Cryptocurrency Explorer and Market Analyzer

Irene Finocchi, Flavio Giorgi, Bardh Prenkaj
finocchi@luiss.it, fgiorgi@luiss.it, bprenkaj@luiss.it

May 9, 2023

Abstract

CEMA (Cryptocurrency Explorer and Market Analyzer) is a tool that is becoming increasingly important in the rapidly-evolving cryptocurrency market. It is a necessary tool for anyone who wants to stay informed about the market and make informed decisions about buying and selling cryptocurrencies. With so many different cryptocurrencies available, and new ones being introduced on a regular basis, it can be challenging for individuals and businesses to stay informed and make informed decisions. CEMA provides a reliable and comprehensive source of information about the cryptocurrency market. It allows users to easily access and analyze data about different cryptocurrencies, including historical price data, trading volumes, market capitalization, and more. Additionally, it helps users to make informed decisions about buying and selling cryptocurrencies by providing data and insights about different factors such as market trends, volatility, and regulatory developments. In short, CEMA is a valuable tool that can help individuals and businesses navigate the complex and rapidly-evolving cryptocurrency market.

Keywords: cryptocurrency market; time series analysis; market trends, volatility and regulations; blockchain

1 Context and motivation

CEMA (Cryptocurrency Explorer and Market Analyzer) is a tool that is becoming increasingly important in the rapidly-evolving cryptocurrency market. With so many different cryptocurrencies available, and new ones being introduced on a regular basis, it can be challenging for individuals and businesses to stay informed and make informed decisions.

One of the primary motivations for the development of CEMA is the need for a reliable and comprehensive source of information about the cryptocurrency market. CEMA provides users with an easy way to access and analyze data about different cryptocurrencies, including historical price data, trading volumes, market capitalization, and more.

Another important motivation for CEMA is the need for a tool that can help users make informed decisions about buying and selling cryptocurrencies. With so many different factors to consider, such as market trends, volatility, and regulatory developments, it can be difficult for individuals and businesses to know when to buy or sell a particular cryptocurrency. CEMA provides users with the data and insights they need to make informed decisions about buying and selling cryptocurrencies.

In summary, CEMA is a necessary tool for anyone who wants to stay informed about the cryptocurrency market and make informed decisions about buying and selling cryptocurrencies. It can provide users with valuable data and insights about different cryptocurrencies and help them navigate the rapidly-evolving market.

2 Project description

2.1 Introduction

Cryptocurrencies are digital assets that use cryptography for security and operate on a decentralized network, such as the blockchain. They can be used as a medium of exchange, similar to traditional currencies, but it is not controlled by any central authority, such as a government or financial institution.

There are many different types of cryptocurrencies, each with its own set of features and characteristics. Some of the most well-known cryptocurrencies include Bitcoin, Ethereum, and Litecoin. One of the main advantages of cryptocurrency is that it allows for peer-to-peer transactions without the need for a central intermediary, such as a bank. This can make transactions faster and more efficient, reducing the risk of fraud or tampering. However, cryptocurrencies are still a relatively new and complex technology, and they are not without risks.

Table 1: The characteristics of each dataset

Dataset	Size (Megabytes)	Number of rows	Number of cryptos	Average monitoring days per crypto
dataset_small.txt	0.88	24,266	98	247.61 ± 13.74
dataset_medium.txt	1.77	48,259	98	492.44 ± 40.59
dataset_large.txt	2.56	69,365	98	707.81 ± 100.11
dataset_full.txt	4.50	120,783	98	1232.48 ± 486.20

2.2 Project Data

The project requires reading a financial dataset and implementing efficient algorithmic solutions to different problems in Python, also experimenting with a few real datasets of different, increasing sizes. In detail, you will analyze a set of cryptocurrencies and compute several metrics.

Terminology In this paragraph, we are going to see the essential terms we will encounter during the project.

- **Open Price**: is the price at which the first trade of the day took place.
- **Close Price**: is the price at which the last trade of the day took place.
- **Volumes**: the amount of cryptocurrency traded on that day. It can be expressed in US Dollars, Euros, or as the crypto itself.
- **High**: is the highest price reached by the crypto within that day
- **Low**: is the lowest price reached by the crypto within that day

Dataset You are given as input a *.txt* file containing a list of cryptocurrencies and additional details. Each row in the file contains these elements in the following order:

```
crypto_name, day, price, volume
```

These values represent the open price and volume for the `crypto_name` (e.g., Bitcoin) on a given day. Both volumes and prices are float numbers, days is an integer value, and finally, `crypto_name` is a string with the cryptocurrency's name. The volume defines the amount of crypto traded on that day. A value of 0 for volume means that no transactions took place on a given day. If you are familiar with stock or other financial data series, you noticed that we didn't include other values such as *Low, High, and Close Price*. We did this simplification to make the dataset simpler. As you can see, if you inspect the .txt file, all the data series are shuffled. If you are interested, you can find the original dataset on Kaggle.

3 Project Goals

The project is divided into three different phases:

1. **Data reading and preprocessing** (Task 1) - announcement February 7, due February 28 (not mandatory)
2. **Crypto sorting and searching** (Task 2) - announcement February 28, due March 28 (not mandatory)
3. **Crypto correlation graph and minimal correlation pathways** (Task 3) - announcement March 28, due May 14

Notice that you can submit each task within the specified deadlines, but you'll only receive an evaluation for the entire project by the end of the course.

3.1 Task 1: data reading and preprocessing

You are given four datasets: `dataset_small.txt`, `dataset_medium.txt`, `dataset_large.txt`, and `dataset_full.txt` with the characteristics shown in Table 1. You are required to design and implement:

1. **A python function that reads the .txt file**, extracting relevant information which must be then stored in a suitable data structure.
    - The function `read_file` is called ONCE to load the dataset. It can be used to prepare and read the input file (e.g., "`data/dataset_small.txt`")

Table 2: Example of Algorand's minimum, average, and maximum prices in the interval $[1, 30]$

crypto	day	price	volume
Gala	14	0.0003000000142492	12503.0
Algorand	25	0.8238009810447693	117943656.0
Quant	5	0.2447540014982223	6582.0
Quant	30	0.7051290273666382	2786320.0
Gala	9	0.0006300000241026	146418.0
Quant	10	0.2664119899272918	4260.0
Quant	8	0.2389190047979354	8290.0
Gala	19	0.0003800000122282	10678.0
Gala	5	0.0009009999921545	115883.0
Algorand	8	1.3938219547271729	121576369.0
Gala	29	0.0003699999942909	21907.0
Algorand	13	1.1359020471572876	112712391.0
Quant	1	0.2878440022468567	30951.0
Algorand	22	0.8555579781532288	140797532.0
Quant	17	0.2503879964351654	7155.0
Gala	7	0.0009940000018104	255093.0
Gala	27	0.0003450000076554	12965.0
Algorand	16	1.1275609731674194	64939220.0
Algorand	15	1.009680986404419	103703180.0
Algorand	2	3.153618097305298	223603358.0
Gala	3	0.0014919999521225	76880.0

2. **A python function to answer queries of the form** "Which are the min-price, average-price, and max-price values for the crypto X in a specific time interval $[a, b]$ where a is the beginning time and b is the end time?".

   Example: Which are the min, average, max values for **Algorand** in the time interval $[1, 30]$? Your code should answer `(0.64, 1.29, 3.28)` according to the selection in Table 2. Notice that this table is just for illustration purposes and doesn't represent the whole view of **Algorand**'s price trend in the first 30 days.

   - The function `crypto_stats` implements your query algorithm. It receives as input the crypto name - e.g., **Algorand** - and a period of time $[a, b]$, and it outputs the observed minimum, average, and maximum price values reached within the beginning time a and end time b. The order of the output is important for evaluation purposes. So, return a tuple `(min_price,avg_price,max_price)`.

3.2 Task 2: crypto sorting and searching

Notice that the .txt files given to you are completely disordered which isn't useful for analysis purposes. You are required to design and implement:

1. **A python function that sorts the datasets** of all cryptocurrencies according to the day of observation In this way we can plot the trend series of each cryptocurrency and see how the crypto market evolved in time (see Figure 1 for **Algorand**'s price and volume trends ordered in time).

   - The function `sort_data` is used to organize all cryptocurrency data by the number of days observed. The output should be a list of tuples in the format `(crypto_name, price)`. Each cryptocurrency in the dataset must be sorted by the number of days observed. For example, if there are three cryptocurrencies, **Algorand**, **Gala**, and **Quant**, the function should first sort **Algorand** by the number of days observed, then **Gala**, and finally **Quant**. The final output list will contain tuples for **Algorand** first, followed by **Gala** and then **Quant**. It is important to note that the cryptocurrencies must be sorted alphabetically before sorting them by the number of days observed.

2. **A python function to retrieve the maximum** value of a given cryptocurrency in a specific month m.

   - The function `get_max_value` implements your retrieval algorithm. It receives as input the crypto name - e.g., **Algorand** - a data structure containing the crypto information, and the month m. It outputs a tuple in the format `(day, price)`. In other words, you need to return the day in which **Algorand**'s price was the highest within the specific month m accompanied by the price value. Notice that the data structure provided in input can be anything. Additionally, you can assume that months have 30 days for simplicity. *Hint: use the data structure that you output from* `sort_data` *function.*

Figure 1: Algorand's price and volume trends ordered in time.

3. **A python function that searches** for a the closest price with respect to a particular value for a given crypto.

   - The function `search` searches for a specific value in a given data series and returns a tuple with the day and the value for a given cryptocurrency. If the searched value is not present in the data, the function returns the closest value. It compares two values of the data series, one at position i and the other at position j, and returns the value closest to the searched value. For example, consider the prices in Table 2. If we want to search for `value = 0.84` of Algorand's price trend, then the output should be `(22, 0.8555579781532288)`. You can see that we have two candidates for the closest value of 0.84: we have price in the 25^{th} day, and that in the 22^{nd}. The price in the 25^{th} day is 0.8238..., and that in the 22^{nd} is 0.85555... The difference between them and the value we want to search for is, respectively, $d_1 = |0.84 - 0.8238| = 0.0162$ and $d_2 = |0.84 - 0.85555| = 0.0155$. You can see that d_2 is lower than d_1 and, thus, closer to the desired value 0.84. **You must do these comparisons for all days and find the closest price to the desired value.**

3.3 Task 3: crypto correlation graph and minimal correlation pathways

This task will study the portfolio management and the correlations of cryptocurrencies. The **goal** is to answer a fundamental question: *If I have a crypto C in my portfolio, which other cryptos should I avoid/sell to reduce risks?* When investing into the financial market, we usually create a **Portfolio**, a collection of financial investments like stocks, bonds, commodities, exchange traded funds (ETFs), or cryptos. One of the key concepts in portfolio management is the wisdom of **diversification and risk management**.

Let us define the return r_C for a crypto C over a temporal period $[a, b]$ as follows:

$$r_{C,a,b} = \begin{cases} \dfrac{price_{C,b} - price_{C,a}}{price_{C,a}} & \text{if price}_{C,a} > 0, \\ 1 & \text{otherwise} \end{cases}$$

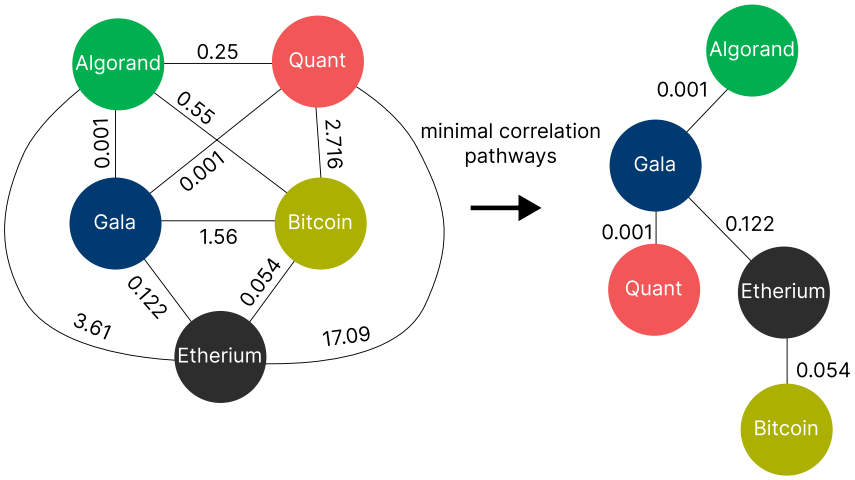

Figure 2: An example of a correlation graph and the minimal correlation pathways algorithm for **Algorand**.

where $price_{C,n}$ denotes the price of crypto C on the n-th day of monitoring. Considering **Algorand**'s trend from Table 2 in the period $[2, 25]$, $r_{Algorand,2,25} = \frac{price_{Algorand,25} - price_{Algorand,2}}{price_{Algorand,2}} = \frac{0.8238 - 3.1536}{3.1536} \simeq -0.7388$. You can do the same thing for the other cryptos for any observation time $[a, b]$.

We can say that two cryptocurrencies are correlated if they have a similar return over the temporal period of interest $[a, b]$. This means that they move together the market. Think of Bitcoin and its influence on the trend of other cryptos: i.e., generally, when Bitcoin reaches a bear market, then the rest of the cryptos are in a downtrend as well; the same is valid if Bitcoin is bullish. Therefore, we can define a distance score between crypto C_i and C_j in the period $[a, b]$ as the squared distance of their returns

$$\sigma_{i,j,a,b} = (r_{i,a,b} - r_{j,a,b})^2$$

The lower $\sigma_{i,j,a,b}$, the higher the correlation.

To study correlations, we often rely on a graph that maps the interrelations between cryptos. Hence, we can build a graph $G_{a,b} = (V, E)$ where V is the set of nodes (cryptos) and E is the set of correlation connections between two cryptos according to the temporal period $[a, b]$. Notice that each edge $e_{i,j} = (i, j, \sigma_{i,j,a,b})$ has a positive weight that represents the correlation between the cryptos C_i and C_j. Because we calculate the correlation between each pair of cryptos in the dataset, the correlation graph that we obtain is complete. In other words, each node is connected to every other node. The left part of Figure 2 illustrates an example of a complete correlation graph. Recall that the lower $\sigma_{i,j,a,b}$ is, the higher the correlation. For this reason, we apply an algorithm that calculates the minimal correlation pathways from a given crypto (e.g., **Algorand**). Notice that this algorithm attempts to minimize the cost of the connections globally. Assuming that we visit the neighboring cryptos in alphabetical order, the tree consisting of the minimal correlation pathways is the one shown in the right part of Figure 2).

You are required to design and implement:

1. **A python function that returns the minimal correlation pathways**. Given a correlation graph and a specific cryptocurrency, we want to know which are the minimal correlation pathways that begin from a particular cryptocurrency.

   - The function `min_correlation_pathways` implements the minimal correlation pathways algorithm. It gets in input a correlation graph, a cryptocurrency, and a temporal period $[a, b]$. It returns a tree, that at each node, selects the lowest cost possible in the path between the root and the currently visited node (crypto). This tree should be returned in the form of a dictionary of tuples in the form of `(crypto_name,correlation_coefficient)`. *Hint: adopt a minimum spanning tree algorithm. Assumption: the cryptos should be visited in alphabetical order.*

2. **A python function that answers the following question** "Given a crypto C, which are all the correlated cryptos at level k in a certain temporal period $[a, b]$?

   Example: Which are the correlated cryptos at level 2 of **Algorand** in [2,25]? Let's suppose that we built the correlation tree of **Algorand** in [2,25] and the result is the one shown in the right side of Figure 2. Therefore, the answer to the previous question should be: **Ethereum** and **Quant**.

   - The function `correlated_cryptos_at_lvl_k` implements your correlation algorithm. It receives as input the crypto name - e.g., **Algorand** - a data structure containing the crypto information, the

time interval $[a, b]$ for which to build the minimal correlation pathways, and the level k of correlation to consider for the other cryptos with respect to **Algorand**. The output should be a list of cryptocurrencies that are correlated to the one given in input at level k.

4 How to open the project and IDEs

You can find the project folder on a GitHub Repository. You can clone the repository in your local machine using the command:

```
$ git clone https://github.com/flaat/Algorithms-2022-2023-Project
```

and you can start coding with your usual IDE or text editor.

Important: you need git on your machine if you haven't git installed, please follow the instruction for Windows, Mac, Linux. If you do not prefer to use the command line (terminal), you can use the Git extension of VSCode that handles all the git commands in its UI. Though, you have to notice that you still need to have an intuition on how git projects work.

4.1 Integrated Development Environment (IDE)

What is an IDE? An IDE, or Integrated Development Environment, is software that enables programmers to consolidate the different aspects of writing a computer program. IDEs increase programmer productivity by combining common activities of writing software into a single application: editing source code, building executables, and debugging. **A proper IDE is an invaluable tool for writing code! It will save you a lot of time.**

Best IDE for Python In python the most known IDEs are PyCharm and VSCode. Both are extremely useful, and both of them offer a wide variety of features. PyCharm is probably the most complete, but it is also cumbersome to load. It has a vast number of functionalities ranging from auto-completing to debugging up to, only for the pro or academic versions, peer coding.

VSCode instead is lightweight, and it is probably more suitable for your current project. It also has many exciting features like auto-completing and version control (GIT) integrated within the IDE.

If you choose to use an IDE you can follow the instruction to clone the repository directly in VSCode or PyCharm. If you have problems cloning the repository directly from the IDE, you can simply use git clone on your local machine, then go to your IDE, go to file, open, and select the repository folder named `Algorithms-2022-2023-Project`.

Here you can find an interesting discussion about Python IDEs between Lex Fridman and Guido van Rossum, the creator of Python.