cryptocurrencyeda

Berkay Bulut

Feb 01, 2022

CONTENTS

This is a Python package to analyze historical cryptocurrency prices and performance through simple exploratory data analysis including calculations and plotting. Data is sourced from the KuCoin API. There are four functions that are included in this python package which are described in more detail below. Cryptocurrency investors and enthusiasts can use this package to analyze cryptocurrencies of interest.

There are existing Python libraries to access information of cryptocurrency such as cryptocompare and cryptofeed. There are also existing Python libraries to visualize financial data such as mplfinance. However, there is no integrated Python library for accessing, analyzing, and visualizing cryptocurrency data altogether. Therefore, we want to build a simple tool that can facilitate simple cryptocurrency data analysis all at once.

CHAPTER

ONE

FUNCTION LIST

The package contains the following four functions:

- `retrieve_data`: downloads historical data from a cryptocurrency exchange using an an http request from a cryptocurrency exchange.
- `plot_price`: generates and visualizes a plot of the price of the cryptocurrenty inputted over a period of time.
- `daily_growth_rate`: performs calculation of daily growth rate of the cryptocurrenty inputted over a period of time.
- `avg_daily_return`: performs calculation of the average daily return of the inputted cryptocurrency price.

CHAPTER

TWO

INSTALLATION AND USAGE

In order to use the package, please follow these steps:

2.1 Create a new conda environment:

```
conda create --name cryptocurrencyeda python=3.9 -y
```

2.2 Activate the environment:

```
conda activate cryptocurrencyeda
```

2.3 Install the package:

```
pip install cryptocurrencyeda
or
pip install git+https://github.com/UBC-MDS/cryptocurrencyeda
```

2.4 Open Python:

```
Python
```

2.5 Import all functions:

```
>>> from cryptocurrencyeda.retrieve_data import retrieve_data
>>> from cryptocurrencyeda.plot_price import plot_price
>>> from cryptocurrencyeda.avg_daily_return import avg_daily_return
>>> from cryptocurrencyeda.daily_growth_rate import daily_growth_rate
```

2.6 Use the functions:

```
>>> retrieve_data(symbol="BTC-USDT",
                  time_period="1day",
                  start_date="2018-01-01",
                  end_date="2022-01-10",
                 )

>>> plot_price(price_df)

>>> daily_growth_rate(price_df, "Close")

>>> avg_daily_return(price_df["Close"])
```

CHAPTER

THREE

DOCUMENTATION

The documentation is hosted on ReadTheDocs here

CHAPTER

FOUR

CONTRIBUTORS

We welcome and recognize all contributions. You can see a list of current contributors in the `contributors tab`. If you are interested in contributing to this project, please check out our CONDUCT.md

- Berkay Bulut
- Cici Du
- Alex Yinan Guo
- Nobby Nguyen

CHAPTER

FIVE

LICENSE

`cryptocurrencyeda` was created by MDS Students from Group-11 for course 524. It is licensed under the terms of the MIT license.

CHAPTER

SIX

CREDITS

cryptocurrencyeda was created with cookiecutter and the py-pkgs-cookiecutter template.

6.1 Example usage

To use cryptocurrencyeda in a project:

6.1.1 Imports

```
from cryptocurrencyeda.retrieve_data import retrieve_data
from cryptocurrencyeda.plot_price import plot_price
from cryptocurrencyeda.avg_daily_return import avg_daily_return
from cryptocurrencyeda.daily_growth_rate import daily_growth_rate
```

6.1.2 Retrieve price data and store it in a data frame

```
price_df = retrieve_data(symbol="BTC-USDT",
                  time_period="1day",
                  start_date="2018-01-01",
                  end_date="2022-01-10",
                 )
```

We can take a look at the head of the data frame.

```
price_df.head()
```

```
     Symbol       Date    Close
0  BTC-USDT 2022-01-10  41816.3
1  BTC-USDT 2022-01-09  41877.6
2  BTC-USDT 2022-01-08  41680.5
3  BTC-USDT 2022-01-07  41561.4
4  BTC-USDT 2022-01-06  43086.1
```

6.1.3 Plot the price data

```
plot_price(price_df)
```

```
alt.Chart(...)
```

6.1.4 Calculate the daily growth rate

```
daily_growth_rate(price_df, "Close")
```

```
        Symbol       Date         Close  daily_growth_rate(%)
0     BTC-USDT 2022-01-10  41816.300000                   NaN
1     BTC-USDT 2022-01-09  41877.600000              0.146594
2     BTC-USDT 2022-01-08  41680.500000             -0.470657
3     BTC-USDT 2022-01-07  41561.400000             -0.285745
4     BTC-USDT 2022-01-06  43086.100000              3.668548
...        ...        ...           ...                   ...
1465  BTC-USDT 2018-01-06  16850.000001              3.990116
1466  BTC-USDT 2018-01-05  17266.663939              2.472783
1467  BTC-USDT 2018-01-04  15194.999997            -11.998056
1468  BTC-USDT 2018-01-03  15380.011104              1.217579
1469  BTC-USDT 2018-01-02  14100.000000             -8.322563

[1470 rows x 4 columns]
```

6.1.5 Calculate the average daily return

```
avg_daily_return(price_df["Close"])
```

```
-18.867460857726357
```

6.2 Changelog

6.2.1 v0.3.6 (2022-01-29)

6.2.2 v0.3.5 (2022-01-29)

Documentation

- Add dependencies jupyter (6f5b88b)
- Add example (bf94577)

6.2.3 v0.3.4 (2022-01-29)

Documentation

- Update readme (cfa3be1)

6.2.4 v0.3.3 (2022-01-29)

6.2.5 v0.3.2 (2022-01-29)

6.2.6 v0.3.1 (2022-01-29)

6.2.7 v0.3.0 (2022-01-28)

Feature

- Update tests (ef0fe07)
- Update function body (f437c15)

Documentation

- Update readme and func body (ddd0006)

6.2.8 v0.2.7 (2022-01-28)

6.2.9 v0.2.6 (2022-01-27)

6.2.10 v0.2.5 (2022-01-27)

Documentation

- Update docs folder (09a0fdf)
- Format README (d29f1bf)
- Update README (ac5f1c1)

6.2.11 v0.2.4 (2022-01-27)

6.2.12 v0.2.3 (2022-01-27)

6.2.13 v0.2.2 (2022-01-27)

6.2.14 v0.2.1 (2022-01-27)

6.2.15 v0.2.0 (2022-01-27)

Feature

- Update func and test (7b89582)
- Add function code (0d52247)
- Add function body and more tests (a59ffde)
- Add test for daily_growth_rate (0659060)

Documentation

- Add documentation for test function (29c877a)
- Edit contributing.md (5a953cf)

6.2.16 v0.1.0 (12/01/2022)

- First release of `cryptocurrencyeda`!

6.3 Contributing

Contributions are welcome, and they are greatly appreciated! Every little bit helps, and credit will always be given.

6.3.1 Contributors

Berkay Bulut Cici Du Alex Yinan Guo Nobby Nguyen

6.3.2 How to contribute

Here's how to set up `cryptocurrencyeda` for local development.

1. Download a copy of `cryptocurrencyeda` locally.
2. Install `cryptocurrencyeda` using `poetry`:

```
$ poetry install
```

3. Use `git` (or similar) to create a branch for local development and make your changes:

```
$ git checkout -b name-of-your-bugfix-or-feature
```

4. When you're done making changes, check that your changes conform to any code formatting requirements and pass any tests.
5. Commit your changes and open a pull request.
6. Another contributor will review your pull request and give feedback or merge your pull request.

6.3.3 Types of Contributions

Report Bugs

If you are reporting a bug, please include:

- Your operating system name and version.
- Any details about your local setup that might be helpful in troubleshooting.
- Detailed steps to reproduce the bug.

Fix Bugs

Look through the GitHub issues for bugs. Anything tagged with "bug" and "help wanted" is open to whoever wants to implement it.

Implement Features

Look through the GitHub issues for features. Anything tagged with "enhancement" and "help wanted" is open to whoever wants to implement it.

Write Documentation

You can never have enough documentation! Please feel free to contribute to any part of the documentation, such as the official docs, docstrings, etc.

Submit Feedback

If you are proposing a feature:

- Explain in detail how it would work.
- Keep the scope as narrow as possible, to make it easier to implement.

6.3.4 Pull Request Guidelines

Before you submit a pull request, check that it meets these guidelines:

1. The pull request should include additional tests if appropriate.
2. If the pull request adds functionality, the docs should be updated.
3. The pull request should work for all currently supported operating systems and versions of Python.

6.3.5 Code of Conduct

Please note that the `cryptocurrencyeda` project is released with a Code of Conduct. By contributing to this project you agree to abide by its terms.

6.4 Code of Conduct for all contributors

We aim towards building a positive, happy and open-minded community which welcomes all ideas from people of all walks of life. Everyone's inputs and ideas are equally valuable and instrumental in building better code, software and ultimately better products. This Code of Conduct documents our expectations for all contributors towards this project, whether you're a direct or indirect contirbutor.

6.4.1 Diversity and Inclusivity

We recognize and emphasize the importance of fostering an open community that welcomes inclusivity and diversity, and are striving towards building a 100% harassment-free environment for everyone, regardless of age, gender identity and expression, nationality, religion, etc.

6.4.2 Expected and unacceptable behaviours

Examples of expcted behaviours include:

- Understanding and respecting other peoples' viewpoints
- Being open to constructive criticism
- Showing gratitude and empathy towards other members
- Being encouraging towards online community to build better software

Examples of unacceptable behavior by participants include:

- Sexual harrassment through the use of text, words or any form of media
- Trolling, insulting/derogatory comments, and personal or political attacks
- Public or private harassment
- Publishing others' private information, such as addresses, without explicit permission

6.4.3 How to report and deal with unacceptable behaviour

Any instances of unacceptable behaviour may be reported by reaching out to the team via this email: berkayb@student.ubc.ca. The project team will review and investigate all complaints, and will respond in a way that it deems appropriate to the circumstances. The project team is obligated to maintain confidentiality with regard to the reporter of an incident. Further details of specific enforcement policies may be posted separately.

6.4.4 Consequences for unacceptable behaviour

Project maintainers who do not follow or enforce the Code of Conduct in good faith may face repercussions such as ones listed here:

- A verbal or emailed warning
- Temporarily removing the reported person from the online community
- Permanently removing the reported person from the online community
- Public announcement of an account of the incident
- Other enforcements if necessary

6.4.5 Our Responsibilities

Project maintainers are responsible for clarifying the standards of acceptable behavior and are expected to take appropriate and fair corrective action in response to any instances of unacceptable behavior. Project maintainers have the right and responsibility to remove, edit, or reject comments, commits, code, wiki edits, issues, and other contributions that are not aligned to this Code of Conduct, or to ban temporarily or permanently any contributor for other behaviors that they deem inappropriate, threatening, offensive, or harmful.

6.4.6 Scope

This Code of Conduct applies both within project spaces and in public spaces when an individual is representing the project or its community. Examples of representing a project or community include using an official project e-mail address, posting via an official social media account, or acting as an appointed representative at an online or offline event. Representation of a project may be further defined and clarified by project maintainers.

6.4.7 Attribution

This Code of Conduct is adapted from the Tidyverse Contributor Covenant Code of Conduct available at [https://github.com/tidyverse/tidyverse.org/blob/master/CODE_OF_CONDUCT.md]

6.5 API Reference

This page contains auto-generated API reference documentation[1].

6.5.1 cryptocurrencyeda

Submodules

cryptocurrencyeda.avg_daily_return

Module Contents

[1] Created with sphinx-autoapi

Functions

avg_daily_return(prices)	Outputs the average daily return of the inputted cryptocurrency price.

cryptocurrencyeda.avg_daily_return.**avg_daily_return**(*prices*)
Outputs the average daily return of the inputted cryptocurrency price.

Parameters **prices** (*array-like*) – Inputted cryptocurrency price.

Returns Average daily return of the cryptocurrency.

Return type float

Examples

```
>>> avg_daily_return("BTCBitcoin")
>>> 0.1
```

cryptocurrencyeda.daily_growth_rate

Module Contents

Functions

daily_growth_rate(df, col_name)	Function to calculate daily growth rate

cryptocurrencyeda.daily_growth_rate.**daily_growth_rate**(*df*, *col_name*)
Function to calculate daily growth rate :param df: Data frame with date and price data. :type df: pandas DataFrame :param col_name: Name of the column holding daily closing price data. :type col_name: str

Returns A dataframe with a new column of daily growth rate

Return type df

Examples

```
>>> daily_growth_rate(price_df, 'Close')
```

cryptocurrencyeda.plot_price

Module Contents

Functions

plot_price(df)	Plot the price of the cryptocurrenty inputted over window specified.

cryptocurrencyeda.plot_price.**plot_price**(*df*)

Plot the price of the cryptocurrenty inputted over window specified.

Parameters **df** (*pandas DataFrame*) – Data frame with cryptocurrency name, date and close price.

Returns

- **plot** (*plot object*) – An altair plot object.
- *Examples*
- **>>> df = retrieve_data(symbol (*str="BTC-USDT",*) – time_period:str="1day",** start_date:str="2018-01-01", end_date:str="2022-01-10",

)
- *>>> plot_price(df)*

cryptocurrencyeda.retrieve_data

Module Contents

Functions

retrieve_data(symbol: str = 'BTC-USDT', time_period: str = '1day', start_date: str = '2018-01-01', end_date: str = '2022-01-10')	Retrieves historical data from the KuCoin API.

cryptocurrencyeda.retrieve_data.**retrieve_data**(*symbol: str = 'BTC-USDT', time_period: str = '1day', start_date: str = '2018-01-01', end_date: str = '2022-01-10'*)

Retrieves historical data from the KuCoin API. Using open API adress "https://openapi-v2.kucoin.com/api/v1/market/history/trade"

Parameters

- **name** (*array-like*) – Inputted cryptocurrency symbol.
- **time_period** (*str*) – Inputted time period. 1min, 3min, 5min, 15min, 30min, 1hour, 2hour, 4hour, 6hour, 8hour, 12hour, 1day, 1week
- **start_date** (*string "%Y-%m-%d"*) – Inputted datetime. Minimum is 2018-01-01
- **end_date** (*string "%Y-%m-%d"*) – Inputted time frame.

Returns Historical data of the cryptocurrency.

Return type pandas.DataFrame

Package Contents

cryptocurrencyeda.__version__

PYTHON MODULE INDEX

C

INDEX

Symbols

A

C

D

M

P

R