

Research Project Installation Manual

vanGoethem, Joren

Maerten, Andreas

January 31, 2022

Contents

1	Data Collector	1
1.1	setup	1
2	Ticker Timescale-swap	1
2.1	Cloning	1
2.2	Build Dependencies	1
2.3	Compiling the Application	1
2.4	Using the Application	2
2.4.1	Changing time rescale ratio	2
2.4.2	Arguments	2
2.5	Verify Data Integrity	2
3	Data Preprocessor	2
3.1	cloning	2
3.2	Build Dependencies	2
3.3	Compiling the Application	3
3.4	Using the Application	3
3.5	Verify Data Integrity	3
4	Model Testing	3
5	References	3

1 Data Collector

1.1 setup

Setting up this repository is fairly simple, you can either clone the repo or download as a zip.

```
1 git clone https://github.com/Research-Project-Crypto/DataCollector.git
```

you will need a binance api key and secret to get started collecting crypto price data but this can be acquired for free by making a binance account and going to the Binance API management page[1], create an API-key and put it inside the crypto.py file.

collecting price data can take a long time depending on how much data you want, to prevent API bans there is timer between each api call. you can always stop the program when you think you have enough data.

To fetch reddit data you will also need to create an application on their site that has an api key and login. This can be found here[2]

2 Ticker Timescale-swap

The ticker timescale-swap application is used for changing the timescale of ticker data.

For example you'll most likely want to convert the most accurate data you have (1min ticker) to something less specific than that (5min / 10min / etc...)

2.1 Cloning

```
1 git clone https://github.com/Research-Project-Crypto/TickerTimescaleSwap.git --  
   recursive
```

If you forgot to clone the repository with the recursive argument you always use the following command:

```
1 git submodule update --init --recursive
```

2.2 Build Dependencies

The following commands are for Arch based systems, they will give you an idea on what's required to port it to other systems.

```
1 pacman -S --noconfirm --needed gcc make  
2 pacman -S --noconfirm --needed premake
```

2.3 Compiling the Application

```
1 generate build instructions  
2 premake5 gmake2  
3  
4 compile the actual application  
5 make config=release
```

2.4 Using the Application

2.4.1 Changing time rescale ratio

By default the time rescale is 1:60, currently you can't dynamically define this through command argument, so you'll have to edit the code directly and recompile. To change the ratio go into the `src/main.cpp` file and locate where the `processor.start()` is called. If you pass a number in the `.start` method you can set the time rescale.

2.4.2 Arguments

Position	Argument
1	Data Input Folder
2	Data Output Folder

Example:

```
1 TickerTimescaleSwap data/input_folder data/output_folder
```

2.5 Verify Data Integrity

Included with this project is a python script with which you can verify the binary output data.

```
1 python3 scripts/binary_reader.py
```

It will loop over all the cells slowly, this mostly to shortly verify calculation mistakes in the program.

3 Data Preprocessor

3.1 cloning

```
1 git clone https://github.com/Research-Project-Crypto/DataPreprocessor.git --  
   recursive
```

If you forgot to clone the repository with the recursive argument you always use the following command:

```
1 git submodule update --init --recursive
```

3.2 Build Dependencies

The following commands are for Arch based systems, they will give you an idea on what's required to port it to other systems.

```
1 pacman -S --noconfirm --needed gcc make  
2 pacman -S --noconfirm --needed premake  
3  
4 yay -S ta-lib  
5 # or  
6 paru -S ta-lib
```

3.3 Compiling the Application

```
1 generate build instructions
2 premake5 gmake2
3
4 compile the actual application
5 make config=release
```

3.4 Using the Application

With Arguments

Position	Argument
----------	----------

1	Data Input Folder
2	Data Output Folder

Example:

```
1 DataProcessor data/input_folder data/output_folder
```

Downside of argument only. With argument only mode you are unable to specify the type of the input data, you can only parse CSV text data.

Using settings.json If you don't give any arguments the application will default to reading the settings from settings.json. only use the is_binary option when you used our time rescaler since this will output to binary files to save disk space.

```
1 {
2     "input": {
3         "input_folder": "data/input",
4         "is_binary": false
5     },
6     "output": {
7         "output_folder": "data/output"
8     }
9 }
```

3.5 Verify Data Integrity

Included with this project is a python script with which you can verify the binary output data.

```
1 python3 scripts/binary_reader.py
```

It will loop over all the cells slowly, this mostly to shortly verify calculation mistakes in the program.

4 Model Testing

5 References

- [1] Binance api management settings:
<https://www.binance.com/en/my/settings/api-management>
- [2] reddit application page <https://www.reddit.com/prefs/apps>