Fast TA

Release 0.1.3

Feb 17, 2021

Introduction

1 Additional Support **3**

2 Introduction **5**
2.1 Fast TA Installation Guide 5

3 Indicators **7**
3.1 Momentum Indicators 7
3.2 Volume Indicators 10
3.3 Volatility Indicators 12

4 Contributing **15**
4.1 Fast TA Testing And Benchmarking 15

5 License **17**
5.1 The MIT License (MIT) 17

6 Credits **19**

Index **21**

Fast TA is an optimized, high-level technical analysis library used to compute technical indicators on financial datasets. It is written entirely in C, and uses SIMD vectorization as well. Fast TA is built with the NumPy C API.

CHAPTER 1

Additional Support

- Ask or search for questions in StackOverflow using the fast-ta tag.
- Report bugs in our issue tracker.

CHAPTER 2

Introduction

2.1 Fast TA Installation Guide

2.1.1 Installing Fast TA

Fast TA runs on Python 3.x under CPython. NOTE: Fast TA requires NumPy.

To install Fast TA, run:

```
pip install fast-ta
```

Report any installation issues in our issue tracker.

Fast TA Installation Guide Get Fast TA installed on your computer.

CHAPTER 3

Indicators

3.1 Momentum Indicators

3.1.1 Awesome Oscillator

`fast_ta.momentum.AO`(*high : np.array*, *low : np.array*, *s : int = 5*, *l : int = 34*) → np.array

Compute the Awesome Oscillator Indicator.

Parameters

- **high** (*np.array*) – High Time Series
- **low** (*np.array*) – Low Time Series
- **s** (*int*) – Short Period
- **l** (*int*) – Long Period

Returns Awesome Oscillator Indicator

Return type np.array

3.1.2 KAMA

`fast_ta.momentum.KAMA`(*close : np.array*, *n : int = 10*, *f : int = 2*, *s : int = 30*) → np.array

Compute the KAMA Indicator.

Parameters

- **close** (*np.array*) – Close Time Series
- **n** (*int*) – Period
- **f** (*int*) – Fast EMA Periods
- **s** (*int*) – Slow EMA Periods

Returns KAMA Indicator

Return type np.array

3.1.3 ROC

`fast_ta.momentum.ROC(close : np.array, n : int = 12)` → np.array

Compute the ROC Indicator.

Parameters

- **close** (*np.array*) – Close Time Series
- **n** (*int*) – Period

Returns ROC Indicator

Return type np.array

3.1.4 RSI

`fast_ta.momentum.RSI(close : np.array, n : int = 14, threads : int = 1)` → np.array

Compute the Relative Strength Indicator.

Parameters

- **close** (*np.array*) – Close Time Series
- **n** (*int*) – Period
- **threads** (*int*) – Number of Threads to Use During Computation (Experimental)

Returns Relative Strength Indicator

Return type np.array

3.1.5 Stochastic Oscillator

`fast_ta.momentum.StochasticOscillator(high : np.array, low : np.array, close : np.array, n : int = 14, d_n : int = 3)` → np.array

Compute the Stochastic Oscillator Indicator.

Parameters

- **high** (*np.array*) – High Time Series
- **low** (*np.array*) – Low Time Series
- **close** (*np.array*) – Close Time Series
- **n** (*int*) – Period
- **d_n** (*int*) – SMA Period

Returns Stochastic Oscillator Indicator And Signal Line

Return type (np.array, np.array)

3.1.6 TSI

`fast_ta.momentum.TSI`(*close : np.array*, *r : int = 25*, *s : int = 13*) → np.array

Compute the True Strength Indicator.

Parameters

- **close** (*np.array*) – Close Time Series
- **r** (*int*) – First EMA Period
- **s** (*int*) – Second EMA Period

Returns True Strength Indicator

Return type np.array

3.1.7 Ultimate Oscillator

`fast_ta.momentum.UltimateOscillator`(*high : np.array*, *low : np.array*, *close : np.array*, *s : int = 7*, *m : int = 14*, *l : int = 28*, *ws : float = 4*, *wm : float = 2*, *wl : float = 1*) → np.array

Compute the Ultimate Oscillator Indicator.

Parameters

- **high** (*np.array*) – High Time Series
- **low** (*np.array*) – Low Time Series
- **close** (*np.array*) – Close Time Series
- **s** (*int*) – Short Period
- **m** (*int*) – Medium Period
- **l** (*int*) – Long Period
- **ws** (*float*) – Short Period Weight
- **wm** (*float*) – Medium Period Weight
- **wl** (*float*) – Long Period Weight

Returns Ultimate Oscillator Indicator

Return type np.array

3.1.8 Williams %R

`fast_ta.momentum.WilliamsR`(*high : np.array*, *low : np.array*, *close : np.array*, *n : int = 14*) → np.array

Compute the Williams %R Indicator.

Parameters

- **high** (*np.array*) – High Time Series
- **low** (*np.array*) – Low Time Series
- **close** (*np.array*) – Close Time Series
- **n** (*int*) – Period

Returns Williams %R Indicator

Return type np.array

3.2 Volume Indicators

3.2.1 Accumulation/Distribution Index (ADI)

`fast_ta.volume.ADI`(*high : np.array*, *low : np.array*, *close : np.array*, *volume : np.array*) → np.array

Compute the Accumulation/Distribution Index Indicator.

Parameters

- **high** (*np.array*) – High Time Series
- **low** (*np.array*) – Low Time Series
- **close** (*np.array*) – Close Time Series
- **volume** (*np.array*) – Volume Time Series

Returns Accumulation/Distribution Index Indicator

Return type np.array

3.2.2 Chaikin Money Flow (CMF)

`fast_ta.volume.CMF`(*high : np.array*, *low : np.array*, *close : np.array*, *volume : np.array*, *n : int = 20*) → np.array

Compute the Chaikin Money Flow (CMF) Indicator.

Parameters

- **high** (*np.array*) – High Time Series
- **low** (*np.array*) – Low Time Series
- **close** (*np.array*) – Close Time Series
- **volume** (*np.array*) – Volume Time Series
- **n** (*int*) – Period

Returns Chaikin Money Flow (CMF) Indicator

Return type np.array

3.2.3 Ease of movement (EoM, EMV)

`fast_ta.volume.EMV`(*high : np.array*, *low : np.array*, *volume : np.array*, *n : int = 14) -> (np.array*, *np.array*)

Compute the Ease of movement (EoM, EMV) Indicator.

Parameters

- **high** (*np.array*) – High Time Series
- **low** (*np.array*) – Low Time Series
- **volume** (*np.array*) – Volume Time Series

- **n** (*int*) – Period

Returns Ease of movement (EoM, EMV) Indicator, n-Period SMA of EMV Indicator

Return type (np.array, np.array)

3.2.4 Force Index (FI)

`fast_ta.volume.FI`(*close : np.array*, *volume : np.array*, *n : int = 13*) → np.array

Compute the Force Index (FI) Indicator.

Parameters

- **close** (*np.array*) – Close Time Series
- **volume** (*np.array*) – Volume Time Series
- **n** (*int*) – Period

Returns Force Index (FI) Indicator

Return type np.array

3.2.5 Money Flow Index (MFI)

`fast_ta.volume.MFI`(*high : np.array*, *low : np.array*, *close : np.array*, *volume : np.array*, *n : int = 14*) → np.array

Compute the Money Flow Index (MFI) Indicator.

Parameters

- **high** (*np.array*) – High Time Series
- **low** (*np.array*) – Low Time Series
- **close** (*np.array*) – Close Time Series
- **volume** (*np.array*) – Volume Time Series
- **n** (*int*) – Period

Returns Money Flow Index (MFI) Indicator

Return type np.array

3.2.6 Negative Volume Index (NVI)

`fast_ta.volume.NVI`(*close : np.array*, *volume : np.array*) → np.array

Compute the Negative Volume Index (NVI) Indicator.

Parameters

- **close** (*np.array*) – Close Time Series
- **volume** (*np.array*) – Volume Time Series

Returns Negative Volume Index (NVI) Indicator

Return type np.array

3.2.7 On-Balance Volume (OBV)

`fast_ta.volume.OBV`(*close : np.array*, *volume : np.array*) → np.array

Compute the On-Balance Volume (OBV) Indicator.

Parameters

- **close** (*np.array*) – Close Time Series
- **volume** (*np.array*) – Volume Time Series

Returns On-Balance Volume (OBV) Indicator

Return type np.array

3.2.8 Volume-Price Trend (VPT)

`fast_ta.volume.VPT`(*close : np.array*, *volume : np.array*) → np.array

Compute the Volume-Price Trend (VPT) Indicator.

Parameters

- **close** (*np.array*) – Close Time Series
- **volume** (*np.array*) – Volume Time Series

Returns Volume-Price Trend (VPT) Indicator

Return type np.array

3.2.9 Volume Weighted Average Price (VWAP)

`fast_ta.volume.VWAP`(*high : np.array*, *low : np.array*, *close : np.array*, *volume : np.array*, *n : int = 14*) → np.array

Compute the Volume Weighted Average Price (VWAP) Indicator.

Parameters

- **high** (*np.array*) – High Time Series
- **low** (*np.array*) – Low Time Series
- **close** (*np.array*) – Close Time Series
- **volume** (*np.array*) – Volume Time Series
- **n** (*int*) – Period

Returns Volume Weighted Average Price (VWAP) Indicator

Return type np.array

3.3 Volatility Indicators

3.3.1 Average True Range (ATR)

`fast_ta.volatility.ATR`(*high : np.array*, *low : np.array*, *close : np.array*, *n : int = 14*) → np.array

Compute the Average True Range (ATR) Indicator.

Parameters

- **high** (*np.array*) – High Time Series
- **low** (*np.array*) – Low Time Series
- **close** (*np.array*) – Close Time Series
- **n** (*int*) – Period

Returns Average True Range (ATR) Indicator

Return type np.array

3.3.2 Bollinger Bands (BOL)

`fast_ta.volatility.BOL`(*close : np.array*, *n : int = 20*, *ndev : float = 2.0*) → np.array

Compute the Bollinger Bands (BOL) Indicator.

Parameters

- **close** (*np.array*) – Close Time Series
- **n** (*int*) – Period
- **ndev** (*int*) – Standard Deviation Factor

Returns Bollinger Bands (BOL) Indicator

Return type np.array

3.3.3 Donchian Channel (DC)

`fast_ta.volatility.DC`(*high : np.array*, *low : np.array*, *n : int = 14*) → np.array

Compute the Donchian Channel (DC) Indicator.

Parameters

- **high** (*np.array*) – High Time Series
- **low** (*np.array*) – Low Time Series
- **n** (*int*) – Period

Returns Lower, Middle, and Upper Donchian Channel (DC) Indicators

Return type np.array

3.3.4 Keltner Channel (KC)

`fast_ta.volatility.KC`(*high : np.array*, *low : np.array*, *close : np.array*, *n1 : int = 14*, *n2 : int = 10*, *num_channels : int = 1*) → np.array

Compute the Keltner Channel (KC) Indicator.

Parameters

- **high** (*np.array*) – High Time Series
- **low** (*np.array*) – Low Time Series
- **close** (*np.array*) – Close Time Series
- **n1** (*int*) – EMA Period
- **n2** (*int*) – ATR Period

- **num_channels** (*int*) – Number of Bands In Each Direction Around EMA

Returns Keltner Channel (KC) Indicator Lines

Return type np.array

Momentum Indicators View the documentation for the momentum indicators.

Volume Indicators View the documentation for the volume indicators.

Volatility Indicators View the documentation for the volatility indicators.

CHAPTER 4

Contributing

4.1 Fast TA Testing And Benchmarking

4.1.1 Testing

The tests.py script located in the tests folder is the script we use to test the Fast TA library.

To view help, simply run:

```
python3.x tests/tests.py
```

NOTE: The TA library is required for testing.

4.1.2 Benchmarking

Withing the benchmarks folder, there are scripts written to benchmark the relative performance between the Fast TA and TA libraries.

To execute all benchmarks, simply run:

```
cd benchmarks && ./run_all.sh
```

NOTE: The TA library is required for benchmarking.

Fast TA Testing And Benchmarking Learn how to test and benchmark the Fast TA library.

CHAPTER 5

License

5.1 The MIT License (MIT)

Copyright 2020 Cristian Bicheru, Calder White

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

The MIT License (MIT) The Fast TA Library is licensed under the MIT License.

CHAPTER 6

Credits

Developed with passion by the Fast TA Team.

Index

F

`fast_ta.momentum.AO()` (*built-in function*), 7
`fast_ta.momentum.KAMA()` (*built-in function*), 7
`fast_ta.momentum.ROC()` (*built-in function*), 8
`fast_ta.momentum.RSI()` (*built-in function*), 8
`fast_ta.momentum.StochasticOscillator()` (*built-in function*), 8
`fast_ta.momentum.TSI()` (*built-in function*), 9
`fast_ta.momentum.UltimateOscillator()` (*built-in function*), 9
`fast_ta.momentum.WilliamsR()` (*built-in function*), 9
`fast_ta.volatility.ATR()` (*built-in function*), 12
`fast_ta.volatility.BOL()` (*built-in function*), 13
`fast_ta.volatility.DC()` (*built-in function*), 13
`fast_ta.volatility.KC()` (*built-in function*), 13
`fast_ta.volume.ADI()` (*built-in function*), 10
`fast_ta.volume.CMF()` (*built-in function*), 10
`fast_ta.volume.EMV()` (*built-in function*), 10
`fast_ta.volume.FI()` (*built-in function*), 11
`fast_ta.volume.MFI()` (*built-in function*), 11
`fast_ta.volume.NVI()` (*built-in function*), 11
`fast_ta.volume.OBV()` (*built-in function*), 12
`fast_ta.volume.VPT()` (*built-in function*), 12
`fast_ta.volume.VWAP()` (*built-in function*), 12