Luiss
Libera Università Internazionale degli Studi Sociali Guido Carli

Algorithms A.Y. 2022/2023

Software Project v3 – A Cryptocurrency Explorer and Market Analyzer

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

4 February 2023

LUISS

Dipartimento di Impresa e Management

Group Project Work

The project requires to:

- read a cryptocurrency dataset
- implement *efficient* algorithmic solutions to different problems

We release three different parts:

1. February 7 – due to February 28 (not mandatory)
2. February 28 – due to April 4 (not mandatory)
3. **April 4 – due May 14 (mandatory)**

Group Project Work - Task 3

For the final task we study portfolio management and correlations between cryptos!

The goal is to answer a fundamental question:

Group Project Work - Task 3

For the final task we study portfolio management and correlations between cryptos!

The goal is to answer a fundamental question:

If I have a crypto *x* in my portfolio, which crypto should I avoid/sell to reduce risk?

Group Project Work - Task 3

For the final task we study portfolio management and correlations between cryptos!

The goal is to answer a fundamental question:

If I have a crypto *x* in my portfolio, which crypto should I avoid/sell to reduce risk?

This method works very well with stocks, a bit less with crypto but it is ok.

Portfolio and Correlation

When investing into the financial market we usually create a ***Portfolio.***

A portfolio is a collection (i.e., a set) 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!**

Portfolio and Correlation

Which is the best couple of stocks to reduce risk?

PepsiCo	162,22 $	-6,90 $	↓4,08%
The Coca-Cola Co...	61,97 $	+0,16 $	↑0,26%
NVIDIA	234,70 $	-3,64 $	↓1,53%

Stock correlation

We first define the return r_s for a stock s over a period of N = {0, .. , n} days as follows:

$$r_s = \begin{cases} (x_n - x_0) / x_0 & if \ \ x_0 > 0 \\ 1 & else \end{cases}$$

For AAPL for example we have:

Stock	Day	Price	Volume
AAPL	371	200	0
TSLA	369	275	13038300
TSLA	370	273	0
TSLA	371	273	0
AAPL	369	197	18526600
AAPL	370	200	0
AAPL	365	191	27862000
TSLA	365	289	8110400
TSLA	366	286	5478900
TSLA	367	292	7929900
TSLA	368	268	23720700
AAPL	366	194	22765700
AAPL	367	195	23271800
AAPL	368	196	19114300

Stock correlation

We first define the return r_s for a stock s over a period of N = {0, .. , n} days as follows:

$$r_s = \begin{cases} (x_n - x_0) / x_0 & if \ \ x_0 > 0 \\ 1 & else \end{cases}$$

For AAPL for example we have:

(200 - 191) / 191 = **0.047**

For TSLA we have:

Stock	Day	Price	Volume
AAPL	371	200	0
TSLA	369	275	13038300
TSLA	370	273	0
TSLA	371	273	0
AAPL	369	197	18526600
AAPL	370	200	0
AAPL	365	191	27862000
TSLA	365	289	8110400
TSLA	366	286	5478900
TSLA	367	292	7929900
TSLA	368	268	23720700
AAPL	366	194	22765700
AAPL	367	195	23271800
AAPL	368	196	19114300

Stock correlation

We first define the return r_s for a stock s over a period of N = {0, .. , n} days as follows:

$$r_s = \begin{cases} (x_n - x_0) / x_0 & if \ \ x_0 > 0 \\ 1 & else \end{cases}$$

For AAPL for example we have:

(200 - 191) / 191 = **0.047**

For TSLA we have:

(273– 289) / 289 = **-0.055**

Stock	Day	Price	Volume
AAPL	371	200	0
TSLA	369	275	13038300
TSLA	370	273	0
TSLA	371	273	0
AAPL	369	197	18526600
AAPL	370	200	0
AAPL	365	191	27862000
TSLA	365	289	8110400
TSLA	366	286	5478900
TSLA	367	292	7929900
TSLA	368	268	23720700
AAPL	366	194	22765700
AAPL	367	195	23271800
AAPL	368	196	19114300

Stock correlation

We first define the return r_s for a stock s over a period of N = {0, .. , n} days as follows:

$$r_s = \begin{cases} (x_n - x_0) / x_0 & if \;\; x_0 > 0 \\ 1 & else \end{cases}$$

We compute the return because two stocks are correlated if they have a similar return over the entire period *.

This means that they move together in the market:

*** *A very simplified definition of correlation!!***

LUISS

Stock correlation

Therefore, we can define a distance score (similarity) between stock i and stock j as the absolute different of their returns:

$$c_{ij} = \quad | r_i - r_j |$$

Stock correlation

Therefore, we can define a distance score (similarity) between stock i and stock j as the absolute different of their returns:

$$c_{ij} = \quad |\, r_i - r_j \,|$$

According to the previous example, the score between NVIDIA and TESLA is $|0.52 - 0.46| = 0.06$ (very similar) while the score between AMAZON and NVIDIA is $|-0.08 - 0.52| = 0.6$ (they are much more distant).

Stock correlation

Therefore, we can define a distance score (similarity) between stock i and stock j as the absolute different of their returns:

$$c_{ij} = \quad |\, r_i - r_j \,|$$

According to the previous example, the score between NVIDIA and TESLA is $|0.52 - 0.46| = 0.06$ (very similar) while the score between AMAZON and NVIDIA is $|-0.08 - 0.52| = 0.6$ (they are much more distant).

LUISS

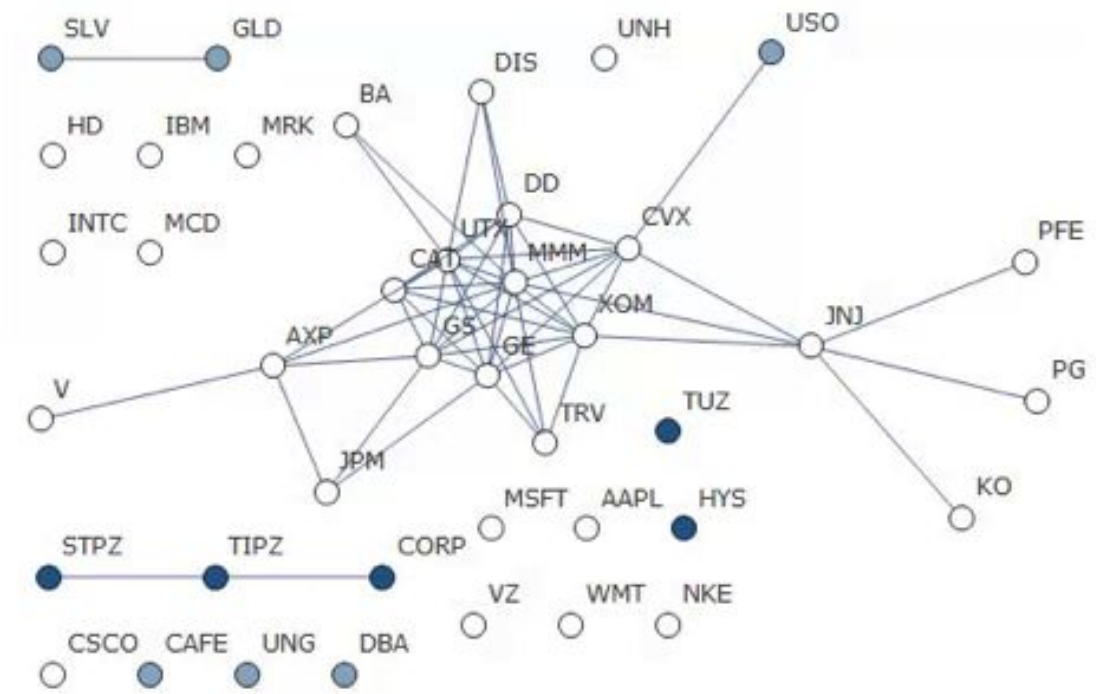
Thus: the less the score the higher the correlation! For simplicity, we'll say that two stocks are correlated if their score c_{ij} is less than a threshold t (i.e., $c_{ij} < t$).

Stock Correlation Graph

To study correlation we often use a graph that maps the interrelations between assets that are correlated at some specified threshold **t** (0.5 of higher, in this illustration).

We can build a Graph G(V, E)

Where :
- Vertices or Nodes **V** = stocks
- Edge **E** = correlations.

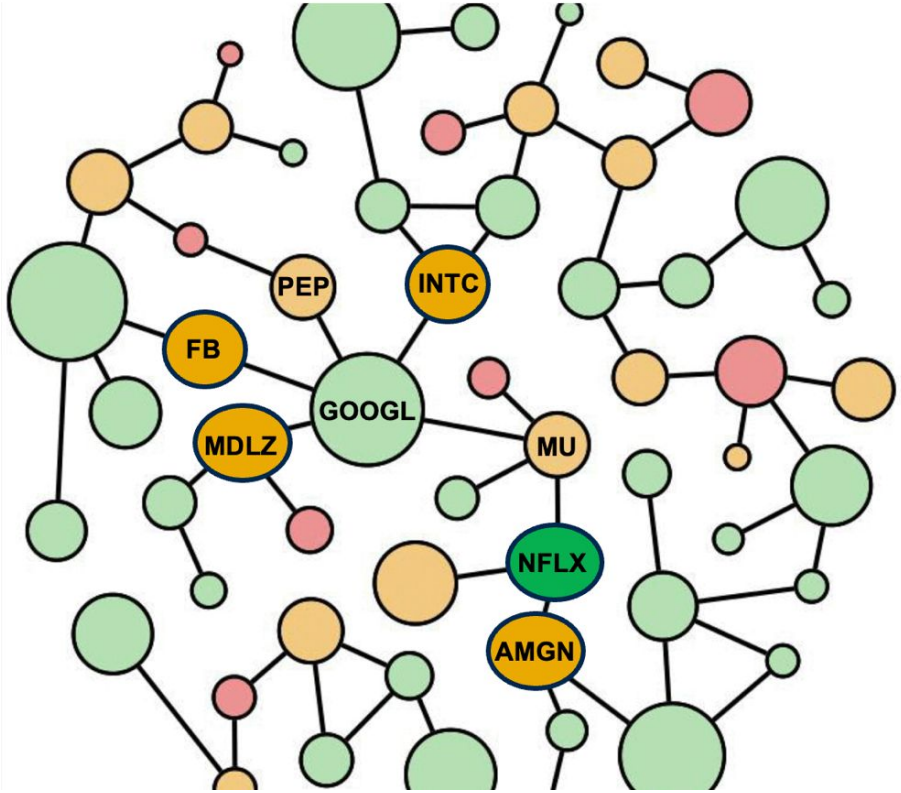


Stock Correlation Graph

Why graphs? Graph are easy to read!

For example, GOOGL is directly correlated with MDLZ, FB, MU, INTC, and PEP. Nevertheless, MU is correlated also with NFLX!
Therefore, we know that:

- If a crash happens on FB is very likely to have a crash also on GOOGL
- If a crash happens on NFLX is very likely to have a crash on MU, and therefore the crash propagates on GOOGL as well!

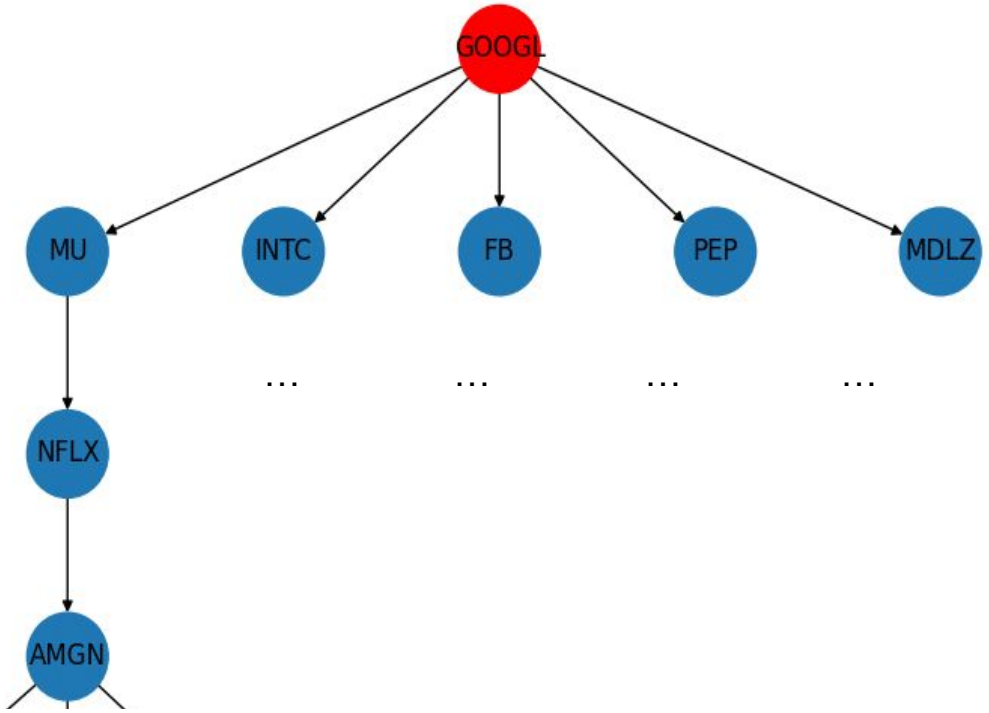


Stock Correlation Tree

We can build a **correlation Tree** as follows:

- The tree has GOOGL as root;
- At level 1 we have the stock that are directly correlated (MDLZ, FB, MU, INTC, and PEP);
- At level 2 we have the stocks that are correlated through a single node (e.g., NFLX is correlated through MU);
- At level 3 the stocks that are correlated through 2 nodes (e.g., AMGN is correlated to NFLX, that is correlated to MU and finally to GOOGL);

In general, at level i we have the stocks correlated through i-1 nodes to the root!



Group Project Work - Task 3

In the third project release we require to design and implement :

1) **A Python function that find the shortest correlation tree between a crypto and all the other cryptos in a given period of time**.

2) **A Python function that given a correlation tree for the crypto c finds all the cryptos that are correlated with c at level l**

Group Project Work

Report:

- **For the first two parts**, you can write a simple presentation about your implementation (about 2 slides). NOT MANDATORY, but HIGHLY SUGGESTED!
- **The final release is MANDATORY TO PASS the Project, and you must provide a presentation with at most 8 slides**

Describe your algorithmic idea, main implementation details, and experiments. You should try to analyze the asymptotic cost of your implementation.

Group Project Work

Score:

- The project contributes up to 8 points (added to the theory score).
- If you miss the May deadline, the maximum grade is lowered: you can achieve max 6 points (if you deliver the project by the second exam session), max 5 points (third session), max 4 points (fourth - and last - session)
- For **top projects**, we might consider assigning an extra **1-point bonus**.

You should work in groups of 3 students.

More details on Luiss-learn and Project guidelines pdf.

Software Project - Input Data

How to store financial data in a simple way?

You are given as input a *.txt* file containing a list of stocks and additional details.

Each line has:

crypto_name, day, price, volume

The values represent the price and volume for the *crypto_name* (e.g., ALGO) in that day.

Crypto	Day	Price	Volume
Gala	458	45	5559100
1inch	507	288	1938100
Etherium	464	75	3553000
Bitcoin	723	65	18966800
Gala	397	97	1314100
Algorand	588	1290	0
Algorand	581	1290	0
Etherium	727	504	0
Tether	643	1398	0

Software Project - Input Data

You are given four datasets*: dataset_small.txt, dataset_medium.txt, dataset_large.txt, dataset_full.txt*

In the first release we suggest to use only the “dataset_small.txt”.

Filename	Size	Rows	Cryptos	Average monitoring days per crypto
dataset_small.txt	901KB	~24k	98	247
dataset_medium.txt	~1.8MB	~48k	98	492
dataset_large.txt	~2.6MB	~69k	98	707
dataset_full.txt	~4.60MB	~120K	98	1232

Group Project Work - Implementation

As always we provide a skeleton of the code.
You should modify the code we provide, adding the missing parts.
In particular, you have to implement the functions in **“group0.py”**.
You can use this file to also add your helper functions.

You can download the code from Luiss-learn or Github:
https://github.com/flaat/Algorithms-2022-2023-Project

Group Project Work : Implementation

The python file **“group0.py”:**

The function **min_correlation_pathways** must return a dictionary containing as keys the nodes and as values a list containing the children of each node.

Remember that the weights on the edges are the correlation coefficient for each pair of cryptos!

```
108 def min_correlation_pathways(data,
109                                crypto: str,
110                                interval: Tuple[int,int]) -> Dict[str, List[str]]:
111     """
112     This function builds a minimal correlation pathways tree on the given
113     data structure for a specific cryptocurrency in a designated temporal
114     period. For each node x, the sum of the weights in the path from the root
115     to x must be minimal.
116
117     Parameters:
118     :data: A data structure that contains the information of all cryptos.
119     :crypto: The crypto name for which to build the tree.
120     :interval: The temporal period for which to build the tree.
121                It's in the form [x,y] where x is the beginning time and y is the end
122                time.
123
124     @return: The minimal correlation pathways tree
125     """
126     # TODO: Implement here your solution
127
128     return None
129
```

LUISS

Group Project Work : Implementation

The python file **“group0.py”:**

The function **correlated_cryptos_at_lvl_k** must return a list containing all the cryptos that are correlated at level l with the crypto given as input

```
131  def correlated_cryptos_at_lvl_k(data,
132                                  crypto: str,
133                                  level: int,
134                                  interval: Tuple[int,int]) -> List[str]:
135      """
136      This function retrieves the cryptocurrencies related to the one given in input
137      at a particular level of correlation in a designated temporal period [x,y].
138
139      Parameters:
140      :data: A data structure that contains the information of all cryptos.
141      :crypto: The crypto name to search correlations for.
142      :level: The level at which the correlated cryptos should stand at.
143      :interval: The temporal period for which to build the correlation tree.
144                 It's in the form [x,y] where x is the beginning time and y is the end
145                 time.
146
147      @return: A list of cryptocurrencies
148      """
149      # TODO: Implement here your solution
150      return None
```

LUISS

Group Project Work : Implementation

The python file **“group0.py”:**

The function **get_max_value** receives as input the crypto name (e.g., *”Gala”*) and the month, it outputs the maximum price of the crypto in that particular month and the day in which it was reached.

```python
def get_max_value(data, crypto: str, month: int) -> Tuple[int, float]:
    """
    This function must return the maximum price for a given crypto in
    a specific month.

    Parameters:

    :data: A data structure containing the information about the cryptos.
    :crypto: The crypto for which to search the maximum value.
    :month: The month in which to search for the maximum value.
          You, last month • changed the descr of get_max_value …
    Assumption: each month contains 30 days. Notice that the month can be
    a natural number in [1,inf). Example the 13th month represents the first
    month of the second year of monitoring; the 14th month represents the
    second month of the second year of monitoring, and so on.

    @return: A tuple containing the day in which the crypto reached the maximum price,
             along with the maximum value for that crypto
    """

    # TODO: Implement here your solution

    return (None, None)
```

LUISS

Group Project Work : Implementation

The python file **“group0.py”:**

The function **search** performs the modified searching algorithm described before! It takes in input the sorted data structure from **sort_data**

```python
def search(data, value: float, crypto: str) -> Tuple[int, float]:
    """
    This function searches for a specific price in a given data series and
    returns a tuple with the day and the price for a given cryptocurrency.
    If the searched value is not present in the data, the function returns the
    closest price. It compares two values of the data series, one at position i
    and the other at position j, and returns the price closest to the searched value.

    N.B.: If you have more than one possible day whose corresponding price is closest
    to the value in input, return the minimum day.

    Parameters:
    :data: A data structure that contains the value of price and volume of all cryptos.
    :value: The price value to be searched in the data.
    :crypto: The crypto name to search the value for.

    @return: A tuple containing the day on which the cryptocurrency reached the closest price
             and the closest price.
    """

    # TODO: Implement here your solution

      flaviogiorgi, 2 months ago • Added: grader.py, functions to implements
    return (None, None)
```

Thank you!

LUISS