

MAT101: Programming

Group Project: Analyzing Bitcoin rates and transactions

Loreno Heer

Deadline: December 4th, 2017

1 Objectives

The goal of this project is to explore Bitcoin price history and analyze the data using Python. In this project you will use a public API to acquire data about the price history of bitcoin. You will learn how to process this data in python and how to draw plots. Furthermore you will be able to monitor bitcoin addresses for transactions (using a public web service) and get other useful information from the bitcoin network. If you have completed the main tasks you can come up with some extra functionality of your choosing.

2 Prerequisites

Every member of the group needs to know the basic of the python pandas library (working with dataframes): <https://pandas.pydata.org/pandas-docs/stable/10min.html>.

Every member of the group should know the basics of creating plots with matplotlib https://matplotlib.org/users/pyplot_tutorial.html.

We use Quandl (a free online service) to get data about bitcoin. Quandl provides an API that can be conveniently accessed with Python. Take a look at the documentation: <https://docs.quandl.com/docs/python>. You do not need to understand this, but if you want to implement extra functionality, like getting historic rates of other cryptocurrencies then take a look. Note that you can search which databases are available on quandl like this: <https://www.quandl.com/search?query=bitcoin>.

Every group should make a (free account) at <https://www.quandl.com/> to get an API key. Otherwise you can only request historic data 50 times. Place the API key on top of the helperlib.py file, there is a comment indicating where to put it.

The project uses the Blockchain.info API to get data about live transactions. For example which address currently contains how much bitcoin. Where coins have been sent and so on. Some basic functionality is implemented in the helperlib.py file. Take a look and experiment with the functions.

Every member should document what they program, also indicate which part of the code has been written by whom.

3 Guidelines

Task 1 (Easy)

To get to know the code a bit, open the helperlib.py file in IDLE, place your API key there (see above), and run it (F5). You should now be able to fetch data like this from the python shell:

```
pr = get.bitcoin_price.usd() # this will fetch the entire available history
pr.head() # display the first few days
pr.tail() # display the last few days
```

or

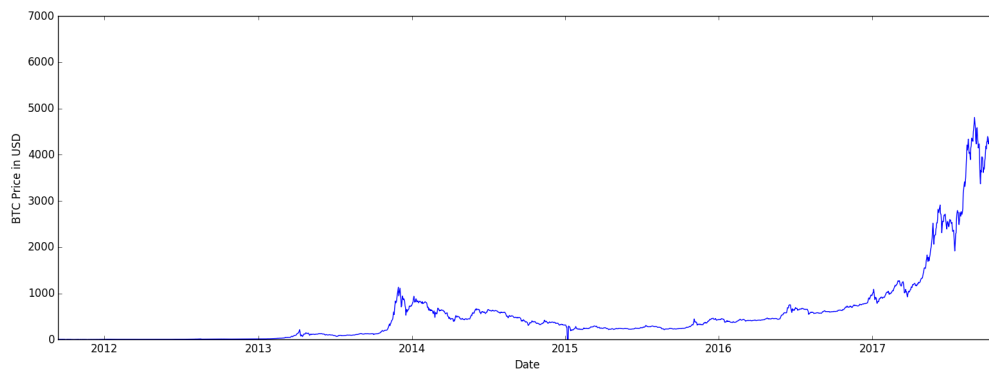
```
pr = get_bitcoin_price_usd_from_to('2017-09-01', '2017-10-01')
pr['Volume (Currency)'] # Trading volume
pr['Weighted Price'] # average
pr['High'] # highest price of the day
max(pr['Weighted Price']) # maximum of the price in the timeframe
```

Play around and try to understand these functions. Write down what you did and which results you got.

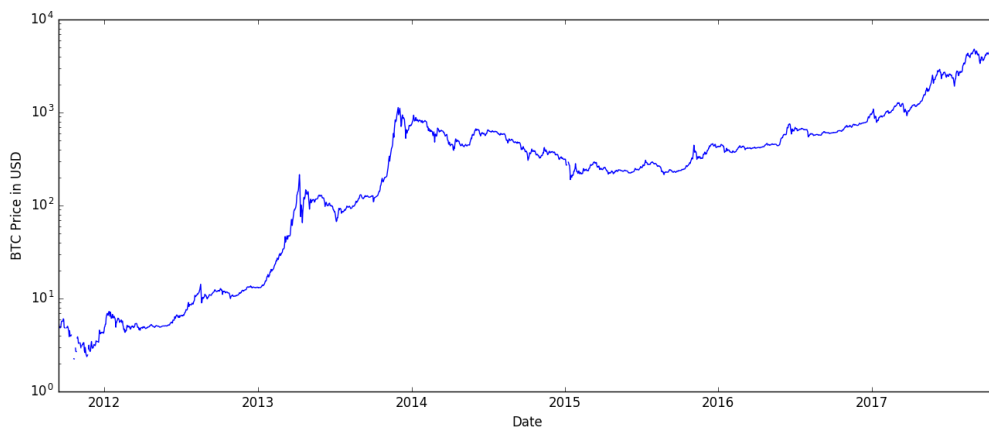
The following tasks are to be implemented in the file called `template.py`:

Task 2 (Easy)

Implement the function which plots the entire history or the specified year. Make sure you label the axes correctly and the plot looks nice. This could look like this for example: or if using a



logarithmic scale for the price, then like this:



Task 3 (Easy)

Use the function `getAddressBalanceBTC` from the `helperlib` to implement a function called `watchBtcAddress` (see template) which periodically checks the given address for a change in balance and prints it to the standard output. (optionally) Write a small gui using the `Tkinter` library which allows you to do the same thing in an interactive way.

Fun fact: The first commercial transaction of bitcoin took place in May 2010 when Laszlo Hanyecz bought two Pizzas for 10000 bitcoin. At that time the pizzas were worth approximately 41\$. The address of the transaction is: 17SkEw2md5avVNyYgj6RiXuQKNwkXaxFyQ

Task 4 (Difficult)

Have a look at curve fitting <https://sites.google.com/site/aslugsguidetopython/data-analysis/curve-fitting> and extrapolation <https://stackoverflow.com/questions/34159342/extrapolate-pandas-dataframe> of data using Python. Use a 3rd degree polynomial to fit and extrapolate the data (in the future and or the past). Make sure the plot looks nice.

Task 5 (Medium)

There are other cryptocurrencies besides bitcoin, currently the most traded ones of those are Litecoin, Zcash, Monero and Ethereum. The helperlib already contains code to receive the price history for those coins:

```
btc_price = get_bitcoin_price_usd()
ltc_price = get_litecoin_price_usd()
xmr_price = get_monero_price_usd()
zec_price = get_zcash_price_usd()
eth_price = get_ethereum_price_usd()
```

Create a plot that shows the various currencies beside each other (use different colors). Note that not all of them have data in the same timeframe. (Hint: The same has been done here using different libraries: <https://blog.patricktriest.com/analyzing-cryptocurrencies-python/>) See the link and create a correlation matrix for the year 2016 and 2017.

Task 6

This is an open-ended project. As a group come up with an additional task. Have a look here for some inspiration: <https://blog.patricktriest.com/analyzing-cryptocurrencies-python/> If you like to do something more difficult have a look at https://blockchain.info/api/blockchain_wallet_api the wallet API or at https://blockchain.info/api/api_receive. But this will require some knowledge of javascript as well.

4 Example Programs

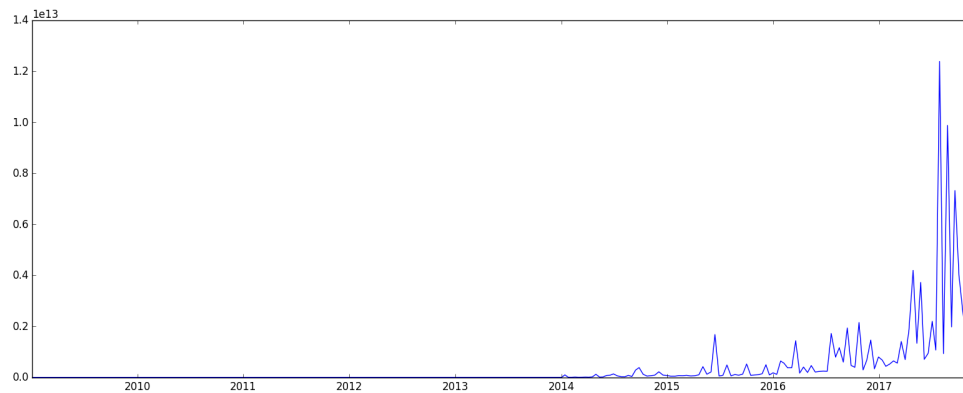
I added some example code in the folder.

4.1 Example 1: Bitcoin Mining difficulty

The blockchain api lets one query bitcoin blocks back to the first (genesis) block. Bitcoin adjusts its difficulty approximately every 2016 blocks. The mining difficulty is adjusted by increasing/decreasing some number. The hash of a mined block has to be lower than this number. Therefore decreasing the number increases the mining difficulty. One can determine the approximate difficulty by reading the hash of a given block and looking at the size of the number compared to the highest possible number. There is an example program provided which reads blocks in intervals of 2016 blocks, reads their hash and computes the mining difficulty at a given time. This information is then stored in a pandas data frame and plotted. The resulting plot looks like this:

5 Useful Links

- <https://coinmap.org/> (A map of local bitcoin businesses and bitcoin ATM's where you can exchange bitcoin to chf and vice versa)



- <https://pandas.pydata.org/pandas-docs/stable/10min.html> (10 Minutes to pandas, tutorial)
- <https://pandas.pydata.org/pandas-docs/stable/tutorials.html> (Extended tutorials for Pandas)
- https://matplotlib.org/users/pyplot_tutorial.html (Pyplot tutorial)
- <https://pythonhosted.org/spyder/> (Spyder Python IDE)
- <http://www.righto.com/2014/02/ascii-bernanke-wikileaks-photographs.html> Hidden messages in the bitcoin blockchain
- <https://satoshidice.com/> Bitcoin Online Casino (Has some frequently used addresses)

6 General Notes

- The goal of this project is to experience programming in a group. Discuss the project as a group and then divide the tasks among yourselves.
- You should discuss your progress with the supervisor of the project. Whenever you have questions about your project, feel free to ask them during the exercise class or post them in the forum.
- Once you have written your code you should briefly describe your results. You should include interesting examples and illustrations (if they are part of your project). This description may be very short; it should certainly not be more than one page of text.
- To hand in your project, just send an email to the supervisor of your project. Make sure that it is clear who is responsible for which task.
- It is important that you understand the entire code of your group, not just the part that you have written yourself. In particular, you should be familiar with the prerequisites.
- During the last week of the semester, every group will have a 15-minute oral exam on their project. The main examiner will be the supervisor of your project, but another assistant will also be in attendance. Each member of the group should prepare a 2-minute presentation of their own code and be ready to answer questions about the entire project.
- For the project you will be graded as a group, but for the oral exam you will be graded individually. Together the project and the oral exam account for 40 percent of your final grade: 30 percent for the project and 10 percent for the oral exam.

- The oral exams will take place during the exercise classes, i.e. on December 12th and December 15th. On December 4th you will be informed about the time slot of your oral exam.