twstock Documentation

發🄵 *1.0.1*

Louie Lu

2021 年 11 月 20 日

Contents

CHAPTER 1

twstock - 台灣股市股票價格🄵取

twstock 是一個簡潔好用的台灣股市程式，透過 twstock，您可以簡單的查詢各類股票之資訊以及即時的股票狀🄵。

twstock 設計僅支援 Python 3 以上之版本，目前最新版 Python 3.6.2 效能已有 Python 2.7 之能力，甚有超過之指標，作者在這邊也鼓勵使用者轉向使用 Python 3。

使用上有問題可以透過 issues 或是 twstock gitter.im 詢問。

1.1 使用指南 - User's Guide

1.1.1 準備工作

工欲善其事，必先利其器。在開始旅程之前，先讓我們准備好工作環境。以下的步驟會將所需的環境准備好，請照著使用，謝謝。

備🄵: twstock 要求需要 `Python 3` 以上之版本!

🄵烈建議使用 Python 3，🄵管 Python 2 了。

快速安裝 twstock (透過 pip)

如果你想要快速安裝 *twstock*，可以透過 *pip* 來安裝:

```
$ pip install --user twstock
```

如果想要安裝到所有使用者身上，請加上 *sudo*:

```
$ sudo pip install twstock
```

取得 twstock 原始碼

`twstock` 之原始碼存放於 github，你可以透過 `git clone` 獲得原始碼，或是自上方 github 網頁下載。以下將示範如何透過 `git clone` 取得原始碼:

```
$ git clone https://github.com/mlouielu/twstock
$ cd twstock
$ ls
docs  flit.ini  LICENSE  MANIFEST.in  README.md  requirements.txt
setup.py  test  twstock
```

手動安裝 twstock (透過原始碼)

取得原始碼後🄵且進入 *twstock* 之資料夾後，如果🄵有安裝過 `flit`，請先透過 `pip` 安裝:

```
$ pip install flit
```

接著透過 `flit` 即可安裝 *twstock*:

```
$ flit install
```

1.1.2 快速上手

下面將透過 Python REPL 來學習如何使用 `twstock`。

更新 TPEX / TWSE Codes

如果你是第一次使用 twstock，你可能會需要更新 TPEX/TWSE Codes。

你可以透過下面兩種方式更新:

以 CLI 更新:

```
$ twstock -U
```

以 Python 更新:

```
>>> import twstock
>>> twstock.__update_codes()
```

認識 Stock

在 twstock 之中，我們可以使用*Stock* 來取得歷史股票資訊。

歷史資料

舉例而言:

```
>>> import twstock
>>> stock = twstock.Stock('2330')
>>> stock.sid  # 回傳股票代號
'2330'
>>> stock.price  # 回傳各日之收盤價
[207.5, 208.0, 207.0, 208.0, 211.5, 213.0, 216.5, 215.5, 218.0,
 217.0, 215.0, 211.5, 208.5, 210.0, 208.5, 209.0, 207.0, 208.5,
 207.5, 206.0, 206.0, 212.0, 210.5, 214.5, 213.0, 213.0, 214.0,
 214.5, 215.5, 214.0, 214.5]
>>> stock.high  # 回傳各日之最高價
[210.0, 208.5, 209.5, 208.0, 212.0, 213.0, 218.0, 217.0, 218.0,
 218.5, 215.0, 214.0, 210.0, 210.5, 208.5, 209.0, 208.5, 208.5,
 208.5, 207.5, 207.0, 212.0, 212.5, 216.0, 214.5, 215.5, 214.0,
 215.0, 215.5, 215.0, 214.5]
```

在*Stock* 之中的資料，愈前面之資料越舊，愈後面之資料愈新，可以透過 date 取得各個資料集之中相對應的日期:

```
>>> stock.date  # 回傳資料之對應日期
[datetime.datetime(2017, 6, 12, 0, 0),
 datetime.datetime(2017, 6, 13, 0, 0),
 datetime.datetime(2017, 6, 14, 0, 0),
 datetime.datetime(2017, 6, 15, 0, 0),
 ...,
 datetime.datetime(2017, 7, 21, 0, 0),
 datetime.datetime(2017, 7, 24, 0, 0)]
```

獲取其他日期之資料

同時，*Stock* 預設建立時會取得近 31 日開盤之資料，如果需要其他日期之資料，可透過不同之 fetch 功能獲得:

```
>>> stock.fetch(2015, 7)  # 獲取 2015 年 7 月之股票資料
>>> stock.fetch(2010, 5)  # 獲取 2010 年 5 月之股票資料
>>> stock.fetch_31()      # 獲取近 31 日開盤之股票資料
>>> stock.fetch_from(2000, 10)  # 獲取 2000 年 10 月至今日之股票資料
```

基本股票資訊分析

Stock 🄵建基本股票分析功能，可以透過這些 method 來使用:

```
>>> stock.moving_average(stock.price, 5)  # 計算五日平均價格
[208.4, 209.5, 211.2, 212.9, 214.9, 216.0, 216.4, 215.4,
 214.0, 212.4, 210.7, 209.5, 208.6, 208.6, 208.1, 207.6,
 207.0, 208.0, 208.4, 209.8, 211.2, 212.6, 213.0, 213.8,
 214.0, 214.2, 214.5]
>>> stock.moving_average(stock.capacity, 5)  # 計算五日平均交易量
[40904388.2, 31779953.2, 27540112.6, 28800229.2, 30121867.6,
 31487778.6, 40018023.8, 43162160.8, 44540048.6, 44730965.6,
 43135743.0, 35320904.6, 30738402.0, 24976223.8, 22618522.2,
 20590067.6, 19042051.8, 21642392.4, 22327332.0, 29302556.6,
 29461849.0, 31076569.0, 27909064.6, 26663795.2, 20795579.4,
 19407173.2, 19127688.4]
>>> stock.ma_bias_ratio(5, 10)  # 計算五日、十日乖離值
[3.8000000000000114, 3.450000000000017, 2.0999999999999943,
 0.5500000000000114, -1.25, -2.6500000000000057,
 -3.4499999999999886, -3.4000000000000057, -2.700000000000017,
 -2.1500000000000057, -1.5500000000000114, -1.25,
 -0.30000000000001137, -0.09999999999999432,
 0.8500000000000227, 1.799999999999983, 2.799999999999983, 2.5,
 2.700000000000017, 2.0999999999999943, 1.5, 0.9499999999999886]
```

認識 BestFourPoint

BestFourPoint 四大買賣點判斷來自 toomore/grs 之中的一個功能，透過四大買賣點來判斷是否要買賣股票。四個買賣點分🄵🄵:

- 量大收紅 / 量大收黑

- 量縮價不跌 / 量縮價跌
- 三日均價由下往上 / 三日均價由上往下
- 三日均價大於六日均價 / 三日均價小於六日均價

使用範例如下:

```
>>> stock = twstock.Stock('2330')
>>> bfp = twstock.BestFourPoint(stock)
>>> bfp.best_four_point_to_buy()    # 判斷是否🄵四大買點
' 量大收紅，三日均價大於六日均價'
>>> bfp.best_four_point_to_sell()   # 判斷是否🄵四大賣點
False
>>> bfp.best_four_point()           # 綜合判斷
(True, ' 量大收紅，三日均價大於六日均價')
```

備🄵: BestFourPoint 是 Stock 的一層 wrapper，如果更動 Stock 之資料，將會直接影響 BestFourPoint 之結果，請特🄵注意。

認識 realtime

realtime 可以取得當前股票市場之即時資訊，可查詢上市以及上櫃之資料。同時可以透過*realtime.mock* 來設定是否使用假資料。

取得單一股票之即時資料

使用*realtime* 取得台積電 (2330) 之即時股票資料:

```
>>> import twstock
>>> stock = twstock.realtime.get('2330')  # 查詢上市股票之即時資料
{
   "timestamp": 1500877800.0,
   "info": {
      "code": "2330",
      "channel": "2330.tw",
      "name": " 台積電",
      "fullname": " 台灣積體電路🄵造股份有限公司",
      "time": "2017-07-24 14:30:00"
   },
   "realtime": {
```

(continues on next page)

(繼續上一頁)

```
      "latest_trade_price": "214.50",
      "trade_volume": "4437",
      "accumulate_trade_volume": "19955",
      "best_bid_price": [
            "214.00",
            "213.50",
            "213.00",
            "212.50",
            "212.00"
      ],
      "best_bid_volume": [
            "29",
            "1621",
            "2056",
            "1337",
            "1673"
      ],
      "best_ask_price": [
            "214.50",
            "215.00",
            "215.50",
            "216.00",
            "216.50"
      ],
      "best_ask_volume": [
            "736",
            "3116",
            "995",
            "1065",
            "684"
      ],
      "open": "213.50",
      "high": "214.50",
      "low": "213.00"
   },
   "success": true
}
>>> stock = twstock.realtime.get('6223')  # 查詢上櫃股票之即時資料
>>> stock
{'timestamp': 1500877800.0, 'info': {'code': '6223', 'channel': '6223.tw',
```

(continues on next page)

(繼續上一頁)

```
'name': ' 旺矽', 'fullname': ' 旺矽科技股份有限公司', 'time': '2017-07-24 14:30:00'},
'realtime': ..., 'success': True}
```

透過 *success* 確認資料之正確性

使用*realtime* 之資料時，需先確認 success 是否🄵 True，如果🄵 False 代表取得之資料有誤或是有錯誤🄵生，請再度參照 rtmessage 取得錯誤訊息、rtcode 取得錯誤代碼:

```
>>> stock = twstock.realtime.get('2330')
>>> stock['success']
True
>>> stock = twstock.realtime.get('')
>>> stock['success']
False
>>> stock
{'rtmessage': 'Information Data Not Found.', 'rtcode': '9999',
 'success': False}
>>> stock = twstock.realtime.get('9999')
>>> stock['success']
False
>>> stock
{'msgArray': [], 'userDelay': 0, 'rtmessage': 'Empty Query.',
 'referer': '', 'queryTime': {'sysTime': '17:27:02',
'sessionLatestTime': -1, 'sysDate': '20170724', 'sessionKey':
'tse_9999.tw_20170724|', 'sessionFromTime': -1, 'stockInfoItem': 1719,
'showChart': False, 'sessionStr': 'UserSession', 'stockInfo': 277019},
'rtcode': '5001', 'success': False}
```

多股票即時資料查詢

realtime 支援多個股票同時查詢:

```
>>> stocks = twstock.realtime.get(['2330', '2337', '2409'])
>>> stocks['success']
>>> stocks
{'2330': {'timestamp': 1500877800.0, ..., 'success': True},
 '2337': {'timestamp': 1500877800.0, ..., 'success': True},
 '2409': {'timestamp': 1500877800.0, ..., 'success': True},
 'success': True}
```

(continues on next page)

(繼續上一頁)

```
>>> stocks['2330']['success']
True
```

使用 mock

```
>>> twstock.realtime.mock = True
>>> twstock.realtime.get('2337')
```

認識 Codes

codes 提供了台灣股票代號之查詢，分🄵🄵*codes.tpex*、*codes.twse*、*codes.codes*。

查詢代號是否🄵上市股票:

```
>>> import twstock
>>> '2330' in twstock.twse
True
>>> '6223' in twstock.twse
False
```

查詢代號是否🄵上櫃股票:

```
>>> '2330' in twstock.tpex
False
>>> '6223' in twstock.tpex
True
```

查詢代號是否🄵台灣股票代號:

```
>>> '2330' in twstock.codes
True
>>> '6223' in twstock.codes
True
```

認識 Legacy

Legacy 用於初期自 toomore/grs 銜接驗證使用，包含兩組 grs 重要功能之驗證，分🄵🄵 LegacyAnalytics 以及 LegacyBestFourPoint。

認識 CLI tools

twstock 🅵建兩組 command line tools 可以使用，分🅵🅵查詢股票資訊以及四大買賣判斷之功能:

```
$ twstock -s 2330 6223
------------- 2330 ----------------
high : 215.0 214.0 210.0 210.5 208.5
low  : 212.0 211.0 208.0 208.5 206.5
price: 215.0 211.5 208.5 210.0 208.5
------------- 2337 ----------------
high :  16.2  16.8  16.4 16.75 16.75
low  :  15.8  16.1 15.15  16.3 16.25
price: 15.95 16.25 16.25  16.6  16.7

$ twstock -b 2330
四大買賣點判斷 Best Four Point
----------------------------
2330: Buy   量大收紅
6223: Sell  量縮價跌，三日均價小於六日均價
```

1.2 API 參照索引 - API Reference

如果你正在尋找某個特定指令、class 或 method 的介紹，請參照這部份的文件。

1.2.1 API Reference

stock —股票歷史資訊

stock 包含三個重要的元素: *DATATUPLE* 負責建立歷史股票資料之 namedtuple、*BaseFetcher* 作🅵*TWSEFetcher* 以及*TPEXFetcher* 之基底 class、*Stock* 封裝整個歷史股票資訊供使用者使用，同時*Stock* 會針對上市或上櫃的股票代號自動給予正確的 fetcher。

DATATUPLE

class stock.**DATATUPLE**(*date*, *capacity*, *turnover*, *open*, *high*, *low*, *close*, *change*, *transaction*)

歷史資料之 *nametuple*。

Attributes:

date

datetime.datetime 格式之時間，例如 datetime.datetime(2017, 6, 12, 0, 0)。

capacity
總成交股數 (單位: 股)。

turnover
總成交金額 (單位: 新台幣/元)。

open
開盤價。

high
盤中最高價

low
盤中最低價。

close
收盤價。

change
漲跌價差。

transaction
成交筆數。

Stock

class stock.**Stock**(*sid: str*, *initial_fetch: bool=True*)
有關股票歷史資訊 (開/收盤價，交易量，日期…etc) 以及簡易股票分析。建立*Stock* 實例時，若 `initial_fetch` 🄵 `True` (預設)，會自動呼叫*fetch_31()* 抓取近 31 日之歷史股票資料。

Class attributes are:

sid
股票代號。

fetcher
抓取方式之 instance，程式會自動判斷上櫃或上市，使用相對應之 fetcher。

raw_data
經由*TWSEFetcher* 或是*TPEXFetcher* 抓取之原始資料。

data
將*raw_data* 透過*DATATUPLE* 處理之歷史股票資料。

Fetcher method:

fetch(*self*, *year: int*, *month: int*)
🄵取該年、月份之歷史股票資料。

fetch_from(*self*, *year: int*, *month: int*)
🄵取自該年、月至今日之歷史股票資料。

fetch_31(*self*)
🄵取近 31 日開盤之歷史股票資料。

分析 method:

continuous(*self*, *data*)
data 之持續上升天數。

moving_average(*self*, *data: list*, *days: int*)
data 之 days 日均數值。

ma_bias_ratio(*self*, *day1*, *day2*)
計算 day1 日以及 day2 之乖離值。

ma_bias_ratio_pivot(*self*, *data*, *sample_size=5*, *position=False*)
判斷正負乖離。

Fetcher

class stock.BaseFetcher

fetch(*self*, *year*, *month*, *sid*, *retry*)
抓取相對應年月份之股票資料。

_convert_date(*self*, *date*)
回傳西元記年，將民國記年轉🄵🄵西元記年。舉例而言:

```
>>> date = self._convert_date('106/05/01')
>>> print(date)
'2017/05/01'
```

_make_datatuple(*self*, *data*)
將相對應之單日資料轉🄵🄵*DATATUPLE*。會將對應之資料轉🄵🄵對應型態。

purify(*self*, *original_data: list*)
將 original_data 🄵之所有資料轉🄵🄵*DATATUPLE* 型態。

class stock.TWSEFetcher(*BaseFetcher*)
台灣上市股票抓取

class stock.TPEXFetcher(*BaseFetcher*)
台灣上櫃股票抓取

analytics – 股票分析模組

analytics 提供了*Analytics* 股票基本分析以及*BestFourPoint* 四大買賣點分析。*Analytics* 會直接由*stock.Stock* 繼承，因此可以在*stock.Stock* 之中直接使用。

備🄵: *analytics* 之分析，僅適用於*stock.Stock* 歷史資料，無法針對*realtime* 之資料進行分析。

基本分析模組 - Analytics

class analytics.Analytics

基本股票分析模組。

continuous(*data*)

參數 data (*list*) – 資料樣本

分析 data 持續上升之天數。

moving_average(*data*, *days*)

參數

- data (*list*) – 資料樣本
- days (*int*) – 天數

分析 data 中之 days 日之平均數:

```
>>> stock.moving_average(stock.price, 5)     # 分析 5 日均價
>>> stock.moving_average(stock.capacity, 5)  # 分析 5 日均量
```

ma_bias_ratio(*day1*, *day2*)

參數

- day1 (*int*) – n 日
- day2 (*int*) – m 日

分析乖離率 (均價), day1 - day2

ma_bias_ratio_pivot(*data*, *sample_size=5*, *positive=False*)

參數

- data (*list*) – 資料樣本, 通常使用 price
- sample_size (*int*) – 計算的區間樣本數量
- positive (*bool*) – 正乖離 (True), 負乖離 (False)

計算正負乖離轉折位置

四大買賣點分析 - BestFourPoint

class analytics.BestFourPoint(*stock*)

參數 **stock** (*stock.Stock*) – 欲分析之股票

四大買賣點判斷

bias_ratio(*position=False*)

參數 **positive** (*bool*) – 正乖離 (True), 負乖離 (False)

判斷 3, 6 日正負乖離率

plus_bias_ratio()

判斷 3, 6 日正乖離率

mins_bias_ratio()

判斷 3, 6 日負乖離率

best_buy_1()

傳回型態 bool

判斷買點一: 是否量大收紅

best_buy_2()

傳回型態 bool

判斷買點二: 是否量縮價不跌

best_buy_3()

傳回型態 bool

判斷買點三: 是否三日均價由下往上

best_buy_4()

傳回型態 bool

判斷買點四: 三日均價大於六日均價

best_sell_1()

傳回型態 bool

判斷賣點一: 是否量大收黑

best_sell_2()

傳回型態 bool

判斷賣點二: 是否量縮價跌

best_sell_3()

傳回型態 bool

判斷賣點三: 是否三日均價由上往下

`best_sell_4()`

傳回型態 bool

判斷賣點四: 三日均價小於六日均價

`best_four_point_to_buy()`

傳回型態 bool, str

如果🄵買點，回傳所有符合之買點原則，否則回傳 False

`best_four_point_to_sell()`

傳回型態 bool, str

如果🄵賣點，回傳所有符合之賣點原則，否則回傳 False

`best_four_point()`

傳回型態 (bool, str), None

如果🄵買點，回傳 `(True, msg)`，如果🄵賣點，回傳 `(False, msg)`，如果皆不符合，回傳 `None`。

realtime —即時股票資訊

realtime 封裝了 基本市🄵報導網站 的即時股票資訊 API，透過*get()* 來取得相關股票即時資訊。

Attributes:

`realtime.mock`

透過*mock* 來設定是否使用假資料:

```
>>> twstock.realtime.mock = True   # 使用假資料
>>> twstock.realtime.mock = False  # 使用正常資料
```

Methods:

`realtime.get(`*stocks*`)`

提供包裝後之股票即時資料。

參數 **stocks** (str or list[str]) – 欲查詢之股票代號，多重搜尋請放入 list 之中:

```
>>> realtime.get('2330')            # 單一查詢
>>> realtime.get(['2330', '6223'])  # 多重查詢
```

傳回 dict – 回傳資料之格式如下

單一代碼:

```
>>> realtime.get('2330')
{
   'timestamp': 1500877800.0,
   'info': {
      'code': '2330',
      'channel': '2330.tw'
      'name': ' 台積電',
      'fullname': ' 台灣積體電路🄵造股份有限公司',
      'time': '2017-07-24 14:30:00'
   },
   'realtime': {
      'latest_trade_price': '214.50',
      'trade_volume': '4437',
      'accumulate_trade_volume':'19955',
      'best_bid_price': ['214.00', '213.50', '213.00', '212.50', '212.00'],
      'best_bid_volume': ['29', '1621', '2056', '1337', '1673'],
      'best_ask_price': ['214.50', '215.00', '215.50', '216.00', '216.50'],
      'best_ask_volume': ['736', '3116', '995', '1065', '684'],
      'open': '213.50',
      'high': '214.50',
      'low': '213.00'
   },
   'success': True
}
```

多重代碼:

```
>>> realtime.get(['2330', '2337'])
{
   '2330': ...,
   '2337': ...,
   'success': True
}
```

codes – 台灣股票代碼

codes 提供了台灣股票代碼的查詢功能，一般使用者🄴不會直接面對這個模組，僅會透過 twstock.codes、twstock.tpex、twstock.twse 接觸已解析完成之代號。

class codes.**StockCodeInfo**(*type*, *code*, *name*, *ISIN*, *start*, *market*, *group*, *CFI*)

各股票代號之資訊

`codes.codes`
: 台灣上市上櫃股票代號

`codes.tpex`
: 台灣上櫃股票代號

`codes.twse`
: 台灣上市股票代號

1.3 額外資訊 - Additional Notes

1.3.1 開源貢獻 - Contributing

twstock 非常歡迎您來貢獻，參與開發與除錯。

如果你有任何的建議、功能要求或是 bug 回報，請在 GitHub issues 開🄵一個新的 issue。如果要上傳 patch，請使用 GitHub pull request 來上傳 patch。當你的 patch 被合🄵時，你會自動被加入到 Contributors List 當中。

Coding Style Guide

Coding Style 請遵循，🄵參照 PEP 8。

1.3.2 LICENSE

CHAPTER 2

Indices and tables

- genindex
- modindex
- search

Python 模組索引

索引

符號

A

B

C

D

F

G

H

L

M

O

P

R

S

T