## CONTENTS

1. **Installation (python >= v3.6)**  
   - 3
2. **Examples**  
   - 5
3. **Motivation**  
   - 7
4. **Contents**  
   - 9
   - 4.1 Documentation  
     - 9
5. **Indices and tables**  
   - 49

Python Module Index  
- 51

Index  
- 53
It is a Technical Analysis library to financial time series datasets (open, close, high, low, volume). You can use it to do feature engineering from financial datasets. It is builded on Python Pandas library.
CHAPTER ONE

INSTALLATION (PYTHON >= V3.6)

```bash
> virtualenv -p python3 virtualenvironment
> source virtualenvironment/bin/activate
> pip install ta
```
Example adding all features:

```python
import pandas as pd
import ta

# Load datas
df = pd.read_csv('ta/tests/data/datas.csv', sep=',')

# Clean NaN values
df = ta.utils.dropna(df)

# Add ta features filling NaN values
df = ta.add_all_ta_features(
    df, open="Open", high="High", low="Low", close="Close", volume="Volume_BTC",
   fillna=True)
```

Example adding a particular feature:

```python
import pandas as pd
import ta

# Load datas
df = pd.read_csv('ta/tests/data/datas.csv', sep=',')

# Clean NaN values
df = ta.utils.dropna(df)

# Initialize Bollinger Bands Indicator
indicator_bb = ta.volatility.BollingerBands(close=df["Close"], n=20, ndev=2)

# Add Bollinger Bands features
df['bb_bbm'] = indicator_bb.bollinger_mavg()
df['bb_bbh'] = indicator_bb.bollinger_hband()
df['bb_bbl'] = indicator_bb.bollinger_lband()

# Add Bollinger Band high indicator
df['bb_bbhi'] = indicator_bb.bollinger_hband_indicator()

# Add Bollinger Band low indicator
df['bb_bbli'] = indicator_bb.bollinger_lband_indicator()
```
CHAPTER
THREE

MOTIVATION

• **English:** https://towardsdatascience.com/technical-analysis-library-to-financial-datasets-with-pandas-python-4b2b390d3543

• **Spanish:** https://medium.com/datos-y-ciencia/biblioteca-de-an%C3%A1lisis-t%C3%A9cnico-sobre-series-temporales-financieras-para-machine-learning-con-cb28f9427d0
4.1 Documentation

It is a technical analysis library to financial time series datasets. You can use it to do feature engineering from financial datasets. It is builted on pandas python library.

4.1.1 Momentum Indicators

Momentum Indicators.

```python
class ta.momentum.AwesomeOscillatorIndicator(high: pandas.core.series.Series, low: pandas.core.series.Series, s: int = 5, len: int = 34, fillna: bool = False)
```

Awesome Oscillator

From: https://www.tradingview.com/wiki/Awesome_Oscillator_(AO)

The Awesome Oscillator is an indicator used to measure market momentum. AO calculates the difference of a 34 Period and 5 Period Simple Moving Averages. The Simple Moving Averages that are used are not calculated using closing price but rather each bar’s midpoints. AO is generally used to affirm trends or to anticipate possible reversals.

From: https://www.ifcm.co.uk/ntx-indicators/awesome-oscillator

Awesome Oscillator is a 34-period simple moving average, plotted through the central points of the bars (H+L)/2, and subtracted from the 5-period simple moving average, graphed across the central points of the bars (H+L)/2.

MEDIAN PRICE = (HIGH+LOW)/2
AO = SMA(MEDIAN PRICE, 5)-SMA(MEDIAN PRICE, 34)

where

SMA — Simple Moving Average.

Parameters

- **high (pandas.Series)** — dataset ‘High’ column.
- **low (pandas.Series)** — dataset ‘Low’ column.
- **s (int)** — short period.
- **len (int)** — long period.
- **fillna (bool)** — if True, fill nan values with -50.
Kaufman’s Adaptive Moving Average (KAMA)

Moving average designed to account for market noise or volatility. KAMA will closely follow prices when the price swings are relatively small and the noise is low. KAMA will adjust when the price swings widen and follow prices from a greater distance. This trend-following indicator can be used to identify the overall trend, time turning points and filter price movements.

https://www.tradingview.com/ideas/kama/

Parameters

- **close** *(pandas.Series)* – dataset ‘Close’ column.
- **n** *(int)* – n period.
- **pow1** *(int)* – number of periods for the fastest EMA constant.
- **pow2** *(int)* – number of periods for the slowest EMA constant.
- **fillna** *(bool)* – if True, fill nan values.

Money Flow Index (MFI)

Uses both price and volume to measure buying and selling pressure. It is positive when the typical price rises (buying pressure) and negative when the typical price declines (selling pressure). A ratio of positive and negative money flow is then plugged into an RSI formula to create an oscillator that moves between zero and one hundred.


Parameters

- **high** *(pandas.Series)* – dataset ‘High’ column.
- **close** *(pandas.Series)* – dataset ‘Close’ column.
- **n** *(int)* – n period.
- **fillna** *(bool)* – if True, fill nan values.
class `ta.momentum.ROCIndicator` (close: pandas.core.series.Series, n: int = 12, fillna: bool = False)

Rate of Change (ROC)

The Rate-of-Change (ROC) indicator, which is also referred to as simply Momentum, is a pure momentum oscillator that measures the percent change in price from one period to the next. The ROC calculation compares the current price with the price “n” periods ago. The plot forms an oscillator that fluctuates above and below the zero line as the Rate-of-Change moves from positive to negative. As a momentum oscillator, ROC signals include centerline crossovers, divergences and overbought-oversold readings. Divergences fail to foreshadow reversals more often than not, so this article will forgo a detailed discussion on them. Even though centerline crossovers are prone to whipsaw, especially short-term, these crossovers can be used to identify the overall trend. Identifying overbought or oversold extremes comes naturally to the Rate-of-Change oscillator.


Parameters

- close (pandas.Series) – dataset ‘Close’ column.
- n (int) – n period.
- fillna (bool) – if True, fill nan values.

roc() → pandas.core.series.Series

Rate of Change (ROC)

Returns New feature generated.

Return type pandas.Series

class `ta.momentum.RSIIndicator` (close: pandas.core.series.Series, n: int = 14, fillna: bool = False)

Relative Strength Index (RSI)

Compares the magnitude of recent gains and losses over a specified time period to measure speed and change of price movements of a security. It is primarily used to attempt to identify overbought or oversold conditions in the trading of an asset.

https://www.investopedia.com/terms/r/rsi.asp

Parameters

- close (pandas.Series) – dataset ‘Close’ column.
- n (int) – n period.
- fillna (bool) – if True, fill nan values.

rsi() → pandas.core.series.Series

Relative Strength Index (RSI)

Returns New feature generated.

Return type pandas.Series


Stochastic Oscillator

Developed in the late 1950s by George Lane. The stochastic oscillator presents the location of the closing price of a stock in relation to the high and low range of the price of a stock over a period of time, typically a 14-day period.

Parameters

- **close** (*pandas.Series*) – dataset ‘Close’ column.
- **high** (*pandas.Series*) – dataset ‘High’ column.
- **n** (*int*) – n period.
- **d_n** (*int*) – sma period over stoch_k.
- **fillna** (*bool*) – if True, fill nan values.

`stoch()` → *pandas.core.series.Series*

Stochastic Oscillator

**Returns**
New feature generated.

**Return type**
*pandas.Series*

`stoch_signal()` → *pandas.core.series.Series*

Signal Stochastic Oscillator

**Returns**
New feature generated.

**Return type**
*pandas.Series*

```python
class ta.momentum.TSIIndicator(close: pandas.core.series.Series, r: int = 25, s: int = 13,fillna: bool = False)
```

True strength index (TSI)

Shows both trend direction and overbought/oversold conditions.


```
BP = Close - Minimum(Low or Prior Close). TR = Maximum(High or Prior Close) - Minimum(Low or Prior Close) Average7 = (7-period BP Sum) / (7-period TR Sum) Average14 = (14-period BP Sum) / (14-period TR Sum) Average28 = (28-period BP Sum) / (28-period TR Sum)
```
UO = 100 x [(4 x Average7)+(2 x Average14)+Average28]/(4+2+1)

Parameters
- **high** (pandas.Series) – dataset ‘High’ column.
- **low** (pandas.Series) – dataset ‘Low’ column.
- **close** (pandas.Series) – dataset ‘Close’ column.
- **s** (int) – short period.
- **m** (int) – medium period.
- **len** (int) – long period.
- **ws** (float) – weight of short BP average for UO.
- **wm** (float) – weight of medium BP average for UO.
- **wl** (float) – weight of long BP average for UO.
- **fillna** (bool) – if True, fill nan values with 50.

uo() → pandas.core.series.Series
Ultimate Oscillator

Returns New feature generated.

Return type pandas.Series


Williams %R

Developed by Larry Williams, Williams %R is a momentum indicator that is the inverse of the Fast Stochastic Oscillator. Also referred to as %R, Williams %R reflects the level of the close relative to the highest high for the look-back period. In contrast, the Stochastic Oscillator reflects the level of the close relative to the lowest low. %R corrects for the inversion by multiplying the raw value by -100. As a result, the Fast Stochastic Oscillator and Williams %R produce the exact same lines, only the scaling is different. Williams %R oscillates from 0 to -100.

Readings from 0 to -20 are considered overbought. Readings from -80 to -100 are considered oversold.

Unsurprisingly, signals derived from the Stochastic Oscillator are also applicable to Williams %R.

%R = (Highest High - Close)/(Highest High - Lowest Low) * -100

Lowest Low = lowest low for the look-back period
Highest High = highest high for the look-back period %R is multiplied by -100 correct the inversion and move the decimal.

https://school.stockcharts.com/doku.php?id=technical_indicators:williams_r

The Williams %R oscillates from 0 to -100. When the indicator produces readings from 0 to -20, this indicates overbought market conditions. When readings are -80 to -100, it indicates oversold market conditions.

Parameters
- **high** (pandas.Series) – dataset ‘High’ column.
- **low** (pandas.Series) – dataset ‘Low’ column.
- **close** (pandas.Series) – dataset ‘Close’ column.
- **lbp** (int) – lookback period.
- **fillna** (bool) – if True, fill nan values with -50.
wr() → pandas.core.series.Series
Williams %R

Returns New feature generated.

Return type pandas.Series

ta.momentum.ao(high, low, s=5, len=34,fillna=False)
Awesome Oscillator

From: https://www.tradingview.com/wiki/Awesome_Oscillator_(AO)

The Awesome Oscillator is an indicator used to measure market momentum. AO calculates the difference of a
34 Period and 5 Period Simple Moving Averages. The Simple Moving Averages that are used are not calculated
using closing price but rather each bar’s midpoints. AO is generally used to affirm trends or to anticipate possible
reversals.

From: https://www.ifcm.co.uk/ntx-indicators/awesome-oscillator

Awesome Oscillator is a 34-period simple moving average, plotted through the central points of the bars
(H+L)/2, and subtracted from the 5-period simple moving average, graphed across the central points of the
bars (H+L)/2.

MEDIAN PRICE = (HIGH+LOW)/2

AO = SMA(MEDIAN PRICE, 5)-SMA(MEDIAN PRICE, 34)

where

SMA — Simple Moving Average.

Parameters

• high (pandas.Series) – dataset ‘High’ column.
• low (pandas.Series) – dataset ‘Low’ column.
• s (int) – short period.
• len (int) – long period.
• fillna (bool) – if True, fill nan values with -50.

Returns New feature generated.

Return type pandas.Series

ta.momentum.kama(close, n=10, pow1=2, pow2=30,fillna=False)
Kaufman’s Adaptive Moving Average (KAMA)

Moving average designed to account for market noise or volatility. KAMA will closely follow prices when the
price swings are relatively small and the noise is low. KAMA will adjust when the price swings widen and
follow prices from a greater distance. This trend-following indicator can be used to identify the overall trend,
time turning points and filter price movements.

https://www.tradingview.com/ideas/kama/

Parameters

• close (pandas.Series) – dataset ‘Close’ column.
• n (int) – n number of periods for the efficiency ratio.
• pow1 (int) – number of periods for the fastest EMA constant.
• pow2 (int) – number of periods for the slowest EMA constant.
• fillna (bool) – if True, fill nan values.
**ta.momentum.money_flow_index** *(high, low, close, volume, n=14, fillna=False)*

Money Flow Index (MFI)

Uses both price and volume to measure buying and selling pressure. It is positive when the typical price rises (buying pressure) and negative when the typical price declines (selling pressure). A ratio of positive and negative money flow is then plugged into an RSI formula to create an oscillator that moves between zero and one hundred.


**Parameters**

- **high** *(pandas.Series)* – dataset ‘High’ column.
- **close** *(pandas.Series)* – dataset ‘Close’ column.
- **n** *(int)* – n period.
- **fillna** *(bool)* – if True, fill nan values.

**Returns** New feature generated.

**Return type** pandas.Series

**ta.momentum.roc** *(close, n=12, fillna=False)*

Rate of Change (ROC)

The Rate-of-Change (ROC) indicator, which is also referred to as simply Momentum, is a pure momentum oscillator that measures the percent change in price from one period to the next. The ROC calculation compares the current price with the price “n” periods ago. The plot forms an oscillator that fluctuates above and below the zero line as the Rate-of-Change moves from positive to negative. As a momentum oscillator, ROC signals include centerline crossovers, divergences and overbought-oversold readings. Divergences fail to foreshadow reversals more often than not, so this article will forgo a detailed discussion on them. Even though centerline crossovers are prone to whipsaw, especially short-term, these crossovers can be used to identify the overall trend. Identifying overbought or oversold extremes comes naturally to the Rate-of-Change oscillator.


**Parameters**

- **close** *(pandas.Series)* – dataset ‘Close’ column.
- **n** *(int)* – n periods.
- **fillna** *(bool)* – if True, fill nan values.

**Returns** New feature generated.

**Return type** pandas.Series

**ta.momentum.rsi** *(close, n=14, fillna=False)*

Relative Strength Index (RSI)

Compares the magnitude of recent gains and losses over a specified time period to measure speed and change of price movements of a security. It is primarily used to attempt to identify overbought or oversold conditions in the trading of an asset.

https://www.investopedia.com/terms/r/rsi.asp

**Parameters**

---

4.1. Documentation
- close (`pandas.Series`) – dataset ‘Close’ column.
- n (`int`) – n period.
- fillna (`bool`) – if True, fill nan values.

**Returns** New feature generated.

**Return type** `pandas.Series`

```python
ta.momentum.stoch(high, low, close, n=14, fillna=False)
```

Stochastic Oscillator

Developed in the late 1950s by George Lane. The stochastic oscillator presents the location of the closing price of a stock in relation to the high and low range of the price of a stock over a period of time, typically a 14-day period.

https://www.investopedia.com/terms/s/stochasticoscillator.asp

**Parameters**

- high (`pandas.Series`) – dataset ‘High’ column.
- low (`pandas.Series`) – dataset ‘Low’ column.
- close (`pandas.Series`) – dataset ‘Close’ column.
- n (`int`) – n period.
- fillna (`bool`) – if True, fill nan values.

**Returns** New feature generated.

**Return type** `pandas.Series`

```python
ta.momentum.stoch_signal(high, low, close, n=14, d_n=3, fillna=False)
```

Stochastic Oscillator Signal

Shows SMA of Stochastic Oscillator. Typically a 3 day SMA.

https://www.investopedia.com/terms/s/stochasticoscillator.asp

**Parameters**

- high (`pandas.Series`) – dataset ‘High’ column.
- low (`pandas.Series`) – dataset ‘Low’ column.
- close (`pandas.Series`) – dataset ‘Close’ column.
- n (`int`) – n period.
- d_n (`int`) – sma period over stoch_k
- fillna (`bool`) – if True, fill nan values.

**Returns** New feature generated.

**Return type** `pandas.Series`

```python
ta.momentum.tsi(close, r=25, s=13, fillna=False)
```

True strength index (TSI)

Shows both trend direction and overbought/oversold conditions.

https://en.wikipedia.org/wiki/True_strength_index

**Parameters**

- close (`pandas.Series`) – dataset ‘Close’ column.
• r (int) – high period.
• s (int) – low period.
• fillna (bool) – if True, fill nan values.

Returns New feature generated.
Return type pandas.Series

```
ta.momentum.uo(high, low, close, s=7, m=14, len=28, ws=4.0, wm=2.0, wl=1.0, fillna=False)
```
Ultimate Oscillator

Larry Williams’ (1976) signal, a momentum oscillator designed to capture momentum across three different timeframes.


BP = Close - Minimum(Low or Prior Close). TR = Maximum(High or Prior Close) - Minimum(Low or Prior Close) Average7 = (7-period BP Sum) / (7-period TR Sum) Average14 = (14-period BP Sum) / (14-period TR Sum) Average28 = (28-period BP Sum) / (28-period TR Sum)

UO = 100 x [(4 x Average7)+(2 x Average14)+Average28]/(4+2+1)

Parameters
• high (pandas.Series) – dataset ‘High’ column.
• low (pandas.Series) – dataset ‘Low’ column.
• close (pandas.Series) – dataset ‘Close’ column.
• s (int) – short period.
• m (int) – medium period.
• len (int) – long period.
• ws (float) – weight of short BP average for UO.
• wm (float) – weight of medium BP average for UO.
• wl (float) – weight of long BP average for UO.
• fillna (bool) – if True, fill nan values with 50.

Returns New feature generated.
Return type pandas.Series

```
ta.momentum.wr(high, low, close, lbp=14, fillna=False)
```
Williams %R


Developed by Larry Williams, Williams %R is a momentum indicator that is the inverse of the Fast Stochastic Oscillator. Also referred to as %R, Williams %R reflects the level of the close relative to the highest high for the look-back period. In contrast, the Stochastic Oscillator reflects the level of the close relative to the lowest low. %R corrects for the inversion by multiplying the raw value by -100. As a result, the Fast Stochastic Oscillator and Williams %R produce the exact same lines, only the scaling is different. Williams %R oscillates from 0 to -100.

Readings from 0 to -20 are considered overbought. Readings from -80 to -100 are considered oversold.

Unsurprisingly, signals derived from the Stochastic Oscillator are also applicable to Williams %R.

%R = (Highest High - Close)/(Highest High - Lowest Low) * -100
Lowest Low = lowest low for the look-back period
Highest High = highest high for the look-back period
\( \%R \) is multiplied by -100, correct the inversion and move the decimal.

From: https://www.investopedia.com/terms/w/williamsr.asp

The Williams \( \%R \) oscillates from 0 to -100. When the indicator produces readings from 0 to -20, this indicates overbought market conditions. When readings are -80 to -100, it indicates oversold market conditions.

**Parameters**

- `high` (pandas.Series) – dataset ‘High’ column.
- `low` (pandas.Series) – dataset ‘Low’ column.
- `close` (pandas.Series) – dataset ‘Close’ column.
- `lbp` (int) – lookback period.
- `fillna` (bool) – if True, fill nan values with -50.

**Returns** New feature generated.

**Return type** pandas.Series

### 4.1.2 Volume Indicators

Volume Indicators.


Accumulation/Distribution Index (ADI)

Acting as leading indicator of price movements.

https://school.stockcharts.com/doku.php?id=technical_indicators:accumulation_distribution_line

**Parameters**

- `high` (pandas.Series) – dataset ‘High’ column.
- `low` (pandas.Series) – dataset ‘Low’ column.
- `close` (pandas.Series) – dataset ‘Close’ column.
- `fillna` (bool) – if True, fill nan values.

**acc_dist_index()** → pandas.core.series.Series

Accumulation/Distribution Index (ADI)

**Returns** New feature generated.

**Return type** pandas.Series


Chaikin Money Flow (CMF)

It measures the amount of Money Flow Volume over a specific period.
Parameters

- **high** (*pandas.Series*) – dataset ‘High’ column.
- **close** (*pandas.Series*) – dataset ‘Close’ column.
- **n** (*int*) – n period.
- **fillna** (*bool*) – if True, fill nan values.

```
chaikin_money_flow() → pandas.core.series.Series
Chaikin Money Flow (CMF)
```

**Returns**
New feature generated.

**Return type**
pandas.Series

---

**Ease of movement (EoM, EMV)**

It relate an asset’s price change to its volume and is particularly useful for assessing the strength of a trend.


Parameters

- **high** (*pandas.Series*) – dataset ‘High’ column.
- **n** (*int*) – n period.
- **fillna** (*bool*) – if True, fill nan values.

```
ease_of_movement() → pandas.core.series.Series
Ease of movement (EoM, EMV)
```

**Returns**
New feature generated.

**Return type**
pandas.Series

---

**Signal Ease of movement (EoM, EMV)**

```
sma_ease_of_movement() → pandas.core.series.Series
Signal Ease of movement (EoM, EMV)
```

**Returns**
New feature generated.

**Return type**
pandas.Series

---

**Force Index (FI)**

It illustrates how strong the actual buying or selling pressure is. High positive values mean there is a strong rising trend, and low values signify a strong downward trend.


---

4.1. Documentation
Parameters

- **high** (*pandas.Series*) – dataset ‘High’ column.
- **close** (*pandas.Series*) – dataset ‘Close’ column.
- **n** (*int*) – n period.
- **fillna** (*bool*) – if True, fill nan values.

**force_index** () → pandas.core.series.Series

Force Index (FI)

**Returns**

New feature generated.

**Return type**

*pandas.Series*

class ta.volume.NegativeVolumeIndexIndicator

```
```

Negative Volume Index (NVI)


Parameters

- **close** (*pandas.Series*) – dataset ‘Close’ column.
- **fillna** (*bool*) – if True, fill nan values with 1000.

**negative_volume_index** () → pandas.core.series.Series

Negative Volume Index (NVI)

**Returns**

New feature generated.

**Return type**

*pandas.Series*

class ta.volume.OnBalanceVolumeIndicator

```
```

On-balance volume (OBV)

It relates price and volume in the stock market. OBV is based on a cumulative total volume.


Parameters

- **close** (*pandas.Series*) – dataset ‘Close’ column.
- **fillna** (*bool*) – if True, fill nan values.

**on_balance_volume** () → pandas.core.series.Series

On-balance volume (OBV)

**Returns**

New feature generated.

**Return type**

*pandas.Series*

Volume-price trend (VPT)

Is based on a running cumulative volume that adds or subtracts a multiple of the percentage change in share price trend and current volume, depending upon the investment’s upward or downward movements.

https://en.wikipedia.org/wiki/Volume%E2%80%93price_trend

**Parameters**
- `close (pandas.Series)` – dataset ‘Close’ column.
- `fillna (bool)` – if True, fill nan values.

**volume_price_trend** () → pandas.core.series.Series

Volume-price trend (VPT)

**Returns** New feature generated.

**Return type** pandas.Series

**ta.volume.acc_dist_index**(*high, low, close, volume, fillna=False*)

Accumulation/Distribution Index (ADI)

Acting as leading indicator of price movements.

https://en.wikipedia.org/wiki/Accumulation/distribution_index

**Parameters**
- `high (pandas.Series)` – dataset ‘High’ column.
- `low (pandas.Series)` – dataset ‘Low’ column.
- `close (pandas.Series)` – dataset ‘Close’ column.
- `fillna (bool)` – if True, fill nan values.

**Returns** New feature generated.

**Return type** pandas.Series

**ta.volume.chaikin_money_flow**(*high, low, close, volume, n=20, fillna=False*)

Chaikin Money Flow (CMF)

It measures the amount of Money Flow Volume over a specific period.


**Parameters**
- `high (pandas.Series)` – dataset ‘High’ column.
- `low (pandas.Series)` – dataset ‘Low’ column.
- `close (pandas.Series)` – dataset ‘Close’ column.
- `n (int)` – n period.
- `fillna (bool)` – if True, fill nan values.

**Returns** New feature generated.
ta.volume.ease_of_movement(high, low, volume, n=14, fillna=False)

Ease of movement (EoM, EMV)

It relate an asset’s price change to its volume and is particularly useful for assessing the strength of a trend.

Parameters

- high (pandas.Series) – dataset ‘High’ column.
- low (pandas.Series) – dataset ‘Low’ column.
- n (int) – n period.
- fillna (bool) – if True, fill nan values.

Returns New feature generated.

Return type pandas.Series

Parameters

- close (pandas.Series) – dataset ‘Close’ column.
- n (int) – n period.
- fillna (bool) – if True, fill nan values.

Returns New feature generated.

Return type pandas.Series

Parameters

- close (pandas.Series) – dataset ‘Close’ column.
- fillna (bool) – if True, fill nan values.

Returns New feature generated.

Return type pandas.Series

If today’s volume is less than yesterday’s volume then:  nvi(t) = nvi(t-1) * ( 1 + (close(t) - close(t-1)) / close(t-1) )

Else  nvi(t) = nvi(t-1)
Please note: the “stockcharts.com” example calculation just adds the percentage change of price to previous NVI when volumes decline; other sources indicate that the same percentage of the previous NVI value should be added, which is what is implemented here.

Parameters

- close (pandas.Series) – dataset ‘Close’ column.
- fillna (bool) – if True, fill nan values with 1000.

Returns New feature generated.

Return type pandas.Series

See also:
https://en.wikipedia.org/wiki/Negative_volume_index

```
import ta

ta.volume.on_balance_volume(close, volume, fillna=False)
```

On-balance volume (OBV)

It relates price and volume in the stock market. OBV is based on a cumulative total volume.


Parameters

- close (pandas.Series) – dataset ‘Close’ column.
- fillna (bool) – if True, fill nan values.

Returns New feature generated.

Return type pandas.Series

```
import ta

ta.volume.put_call_ratio()
```

Put/Call ratio (PCR) https://en.wikipedia.org/wiki/Put_call_ratio

```
import ta

ta.volume.sma_ease_of_movement(high, low, volume, n=14, fillna=False)
```

Ease of movement (EoM, EMV)

It relate an asset’s price change to its volume and is particularly useful for assessing the strength of a trend.


Parameters

- high (pandas.Series) – dataset ‘High’ column.
- low (pandas.Series) – dataset ‘Low’ column.
- n (int) – n period.
- fillna (bool) – if True, fill nan values.

Returns New feature generated.

Return type pandas.Series

```
import ta

ta.volume.volume_price_trend(close, volume, fillna=False)
```

Volume-price trend (VPT)
Is based on a running cumulative volume that adds or substracts a multiple of the Percentage change in share price trend and current volume, depending upon the investment’s upward or downward movements.

https://en.wikipedia.org/wiki/Volume%E2%80%93price_trend

Parameters
- **close** *(pandas.Series)* – dataset ‘Close’ column.
- **fillna** *(bool)* – if True, fill nan values.

Returns New feature generated.

Return type pandas.Series

### 4.1.3 Volatility Indicators

Volatility Indicators.


Average True Range (ATR)

The indicator provide an indication of the degree of price volatility. Strong moves, in either direction, are often accompanied by large ranges, or large True Ranges.


Parameters
- **high** *(pandas.Series)* – dataset ‘High’ column.
- **close** *(pandas.Series)* – dataset ‘Close’ column.
- **n** *(int)* – n period.
- **fillna** *(bool)* – if True, fill nan values.

**average_true_range** () → pandas.core.series.Series

Average True Range (ATR)

Returns New feature generated.

Return type pandas.Series


Bollinger Bands


Parameters
- **close** *(pandas.Series)* – dataset ‘Close’ column.
- **n** *(int)* – n period.
- **ndev** *(int)* – n factor standard deviation
- **fillna** *(bool)* – if True, fill nan values.
bollinger_hband() → pandas.core.series.Series
Bollinger Channel High Band
Returns New feature generated.
Return type pandas.Series

bollinger_hband_indicator() → pandas.core.series.Series
Bollinger Channel Indicator Crossing High Band
Returns New feature generated.
Return type pandas.Series

bollinger_lband() → pandas.core.series.Series
Bollinger Channel Low Band
Returns New feature generated.
Return type pandas.Series

bollinger_lband_indicator() → pandas.core.series.Series
Bollinger Channel Indicator Crossing Low Band
Returns New feature generated.
Return type pandas.Series

bollinger_mavg() → pandas.core.series.Series
Bollinger Channel Middle Band
Returns New feature generated.
Return type pandas.Series

bollinger_wband() → pandas.core.series.Series
Bollinger Channel Width Band
Returns New feature generated.
Return type pandas.Series

class ta.volatility.DonchianChannel(close: pandas.core.series.Series, n: int = 20, fillna: bool = False)
Donchian Channel
https://www.investopedia.com/terms/d/donchianchannels.asp

Parameters
- close (pandas.Series) – dataset ‘Close’ column.
- n (int) – n period.
- ndev (int) – n factor standard deviation
- fillna (bool) – if True, fill nan values.

donchian_channel_hband() → pandas.core.series.Series
Donchian Channel High Band
Returns New feature generated.
Return type pandas.Series

donchian_channel_hband_indicator() → pandas.core.series.Series
Donchian Channel Indicator Crossing High Band
Returns New feature generated.
Return type  pandas.Series

donchian_channel_lband()  \rightarrow  pandas.core.series.Series
Donchian Channel Low Band

Returns  New feature generated.

Return type  pandas.Series

donchian_channel_lband_indicator()  \rightarrow  pandas.core.series.Series
Donchian Channel Indicator Crossing Low Band

Returns  New feature generated.

Return type  pandas.Series

Keltner Channels are a trend following indicator used to identify reversals with channel breakouts and channel direction. Channels can also be used to identify overbought and oversold levels when the trend is flat.


Parameters
• high (pandas.Series) – dataset ‘High’ column.
• low (pandas.Series) – dataset ‘Low’ column.
• close (pandas.Series) – dataset ‘Close’ column.
• n (int) – n period.
• fillna (bool) – if True, fill nan values.

keltner_channel_central()  \rightarrow  pandas.core.series.Series
Keltner Channel Middle Band

Returns  New feature generated.

Return type  pandas.Series

keltner_channel_hband()  \rightarrow  pandas.core.series.Series
Keltner Channel High Band

Returns  New feature generated.

Return type  pandas.Series

keltner_channel_hband_indicator()  \rightarrow  pandas.core.series.Series
Keltner Channel Indicator Crossing High Band

Returns  New feature generated.

Return type  pandas.Series

keltner_channel_lband()  \rightarrow  pandas.core.series.Series
Keltner Channel Low Band

Returns  New feature generated.

Return type  pandas.Series

keltner_channel_lband_indicator()  \rightarrow  pandas.core.series.Series
Keltner Channel Indicator Crossing Low Band

Returns  New feature generated.
Return type  pandas.Series
ta.volatility.average_true_range(high, low, close, n=14, fillna=False)
Average True Range (ATR)

The indicator provides an indication of the degree of price volatility. Strong moves, in either direction, are often accompanied by large ranges, or large True Ranges.


Parameters
• high (pandas.Series) – dataset ‘High’ column.
• low (pandas.Series) – dataset ‘Low’ column.
• close (pandas.Series) – dataset ‘Close’ column.
• n (int) – n period.
•fillna (bool) – if True, fill nan values.

Returns New feature generated.
Return type  pandas.Series

Bollinger Hband (BB)

Upper band at K times an N-period standard deviation above the moving average (MA + Kdeviation).
https://en.wikipedia.org/wiki/Bollinger_Bands

Parameters
• close (pandas.Series) – dataset ‘Close’ column.
• n (int) – n period.
• ndev (int) – n factor standard deviation
•fillna (bool) – if True, fill nan values.

Returns New feature generated.
Return type  pandas.Series

Bollinger Hband Indicator

Returns 1, if close is higher than bollinger high band. Else, return 0.
https://en.wikipedia.org/wiki/Bollinger_Bands

Parameters
• close (pandas.Series) – dataset ‘Close’ column.
• n (int) – n period.
• ndev (int) – n factor standard deviation
•fillna (bool) – if True, fill nan values.

Returns New feature generated.
Return type  pandas.Series
ta.volatility.bollinger_lband(close, n=20, ndev=2, fillna=False)

Bollinger Bands (BB)

Lower band at K times an N-period standard deviation below the moving average (MA Kdeviation).

https://en.wikipedia.org/wiki/Bollinger_Bands

Parameters

- close (pandas.Series) – dataset ‘Close’ column.
- n (int) – n period.
- ndev (int) – n factor standard deviation
- fillna (bool) – if True, fill nan values.

Returns New feature generated.

Return type pandas.Series

ta.volatility.bollinger_lband_indicator(close, n=20, ndev=2, fillna=False)

Bollinger Low Band Indicator

Returns 1, if close is lower than bollinger low band. Else, return 0.

https://en.wikipedia.org/wiki/Bollinger_Bands

Parameters

- close (pandas.Series) – dataset ‘Close’ column.
- n (int) – n period.
- ndev (int) – n factor standard deviation
- fillna (bool) – if True, fill nan values.

Returns New feature generated.

Return type pandas.Series

ta.volatility.bollinger_mavg(close, n=20, fillna=False)

Bollinger Bands (BB)

N-period simple moving average (MA).

https://en.wikipedia.org/wiki/Bollinger_Bands

Parameters

- close (pandas.Series) – dataset ‘Close’ column.
- n (int) – n period.
- fillna (bool) – if True, fill nan values.

Returns New feature generated.

Return type pandas.Series

ta.volatility.donchian_channel_hband(close, n=20, fillna=False)

Donchian channel (DC)

The upper band marks the highest price of an issue for n periods.

https://www.investopedia.com/terms/d/donchianchannels.asp

Parameters
• close (pandas.Series) – dataset ‘Close’ column.
• n (int) – n period.
•fillna (bool) – if True, fill nan values.

Returns New feature generated.
Return type pandas.Series

ta.volatility.donchian_channel_hband_indicator(close, n=20, fillna=False)
Donchian High Band Indicator

Returns 1, if close is higher than donchian high band channel. Else, return 0.
https://www.investopedia.com/terms/d/donchianchannels.asp

Parameters
• close (pandas.Series) – dataset ‘Close’ column.
• n (int) – n period.
•fillna (bool) – if True, fill nan values.

Returns New feature generated.
Return type pandas.Series

ta.volatility.donchian_channel_lband(close, n=20, fillna=False)
Donchian channel (DC)
The lower band marks the lowest price for n periods.
https://www.investopedia.com/terms/d/donchianchannels.asp

Parameters
• close (pandas.Series) – dataset ‘Close’ column.
• n (int) – n period.
•fillna (bool) – if True, fill nan values.

Returns New feature generated.
Return type pandas.Series

ta.volatility.donchian_channel_lband_indicator(close, n=20, fillna=False)
Donchian Low Band Indicator

Returns 1, if close is lower than donchian low band channel. Else, return 0.
https://www.investopedia.com/terms/d/donchianchannels.asp

Parameters
• close (pandas.Series) – dataset ‘Close’ column.
• n (int) – n period.
•fillna (bool) – if True, fill nan values.

Returns New feature generated.
Return type pandas.Series
**ta.volatility.keltner_channel_central** *(high, low, close, n=10, fillna=False)*

Keltner channel (KC)

Showing a simple moving average line (central) of typical price.

https://en.wikipedia.org/wiki/Keltner_channel

**Parameters**

- **high** *(pandas.Series)* – dataset ‘High’ column.
- **close** *(pandas.Series)* – dataset ‘Close’ column.
- **n** *(int)* – n period.
- **fillna** *(bool)* – if True, fill nan values.

**Returns** New feature generated.

**Return type** pandas.Series

**ta.volatility.keltner_channel_hband** *(high, low, close, n=10, fillna=False)*

Keltner channel (KC)

Showing a simple moving average line (high) of typical price.

https://en.wikipedia.org/wiki/Keltner_channel

**Parameters**

- **high** *(pandas.Series)* – dataset ‘High’ column.
- **close** *(pandas.Series)* – dataset ‘Close’ column.
- **n** *(int)* – n period.
- **fillna** *(bool)* – if True, fill nan values.

**Returns** New feature generated.

**Return type** pandas.Series

**ta.volatility.keltner_channel_hband_indicator** *(high, low, close, n=10, fillna=False)*

Keltner Channel High Band Indicator (KC)

Returns 1, if close is higher than keltner high band channel. Else, return 0.

https://en.wikipedia.org/wiki/Keltner_channel

**Parameters**

- **high** *(pandas.Series)* – dataset ‘High’ column.
- **close** *(pandas.Series)* – dataset ‘Close’ column.
- **n** *(int)* – n period.
- **fillna** *(bool)* – if True, fill nan values.

**Returns** New feature generated.

**Return type** pandas.Series
ta.volatility.keltner_channel_lband(high, low, close, n=10, fillna=False)

Keltner channel (KC)

Showing a simple moving average line (low) of typical price.
https://en.wikipedia.org/wiki/Keltner_channel

Parameters
- high (pandas.Series) – dataset ‘High’ column.
- low (pandas.Series) – dataset ‘Low’ column.
- close (pandas.Series) – dataset ‘Close’ column.
- n (int) – n period.
- fillna (bool) – if True, fill nan values.

Returns New feature generated.
Return type pandas.Series

ta.volatility.keltner_channel_lband_indicator(high, low, close, n=10, fillna=False)

Keltner Channel Low Band Indicator (KC)

Returns 1, if close is lower than keltner low band channel. Else, return 0.
https://en.wikipedia.org/wiki/Keltner_channel

Parameters
- high (pandas.Series) – dataset ‘High’ column.
- low (pandas.Series) – dataset ‘Low’ column.
- close (pandas.Series) – dataset ‘Close’ column.
- n (int) – n period.
- fillna (bool) – if True, fill nan values.

Returns New feature generated.
Return type pandas.Series

4.1.4 Trend Indicators

Trend Indicators.


Average Directional Movement Index (ADX)

The Plus Directional Indicator (+DI) and Minus Directional Indicator (-DI) are derived from smoothed averages of these differences, and measure trend direction over time. These two indicators are often referred to collectively as the Directional Movement Indicator (DMI).

The Average Directional Index (ADX) is in turn derived from the smoothed averages of the difference between +DI and -DI, and measures the strength of the trend (regardless of direction) over time.

Using these three indicators together, chartists can determine both the direction and strength of the trend.

Parameters
- **high** (`pandas.Series`) – dataset ‘High’ column.
- **low** (`pandas.Series`) – dataset ‘Low’ column.
- **close** (`pandas.Series`) – dataset ‘Close’ column.
- **n** (`int`) – n period.
- **fillna** (`bool`) – if True, fill nan values.

### adx() → pandas.core.series.Series
Average Directional Index (ADX)

Returns New feature generated.

Return type pandas.Series

### adx_neg() → pandas.core.series.Series
Minus Directional Indicator (-DI)

Returns New feature generated.

Return type pandas.Series

### adx_pos() → pandas.core.series.Series
Plus Directional Indicator (+DI)

Returns New feature generated.

Return type pandas.Series

### class ta.trend.AroonIndicator (close: pandas.core.series.Series, n: int = 25, fillna: bool = False)
Aroon Indicator

Identify when trends are likely to change direction.

Aroon Up = ((N - Days Since N-day High) / N) x 100
Aroon Down = ((N - Days Since N-day Low) / N) x 100
Aroon Indicator = Aroon Up - Aroon Down

https://www.investopedia.com/terms/a/aroon.asp

Parameters

- **close** (`pandas.Series`) – dataset ‘Close’ column.
- **n** (`int`) – n period.
- **fillna** (`bool`) – if True, fill nan values.

### aroon_down() → pandas.core.series.Series
Aroon Down Channel

Returns New feature generated.

Return type pandas.Series

### aroon_indicator() → pandas.core.series.Series
Aroon Indicator

Returns New feature generated.

Return type pandas.Series

### aroon_up() → pandas.core.series.Series
Aroon Up Channel

Returns New feature generated.

Return type pandas.Series

Commodity Channel Index (CCI)

CCI measures the difference between a security’s price change and its average price change. High positive readings indicate that prices are well above their average, which is a show of strength. Low negative readings indicate that prices are well below their average, which is a show of weakness.


Parameters

• high (pandas.Series) – dataset ‘High’ column.
• low (pandas.Series) – dataset ‘Low’ column.
• close (pandas.Series) – dataset ‘Close’ column.
• n (int) – n period.
• c (int) – constant.
• fillna (bool) – if True, fill nan values.

cci() → pandas.core.series.Series

Commodity Channel Index (CCI)

Returns New feature generated.

Return type pandas.Series

class ta.trend.DPOIndicator(close: pandas.core.series.Series, n: int = 20,fillna: bool = False)

Detrended Price Oscillator (DPO)

Is an indicator designed to remove trend from price and make it easier to identify cycles.


Parameters

• close (pandas.Series) – dataset ‘Close’ column.
• n (int) – n period.
• fillna (bool) – if True, fill nan values.

dpo() → pandas.core.series.Series

Detrended Price Oscillator (DPO)

Returns New feature generated.

Return type pandas.Series

class ta.trend.EMAIndicator(close: pandas.core.series.Series, n: int = 14,fillna: bool = False)

EMA - Exponential Moving Average

Parameters

• close (pandas.Series) – dataset ‘Close’ column.
• n (int) – n period.
• fillna (bool) – if True, fill nan values.

eaMA_indicator() → pandas.core.series.Series

Exponential Moving Average (EMA)

Returns New feature generated.
Return type pandas.Series


Ichimoku Kinkō Hyō (Ichimoku)

Parameters

• high (pandas.Series) – dataset ‘High’ column.
• low (pandas.Series) – dataset ‘Low’ column.
• n1 (int) – n1 low period.
• n2 (int) – n2 medium period.
• n3 (int) – n3 high period.
• visual (bool) – if True, shift n2 values.
• fillna (bool) – if True, fill nan values.

ichimoku_a() → pandas.core.series.Series
Senkou Span A (Leading Span A)

Returns New feature generated.

Return type pandas.Series

ichimoku_b() → pandas.core.series.Series
Senkou Span B (Leading Span B)

Returns New feature generated.

Return type pandas.Series


KST Oscillator (KST Signal)

It is useful to identify major stock market cycle junctures because its formula is weighed to be more greatly influenced by the longer and more dominant time spans, in order to better reflect the primary swings of stock market cycle.


Parameters

• close (pandas.Series) – dataset ‘Close’ column.
• r1 (int) – r1 period.
• r2 (int) – r2 period.
• r3 (int) – r3 period.
• r4 (int) – r4 period.
• n1 (int) – n1 smoothed period.
• n2 (int) – n2 smoothed period.
• n3 (int) – n3 smoothed period.
• n4 (int) – n4 smoothed period.
• `nsig (int)` – n period to signal.
• `fillna (bool)` – if True, fill nan values.

`kst ()` → `pandas.core.series.Series`
Know Sure Thing (KST)

Returns New feature generated.
Return type `pandas.Series`

`kst_diff ()` → `pandas.core.series.Series`
Diff Know Sure Thing (KST)

KST - Signal_KST

Returns New feature generated.
Return type `pandas.Series`

`kst_sig ()` → `pandas.core.series.Series`
Signal Line Know Sure Thing (KST)

nsig-period SMA of KST

Returns New feature generated.
Return type `pandas.Series`

`class ta.trend.MACD (close: pandas.core.series.Series, n_slow: int = 26, n_fast: int = 12, n_sign: int = 9, fillna: bool = False)`
Moving Average Convergence Divergence (MACD)

Is a trend-following momentum indicator that shows the relationship between two moving averages of prices.


Parameters

• `close (pandas.Series)` – dataset ‘Close’ column.
• `n_fast (int)` – n period short-term.
• `n_slow (int)` – n period long-term.
• `n_sign (int)` – n period to signal.
• `fillna (bool)` – if True, fill nan values.

`macd ()` → `pandas.core.series.Series`
MACD Line

Returns New feature generated.
Return type `pandas.Series`

`macd_diff ()` → `pandas.core.series.Series`
MACD Histogram

Returns New feature generated.
Return type `pandas.Series`

`macd_signal ()` → `pandas.core.series.Series`
Signal Line

Returns New feature generated.
Return type pandas.Series


Mass Index (MI)

It uses the high-low range to identify trend reversals based on range expansions. It identifies range bulges that can foreshadow a reversal of the current trend.


Parameters

• high (pandas.Series) – dataset ‘High’ column.
• low (pandas.Series) – dataset ‘Low’ column.
• n (int) – n low period.
• n2 (int) – n high period.
• fillna (bool) – if True, fill nan values.

mass_index() → pandas.core.series.Series

Mass Index (MI)

Returns New feature generated.

Return type pandas.Series

class ta.trend.PSARIndicator(high: pandas.core.series.Series, low: pandas.core.series.Series, close: pandas.core.series.Series, step: float = 0.02, max_step: float = 0.2, fillna: bool = False)

Parabolic Stop and Reverse (Parabolic SAR)

The Parabolic Stop and Reverse, more commonly known as the Parabolic SAR, is a trend-following indicator developed by J. Welles Wilder. The Parabolic SAR is displayed as a single parabolic line (or dots) underneath the price bars in an uptrend, and above the price bars in a downtrend.


Parameters

• high (pandas.Series) – dataset ‘High’ column.
• low (pandas.Series) – dataset ‘Low’ column.
• close (pandas.Series) – dataset ‘Close’ column.
• step (float) – the Acceleration Factor used to compute the SAR.
• max_step (float) – the maximum value allowed for the Acceleration Factor.

psar() → pandas.core.series.Series

PSAR value

Returns New feature generated.

Return type pandas.Series

psar_down() → pandas.core.series.Series

PSAR down trend value

Returns New feature generated.

Return type pandas.Series
psar_down_indicator() → pandas.core.series.Series
PSAR down trend value

Returns  New feature generated.
Return type  pandas.Series

psar_up() → pandas.core.series.Series
PSAR up trend value

Returns  New feature generated.
Return type  pandas.Series

psar_up_indicator() → pandas.core.series.Series
PSAR up trend value

Returns  New feature generated.
Return type  pandas.Series

class ta.trend.TRIXIndicator(close: pandas.core.series.Series, n: int = 15, fillna: bool = False)
Trix (TRIX)

Shows the percent rate of change of a triple exponentially smoothed moving average.

Parameters

• close (pandas.Series) – dataset ‘Close’ column.
• n (int) – n period.
• fillna (bool) – if True, fill nan values.

trix() → pandas.core.series.Series
Trix (TRIX)

Returns  New feature generated.
Return type  pandas.Series

Vortex Indicator (VI)

It consists of two oscillators that capture positive and negative trend movement. A bullish signal triggers when the positive trend indicator crosses above the negative trend indicator or a key level.

Parameters

• high (pandas.Series) – dataset ‘High’ column.
• low (pandas.Series) – dataset ‘Low’ column.
• close (pandas.Series) – dataset ‘Close’ column.
• n (int) – n period.
• fillna (bool) – if True, fill nan values.

vortex_indicator_diff()
Diff VI

Returns  New feature generated.
vortex_indicator_neg()

- VI

Returns New feature generated.

Return type pandas.Series

vortex_indicator_pos()

+ VI

Returns New feature generated.

Return type pandas.Series

ta.trend.adx(high, low, close, n=14, fillna=False)

Average Directional Movement Index (ADX)

The Plus Directional Indicator (+DI) and Minus Directional Indicator (-DI) are derived from smoothed averages of these differences, and measure trend direction over time. These two indicators are often referred to collectively as the Directional Movement Indicator (DMI).

The Average Directional Index (ADX) is in turn derived from the smoothed averages of the difference between +DI and -DI, and measures the strength of the trend (regardless of direction) over time.

Using these three indicators together, chartists can determine both the direction and strength of the trend.

Parameters

• high (pandas.Series) – dataset ‘High’ column.
• low (pandas.Series) – dataset ‘Low’ column.
• close (pandas.Series) – dataset ‘Close’ column.
• n (int) – n period.
• fillna (bool) – if True, fill nan values.

Returns New feature generated.

Return type pandas.Series

ta.trend.adx_neg(high, low, close, n=14, fillna=False)

Average Directional Movement Index Negative (ADX)

The Plus Directional Indicator (+DI) and Minus Directional Indicator (-DI) are derived from smoothed averages of these differences, and measure trend direction over time. These two indicators are often referred to collectively as the Directional Movement Indicator (DMI).

The Average Directional Index (ADX) is in turn derived from the smoothed averages of the difference between +DI and -DI, and measures the strength of the trend (regardless of direction) over time.

Using these three indicators together, chartists can determine both the direction and strength of the trend.

Parameters

• high (pandas.Series) – dataset ‘High’ column.
• low (pandas.Series) – dataset ‘Low’ column.
• close (pandas.Series) – dataset ‘Close’ column.
### ta.trend.adx_pos(high, low, close, n=14, fillna=False)

**Average Directional Movement Index Positive (ADX)**

The Plus Directional Indicator (+DI) and Minus Directional Indicator (-DI) are derived from smoothed averages of these differences, and measure trend direction over time. These two indicators are often referred to collectively as the Directional Movement Indicator (DMI).

The Average Directional Index (ADX) is in turn derived from the smoothed averages of the difference between +DI and -DI, and measures the strength of the trend (regardless of direction) over time.

Using these three indicators together, chartists can determine both the direction and strength of the trend.

**Parameters**

- **high** (pandas.Series) – dataset ‘High’ column.
- **low** (pandas.Series) – dataset ‘Low’ column.
- **close** (pandas.Series) – dataset ‘Close’ column.
- **n** (int) – n period.
- **fillna** (bool) – if True, fill nan values.

**Returns** New feature generated.

**Return type** pandas.Series

### ta.trend.aroon_down(close, n=25, fillna=False)

**Aroon Indicator (AI)**

Identify when trends are likely to change direction (downtrend).

Aroon Down - ((N - Days Since N-day Low) / N) x 100

https://www.investopedia.com/terms/a/aroon.asp

**Parameters**

- **close** (pandas.Series) – dataset ‘Close’ column.
- **n** (int) – n period.
- **fillna** (bool) – if True, fill nan values.

**Returns** New feature generated.

**Return type** pandas.Series

### ta.trend.aroon_up(close, n=25, fillna=False)

**Aroon Indicator (AI)**

Identify when trends are likely to change direction (uptrend).

Aroon Up - ((N - Days Since N-day High) / N) x 100

https://www.investopedia.com/terms/a/aroon.asp

**Parameters**
• `close (pandas.Series)` – dataset ‘Close’ column.
• `n (int)` – n period.
• `fillna (bool)` – if True, fill nan values.

**Returns** New feature generated.

**Return type** pandas.Series

``ta.trend.cci (high, low, close, n=20, c=0.015, fillna=False)``

Commodity Channel Index (CCI)

CCI measures the difference between a security’s price change and its average price change. High positive readings indicate that prices are well above their average, which is a show of strength. Low negative readings indicate that prices are well below their average, which is a show of weakness.


**Parameters**

• `high (pandas.Series)` – dataset ‘High’ column.
• `low (pandas.Series)` – dataset ‘Low’ column.
• `close (pandas.Series)` – dataset ‘Close’ column.
• `n (int)` – n periods.
• `c (int)` – constant.
• `fillna (bool)` – if True, fill nan values.

**Returns** New feature generated.

**Return type** pandas.Series

``ta.trend.dpo (close, n=20, fillna=False)``

Detrended Price Oscillator (DPO)

Is an indicator designed to remove trend from price and make it easier to identify cycles.


**Parameters**

• `close (pandas.Series)` – dataset ‘Close’ column.
• `n (int)` – n period.
• `fillna (bool)` – if True, fill nan values.

**Returns** New feature generated.

**Return type** pandas.Series

``ta.trend.emacalculator (close, n=12, fillna=False)``

Exponential Moving Average (EMA)

**Returns** New feature generated.

**Return type** pandas.Series

``ta.trend.ichimoku_a (high, low, n1=9, n2=26, visual=False, fillna=False)``

Ichimoku Kinkō Hyō (Ichimoku)

It identifies the trend and look for potential signals within that trend.

Parameters

- **high** *(pandas.Series)* – dataset ‘High’ column.
- **n1** *(int)* – n1 low period.
- **n2** *(int)* – n2 medium period.
- **visual** *(bool)* – if True, shift n2 values.
- **fillna** *(bool)* – if True, fill nan values.

**Returns** New feature generated.

**Return type** pandas.Series

```python
ta.trend.ichimoku_b(high, low, n2=26, n3=52, visual=False, fillna=False)
```

Ichimoku Kinkō Hyō (Ichimoku)

It identifies the trend and look for potential signals within that trend.


Parameters

- **high** *(pandas.Series)* – dataset ‘High’ column.
- **n2** *(int)* – n2 medium period.
- **n3** *(int)* – n3 high period.
- **visual** *(bool)* – if True, shift n2 values.
- **fillna** *(bool)* – if True, fill nan values.

**Returns** New feature generated.

**Return type** pandas.Series

```python
ta.trend.kst(close, r1=10, r2=15, r3=20, r4=30, n1=10, n2=10, n3=10, n4=15, fillna=False)
```

KST Oscillator (KST)

It is useful to identify major stock market cycle junctures because its formula is weighed to be more greatly influenced by the longer and more dominant time spans, in order to better reflect the primary swings of stock market cycle.

https://en.wikipedia.org/wiki/KST_oscillator

Parameters

- **close** *(pandas.Series)* – dataset ‘Close’ column.
- **r1** *(int)* – r1 period.
- **r2** *(int)* – r2 period.
- **r3** *(int)* – r3 period.
- **r4** *(int)* – r4 period.
- **n1** *(int)* – n1 smoothed period.
- **n2** *(int)* – n2 smoothed period.
- **n3** *(int)* – n3 smoothed period.
- **n4** (int) – n4 smoothed period.
- **fillna** (bool) – if True, fill nan values.

**Returns** New feature generated.

**Return type** pandas.Series

```python
ta.trend.kst_sig(close, r1=10, r2=15, r3=20, r4=30, n1=10, n2=10, n3=10, n4=15, nsig=9, fillna=False)
```

KST Oscillator (KST Signal)

It is useful to identify major stock market cycle junctures because its formula is weighed to be more greatly influenced by the longer and more dominant time spans, in order to better reflect the primary swings of stock market cycle.


**Parameters**

- **close** (pandas.Series) – dataset ‘Close’ column.
- **r1** (int) – r1 period.
- **r2** (int) – r2 period.
- **r3** (int) – r3 period.
- **r4** (int) – r4 period.
- **n1** (int) – n1 smoothed period.
- **n2** (int) – n2 smoothed period.
- **n3** (int) – n3 smoothed period.
- **n4** (int) – n4 smoothed period.
- **nsig** (int) – n period to signal.
- **fillna** (bool) – if True, fill nan values.

**Returns** New feature generated.

**Return type** pandas.Series

```python
ta.trend.macd(close, n_slow=26, n_fast=12, fillna=False)
```

Moving Average Convergence Divergence (MACD)

Is a trend-following momentum indicator that shows the relationship between two moving averages of prices.

https://en.wikipedia.org/wiki/MACD

**Parameters**

- **close** (pandas.Series) – dataset ‘Close’ column.
- **n_fast** (int) – n period short-term.
- **n_slow** (int) – n period long-term.
- **fillna** (bool) – if True, fill nan values.

**Returns** New feature generated.

**Return type** pandas.Series
**Moving Average Convergence Divergence (MACD Diff)**

Shows the relationship between MACD and MACD Signal.

https://en.wikipedia.org/wiki/MACD

**Parameters**

- `close` (pandas.Series) – dataset ‘Close’ column.
- `n_fast` (int) – n period short-term.
- `n_slow` (int) – n period long-term.
- `n_sign` (int) – n period to signal.
- `fillna` (bool) – if True, fill nan values.

**Returns** New feature generated.

**Return type** pandas.Series

**Moving Average Convergence Divergence (MACD Signal)**

Shows EMA of MACD.

https://en.wikipedia.org/wiki/MACD

**Parameters**

- `close` (pandas.Series) – dataset ‘Close’ column.
- `n_fast` (int) – n period short-term.
- `n_slow` (int) – n period long-term.
- `n_sign` (int) – n period to signal.
- `fillna` (bool) – if True, fill nan values.

**Returns** New feature generated.

**Return type** pandas.Series

**Mass Index (MI)**

It uses the high-low range to identify trend reversals based on range expansions. It identifies range bulges that can foreshadow a reversal of the current trend.


**Parameters**

- `high` (pandas.Series) – dataset ‘High’ column.
- `low` (pandas.Series) – dataset ‘Low’ column.
- `n` (int) – n low period.
- `n2` (int) – n high period.
- `fillna` (bool) – if True, fill nan values.

**Returns** New feature generated.

**Return type** pandas.Series
ta.trend.psar_down(high, low, close, step=0.02, max_step=0.2)
Parabolic Stop and Reverse (Parabolic SAR)
Returns the PSAR series with non-N/A values for downward trends

Parameters
• high (pandas.Series) – dataset ‘High’ column.
• low (pandas.Series) – dataset ‘Low’ column.
• close (pandas.Series) – dataset ‘Close’ column.
• step (float) – the Acceleration Factor used to compute the SAR.
• max_step (float) – the maximum value allowed for the Acceleration Factor.

Returns New feature generated.
Return type pandas.Series

ta.trend.psar_down_indicator(high, low, close, step=0.02, max_step=0.2)
Parabolic Stop and Reverse (Parabolic SAR) Downward Trend Indicator
Returns 1, if there is a reversal towards an downward trend. Else, returns 0.

Parameters
• high (pandas.Series) – dataset ‘High’ column.
• low (pandas.Series) – dataset ‘Low’ column.
• close (pandas.Series) – dataset ‘Close’ column.
• step (float) – the Acceleration Factor used to compute the SAR.
• max_step (float) – the maximum value allowed for the Acceleration Factor.

Returns New feature generated.
Return type pandas.Series

ta.trend.psar_up(high, low, close, step=0.02, max_step=0.2)
Parabolic Stop and Reverse (Parabolic SAR)
Returns the PSAR series with non-N/A values for upward trends

Parameters
• high (pandas.Series) – dataset ‘High’ column.
• low (pandas.Series) – dataset ‘Low’ column.
• close (pandas.Series) – dataset ‘Close’ column.
• step (float) – the Acceleration Factor used to compute the SAR.
• max_step (float) – the maximum value allowed for the Acceleration Factor.

Returns New feature generated.
Return type pandas.Series
ta.trend.psar_up_indicator(high, low, close, step=0.02, max_step=0.2)
Parabolic Stop and Reverse (Parabolic SAR) Upward Trend Indicator

Returns 1, if there is a reversal towards an upward trend. Else, returns 0.

Parameters
- high (pandas.Series) – dataset ‘High’ column.
- low (pandas.Series) – dataset ‘Low’ column.
- close (pandas.Series) – dataset ‘Close’ column.
- step (float) – the Acceleration Factor used to compute the SAR.
- max_step (float) – the maximum value allowed for the Acceleration Factor.

Returns New feature generated.
Return type pandas.Series

ta.trend.trix(close, n=15, fillna=False)
Trix (TRIX)

Shows the percent rate of change of a triple exponentially smoothed moving average.

Parameters
- close (pandas.Series) – dataset ‘Close’ column.
- n (int) – n period.
- fillna (bool) – if True, fill nan values.

Returns New feature generated.
Return type pandas.Series

ta.trend.vortex_indicator_neg(high, low, close, n=14, fillna=False)
Vortex Indicator (VI)

It consists of two oscillators that capture positive and negative trend movement. A bearish signal triggers when the negative trend indicator crosses above the positive trend indicator or a key level.

Parameters
- high (pandas.Series) – dataset ‘High’ column.
- low (pandas.Series) – dataset ‘Low’ column.
- close (pandas.Series) – dataset ‘Close’ column.
- n (int) – n period.
- fillna (bool) – if True, fill nan values.

Returns New feature generated.
Return type pandas.Series

4.1. Documentation
It consists of two oscillators that capture positive and negative trend movement. A bullish signal triggers when the positive trend indicator crosses above the negative trend indicator or a key level.


Parameters
- **n** ([int](http://docs.python.org/3/library/stdtypes.html#int)) – n period.
- **fillna** ([bool](http://docs.python.org/3/library/stdtypes.html#bool)) – if True, fill nan values.

Returns New feature generated.

Return type pandas.Series

### 4.1.5 Others Indicators

Others Indicators.

**class ta.others.CumulativeReturnIndicator**

Cumulative Return (CR)

Parameters
- **fillna** ([bool](http://docs.python.org/3/library/stdtypes.html#bool)) – if True, fill nan values.

```
cumulative_return() → pandas.core.series.Series
```

Cumulative Return (CR)

Returns New feature generated.

Return type pandas.Series

**class ta.others.DailyLogReturnIndicator**

Daily Log Return (DLR)


Parameters
- **fillna** ([bool](http://docs.python.org/3/library/stdtypes.html#bool)) – if True, fill nan values.

```
daily_log_return() → pandas.core.series.Series
```

Daily Log Return (DLR)

Returns New feature generated.

Return type pandas.Series

**class ta.others.DailyReturnIndicator**

Daily Return (DR)

Parameters
• fillna (bool) – if True, fill nan values.

daily_return () \rightarrow \text{pandas.core.series.Series}
Daily Return (DR)

Returns New feature generated.
Return type pandas.Series
ta.others.cumulative_return (close, fillna=False)
Cumulative Return (CR)

Parameters
• close (pandas.Series) – dataset ‘Close’ column.
• fillna (bool) – if True, fill nan values.

Returns New feature generated.
Return type pandas.Series
ta.others.daily_log_return (close, fillna=False)
Daily Log Return (DLR)


Parameters
• close (pandas.Series) – dataset ‘Close’ column.
• fillna (bool) – if True, fill nan values.

Returns New feature generated.
Return type pandas.Series
ta.others.daily_return (close, fillna=False)
Daily Return (DR)

Parameters
• close (pandas.Series) – dataset ‘Close’ column.
• fillna (bool) – if True, fill nan values.

Returns New feature generated.
Return type pandas.Series
CHAPTER

FIVE

INDICES AND TABLES

- genindex
- modindex
- search
m
momentum, 9

o
others, 46

t
  ta, 9
ta.momentum, 9
ta.others, 46
ta.trend, 31
ta.volatility, 24
ta.volume, 18
trend, 31

v
volatility, 24
volume, 18
INDEX

A
acc_dist_index() (in module ta.volume), 21
acc_dist_index() (ta.volume.AccDistIndexIndicator method), 18
AccDistIndexIndicator (class in ta.volume), 18
adx() (in module ta.trend), 38
adx() (ta.trend.ADXIndicator method), 32
adx_neg() (in module ta.trend), 38
adx_neg() (ta.trend.ADXIndicator method), 32
adx_pos() (in module ta.trend), 39
adx_pos() (ta.trend.ADXIndicator method), 32
ADXIndicator (class in ta.trend), 31
ao() (in module ta.momentum), 14
ao() (ta.momentum.AwesomeOscillatorIndicator method), 9
aroon_down() (in module ta.trend), 39
aroon_down() (ta.trend.AroonIndicator method), 32
aroon_indicator() (ta.trend.AroonIndicator method), 32
aroon_up() (in module ta.trend), 39
aroon_up() (ta.trend.AroonIndicator method), 32
AroonIndicator (class in ta.trend), 32
average_true_range() (in module ta.volatility), 27
average_true_range() (ta.volatility.AverageTrueRange method), 24
AverageTrueRange (class in ta.volatility), 24
AwesomeOscillatorIndicator (class in ta.momentum), 9

B
bollinger_hband() (in module ta.volatility), 27
bollinger_hband() (ta.volatility.BollingerBands method), 24
bollinger_hband_indicator() (in module ta.volatility), 27
bollinger_hband_indicator() (ta.volatility.BollingerBands method), 25
bollinger_lband() (in module ta.volatility), 27
bollinger_lband() (ta.volatility.BollingerBands method), 25
bollinger_lband_indicator() (in module ta.volatility), 28
bollinger_lband_indicator() (ta.volatility.BollingerBands method), 25
bollinger_mavg() (in module ta.volatility), 28
bollinger_mavg() (ta.volatility.BollingerBands method), 25
bollinger_wband() (ta.volatility.BollingerBands method), 25
BollingerBands (class in ta.volatility), 24

C
cci() (in module ta.trend), 40
cci() (ta.trend.CCIIndicator method), 33
CCIIndicator (class in ta.trend), 33
chaikin_money_flow() (in module ta.volume), 21
chaikin_money_flow() (ta.volume.ChaikinMoneyFlowIndicator method), 19
ChaikinMoneyFlowIndicator (class in ta.volume), 18
cumulative_return() (in module ta.others), 47
cumulative_return() (ta.others.CumulativeReturnIndicator method), 46
CumulativeReturnIndicator (class in ta.others), 46

daily_log_return() (in module ta.others), 47
daily_log_return() (ta.others.DailyLogReturnIndicator method), 46
daily_return() (in module ta.others), 47
daily_return() (ta.others.DailyReturnIndicator method), 47
DailyLogReturnIndicator (class in ta.others), 46
DailyReturnIndicator (class in ta.others), 46
donchian_channel_hband() (in module ta.volatility), 28
donchian_channel_hband() (ta.volatility.DonchianChannel method),
Technical Analysis Library in Python Documentation, Release 0.1.4

25
donchian_channel_hband_indicator() (in module ta.volatility), 29

donchian_channel_hband_indicator() (ta.volatility.DonchianChannel method), 25

donchian_channel_lband() (in module ta.volatility), 29

donchian_channel_lband() (ta.volatility.DonchianChannel method), 26

donchian_channel_lband_indicator() (in module ta.volatility), 29

donchian_channel_lband_indicator() (ta.volatility.DonchianChannel method), 26

DonchianChannel (class in ta.volatility), 25
dpo() (in module ta.trend), 40
dpo() (ta.trend.DPOIndicator method), 33
DPOIndicator (class in ta.trend), 33

E
ease_of_movement() (in module ta.volume), 22
ease_of_movement() (ta.volume.EaseOfMovementIndicator method), 19
EaseOfMovementIndicator (class in ta.volume), 19
ess_of_movement() (in module ta.volume), 22
ess_of_movement() (ta.volume.EaseOfMovementIndicator method), 19
EaseOfMovementIndicator (class in ta.volume), 19
ema_indicator() (in module ta.trend), 40
ema_indicator() (ta.trend.EMAIndicator method), 33
EMAIndicator (class in ta.trend), 33

F
force_index() (in module ta.volume), 22
force_index() (ta.volume.ForceIndexIndicator method), 20
ForceIndexIndicator (class in ta.volume), 19

I
ichimoku_a() (in module ta.trend), 40
ichimoku_a() (ta.trend.IchimokuIndicator method), 34
ichimoku_b() (in module ta.trend), 41
ichimoku_b() (ta.trend.IchimokuIndicator method), 34
IchimokuIndicator (class in ta.trend), 34

K
kama() (in module ta.momentum), 14
kama() (ta.momentum.KAMAIndicator method), 10
KAMAIndicator (class in ta.momentum), 10
keltner_channel_central() (in module ta.volatility), 29
keltner_channel_central() (ta.volatility.KeltnerChannel method), 26

KeltnerChannel (class in ta.volatility), 26
kst() (in module ta.trend), 41
kst() (ta.trend.KSTIndicator method), 35
kst_diff() (ta.trend.KSTIndicator method), 35
kst_sig() (in module ta.trend), 42
kst_sig() (ta.trend.KSTIndicator method), 35
KSTIndicator (class in ta.trend), 34

MACD (class in ta.trend), 35
macd() (in module ta.trend), 42
macd() (ta.trend.MACD method), 35
macd_diff() (in module ta.trend), 42
macd_diff() (ta.trend.MACD method), 35
macd_signal() (in module ta.trend), 43
macd_signal() (ta.trend.MACD method), 35
mass_index() (in module ta.trend), 43
mass_index() (ta.trend.MassIndex method), 36
MassIndex (class in ta.trend), 36
MFIIndicator (class in ta.momentum), 10
momentum (module), 9
money_flow_index() (in module ta.momentum), 15
money_flow_index() (ta.momentum.MFIIndicator method), 10

N
negative_volume_index() (in module ta.volume), 22
negative_volume_index() (ta.volume.NegativeVolumeIndexIndicator method), 20
NegativeVolumeIndexIndicator (class in ta.volume), 20

on_balance_volume() (in module ta.volume), 23
on_balance_volume()
    (ta.volume.OnBalanceVolumeIndicator method), 20
OnBalanceVolumeIndicator (class in ta.volume), 20
others (module), 46

P
psar() (ta.trend.PSARIndicator method), 36
psar_down() (in module ta.trend), 43
psar_down() (ta.trend.PSARIndicator method), 36
psar_down_indicator() (in module ta.trend), 44
psar_down_indicator() (ta.trend.PSARIndicator method), 36
psar_up() (in module ta.trend), 44
psar_up() (ta.trend.PSARIndicator method), 37
psar_up_indicator() (in module ta.trend), 44
psar_up_indicator() (ta.trend.PSARIndicator method), 37
PSARIndicator (class in ta.trend), 36
put_call_ratio() (in module ta.volume), 23

R
roc() (in module ta.momentum), 15
roc() (ta.momentum.ROCIndicator method), 11
ROCIndicator (class in ta.momentum), 10
rsi() (in module ta.momentum), 15
rsi() (ta.momentum.RSIIndicator method), 11
RSIIndicator (class in ta.momentum), 11

S
sma_ease_of_movement() (in module ta.volume), 23
sma_ease_of_movement()
    (ta.volume.EaseOfMovementIndicator method), 19
stoch() (in module ta.momentum), 16
stoch() (ta.momentum.StochasticOscillator method), 12
stoch_signal() (in module ta.momentum), 16
stoch_signal() (ta.momentum.StochasticOscillator method), 12
StochasticOscillator (class in ta.momentum), 11

T
ta (module), 9
ta.momentum (module), 9
ta.others (module), 46
ta.trend (module), 31
ta.volatility (module), 24
ta.volume (module), 18
trend (module), 31
trix() (in module ta.trend), 45
trix() (ta.trend.TRIXIndicator method), 37

TRIXIndicator (class in ta.trend), 37
tsi() (in module ta.momentum), 16
tsi() (ta.momentum.TSIIndicator method), 12
TSIIndicator (class in ta.momentum), 12

U
UltimateOscillator (class in ta.momentum), 12
uo() (in module ta.momentum), 17
uo() (ta.momentum.UltimateOscillator method), 13

V
volatility (module), 24
volume (module), 18
volume_price_trend() (in module ta.volume), 23
volume_price_trend()
    (ta.volume.VolumePriceTrendIndicator method), 21
VolumePriceTrendIndicator (class in ta.volume), 20
vortex_indicator_diff() (ta.trend.VortexIndicator method), 37
vortex_indicator_neg() (in module ta.trend), 45
vortex_indicator_neg()
    (ta.trend.VortexIndicator method), 38
vortex_indicator_pos() (in module ta.trend), 45
vortex_indicator_pos()
    (ta.trend.VortexIndicator method), 38
VortexIndicator (class in ta.trend), 37

W
WilliamsRIndicator (class in ta.momentum), 13
wr() (in module ta.momentum), 17
wr() (ta.momentum.WilliamsRIndicator method), 13

Index 55