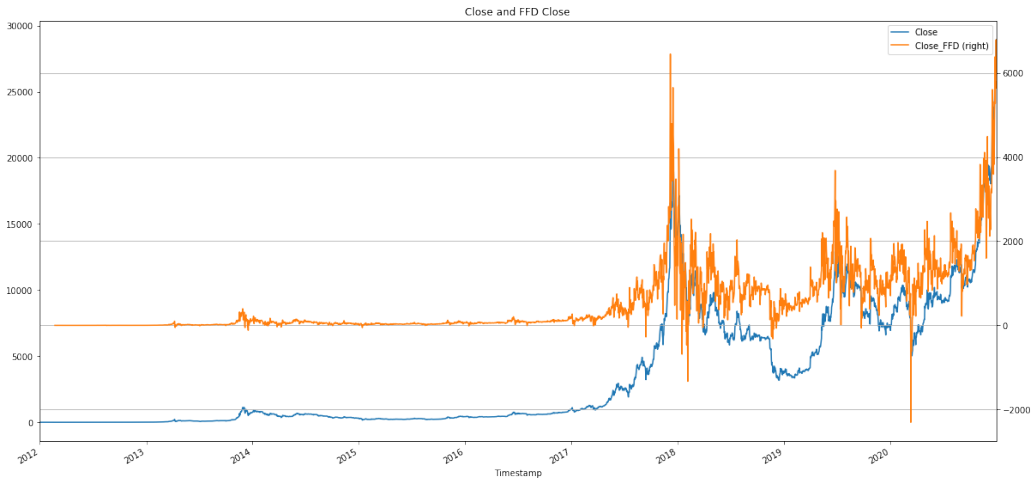

Figure 11: Close price and fractionally differentiated Close price with $d^* = 0.4$ (see table 1).

On the other hand, other features were created out of the Close price series. Those features features are:

- RSI: Relative Strength Indicator. It is a momentum index that measures the strength or weakness of a stock price. Some time windows are used in favor of capturing high speed and low speed trends. See figure 12.
- Autocorrelation: with different lags and time window lengths, the price series autocorrelation looks for repetitive patterns in the signal. Peaks in the autocorrelation signal indicate an occurrence of repetition. See figure 13

- Logarithm of the returns: just what the index expresses. It uses a logarithm transform to control the variance of the series. See figure 14 and the histogram of values.

- Volatility: computed as the standard deviation of the moving average of the log returns. That yields the trend in return variation. Multiple time windows are used to capture different speeds. See figure 15.

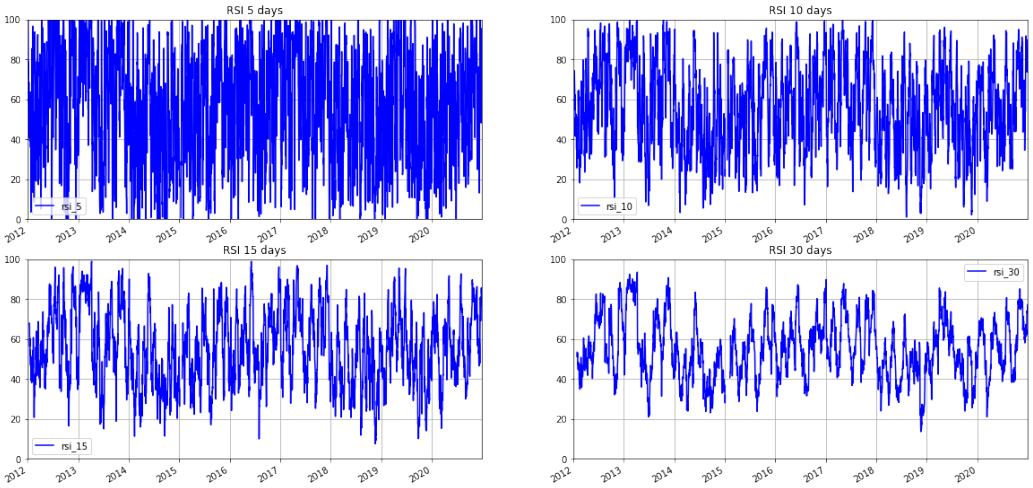

Figure 12: RSI indexes for Close prices.

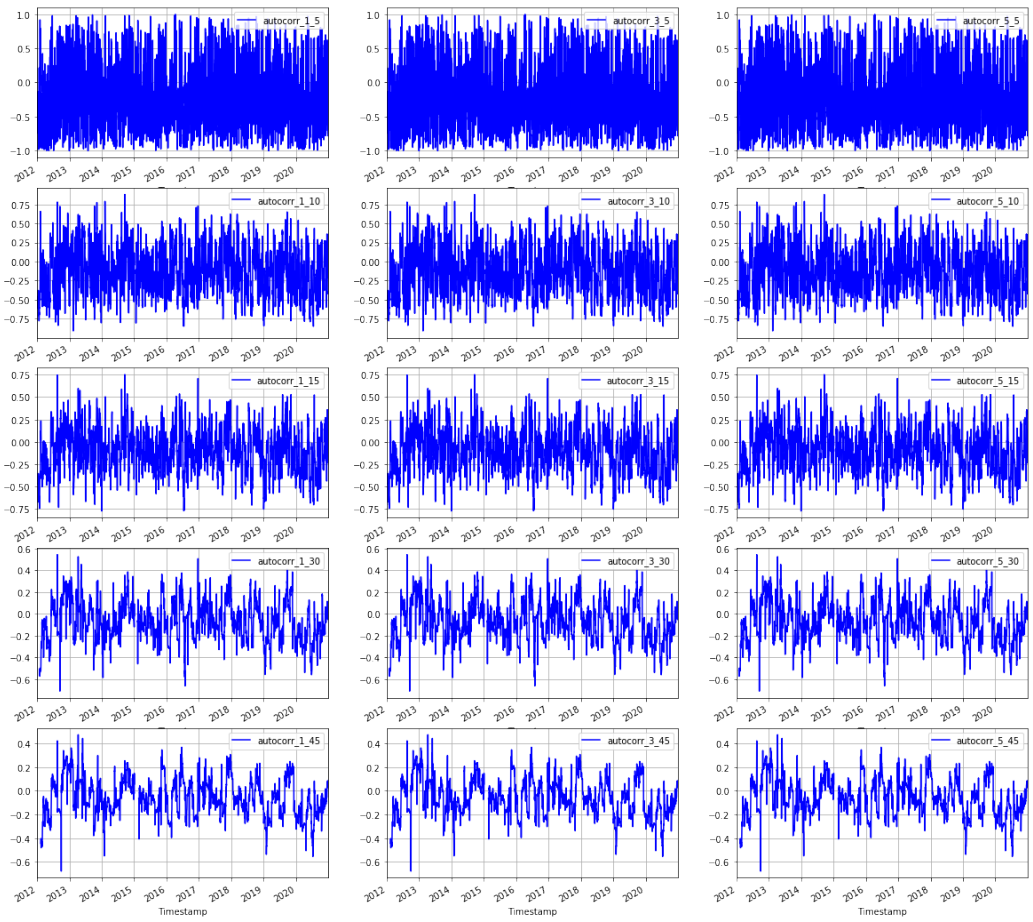

Figure 13: Different autocorrelation signals with different lags and time windows.

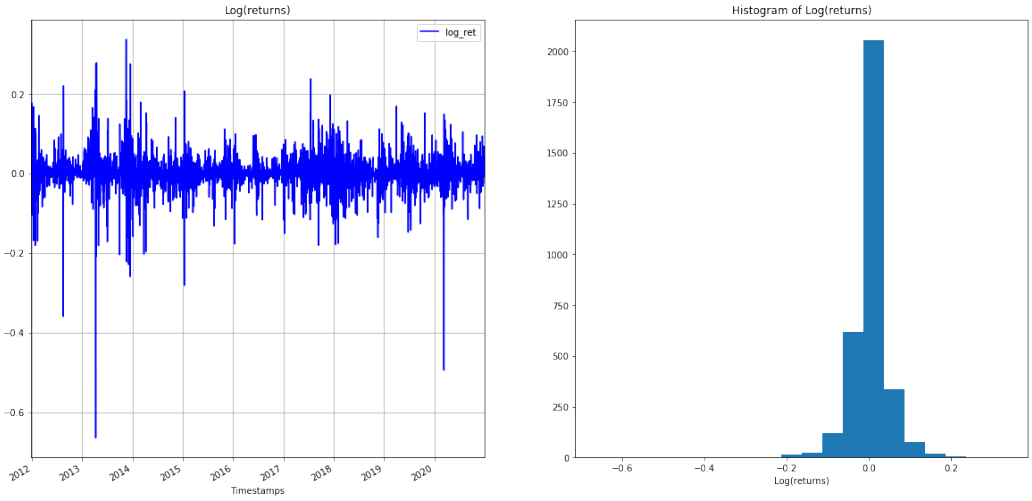

Figure 14: Logarithm of returns on the left, the histogram of the logarithm of returns on the right.

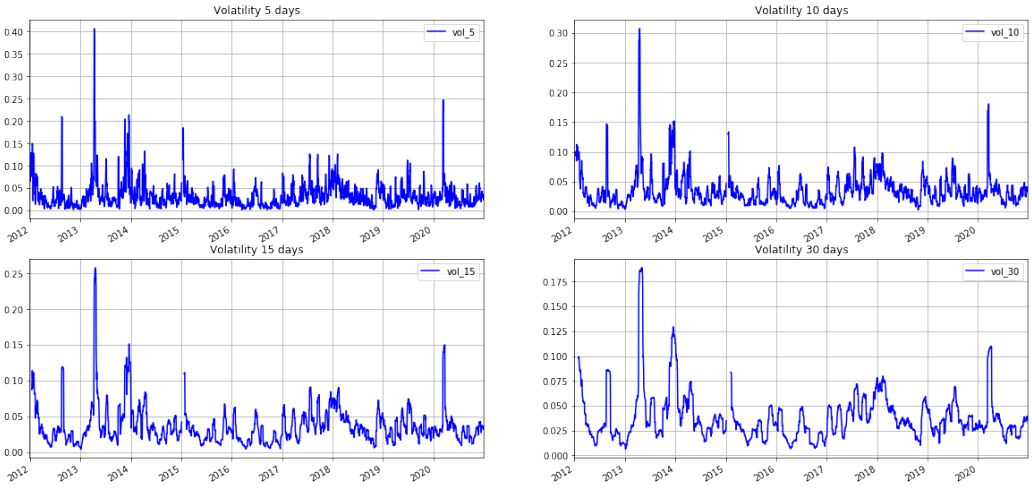

Figure 15: Volatility index.

5.1.2 Domain specific features

The objective of including domain specific features is to add into the secondary model data that is intrinsically representative of the underlying technology and market behavior. This subsection shows and explains the transformations incurred to the listed features in 4.1.2.

Addresses: this group is composed of: *total-addresses*, *new-addresses*, *active-addresses*, *total-addresses*, *sending-addresses*, *receiving-addresses*. We can split these features into two groups, in figure 16 the evolution of total registered addresses in the network is plotted. A logarithm transformation is show as well in green to stabilize its variance and range. Then, in figure 17 we see the evolution of the other features. The reader may observe the strong correlation between these four indexes and validate the observation by inspecting figure 18 which shows the correlation matrix between these four attributes. Because of that, only *active-addresses* will be used to train the model. We computed the fractional differentiation optimum d value for the latter index and turned out to be 0.1.

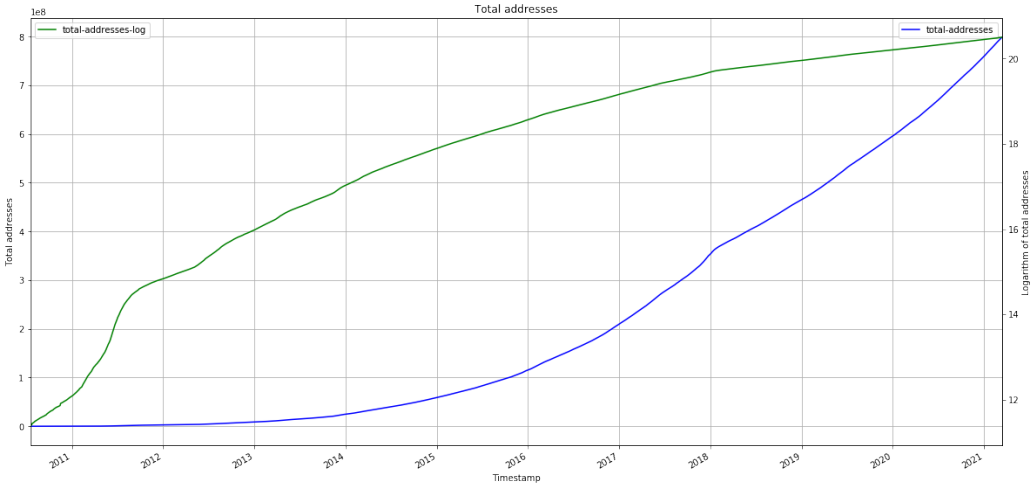

Figure 16: Evolution of total registered addresses over time.

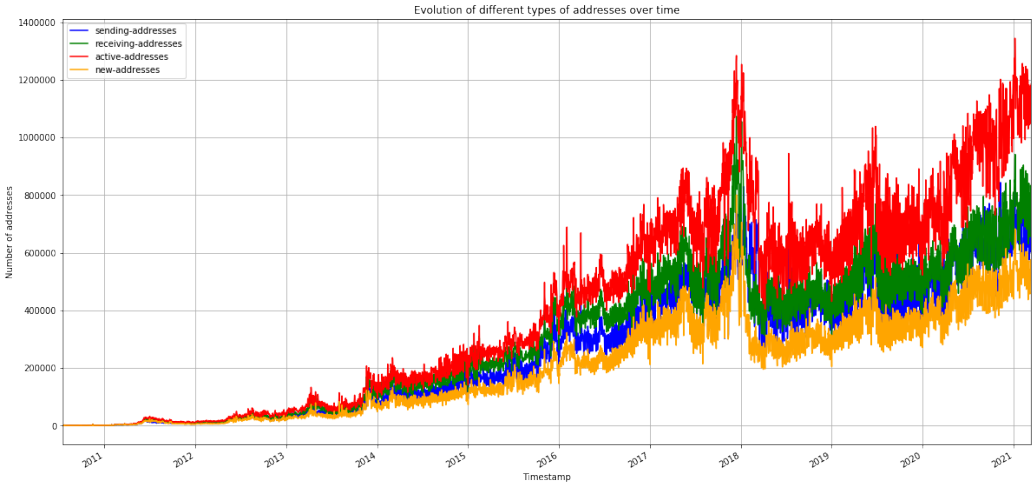

Figure 17: Evolution of *new-addresses*, *active-addresses*, *total-addresses*, *sending-addresses* and *receiving-addresses* over time.

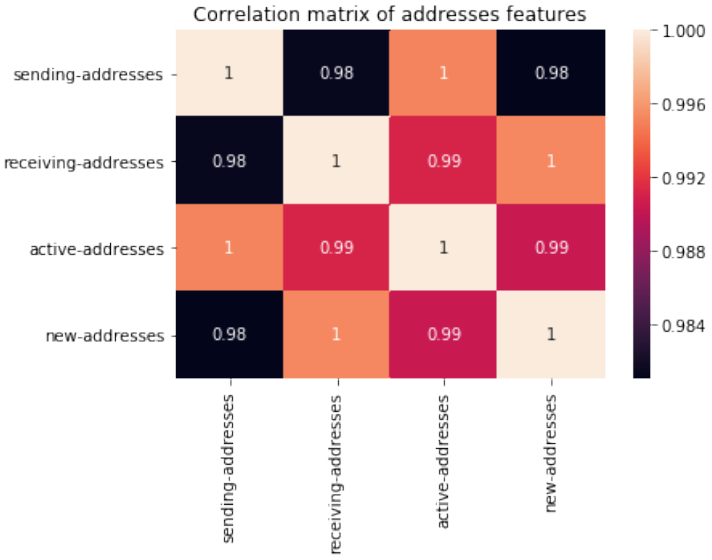

Figure 18: Correlation matrix of *new-addresses*, *active-addresses*, *total-addresses*, *sending-addresses* and *receiving-addresses*.

Blocks: in this group There are: *blocks-mined*, *block-interval-mean*, *block-interval-median*, *block-size-mean* and *block-size-total*. The first three exhibit a quite similar tendency (see figure 19) which is aligned with what they mean: *blocks-mined* describes the number of blocks added to the blockchain and the other two are centrality measures of the production rate. The cor-

relation matrix confirms it, see figure 20. Only *blocks-mined* will be used to avoid adding redundant information. On the other hand, we have the *block-size-mean* and *block-size-total* which are expressed in bytes. Both indexes are expected to be correlated and 21 shows their evolution over time. A fractional differentiation transformation is not enough to stabilize the series and a logarithmic transformation is applied which results in 22. Only the logarithm of *block-size-total* will be used to train the model between these two.

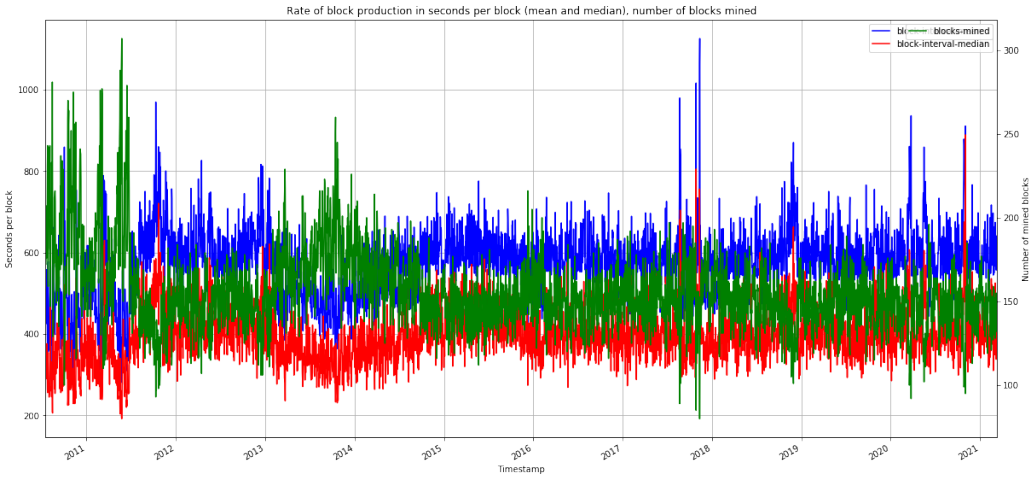

Figure 19: Evolution of *block-interval-mean* and *block-interval-median* are tracked on the left y-axis. *blocks-mined* is tracked on the right y-axis.

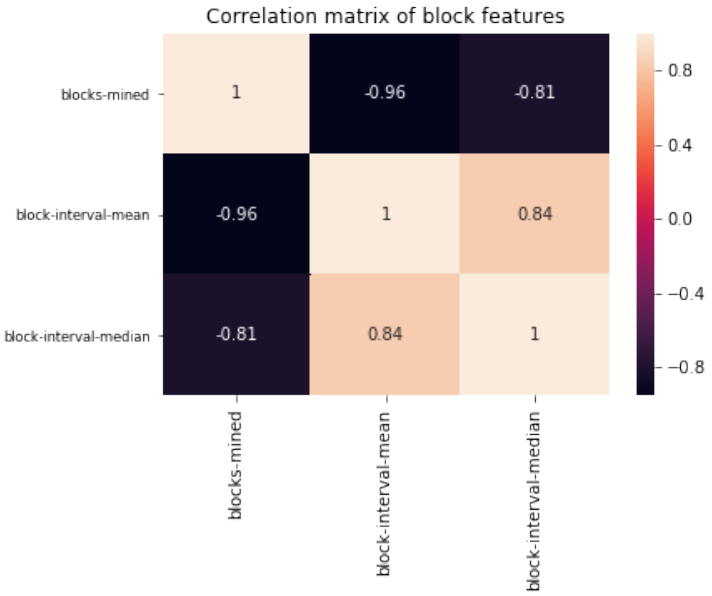

Figure 20: Correlation matrix of *blocks-mined*, *block-interval-mean* and *block-interval-median*.

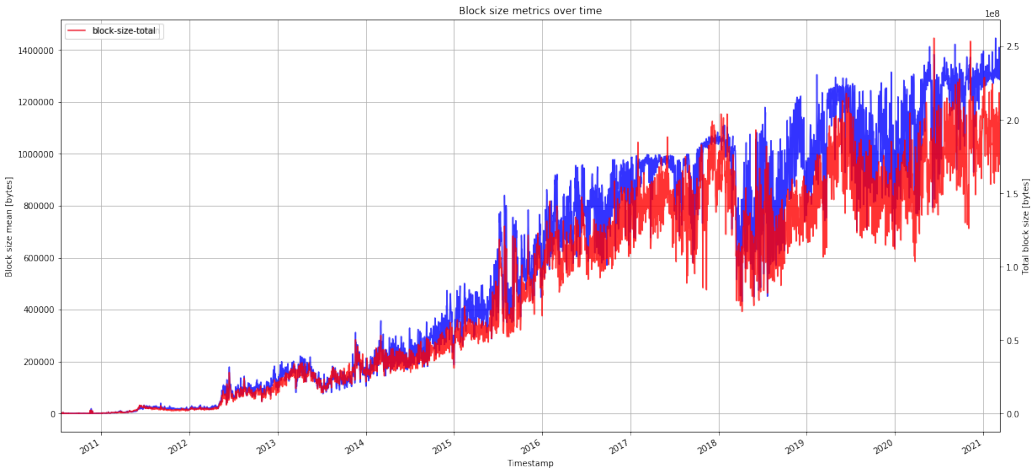

Figure 21: Evolution of *block-size-mean* (in blue and tracked on the left y axis) and *block-size-total* (in red and tracked on the right y axis) over time.

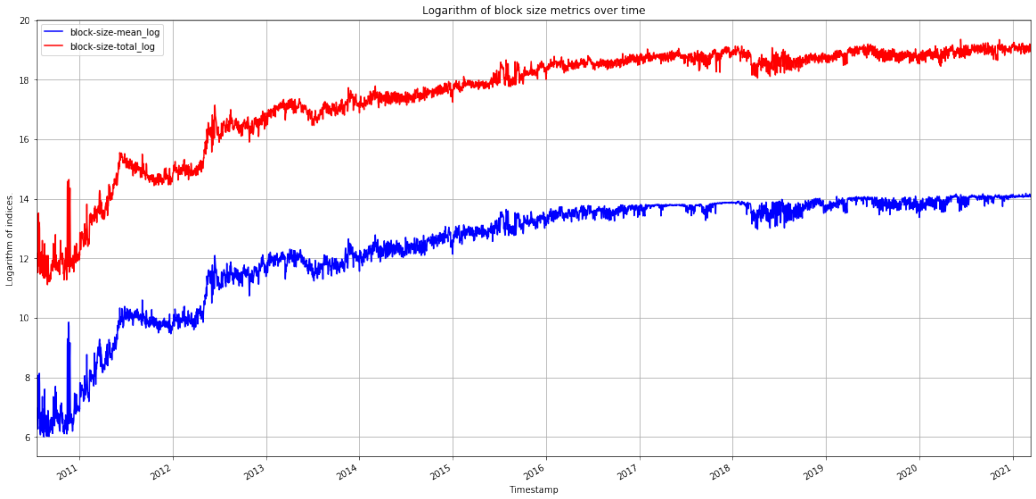

Figure 22: Same features as in 21 but with a logarithmic transformation. Due to the change of scale, both features are tracked on the same left y axis.

Fees: there are two features related to fees: *fees-total* and *fees-mean*. They both remain relatively stable with very low values but there are high valued outliers that get out of range rapidly. Fractional differentiation was tried but ended up in very low values of d that made no significant change. A logarithmic transformation is applied to account the explosive change in local variance when these outliers occur. Note also that because some values in the series are zero, a 1 is added to the logarithm argument to avoid having minus infinity in the series after the transformation. For illustration purposes, figure 23 is shown.

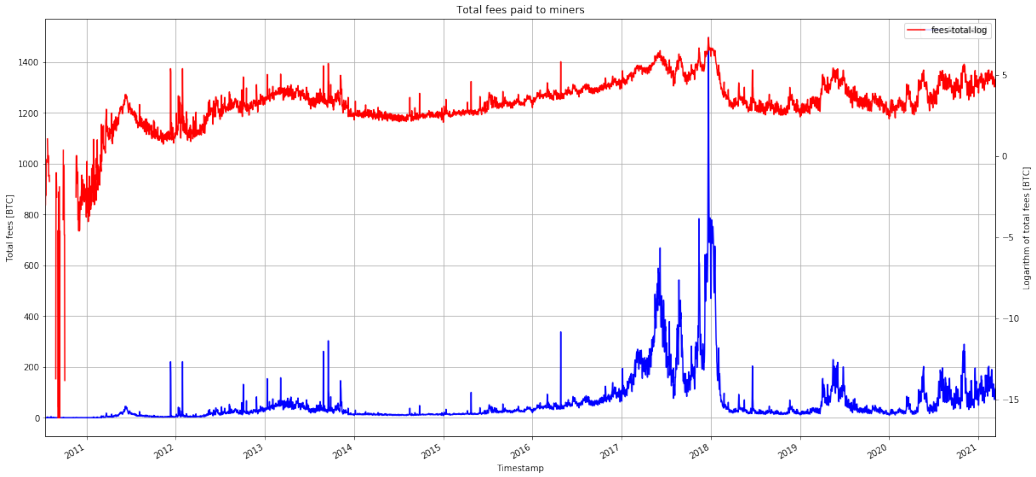

Figure 23: Evolution of *fees-total* (in blue) and its the logarithmic transformation (in red) over time. The left y axis tracks the linear scale and the right y axis tracks the logarithmic scale.

General indicators: in this group we have *sopr*, *ratio*, *daysTillHalving*, *price-drawdown-from-ath*, *market-cap* and *circulating-supply*. The relationship between *market-cap* and *ratio* was already explained in section 3.2 when referring to the stock to flow model so it will be omitted in this case. We will just mention that *market-cap* will suffer a logarithmic transformation to stabilize its range. *sopr* and *price-drawdown-from-ath* are indexes so no further transformation is required (see figures 24 and 25). Moreover, the *circulating-supply* values are in the order of millions and evolves asymptotically to 18 millions as the issuance model expects it to be. In particular, we will create a derivative index from the supply computed as the differentiated series of circulating supply logarithm which shows the speed of issuance (see figure 26).

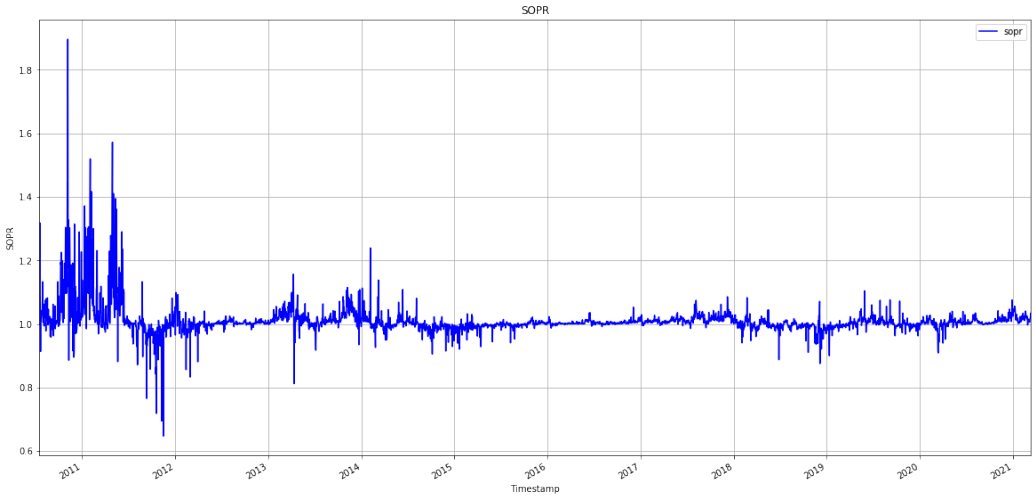

Figure 24: Evolution of *sopr* over time.

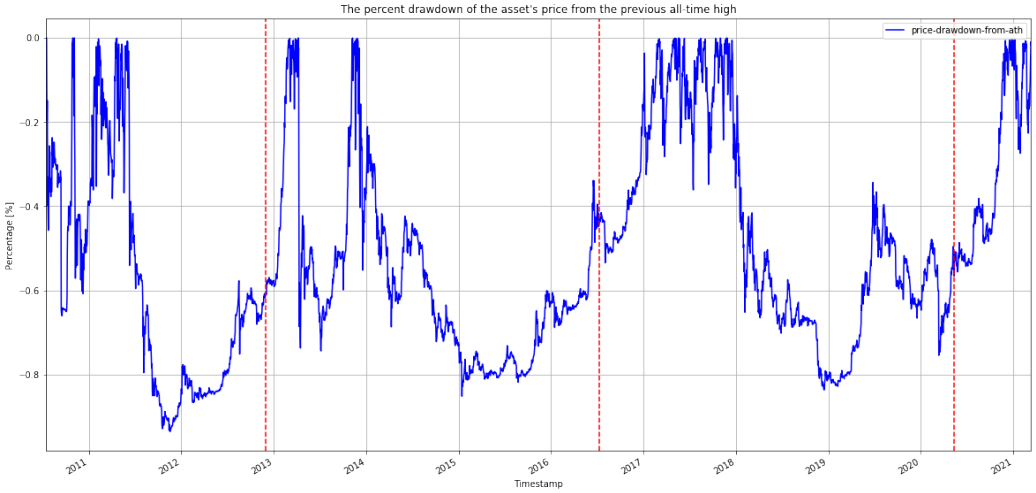

Figure 25: Evolution of *price-drawdown-from-ath* over time. In vertical dashed red lines the halving dates are displayed

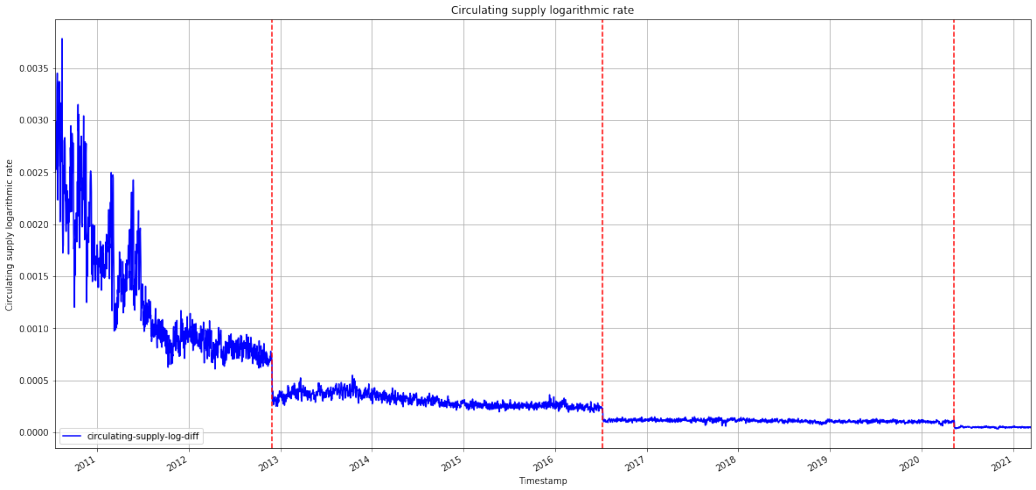

Figure 26: Evolution of the logarithm of issuance over time. In vertical dashed red lines the halving dates are displayed.

Transactions: in this group we have *transaction-size-total*, *transaction-rate*, *transaction-count*, *transaction-size-mean*, *transfer-volume-mean* and *transfer-volume-median*. As it can be seen in figure 27, the three indexes *transaction-size-total*, *transaction-rate*, and *transaction-count* are highly correlated so only one will be taken as input to the model, in particular the *transaction-rate*. A note about this index can be done when looking at the histogram which seems to be bimodal. A derivative index is created from it which groups the values into their deciles and aims to reduce the high frequency changes in the transaction rate that affect the signal on a daily basis. Figure 28 shows it in detail. Moving to *transaction-size-mean* the index exhibits some frequent spikes but those are within range and there is no clear trend so the feature remains as is (29). Central transfer volume features (*transfer-volume-mean* and *transfer-volume-median*) exhibit a high volatility in the first issuance period and then it is more and more stable. A logarithmic transformation is applied to both and even though they try to predict the same, their tendencies are different in values and capture differently the local volatility so both will be preserved (30). Similarly to the processing done to the others, the *transfer-volume-total* is transformed with a logarithm (31).

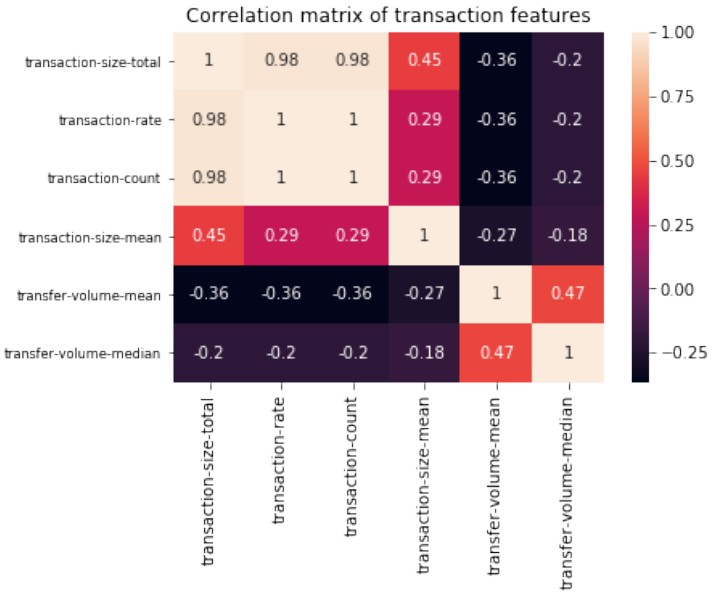

Figure 27: Correlation matrix between the transaction related features

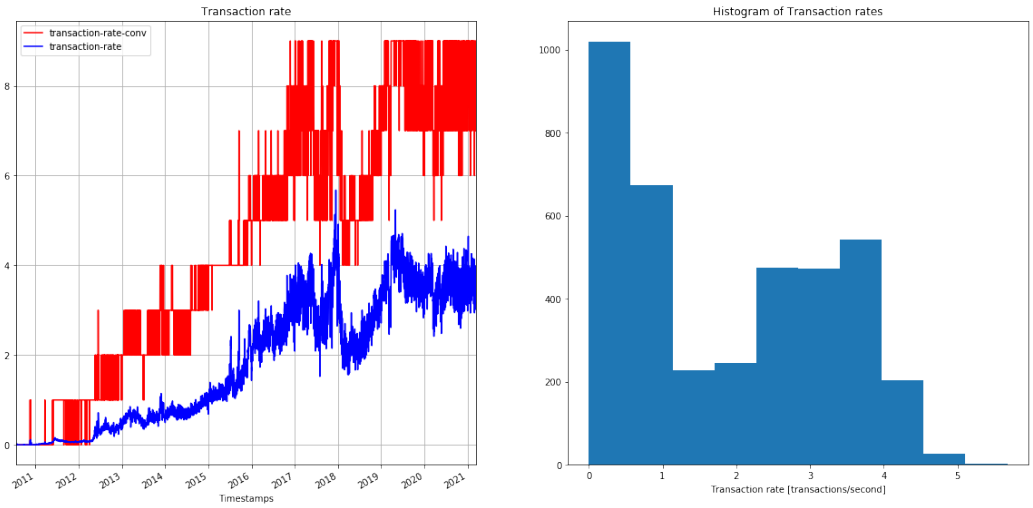

Figure 28: Left: *transaction-rate* (in blue) and decile index of the same feature (in red) over time. Right: histogram of *transaction-rate*.

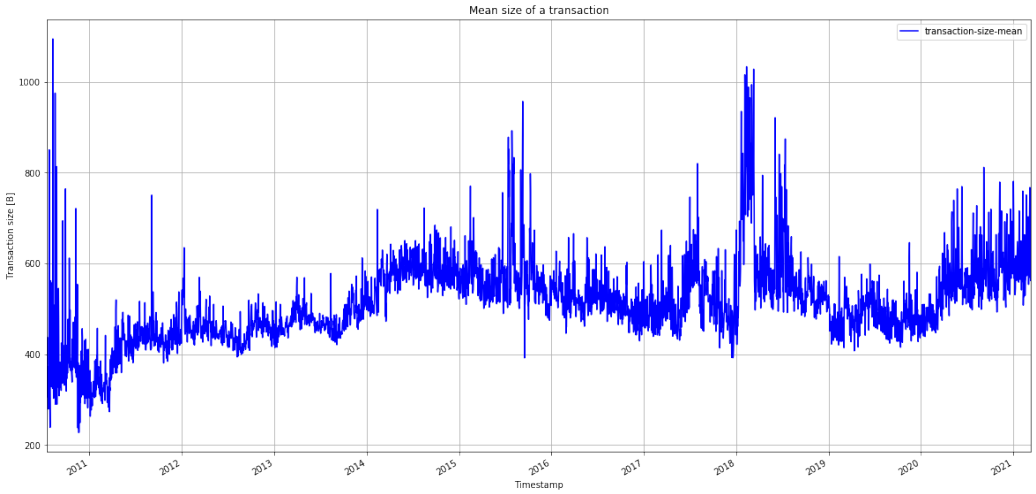

Figure 29: Evolution of *transaction-size-mean* over time.

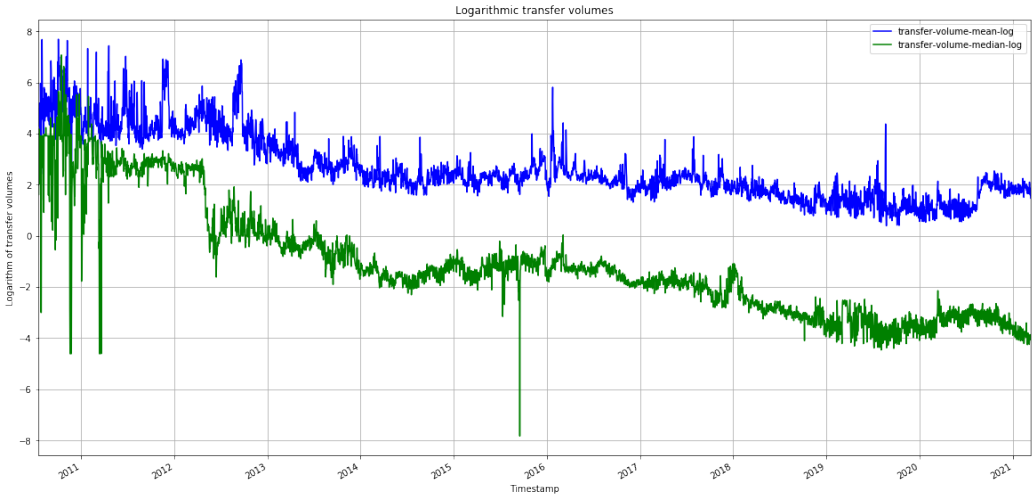

Figure 30: Evolution of the logarithm of *transfer-volume-mean* and *transfer-volume-median*.

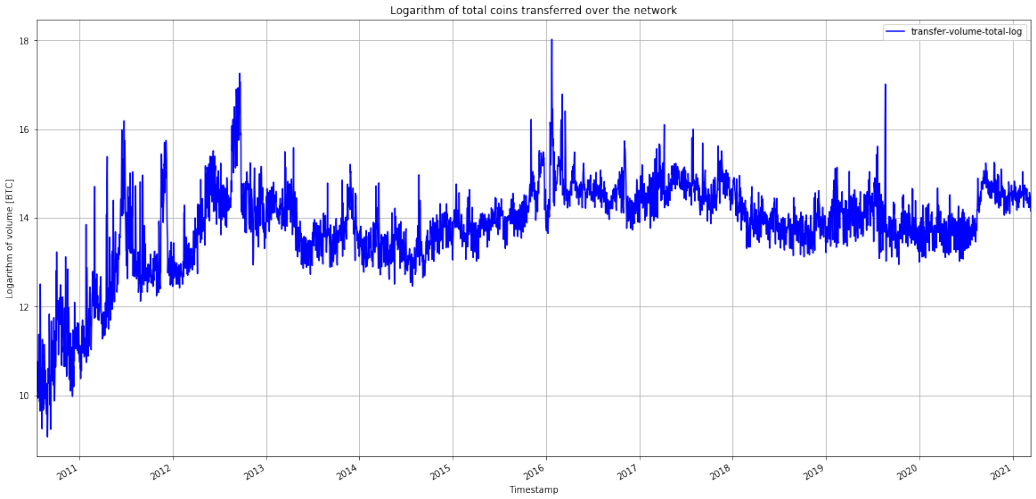

Figure 31: Evolution of the logarithm of *transfer-volume-total* over time.

Unspent / spent transactions: under this final group we can find: *utx-os-created* *utx-os-spent*, *utxo-value-spent-mean*, *utxo-value-spent-median* and *utxo-value-created-mean*. Following the same procedure as with the others, a correlation matrix between these features is created and shown in figure 32. We can find two pairs of highly correlated features: *utx-os-created* with *utx-os-spent*, and *utxo-value-created-mean* with *utxo-value-spent-mean*. Note that *utxo-value-spent-median* is mildly correlated with the others (0.48 and 0.49 respectively) which makes this feature to be kept. Provided that *utx-os-created* is a super set of *utx-os-spent*, the former will be preferred (see figure 33). The same reasoning applies to *utxo-value-created-mean* (see figure 35).

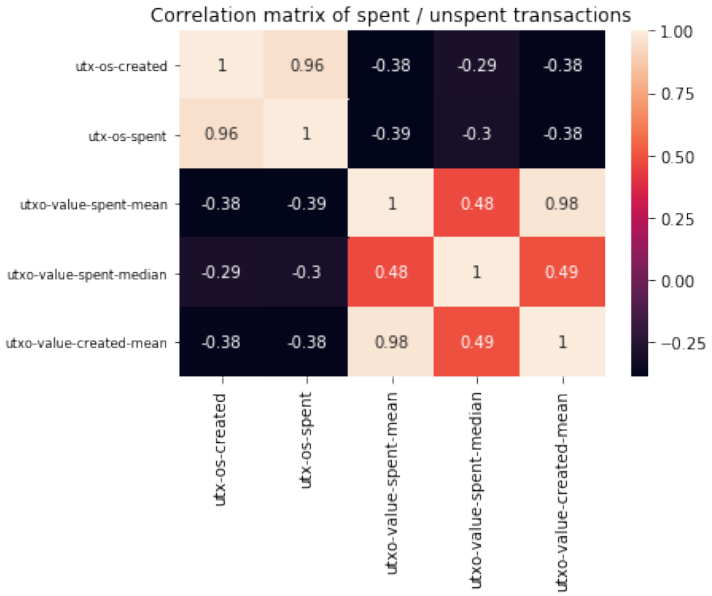

Figure 32: Correlation matrix between unspent / spent transaction features.

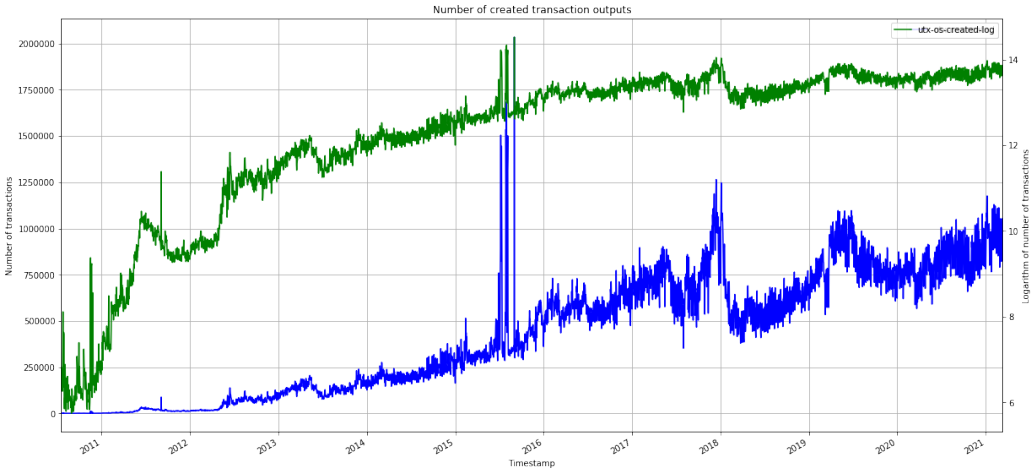

Figure 33: Evolution *utx-os-created* (in blue) and its logarithmic transform (in green) over time.

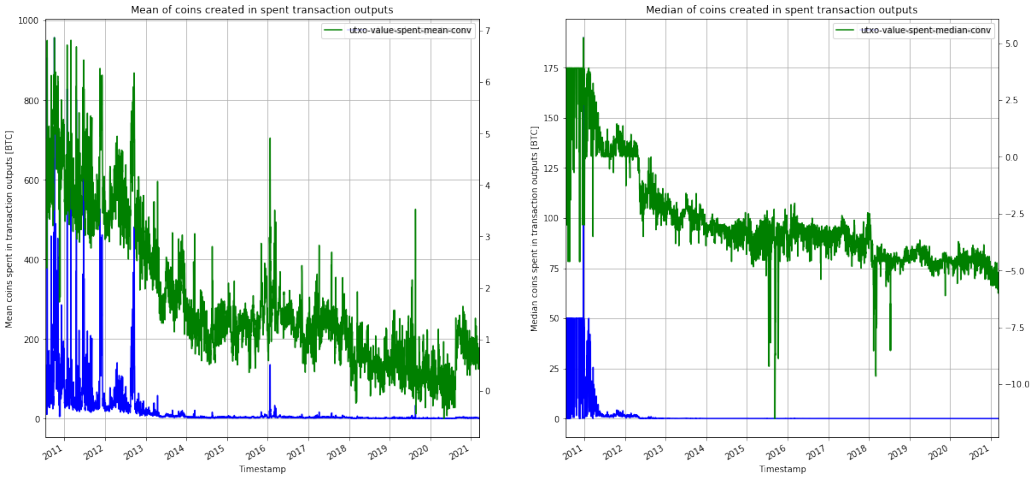

Figure 34: Left: evolution of *utxo-value-created-mean* (blue) and its logarithmic transformation (in green) over time. Right: evolution of *utxo-value-created-median* (blue) and its logarithmic transformation (in green) over time.

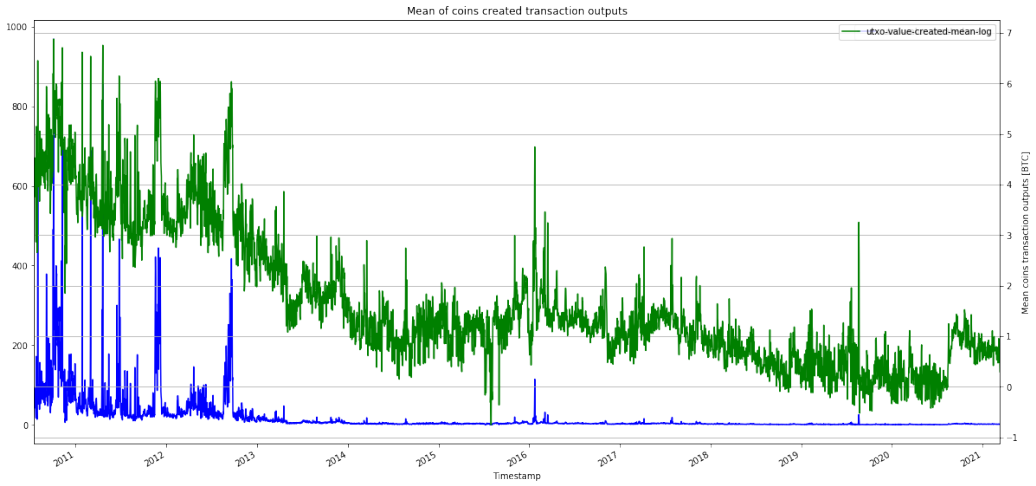

Figure 35: Evolution of *utxo-value-created-mean* (in blue) and its logarithmic transformation (in green) over time.

5.1.3 Supremum Augmented Dickey Fuller

Section 3.2 briefly introduced the concept of structural breaks and the index to be implemented and used in this research project, SADF. In words of Lopez de Prado:

"In developing an ML-based investment strategy, we typically wish to bet when there is a confluence of factors whose predicted outcome offers a

favorable risk-adjusted return. Structural breaks, like transition from one market regime to another, is one example of particular interest."

The problem appears when trying to quantitatively detect how a regime change occurs. In [Pet11] Phillips, Wu and Yu studied Nasdaq index in 1990 prior to the famous DotCom bubble and proposed a new index based on recursive augmented Dickey-Fuller tests for unit root against the alternative of an explosive root (the right-tailed). The objective of this test is to identify the presence of exponential growth or collapse, while assuming an autoregressive specification.

In price series, like the one used in this research project, not only one, but many bubbles are prune to happen. Not all indexes are useful under this assumption as many fail to detect recurrent bubbles. Nevertheless, we will start analyzing the case of just one bubble in the series and then generalize it to many as the it is described in chapter 17 of [Pra18]. Suppose a price series that follows a first order autoregressive process:

$$y_t = \rho y_{t-1} + \epsilon_t$$

where $\epsilon_t \sim N(0, \sigma_y^2)$, i.e. white noise. We can create a test to evaluate the value of ρ whose null hypothesis states that the price series follows a random walk. In other words, $H_0 : \rho = 1$ and the alternative hypothesis is that y_t starts as a random walk but at some point in time t^*T the process becomes explosive, such:

$$H_1 : \begin{cases} \rho = 1 & \text{if t = 1, ..., } \tau^*T \\ \rho > 1 & \text{if t = } \tau^*T\text{, ..., T} \end{cases} \tag{6}$$

where $\tau^* \in (0, 1)$. At the end of the series, i.e. at T, one could try to find the value τ^* where there was a change of regime from random walk to an explosive process. To test this hypothesis, we should consider:

$$\Delta y_t = \delta y_{t-1} D_t[\tau^*] + \epsilon_t$$

where $D_t[\tau^*]$ is a dummy variable that takes the value of 0 when $t < \tau^*T$ and 1 otherwise. $H_0 : \delta = 0$ which is tested against the one-sided alternative $H_1 : \delta > 1$ leading to an statistic:

$$DFC_{\tau^*} = \frac{\hat{\delta}}{\hat{\sigma}_\delta}$$

One needs to determine the value of τ^* because it is unknown. The approach to determine it is to compute the supremum statistic for each possible value of τ^* in the series. That would determine the start of the explosive

process yielding to a bubble. Note that only the beginning of the regime change is determined with this method, i.e. there is no return to a random walk after the bubble starts. This is where the novelty of Phillips, Wu and Yu appears by a few key differences to the autoregressive model and test:

- The regression specification becomes: $\Delta y_t = \alpha + \beta y_{t-1} + \sum_{l=1}^{L} \gamma_l \Delta y_{t-l} + \epsilon_t$
- $H_0 : \beta \leq 0$, and $H_1 : \beta > 0$
- $SADF_t = \sup_{t_0 \in [1, t-\tau]} \{ADF_{t_0,t}\} = \sup_{t_0 \in [1, t-\tau]} \frac{\hat{\beta}_{t_0,t}}{\hat{\sigma}_{\beta_0,t}}$

A few differences with respect to the original, one-bubble model can be noted:

- The regression is changed, there is no more a dummy $D_t[\tau^*]$ variable. Instead, the regression starts at $t_0 \in [1, t-\tau]$ and ends in $t \in [\tau, T]$.
- $SADF_t$ computes the supremum in a double nested loop for every possible value of t_0 and t which are the indexes to segment the series.

The aforementioned characteristics allows SADF to vary provided that it is not computed just for one time (T), but instead for many (every value of $t \in [\tau, T]$).

Getting into the details of the augmented Dickey-Fuller statistic, the confidence value should be set from the sample to yield the best results. In [Pet11] the authors refer to a value close to 4% to deliver the best performance but values between 1% and 5% are recommended. 5% was used in this case.

Implementation notes: Lopez de Prado offers in his book almost the entire algorithm (see chapter 17 of [Pra18]), but leaves behind the outer loop which allows to move forward the $SADF_t$ for each $t \in [\tau, T]$. A modification was introduced in code to avoid excessive and inefficient computation in the inner loop: an upper bound for the window was introduced to so that the range of t becomes $[max(\tau, t - \Delta t_{max}), T]$. Although the asymptotic computational complexity of this series generation is high ($O(n^5)$ at least, see [Pra18] for a detailed analysis), one way to saturate one order is to use Δt_{max} which should be cautiously selected to account for long lasting bubbles.

On a separate note, the algorithm allows researchers to introduce a constant, a linear and a quadratic polynomial regression depending on the type

of series to analyze. All of them were computed in favor of feeding the model with more data and experiment what yields the best results.

Finally, a recommendation in the article [Pet11] and discussed in [Pra18] has been implemented. Instead of using raw prices, the algorithm takes as input log-prices. Conceptually, by applying the logarithmic transformation, the variance of the process gets stabilized and the heteroscedastic assumption about the data can be fulfilled. Long time series are expected to change their price levels but that does not directly cause a regime change and the transformation facilitates to distinguish them. A regime change implies a transition to an explosive behavior, e.g. exponential behavior is identified.

Three pictures are shown to illustrate this index: 36, 37, and 38. They contrast the index with raw bitcoin close price, fractionally differentiated bitcoin price and log prices respectively. Note the progression in the pictures and how the volatility of the log-prices series correlates with SADF spikes 38.

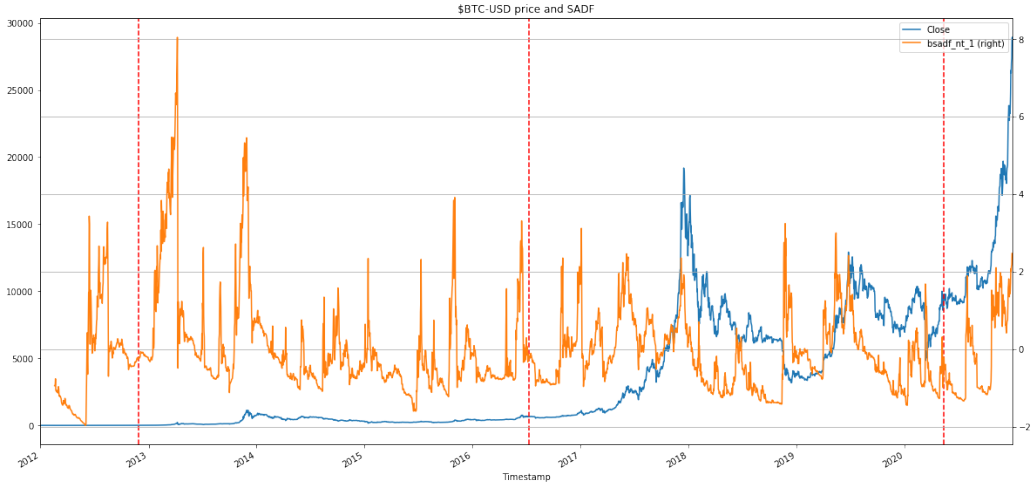

Figure 36: Raw close prices and SADF over time.

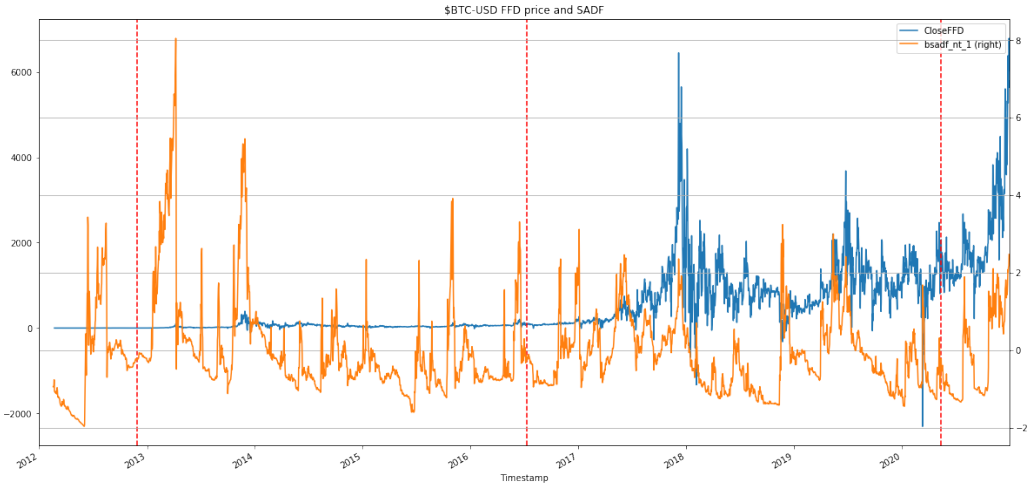

Figure 37: Fractionally differentiated prices and SADF over time.

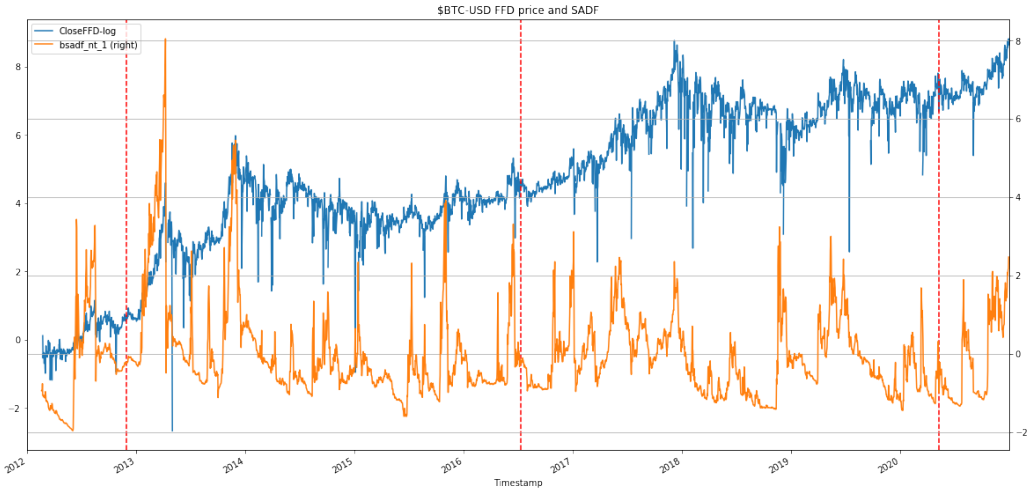

Figure 38: Close log-prices and SADF over time.

5.1.4 Social features

So far, there were explained traditional financial features, then Bitcoin and bitcoin features, and finally structural break features to introduce SADF computation. Nevertheless, nothing has been said about the agents that operate in this market so far. Therefore, what follows shows some figures to illustrate the data gathered from the two sources: Google Trends and Alternative.me.

First, in figure 39 the bitcoin price series is shown together with the

interest index that Google Trends offers. Note that a linear interpolation was used to fit daily values because social data was available only a weekly basis. We can see that there is a better fit between the curve shapes in the third regime and after the third halving on.

Finally, in figure 40 the sentiment index is displayed for the last two periods. Data for the first two two periods was not available. It is worth mentioning that this series is the result of fused data from Twitter, Reddit and other forums which is definitely richer than Google Trends interest index. Google Trends is just used to compensate social information during the first two periods.

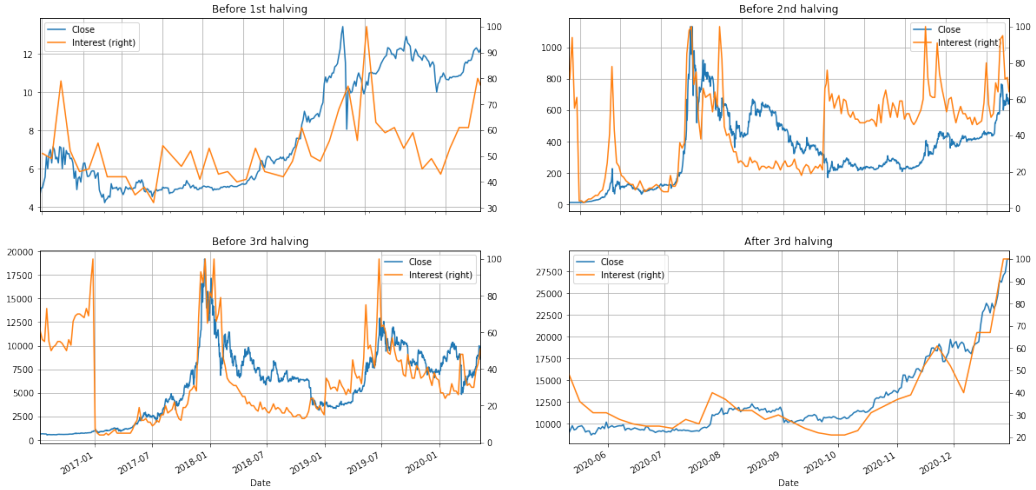

Figure 39: Evolution over each bitcoin regime of the Google Trends interest in the "bitcoin" keyword together with the close price of bitcoin.

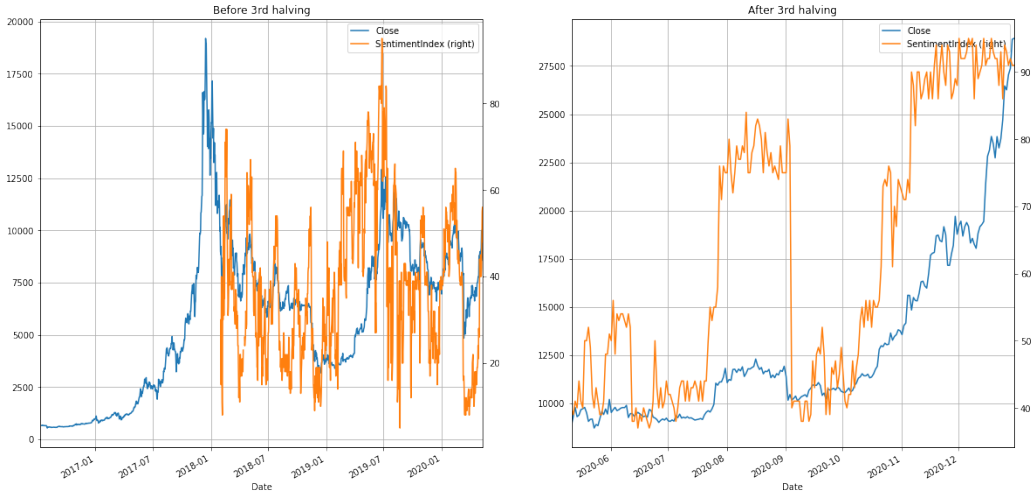

Figure 40: Evolution over the last two bitcoin's regimes of the Alternative.me social interest of bitcoin together with the close price of bitcoin.

5.2 Pipeline

This research project follows a pipeline and general strategies proposed in Lopez de Prado book ([Pra18]) which do not differ significantly from any other machine learning pipeline although there are a few key points worth to mention and explain because they are not intuitively derived by inspection. As commented before, there are two models:

- Primary model: it outputs the buy and sell signals based on a certain set of features. We will further discuss this model in section 5.2.1, however it involves a momentum based strategy computed out of the crossing events of two different speeds moving average signals (as initially introduced in 3.2.1).

- Secondary model: the first model outputs buy and sell signals based on prices, volumes, and other series. This second model is a machine learning model that will define the size of the position. The machine learning model will be explained in detailed section 5.2.2.

- Training: training a machine learning model that uses time series as features requires special care to avoid leakage. Section 5.2.3 describes the involved methods to reduce the impact of leakage while preserving the performance of the model.

- Feature importance: it is expected that as part of the research process, more features than there are really required to develop the secondary

model are used in the training stage. Once model's performance is satisfactory, features could be removed to obtain the right trade off between model complexity and required data. See section 5.2.4 for further details.

- Bet sizing: the secondary model outputs the confidence on the bet that the primary model provided. This stage helps the financial analyst execute trades with consolidated sizes and avoids prompt operations that are not worth because transaction rates significantly damage the operation. See section 5.2.5 for further details.
- Back testing: in this stage the strategy is stressed under different circumstances to evaluate its performance metrics. It is not a research tool to guide the training of the aforementioned models. Instead, it should be used as a benchmark tool and help the researcher to quantitatively compare different strategies. See section 5.2.6 for further details.

In figure 41 the previous stages are displayed in favor of increased clarity. The training and feature selection stages are considered inside the Secondary model process.

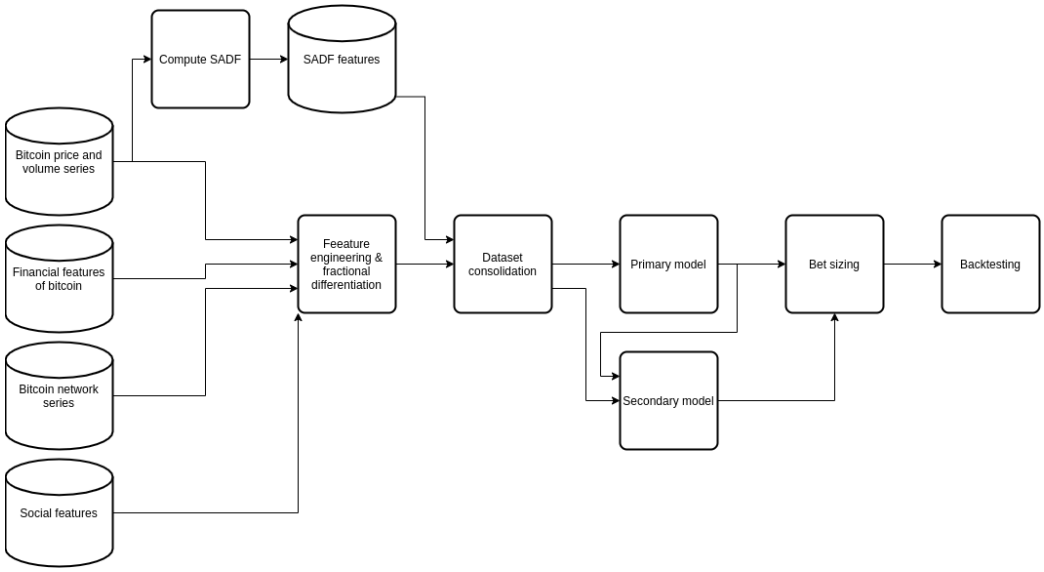

Figure 41: Financial machine learning pipeline used in this research project.

5.2.1 Primary model

The primary model consists of computing two moving average signals with different time windows. The longer the time window, the more samples are

averaged and consequently the slower it varies with price variations. Having two signals, one *fast* and one *slow* provides information of how short and long term price movement vary one with respect to other. In particular, the strategy takes advantage of the crossing points of both signals. When the fast signal crosses above the slow signal, a buy event is generated. When the slow signal crosses above the fast signal, a sell event is generated.

Note that from a daily sampled, positive and real signal as the bitcoin close price in USD is, we obtain another two daily sampled, positive and real signals (fast and slow). The latter two signals generate when they cross the events of the primary model. These events are not equally spaced anymore, and the series is categorical, its values are $\{1, -1\}$ which represent $\{buy, sell\}$ respectively. See figure 42 to show these events over the bitcoin price series.

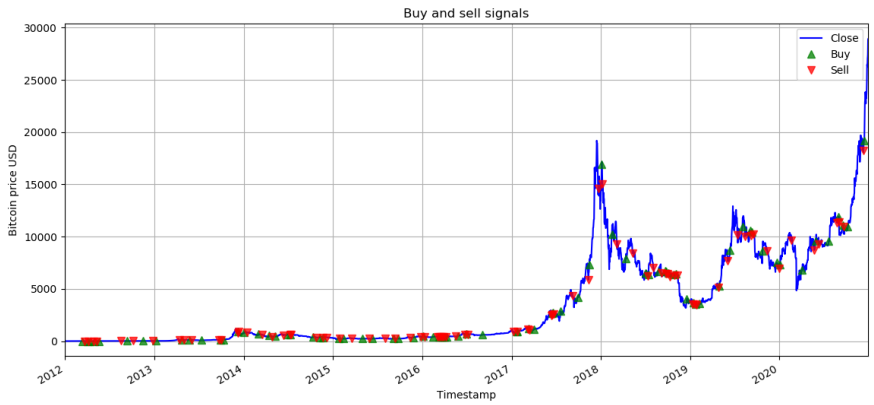

Figure 42: Buy and sell signals over the price series.

The process of obtaining the signal with the bets is called *labeling* because it generates a series of *labels* that determine a concrete action: buy or sell the position. Next, *metalabeling* process comes. It consists in providing a probability to each *label* which will be used to size the bet. To assess whether the *label* is correct or not, the triple barrier method is used:

1. Define a profit taking and stop loss rate for which a buy and sell signals will be considered valid. For a given price and time, a new greater value and lesser value are defined based on both rates before.

2. Define a time constant h (expressed as a positive and integer multiple of the sampling rate, i.e. number of days) that determines for each label, a new time stamp ahead.

3. Determine whether the price signal hits the greater price or the lesser price before reaching the time stamp ahead of h periods from the label's event. When:

    - A 1-valued label gets a cross with the greater price barrier, the metalabel is 1 to indicate a positive label.
    - A -1-valued label gets a cross with the lesser price barrier, the metalabel is 1 to indicate a positive label.
    - A 1-valued label gets a cross with the lesser price barrier, the metalabel is 0 to indicate a positive label.
    - A -1-valued label gets a cross with the greater price barrier, the metalabel is 0 to indicate a positive label.
    - When both 1 and -1 valued labels do not get a corresponding cross with any of the price barriers, the return sign between the price at h sample periods ahead of label's time stamp and the label's price is used. If the sign of the return and the label match, the metalabel is 1, otherwise it is 0.

Labels and metalabels are fundamental series to build the secondary model. The secondary model is a classifier that is trained with metalabels. The predicted probability will help to size the bet on each label. In mathematical terms: $Metalabels = f_{(Labels,\ldots)}$ where f represents the secondary model.

5.2.2 Secondary model

The metalabelling process allows to algorithmically determine the positiveness of the labels. Note that metalabels should not be included as input features of the model because they belong to the future of the label and one model doing that will not be plausible to implement on a real time system. Initially, all features in the feature engineering section 5.1 will be used to build this model.

This research project will use bagging trees. In chapter 6 of [Pra18] there is a discussion about ensemble models and three are compared: bagging trees, random forests and boosting. Boosting is generally superior in terms of bias and variance results with respect to the others but it comes with, in my opinion, an important penalty: the training process should be done sequentially whereas bagging can be parallelized. The author is explicit about the other positive and negative aspects such as tendency to over-fitting or under-fitting, by stating that those claims are relative to how careful the researcher is when developing the training pipeline and curating data.

The focus of this research is not to get into the details of one machine learning model or the other, but to explain the most relevant characteristics of the one to be used and how its characteristics are used in favor of the research problem at hand. Bagging ensembles rely on multiple, B, decision (or regression but in this case we need decision) trees whose outcome is averaged to reduce the high variance of each tree. Decision trees tend to be deep, in favor of bias reduction, but each tree variance is high. Increasing the number of B trees does not imply an immediate shift towards overfitting what makes it an appropriate hyperparameter to adjust in favor of variance reduction. See section 8.2.1 of [JWT17] for a better description of how bagging ensembles work.

Provided that only one dataset is available, a bootstrap sample method is used. Bootstrapping consists of sampling the base dataset with replacement to generate new datasets that will be used to train each decision trees. Each new dataset is expected to be biased and the variance between datasets will be diminished when averaging, or bagging, the results of the trees. Bootstrapping can also be done with the features of the dataset. See section 5.2 of [JWT17] for a concise but comprehensive description to bootstrapping.

The general bootstrapping method for samples assumes that all samples are IID. This is not our case when using the triple barrier method. Note that some samples might co-occur when time windows expand from the label timestamp to h sample periods ahead. In high volatility events, or with high values of h, we should expect events, or labels, to happen while another one is being evaluated. Weighing these labels with respect to unique and without any overlap events is important to get the most out of the bootstrapping sampling procedure. Following Lopez de Prado's analysis, concurrent labels are defined as those that share at least one return attribution in the triple barrier method. Let the concurrency c_t at time t be defined as:

$$c_t = \sum_{i=1}^{I} l_{t,i}$$

Where $l_{t,i}$ is the i-th label value that occurs at t and I is the number of label which co-occur at t. Labels start at a certain time t but span a number of sampling periods that could be h or the time difference with respect to one of the horizontal (price) barriers crossing events. Consequently, a label will contribute to at least two different and consecutive timestamps and up to h consecutive timestamps.

Let the uniqueness of a label i at time t be defined as:

$$u_{t,i} = \frac{l_{t_i}}{c_t}$$

And the average uniqueness will be defined as the averaged uniqueness over the label's lifespan:

$$\bar{u}_{t,i} = \frac{\sum_{t=1}^{T} u_{t,i}}{\sum_{t=1}^{T} l_{t,i}}$$

Label's average uniqueness allows a smarter bootstrap sampling method because it can be used to prioritize events with higher uniqueness over the ones with less uniqueness because the sole fact that they are weird in the dataset.

Lopez de Prado proposes a change to the above uniqueness sample weight. He introduces the bet return also to account for an average of all the bets simultaneously running. To do so, a new weight w_i for label i is proposed:

$$\tilde{w}_i = \Big| \sum_{t=t_{i,0}}^{t_{i,1}} \frac{r_{t-1,t}}{c_t} \Big|$$

$$w_i = \frac{\tilde{w}_i I}{\sum_{j=1}^{I} \tilde{w}_j}$$

Moving from $\tilde{w}_i$ to w_i involves a scale factor to assure: $\sum_{i=1}^{I} w_j = I$. When training the model, the training set / fold (when using cross validation) will incorporate the weight with combined return and uniqueness attribution to differentiate rare as well as rare high-return events from the other concurrent and low-return events. Lopez de Prado also proposes a sequential bootstrap method which updates the probability of each sample in the series every time a row is drawn. This process yields train sets closer to IID but this is not part of this research pipeline.

5.2.3 Training

The training stage involves tweaking both the primary model and the secondary model. The primary model has the following hyperparameters:

- Fast window period: the number of periods to average the price signal for the fast moving average signal.
- Slow window period: the number of periods to average the price signal for the slow moving average signal.
- Stop loss ratio: the price ratio when a label occurs that will determine the lower price barrier to determine the triple-barrier frontier.

- Profit taking ratio: the price ratio when a label occurs that will determine the higher price barrier to determine the triple-barrier frontier.
- Volatility time window: when applying the triple barrier method, a volatility series is computed out of daily returns. Volatility series is the result of an exponentially weighted moving average which takes the time window long samples and computes the standard deviation of it. This series is used together with the minimum return.
- Minimum return: the primary model generates labels which involve really low returns and can be considered noisy samples. This threshold works as a filter to discard these labels before they become part of the primary model output.

On the other hand, we have the secondary model which is a classifier. This classifier will be used to determine the size of the bets. In section 5.2.2 it was mentioned that it will be a Bagging classifier whose parameters are:

- Number of estimators: the number of trees to train (B) and then take the majority vote for each new estimation.
- Number of samples per estimator: the fraction of samples in the dataset to sample with or without bootstrap. This value is set to the average uniqueness of all labels.
- Use bootstrap: whether to use or not bootstrap sampling. This value is set to true. On top of it, when fitting, the particular label uniqueness is used.
- Maximum of features to use: out of all predictors, how many features per estimator to use. No bootstrap is used at the predictor level, just a sampling.

Unless explicitly mentioned, all hyperparameters are optimized. For the primary model, the following criteria is used:

- For certain combinations of hyperparameters, there might not be enough labels and metalabels to train the secondary model training process. Also, the primary model might output a too imbalanced set of labels and metalabels which end up causing trouble when training because of lack of one class of metalabel. The generated model out of that hyperparameters are discarded.

- For certain combinations of hyperparameters, there might be very little amount of samples. These combinations are discarded.
- Using a quite simple secondary model of the same nature but without optimized hyperparameters, we choose the combination of primary model hyperparameters that yield better secondary performance.

Performance in the secondary model is evaluated with negative logarithmic loss. Typically, classifiers use F1-score because they provide a good balance between precision and recall (its the harmonic mean between both). In this case, we are mostly interested in selecting models based on the predicted probabilities because that is used to size bets.

Secondly, once the primary model hyperparameters have been selected, a full model optimization for the secondary model is done. Again, negative logarithmic loss is used to determine the best model. Following Lopez de Prado recommendations in chapter 7 of [Pra18], purged K-fold cross validation with embargo is used to train the secondary model.

Purge: when the classification output at time t, i.e. metalabel, depends on the value of the predictor(s) at two or more sample values, we have an inter-time dependency of the output with several of the predictors that might lead to leakage if they are not properly purged. The purge strategy consists on removing the predictor and classification outputs that are concurrent in adjacent training and test folds. There are three situations that would make two labels concurrent:

- Label i starts inside the triple barrier period of label j.
- Evaluation of metalabel i ends inside the triple barrier period of label j.
- Evaluation of label j starts and ends inside the triple barrier period of label i.

In addition to this concurrency effect, there is another effect that produces information leakage between train-test splits. Suppose a label that is assigned to a test fold and it is generated from data that is spread in multiple labels both in train and test folds even though there is no label concurrency involved, there will be leakage. In this case purge is also required to mitigate leakage.

Embargo: when there is no clear time span that is required to ban from the adjacent training and test folds to avoid leakage due to the nature of the classification output generation process, the embargo technique can be used. It consist of removing a percentage of samples in the train fold that are right after the test fold beginning. Note that there is no need to remove samples from the end of the test fold when it is adjacent to another training fold because those samples will simply not contribute to train the model for the first set of labels. The percentage is usually low, e.g. 1%, and provides enough data pruning to run the training stage without noticeable leakage.

5.2.4 Feature importance

Lopez de Prado's eloquent words in section 8.2 of [Pra18] are very appropriate to illustrate what this section is about.

"Hunters do not blindly eat everything their smart dogs retrieve for them, do they?"

As a financial machine learning researcher once we are satisfied with the performance of a machine learning model it is advised to understand which, how and when features contribute to improve the performance of the model. The author focuses on the *importance* of the features with and without substitution effects. In this research pipeline, only a method to consider substitution effects is implemented.

The method is called Mean Decrease Accuracy and it uses model's estimation accuracy as guiding principle. Description of the method follows:

- Let X of be the feature matrix and let Y be the output vector.
- Let $X_1, X_2, X_3, ..., X_m$ be the columns of X.
- Fit via the desired training process $m + 1$ models and measure their accuracy. One model will be fitted with X as is. And the other m models will be fitted with one randomly permutated column X_i.
- For each of the aforementioned m models, compute the relative loss in accuracy with respect to the base without permutation model.

This method allows to create a rank in which once can inspect the relative loss of performance measured in accuracy, but it could also be F1-score or negative log-loss when working with classifiers. Having a high value means that the predictive importance of the feature is relevant. Even though this method is flexible and adaptable to multiple types of models as it is based on out of sample performance, it comes with some important drawbacks:

- It is relatively slow because it requires the training of $m+1$ models and the evaluation of their performance. When this is done with purged K-fold cross validation with embargo*ed* datasets, it will definitely take time.

- It is susceptible to *substitution effects*. The effect can be described with two or more features that are highly correlated. The performance loss will be similar so a researcher might decide to remove them but with that the overall predictive capacity diminishes more than just keeping one of the features in the set under study.

- A possible result is that all features are detrimental or unimportant for the model what is somewhat hard to interpret.

Feature orthogonalization could help to reduce the substitution effect. Two procedures are proposed. First, a direct application of Frisch–Waugh–Lovell theorem analysis can be used. Each feature is analyzed individually via a linear regression and residues are inspected to derive relative importance. See chapters 2 and 3 of [DM03] for an in depth description of the theorem and applications. Secondly, Principal Component Analysis, i.e. PCA, as suggested in section 8.4.2 of [Pra18] can also be used to determine the set of features whose eigenvalues in the orthogonalized space are greater. Alternatively, the feature space could also be reduced in favor of faster convergence of the machine learning models under use.

Once all features are ranked, the researcher is required to select which features to drop and which ones to keep. The final set of features will be used to build a new model. Heuristic rules are used to drop features but as mentioned before researchers should be aware of the substitution effect. In this research project, the applied rule applied consists in computing the mean loss of negative log-loss and keep those features that produce higher than the mean loss in performance.

Furthermore, researchers often face the requirement of explaining the economic mechanism that generates the excess return of the strategy with respect to a benchmark. Understanding which features contribute more to predict with increased confidence labels is reasonably simpler when extra unimportant features are removed. This stage of the process together with feature engineering are very important to understand how the model behaves.

5.2.5 Bet sizing

Labels and metalabels will be used together with other features to train a secondary model whose output estimated probability for each new label will be used to size the bet for that event. One way to do this is to scale the predicted probability, by the budget total and get value in currency for the bet. However, if we did so, there would not be any awareness about other concurrent bets, so we might misuse the available budget to run other concurrent bets. Lopez de Prado in chapter 10 of [Pra18] proposes the following:

1. Let $p_{(x)}$ be the probability of label x that takes the one of the values in $[-1, 1]$.
2. Run a statistical test where $H_0 : p_{(x=1)} = 0.5$ with the statistic $z = \frac{p_{(x=1)} - 0.5}{\sqrt{0.5(1-0.5)}} = \frac{p_{(x=1)} - 0.5}{0.5}$
3. Let the bet size m be: $m = 2\Phi_{(z)} - 1$ where $\Phi_{(z)}$ is the cumulative distribution function of the standard normal distribution. $m \in [-1; 1]$

Once we have a vector of m values, which has a one to one relationship with each label x, we can average the bet with those concurrent triple barrier windows as the come in the pipeline. Averaging does not change the bet size for open windows, but changes the size for new bets that overlap with open windows.

Finally, a portfolio manager might also consider bet size discretization by means of setting an integer number of equally sized budget partitions. The discretized value of the bet after the average is $m^* = round\left(\frac{m}{d}\right)d$ where $d = \frac{1}{Numberofpartitions}$. And the only remaining step when running the strategy is to scale the budget by m^* to have the final bet size in currency. The goal of this step is to reduce jitter which causes overtrading. Typical number of partitions are below ten.

5.2.6 Back testing

This section focuses on explaining how this exercise would be carried out, the differences with respect to the implemented back testing strategy and how benchmarking metrics are affected. Also, metrics are explained.

Strategy The strategy will start at certain date with an initial budget of $1,000,000 dollars (USD) to spend as both the primary and secondary

model dictate. Every day, new features are gathered and processed together with the historical data. Feed the selected set of columns that conform the secondary model and the primary model determines whether a new label and metalabel should be created and then the secondary model proposes a probability for that metalabel. The bet sizing transformation will scale and determine a dollar level for the bet. Depending on the position the primary model proposes and the bet size, the following steps are conducted:

- When placing a long position and the bet size is not zero:
    1. Bet size dollars are taken from the portfolio at time t.
    2. Bet size dollars are used to buy bitcoins. The BTC/USD rate is the price at the moment where the order takes place. From the bet size dollars, the exchange buy fee is discounted from the bet size.
    3. We hold the long position until one of the three barriers of the triple barrier boundaries is crossed by the price path. The primary model defined by hyperparameter optimization a value for the stop loss, the profit taking and the holding period for the triple barrier, those values are computed based on the bitcoin price level at t.
    4. When the price series crosses any of the barriers, the long position is closed and a sell fee is paid to the exchange.
    5. At this point, the value of the portfolio would be damaged because of the buy and sell fees and benefited from the expected rise in bitcoin valuation measured in dollars. It is worth pointing out that the bet could end up having a net negative effect on the value of the portfolio.
- When placing a short position and the bet size is not zero:
    1. Bet size dollars are taken from the portfolio at time t.
    2. A loan in bitcoins is requested. The bet size amount is used to pay the loan and pay the collateral amount.
    3. Those bitcoins are sold, consequently we get their dollar valuation minus the sell fee of the exchange.
    4. We hold the short position until one of the three barriers of the triple barrier boundaries is crossed by the prices path. The primary model defined by hyperparameter optimization a value for the stop loss, the profit taking and the holding period for the triple barrier, those values are computed based on the bitcoin price level at t.

5. When the price series crosses any of the barriers, the short position is closed. In this case, bitcoins are bought to complement the collateral amount and return the loan interest.
6. At this point, the value of the portfolio would be damaged because of the buy and sell fees and the loan interest. However, it would benefit from the price drop and the short position.

The aforementioned procedures are iterated over the lifespan of the strategy and must be monitored as any other strategy. Also, it is worth mentioning the risks involved in the operation:

- If the price path rapidly crosses the upper price level barrier before a new price sample comes in (when evaluating sell orders) or it crosses the lower price level barrier before a new price sample comes in (when evaluating buy orders), there is an extra cost incurred due to a lack of live price monitor. It can be mitigated by having such monitor and an automatic order placing system.
- If having automatic control to place orders, the price path can simply not be enough to pay off the fees, even though the net return the price only is positive (accounting the transaction cost reduces the net profit from the trade).

The portfolio manager will also have to consider how the collateral valuation varies with time. Note that many bets time windows can overlap which could cause budget starvation if highly confident bets co-occur. Consequently, the collateral valuation will be affected by all short and concurrent positions and with the price variations of the asset under management and will reduce the effective return of each bet because of the compromised savings to pay back the loan. However, it is worth mentioning that in case of a an undesired rise in prices, the collateral will help to reduce the impact.

In this research project, we make the following assumptions:

- Buy and sell exchange fees might vary over time. For simplicity, they are kept the same for the entire back testing period.
- Loan interests vary their every day. For simplicity, they are kept the same for the entire back testing period.
- Collateral budgets are not considered. Instead, a penalty to the loan interest is added.

- Because of considering the collateral budget as an extra interest, the value of the portfolio does not benefit from intra bet windows bitcoin price variation.
- Orders might not get processed during high price corrections. Those scenarios are not contemplated either.

In general, all of these simplifications will imply higher performance which is perceived as excess returns.

As a reference, table 2 contains the fees and rates involved in the strategy.

Fees	
Fee	Value
Buy	0.10%
Sell	0.10%
Loan interest	5%

Table 2: Financial metrics benchmark.

Buy and sell fees were take from Binance [Bin] and loans from DefiRate [Rat] where an average has been done and a 1% extra has been added.

Benchmark A base strategy is required to provide a reference and compare against the output strategy of this research. Buying bitcoins the first day and holding them until the last day of the testing period will be used. This is an obvious choice because machine learning enhanced momentum is used over this asset and would set a valid benchmark for this strategy.

Metrics The secondary model will be measured with two metrics which are appropriate for classification problems. The F1-score as a measure of the accuracy of the classifier. Then, to measure classifier performance, negative logarithmic loss will be used to assess how well predicted probabilities adjust to validation labels.

The strategy will be evaluated by the following metrics which also cover different types:

- Performance metrics:
  - Win loss ratio: the ratio between positive returns and negative returns. The bigger the ratio, the better.
  - Win rate: the ratio of positive returns over the total number of returns. The bigger the ration, the better.

  - Average return: the arithmetic average of all returns. It is an estimate of the true mean return.

- Efficiency metrics:
  - Sharpe Ratio: measures the ratio between the average sample returns minus a risk free interest rate[2], and the sample standard deviation. It provides an estimate of the true Sharpe Ratio.
  - Probabilistic Sharpe Ratio: expresses the probability that estimated Sharpe Ratio (see above) is higher than a benchmark Sharpe Ratio. See below for an in detail discussion.
  - Sortino ratio: it is similar to the Sharpe Ratio in the sense that provides a a risk adjusted return ratio but it penalizes negative returns. It is generally a better choice to compare skewed return distributions. As the Sortino ratio increases the strategy becomes more appealing for a given benchmark target return. See [RH15] for an analysis and explanation of the ratio.

- Implementation shortfall metrics:
  - Return on execution costs: ratio between the sum of all net returns (after operation costs) and the sum of all transaction costs. The bigger the ratio is, the more confident a portfolio manager will be about the strategy. Transaction costs cannot be known beforehand and high values cover for increased costs at execution time.

- General metrics:
  - Correlation to underlying: the correlation between the strategy and the underlying investment universe. As it gets closer to one or minus one, the closer it is to the benchmark strategy.
  - Volatility: the standard deviation of the returns. It is an estimate of the true return standard deviation.

Moreover, it is important to discuss the Probabilistic Sharpe Ratio. It was introduced in [BL12] which compliments the information of the estimated Sharpe Ratio which is usually provided in financial benchmarks. For both strategies, the benchmark and the strategy under test, the true mean and

[2]Using US treasury yield annual curves [Tre], one can observe that risk free interest rates vary from 0.05% to 2.74%. It means that the daily risk free interest rate varies from $2.73x10^{-6}$ to $7.4x10^{-5}$. It was simply considered zero.

standard deviation of the Sharpe Ratio is and will remain unknown. Back testing over past or generated data will only provide an estimate that adjusts to the sample consequently, the Probabilistic Sharpe Ratio helps to understand which is the probability that the estimated Sharpe Ratio is bigger than a set of target ratios.

Furthermore, the estimate of the Sharpe Ratio mean comes from non IID return samples. Hence, Bailey and Lopez de Prado introduced an adjustment by skewness and kurtosis to the standard error used in the statistical test. Then, it is appropriate to use a series of target Sharpe Ratio to compare with and determine the probability that the estimated Sharpe Ratio of our strategy is bigger for proposed target ratios. As a reference, the Sharpe Ratio and the Probabilistic Sharpe Ratio formulas are copied below.

$$\hat{SR} = \frac{\hat{r}}{s_r}$$

Where:

- $\hat{SR}$ is the estimated Sharpe Ratio.
- $\hat{r}$ is the average return.
- $\hat{s_r}$ is the sample standard deviation of the returns.

$$\hat{PSR}_{(SR^*)} = \Phi\left(\frac{(\hat{SR} - SR^*)\sqrt{N-1}}{\sqrt{1 - \hat{\gamma_3}\hat{SR} + \frac{\hat{\gamma_4}-1}{4}\hat{SR}^2}}\right) \tag{7}$$

Where:

- $\hat{PSR}_{(SR^*)}$ is the Probabilistic Sharpe Ratio for the reference Sharpe Ratio SR^*.
- N is the number of returns in the sample to compute the index.
- $\hat{\gamma_3}$ is the sample return skewness.
- $\hat{\gamma_4}$ is the sample return kurtosis.
- $\Phi()$ is the cumulative distribution function of the standard Normal distribution.

Taxation A potentially relevant "transaction cost" relates to taxes. For example, in the US, and depending on the tax bracket, federal taxes for short-term capital gains are taxed at up to 37%[3]. Capital gain taxes would disfavor active trading on a highly volatile asset in our results.

[3]See 26 CFR 601.602: Tax forms and instructions.

6 Results

6.1 Model results

Training The strategy has been optimized using a random search in the hyperparameter space as commented in section 5.2.3. Primary model results are shown in table 3, and secondary model results are shown in table 4.

Primary model	
Parameter	Value
Moving average fast window length	5
Moving average slow window length	30
Profit taking band	0.06
Stop loss taking band	0.02
Triple barrier window length	3
Minimum return	0.005
Volatility window length	50

Table 3: Primary model hyperparameters.

Secondary model	
Parameter	Value
Number of estimators	720
Maximum features ratio when bagging	0.2936
Average uniqueness	0.9502
Cross validation splits	6
Embargo	1%

Table 4: Secondary model hyperparameters.

Labels The output of the primary model is summarized in table 5 where we also show the labels and metalabels count. Also, in 6 a contingency table between labels and metalabels is shown. We can see that even though labels are balanced, there is a bias towards negative metalabels (i.e. do not take the bet) based on the triple barrier parameters.

Labels	
Value	Count
1 (buy signal)	80
-1 (sell signal)	79
Metalabels	
Value	Count
0 (unreliable label)	103
1 (reliable label)	56

Table 5: Primary model labels and metalabels.

Label / Metalabel	0	1
1	51	29
-1	52	27

Table 6: Contingency table between labels and metalabels.

Feature importance Table 7 shows the mean loss in performance per feature after applying the method explained in 5.2.4. Note that only those features with mean loss in performance above the mean loss are shown. See figure 43 for an in detail mean loss of performance and standard deviation per feature permutation.

It is interesting to point the groups of features that remained in the final model:

- Price and volume features with fractional differentiation.
- SADF derived indices with different regression models (constant, linear and quadratic) and with one, two and three lag periods for the autocorrelation.
- Volatility and logarithmic returns.
- Market capitalization.
- Fees, transfer volume and days till halving. These are features related to the bitcoin ecosystem.

However, none of the social features seemed to produce a high impact in the final model.

Feature importance		
Feature	Mean perf. loss	Std. dev. of perf. loss
CloseFFD	0.039714	0.026393
HighFFD	0.018739	0.019605
LowFFD	0.002177	0.001215
OpenFFD	0.006869	0.003897
Volume_(BTC)-log	0.032430	0.015899
Volume_(Currency)-log	0.020350	0.008696
bsadf_ct_1	0.002128	0.002060
bsadf_ctt_1	0.003910	0.003278
bsadf_ctt_2	0.004907	0.004139
bsadf_ctt_3	0.003264	0.004178
bsadf_nt_2	0.003447	0.004077
daysTillHalving	0.003387	0.003194
fees-mean-log	0.004142	0.002743
log_ret	0.013930	0.006140
market-cap-log	0.002412	0.001501
transfer-volume-median-log	0.006073	0.006431
trgt	0.003003	0.007776
vol_15	0.008678	0.003120
vol_5	0.020516	0.015831

Table 7: Feature performance loss of the secondary model. Only selected features are displayed.

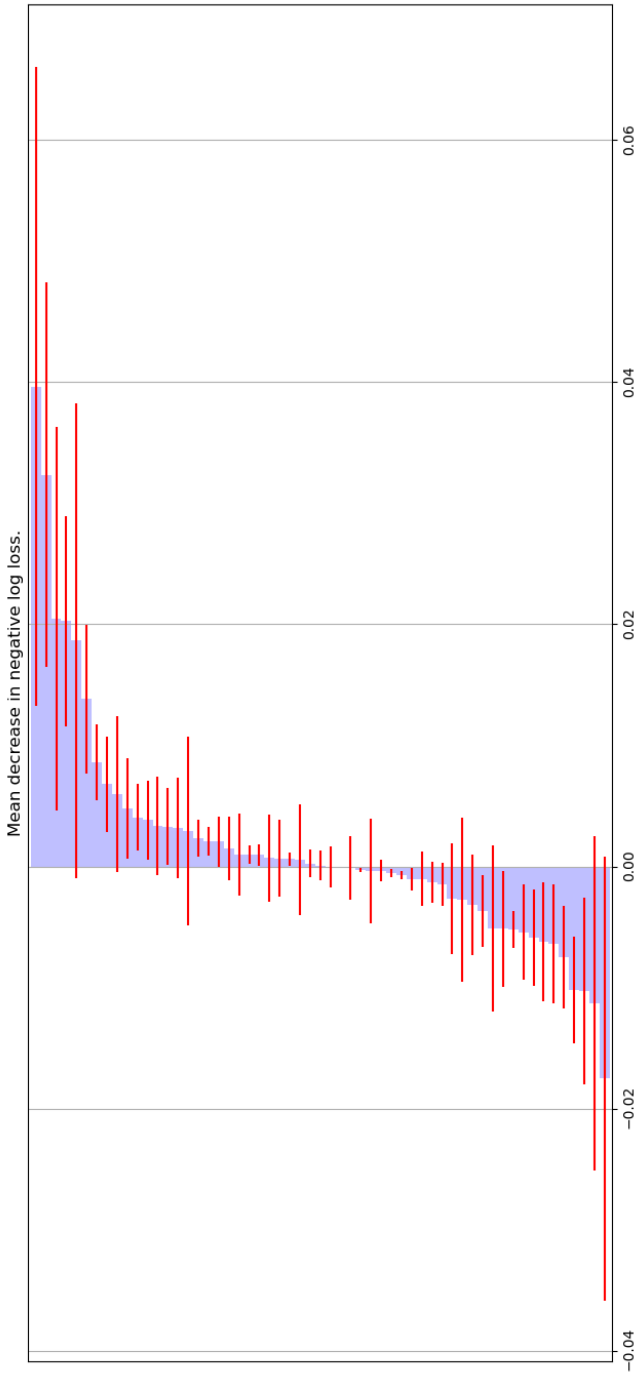

Figure 43: Displays the mean feature importance loss in the bars. Each bar shows one feature. From the top to the bottom, there is one bar per feature in table 7 and the rest are unimportant features. The standard deviation across cross validation folds is shown as a red line centered in the mean of each feature.

A new model was retrained using only these features. That model yields the results in paragraph *Performance metrics*. Also, as commented in 5.2.4 it is important to explain how these features are able to produce excess returns. It can be seen that price related features are the ones that generate the higher mean of performance loss and it means that a lot of market information is in prices already. Secondly, we interpret that transacted volumes, both in currency and in bitcoins provide a lot of information about market tendencies. As outlined in section 5.1.3 SADF related features determine when regime changes in price series occur and even though that information is in prices, SADF offers a series that provides direct information about changes towards explosive behavior. It is not surprising that the day count until halving provides information and that is also connected with the two volatility series. We have seen that close to halvings when issuance changes and produces a considerable change in the network and that definitely affects market volatility. Market capitalization offers an overall high level market metric which provides an aggregated indicator of the market and similarly does the transfer volume which accounts for the amount of currency that is flowing. Making partitions over these features by means of a tree based model as bagging ensembles are allows to generate complex relationship between them and identify patterns.

Performance metrics Table 8 shows the F1-score and negative log loss of the retrained secondary model.

Secondary model performance	
Metric	Value
F1-score	0.1574
Negative log loss	0.6031

Table 8: Secondary model classification metrics.

F1-score evaluates model's accuracy when classifying samples. Negative log-loss evaluates model's accuracy of the the estimated probabilities. The secondary model will not be used as a classifier. Instead, predicted probabilities will determine the bet size.

6.2 Strategy results

Table 9 shows the results of the benchmark and the strategy under test. We can see that the estimate on the Sharpe Ratio is bigger for benchmark than the strategy under test. Later, evaluation of the Probabilistic Sharpe Ratio

will be used to analyze the confidence on the Sharpe Ratio estimation for different reference values. Sortino ratio is bigger for the strategy under test which is aligned with the win loss ratio observation. There are no considerable differences in other indexes with the exception of the volatility which has a rough difference of two magnitude orders.

Two indices were not computed for the benchmark strategy: correlation to underlying (because it would be equal to 1) and the return on execution cost. The latter will just add noise because only two trades will be executed: one at the start of the time series and one at the end. Finally, because of the low observed correlation value we can state that both strategies are doing quite different things.

Financial metrics		
Metric	Buy and hold	Strategy under test
Sharpe ratio	2.2156	1.2289
Sortino ratio	2.9138	5.7169
Win loss ratio	1.1171	8.0973
Win rate	0.5637	0.5833
Average return	0.0073	0.0065
Volatility	1.2062	0.0352
Correlation to underlying	-	0.0454
Return on execution costs	-	21.2264

Table 9: Financial metrics benchmark.

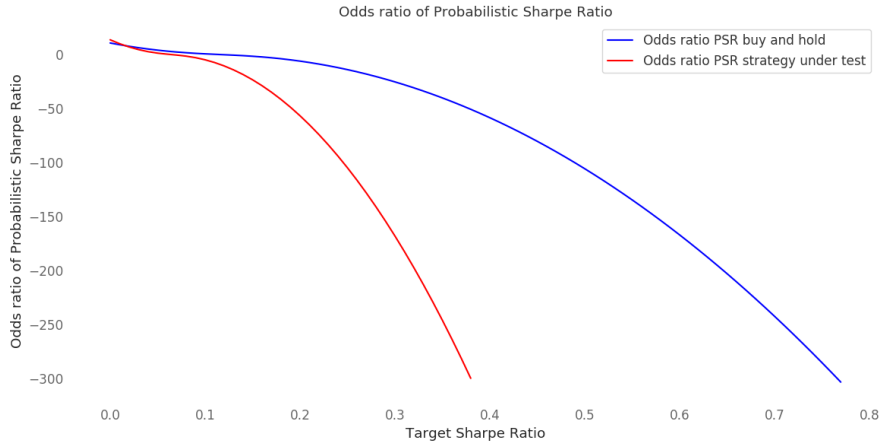

Figure 44: Odds ratio for the Probabilistic Sharpe Ratio of the buy and hold strategy and the strategy under test.

Because the Probabilistic Sharpe Ratio is compared against a sequence of target Sharpe Ratios (0.01 increments between 0. and 1.), it is better shown in terms log-odds ratio of the PSR values in figure 44. It is observed that for both strategies the odds quickly drop to quite low values below Sharpe Ratios above 0.1. This is explained by analyzing the Probabilistic Sharpe Ratio equation (7). The denominator of the test statistic, i.e. the standard error, is simply great because of the extremely large kurtosis and skewness coefficients that both strategies exhibit. See table 10 and remember that the normal distribution has a skewness of 0 and a kurtosis of 3. Together with the table, figures 45 and 46 display histograms for the return distributions.

Moments of returns		
Metric	Buy and hold	Strategy under test
Mean of returns	0.0073	0.0065
Variance of returns	0.0039	$3.4018\ .\ 10^{-6}$
Skewness of returns	0.2973	19.1115
Kurtosis of returns	11.9335	450.5218

Table 10: Moments of the returns for both strategies

Figure 45: Buy and hold return distribution.

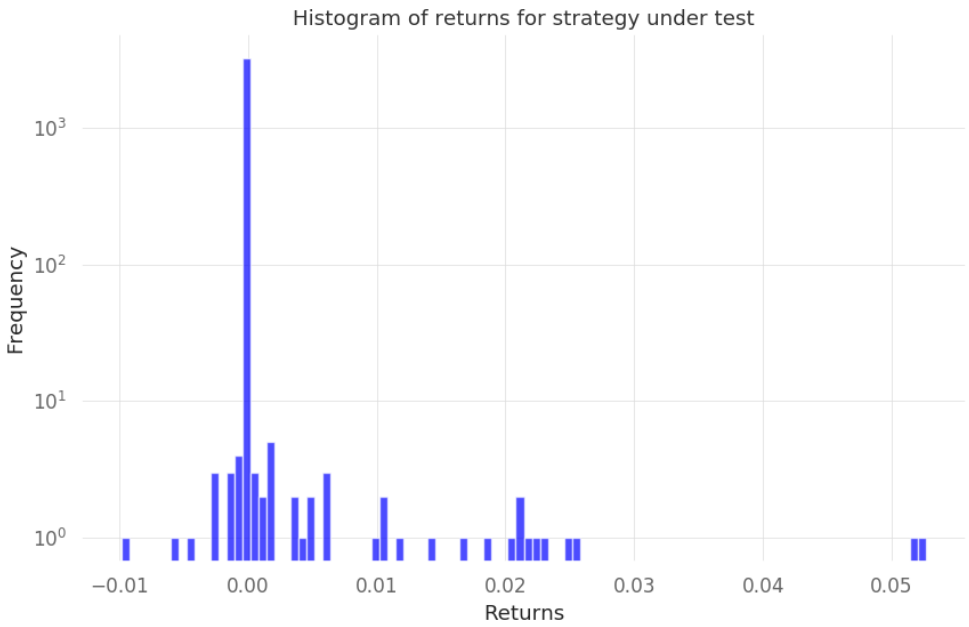

Figure 46: Strategy under test return distribution.

7 Conclusion

Along this research project a trading strategy over bitcoin with a machine learning model to learn the bet size was developed and evaluated in modest but comprehensive back testing benchmark. It was heavily based on the proposed pipeline of Marcos Lopez de Prado and used some relatively novel variations to a standard financial pipeline to test and evaluate a strategy. Careful feature handling, rigorous training and validation techniques are described in each section of this report.

The structure of this pipeline allows to change features, change a primary model, change a secondary model or even the back testing strategy but preserve the other building blocks because the interfaces are the same. The primary model should generate labels and metalabels to train a secondary model. The secondary model will provide a probability to perform the bet sizing. Feature engineering will be necessary to discard irrelevant features and explain the economic mechanism why the strategy produces excess returns. The back testing could be more comprehensive and incorporate new metrics but each stage preserves its interface.

As commented in 6.1 features from all the groups (price, volume, volatility, bitcoin network data, SADF indexes, and social indexes) remained in the purged model except social indexes. Interest and animosity were removed according to the heuristic rule to prune features that did not make a significant impact on the performance loss. This result is important because it refuses one of the preliminary intuitions for the secondary model. It was initially based on the belief that knowing the social interest would affect the performance of the bet sizing process. However, we probably assume that social indexes do not provide significant new data that models can use because it is already incorporated into, e.g. prices. On the contrary, a handful of SADF derived indexes remained and proved to have higher mean loss of performance. Similarly, bitcoin ecosystem features and traditional price and volume features remained in the final model.

Metrics for the secondary model performance are relatively bad ones compared to other disciplines or domains. Also, as a classifier, it is by no means a good one. However, the selection and training criteria that we used was to maximize the negative log-loss of the model because the predicted probability is what actually matters at the moment of implementing the strategy. It is worth noting that even though a primary model with high accuracy can lead important losses if the bet size is not properly computed when the primary model fails.

An increase in the negative log-loss is desired though and might be accomplished with further analysis to the features. Lopez de Prado describes

other techniques in chapter 8 of [Pra18] to fight back the substitution effect that would lead to the removal of important features. Also, with special care to overfitting, more complex machine learning models could be tested but that requires a back testing scheme that stresses much more the strategy.

Section 6 provides a comprehensive set of metrics for each model and the strategy as a whole. Some indicators lean towards the strategy under test as it offers less volatility, higher win loss ratio and Sortino ratio. However, the estimated Sharpe Ratio is smaller. When analyzing the Probabilistic Sharpe ratio both strategies perform really bad and fall rapidly when comparing them with different target ratios. Consequently, we cannot assure that they are considerably different based on the back testing evaluation. It is also important to note that the Probabilistic Sharpe Ratio incorporates a non-normality correction but the huge kurtosis and skewness coefficients make the PSR probability to quickly drop.

8 Future work

In this section, we list some possible variations that could be applied to the pipeline in the search for more performance.

Primary model The primary model is using one of the many implementations of momentum. Other alternatives can be tried using a relatively similar scheme and might lead to better results without affecting anything in the pipeline. Furthermore, other non-momentum based primary models could be tried as well, e.g. mean reversion. Anyway, any change to the primary model will imply retraining the secondary model with the new labels.

The primary model can also be applied to more than one specific asset, like other cryptocurrencies. That would provide more resilience to the secondary model because it will be trained with labels and metalables generated from different asset sources and reduce the risk of overfitting.

Secondary model As mentioned in 7, other models could be evaluated. Not only tree based models, but also neural networks or SVMs. Probably, further feature engineering would be required for the latter two and that would not necessarily be useful for the tree based models. Special attentions should be paid to overfitting though. A more comprehensive back testing strategy would be required aiming to evaluate the strategy under more scenarios and realize if testing data produced in excess model adjustment.

Feature importance Also, as commented in 7, it is recommended to try feature orthogonalization to mitigate the effect of feature substitution (the analog of multi-collinearity in linear models) in the trees. Implementation of that technique was out of the scope of this research project.

About interpretability, as Lopez de Prado points out in [Pra20], only a theory can pin down a cause-effect mechanism that allows you to generate excess returns. However, most interpretability techniques are not suited for identifying causal relationships unless additional assumptions are imposed (see [ZH21]).

At this time, interpretability techniques in machine learning have become a widely used tool for practitioners. However, their outputs must be taken only as approximations of what models are doing, even if those interpretability exercises are sufficient to comply with current regulatory constraints. In this regard, how to do proper inference on the estimates of feature importance remains a current research area. In this regard, techniques for proper

inference on the estimates of feature importance remain a current research area.

In another context, this point was also mentioned by Swadroe [BS16] when he warns us to be skeptical about the persistence of excess returns from technical trading rules. These rules rely solely on historical prices and lack risk-based explanations that cannot be arbitraged away.

Back testing The implemented back testing strategy is not the only one that could be used and probably it is not the most appropriate, although the most common. Splitting the feature space into multiple folds and mixing them to create different time lines that the model could face is one the many alternatives that would help to create different scenarios to evaluate the model.

There are several ways to obtain rich scenarios under which to stress a given strategy. For example, with access to large data sets, Wiese, Knobloch, Korn, and Kretschmer in [Wie+20] implement generative models for these purposes.

It is possible to analyze if more sophisticated back testing techniques are warranted by calculating a metric called Probability of Backtest Overfitting (PBO, [Pra18]). This metric measures the change in performance rankings for our strategies. In particular, one can use them when assessing different trading rules. Intuitively, an optimal trading rule overfits when it is expected to underperform the median of a set of alternative trading rules out of the sample.

Metrics In Andrew Lo [Lo02] it is shown how the simplified scale of from monthly Sharpe Ratios to annual Sharpe Ratios cannot be simply expanded by $\sqrt{12}$ and it depends on its distribution. Application of the proposed equations to the benchmark would correct the informed Sharpe Ratio even more.

9 Appendix

9.1 Code

The code in this project is available on https://github.com/agalbachicar/swing_for_the_fences and licensed under BSD 3-Clause "New" or "Revised" License. You can find in this link what you can and cannot do with it. In the *src* folder you will find:

- bet_sizing.py: contains functions to perform the bet sizing procedures explained in section 5.2.5
- btc_strategy.py: contains many functions and classes that act as wrappers and implement the pipeline.
- cv.py: contains cross validation functions and classes that implement the purged K-fold cross validation with embargo. See section 5.2.3 for a reference about the techniques.
- events.py: contains functions that allow to obtain the labels by means of a momentum strategy. See section 5.2.1.
- feature_importance.py: contains functions to evaluate the mean feature importance loss. See section 5.2.4.
- features.py: contains functions to compute financial features out of a price series. See section 5.1.1.
- frac_diff.py: contains functions that implement the fractional differentiation method. See section 5.1.
- labelling.py: contains functions to implement labels and metalabels for price series. See section 5.2.1.
- load_data.py: contains functions to load and merge datasets.
- mpfin.py: contains functions from chapter 20 of [Pra18] for parallel execution of certain algorithms.
- pipeline_thesis.py: main Python script to run and evaluate different stages of the pipeline.
- sample_weights.py: contains functions to implement sample weights. See section 5.2.2.

- sharpe_ratio_stats.py: contains functions to compute Sharpe Ratio and Probabilistic Sharpe Ration. See section 5.2.6.
- strategy_backtesting.py: implements the back testing strategy and obtains results. See 5.2.6 and 6.
- structural_breaks.py: contains functions to compute SADF series. See section 5.1.3.

src/notebooks folder contains some Jupyter Python notebooks to generate the images in this research project.

datasets folder contains all datasets used in this project. Also, *pickle* files with models, metrics and hyperparameter values are stored there by default for convenience.

doc folder contains the latex project to generate this document.

Bibliography

[Mac69] Charles Mackay. *Memoirs of extraordinary popular delusions and the madness of crowds*. London ; New York : G. Routledge, 1869. ISBN: 978-1119482086.

[Ros58] Frank Rosenblatt. "The Perceptron: A Probabilistic Model for Information Storage and Organization in The Brain". In: *Working paper IDEI* (1958). DOI: https://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.588.3775.

[Hos81] J. M. R. Hosking. "Fractional differencing". In: *Biometrika, Volume 68, Issue 1, April 1981, Pages 165–176* (1981). DOI: https://doi.org/10.1093/biomet/68.1.165.

[GKP94] Ronald L. Graham, Donald E. Knuth, and Oren Patashnik. *Concrete Mathematics, A Foundation for Computer Science*. Addison Wesley Publishing Company, 1994. ISBN: 0-201-55802-5.

[Ham94] James D. Hamilton. *Time Series Analysis*. Princeton University Press, 1994. ISBN: 0-691-04289-6.

[Lo02] Andrew W. Lo. "The Statistics of Sharpe Ratios". In: *Financial Analysts Journal* 58.4 (2002), pp. 36–52. DOI: 10.2469/faj.v58.n4.2453. URL: https://doi.org/10.2469/faj.v58.n4.2453.

[DM03] Russell Davidson and James G. MacKinnon. *Econometric Theory and Methods*. Oxford University Press, 2003. ISBN: 978-0195123722.

[Nak08] Satoshi Nakamoto. "Bitcoin: A Peer-to-Peer Electronic Cash System". In: (2008). DOI: https://bitcoin.org/bitcoin.pdf.

[Pet11] Jun Yu Peter C. B. Phillips Yangru Wu. "Explosive Behavior in the 1990s Nasdaq: When Did Exuberance Escalate Asset Values?" In: *International Economic Review. 52, (1), 201-226. Research Collection School Of Economics* (2011). DOI: https://doi.org/10.1111/j.1468-2354.2010.00625.x.

[BL12] David H. Bailey and Marcos López de Prado. "The Sharpe Ratio Efficient Frontier". In: *Journal of Risk, Vol. 15, No. 2, Winter 2012/13* (2012). DOI: http://dx.doi.org/10.2139/ssrn.1821643.

[BW12] Bruno Biais and Paul Woolley. "High Frequency Trading". In: *Working paper IDEI* (2012). DOI: http://dx.doi.org/10.1002/andp.19053221004.

[CP13] Tobias Moskowitz Clifford Asness and Lasse Heje Pedersen. "Value and Momentum Everywhere". In: (2013). DOI: https://doi.org/10.1111/jofi.12021.

[IM14] Clifford Asness Andrea Frazzini Ronen Israel and Tobias Moskowitz. "Fact, Fiction and Momentum Investing". In: *The journal of Portfolio Management* (2014). DOI: https://doi.org/10.3905/jpm.2014.40.5.075.

[Lop15] Marcos Lopez de Prado. "The Future of Empirical Finance". In: *Journal of Portfolio Management, 41(4), Summer 2015* (2015). DOI: http://dx.doi.org/10.2139/ssrn.2609734.

[MO15] Hannah Ritchie Max Roser and Esteban Ortiz-Ospina. "Internet". In: *Our World in Data* (2015). https://ourworldindata.org/internet.

[RH15] Thomas N. Rollinger and Scott T. Hoffman. "Sortino: A Sharper Ratio". In: (2015). URL: https://www.cmegroup.com/education/files/rr-sortino-a-sharper-ratio.pdf.

[BS16] Andrew L. Berkin and Larry E. Swedroe. *Your Complete Guide to Factor-Based Investing: The Way Smart Money Invests Today*. Buckingham, 2016. ISBN: 978-0692783658.

[BT17] Chris Burniske and Jack Tatar. *Cryptoassets. The Innovative Investor's Guide to Bitcoin and Beyond*. McGraw-Hill Education, 2017. ISBN: 978-1260026672.

[JWT17] Gareth James, Daniela Witten, and Robert Tibshirani Trevor Hastie. *An Introduction to Statistical Learning*. Springer, 2017. ISBN: 978-1-4614-7137-0.

[Tre17] Jerome Friedman Trevor Haastie Robert Tibshirani. *The Elements of Statistical Learning. Data Mining, Inference and Prediction*. Springer, 2017. ISBN: 978-0-387-84857-0.

[Ali18] Amanda Casari Alice Zheng. *Feature Engineering for Machine Learning: Principles and Techniques for Data Scientists*. O'Reilly Media, 2018. ISBN: 978-1491953242.

[Chi18] Office of the Chief Economist. "Cryptocurrencies and Blockchain". In: *World Bank Economic ECA Update* (2018). DOI: http://documents1.worldbank.org/curated/pt/293821525702130886/pdf/Cryptocurrencies-and-blockchain.pdf.

[GZ18] Peter Gomber and Kai Zimmermann. "Algorithmic Trading in Practice". In: (2018). DOI: https://doi.org/10.1093/oxfordhb/9780199844371.013.12.

[Pra18] Marcos Lopez de Prado. *Advances in Financial Machine Learning*. Wiley, 2018. ISBN: 978-1119482086.

[20] "5 Blockchain Trends for 2020". In: *C-Suite Briefing* (2020). DOI: https://www2.deloitte.com/content/dam/Deloitte/ie/Documents/Consulting/Blockchain-Trends-2020-report.pdf.

[DHB20] Matthew F. Dixon, Igor Halperin, and Paul Bilokon. *Machine Learning in Finance*. Springer International Publishing, 2020. ISBN: 978-3-030-41068-1.

[Pra20] Marcos Lopez de Prado. *Machine Learning for Asset Managers (Elements in Quantitative Finance)*. Cambridge University Press, 2020. ISBN: 978-1108792899.

[Wie+20] Magnus Wiese et al. "Quant GANs: deep generation of financial time series". In: *Quantitative Finance* 20.9 (2020), pp. 1419–1440. DOI: 10.1080/14697688.2020.1730426. URL: https://doi.org/10.1080/14697688.2020.1730426.

[ZH21] Qingyuan Zhao and Trevor Hastie. "Causal Interpretations of Black-Box Models". In: *Journal of Business & Economic Statistics* 39.1 (2021), pp. 272–281. DOI: 10.1080/07350015.2019.1624293. URL: https://doi.org/10.1080/07350015.2019.1624293.

[Alt] Alternative.me. *Crypto Fear & Greed Index*. URL: https://alternative.me/crypto/fear-and-greed-index/. (accessed: 04.18.2021).

[Bes] Raynor de Best. *Quantity of cryptocurrencies 2013-2021*. URL: https://www.statista.com/statistics/863917/number-crypto-coins-tokens/. (accessed: 04.02.2021).

[Bin] Binance. *Fee schedule*. URL: https://www.binance.com/en/fee/schedule. (accessed: 05.23.2021).

[Bita] The Bitcoin Core developers Bitcoin Developers. *Bitcoin project*. URL: https://github.com/bitcoin. (accessed: 04.10.2021).

[Bitb] Bitcoincharts. *Bitcoincharts*. URL: https://bitcoincharts.com/charts. (accessed: 04.18.2021).

[Bla] iShares by BlackRock. *iShares MSCI USA Momentum Factor ETF. Fact Sheet as of 03/31/2021*. URL: https://www.ishares.com/us/literature/fact-sheet/mtum-ishares-msci-usa-momentum-factor-etf-fund-fact-sheet-en-us.pdf. (accessed: 05.23.2021).

[CNN] CNN. *Fear & Greed Index*. URL: https://money.cnn.com/data/fear-and-greed/. (accessed: 04.18.2021).

[Coi] CoinMarketCap. *Historical Data for Bitcoin*. URL: https://coinmarketcap.com/es/currencies/bitcoin/historical-data/. (accessed: 04.02.2021).

[Coma] Creative Commons. *Creative Commons Attribution-ShareAlike 4.0 International Public License*. URL: https://creativecommons.org/licenses/by-sa/4.0/legalcode. (accessed: 04.18.2021).

[coma] Bitcoin community. *Block chain*. URL: https://en.bitcoin.it/wiki/Block_chain. (accessed: 04.08.2021).

[comb] Bitcoin community. *Network*. URL: https://en.bitcoin.it/wiki/Network. (accessed: 04.08.2021).

[comc] Bitcoin community. *Wallet*. URL: https://en.bitcoin.it/wiki/Wallet. (accessed: 04.08.2021).

[Comb] James Comtois. *BlackRock reorganizes active equity unit, turns more to computer-based quant strategies*. URL: https://www.pionline.com/article/20170329/ONLINE/170329859/blackrock-reorganizes-active-equity-unit-turns-more-to-computer-based-quant-strategies. (accessed: 29.03.2017).

[Glaa] Glassnode. *Glassnode*. URL: https://glassnode.com/. (accessed: 04.18.2021).

[Glab] Glassnode. *Terms & Conditions*. URL: https://glassnode.com/terms-and-conditions. (accessed: 04.18.2021).

[Hay] Adam Hayes. *Dotcom Bubble*. URL: https://www.investopedia.com/terms/d/dotcom-bubble.asp. (accessed: 04.02.2021).

[Inc] Kaggle Inc. *Kaggle*. URL: https://www.kaggle.com/. (accessed: 04.18.2021).

[Int] ECMA International. *ECMA-404, The JSON data interchange syntax,2nd edition, December 2017*. URL: https://www.ecma-international.org/publications-and-standards/standards/ecma-404/. (accessed: 04.15.2021).

[ITU] ITU. *Internet access through time*. URL: https://www.itu.int/en/ITU-D/Statistics/Pages/stat/default.aspx. (accessed: 2019).

[Kom] Matt Komorowski. *a history of storage cost (update)*. URL: https://mkomo.com/cost-per-gigabyte-update. (accessed: 03.09.2014).

[LLC] Google LLC. *Terms and conditions to use Google Trends data*. URL: https://support.google.com/trends/answer/4365538?hl=es. (accessed: 04.18.2021).

[Moo] Gordon Moore. *Moore's Law and Intel innovation*. URL: https://www.intel.com/content/www/us/en/history/museum-gordon-moore-law.html. (accessed: 04.02.2021).

[Nak] Satoshi Nakamoto. *Bitcoin v0.1 released*. URL: https://www.mail-archive.com/cryptography@metzdowd.com/msg10142.html. (accessed: 04.02.2021).

[Pla] PlanB. *Modeling Bitcoin Value with Scarcity*. URL: https://medium.com/@100trillionUSD/modeling-bitcoins-value-with-scarcity-91fa0fc03e25. (accessed: 04.10.2021).

[Rat] Defi Rate. *Lend*. URL: https://defirate.com/lend/?exchange_table_type=lend. (accessed: 05.23.2021).

[RFC] RFC. *Common Format and MIME Type for Comma-Separated Values (CSV) Files*. URL: https://tools.ietf.org/html/rfc4180. (accessed: 04.15.2021).

[Sin] Manoj Singh. *The 2007-2008 Financial Crisis in Review*. URL: https://www.investopedia.com/terms/d/dotcom-bubble.asp. (accessed: 04.02.2021).

[Tre] U.S. Department of the Treasury. *Daily Treasury Yield Curve Rates*. URL: https://www.treasury.gov/resource-center/data-chart-center/interest-rates/pages/TextView.aspx?data=yieldYear&year=2021. (accessed: 05.23.2021).

[Zie] Mark Zielinski. *Bitcoin Historical Data*. URL: https://www.kaggle.com/mczielinski/bitcoin-historical-data. (accessed: 04.18.2021).