Technical Indicators

Release 0.0.1

Kunal Kini K

Aug 12, 2020

CONTENTS

Technical indicators library provides means to derive stock market technical indicators. Provides multiple ways of deriving technical indicators using raw OHLCV(Open, High, Low, Close, Volume) values.

Provides 2 ways to get the values,

1. You can send a pandas data-frame consisting of required values and you will get a new data-frame with required column appended in return.

   Note: make sure the column names are in lower case and are as follows,

   - Open values should be named 'open'
   - High values should be named 'high'
   - Low values should be named 'low'
   - Close values should be named 'close'
   - Volume values should be named 'volume'

2. You can send numpy arrays or pandas series of required values and you will get a new pandas series in return.

CHAPTER

ONE

INSTALLATION

```
pip install  technical_indicators_lib
```

CHAPTER

TWO

USAGE

```
# import dependencies
import pandas as pd
import numpy as np

# importing an indicator class
from technical_indicators_lib import OBV

# instantiate the class
obv = OBV()

# load data into a dataframe df
df = pd.read_csv("./test/data/test_data.csv")

# Method 1: get the data by sending a dataframe
df = obv.get_value_df(df)


# Method 2: get the data by sending series values
obv_values = obv.get_value_list(df["close"], df["volume"])
```

CHAPTER

THREE

DEVELOPMENT

Want to contribute?

Great. Follow these steps,

```
git clone https://github.com/kunalkini015/technical-indicators.git

cd technical_indicator_lib

pip install -r requirements.txt
```

then you are good to go. You can create a pull request or write to me at kunalkini15@gmail.com

CHAPTER

FOUR

CREDITS

Developed by Kunal Kini K, a software engineer by profession and passion.

If you have any comments, feedbacks or queries, write to me at kunalkini15@gmail.com

4.1 technical_indicators_lib package

class technical_indicators_lib.indicators.**ADI**

Bases: object

ADI -> Accumulation Distribution Index

The name accumulation/distribution comes from the idea that during accumulation buyers are in control and the price will be bid up through the day, or will make a recovery if sold down, in either case more often finishing near the day's high than the low. The opposite applies during distribution.

Links: http://www.ta-guru.com/Book/TechnicalAnalysis/TechnicalIndicators/AccumulationDistribution.php5

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

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

get_value_df(*df: pandas.core.frame.DataFrame*) → pandas.core.frame.DataFrame

Get The expected indicator in a pandas dataframe.

Args: df: pandas Dataframe with high, low, close and volume values

Returns: pandas.DataFrame: new pandas dataframe adding ADI as a new column, preserving the columns which already exists

get_value_list(*high_values: pandas.core.series.Series*, *low_values: pandas.core.series.Series*, *close_values: pandas.core.series.Series*, *volume_values: pandas.core.series.Series*)

Get The expected indicator in a pandas series.

Args: high_values(pandas.Series): 'High' values.

low_values: 'Low' values.

close_values: 'Close' values.

volume_values: 'Volume' values.

Returns: pandas.Series: A pandas Series of ADI values

info()

Provides basic information about the indicator

class technical_indicators_lib.indicators.**ATR**

Bases: object

ATR -> Average True Range

Average True Range is a volatility indicator which provides degree of price of volatility making use of smoothed moving average of true ranges.

Links: http://www.ta-guru.com/Book/TechnicalAnalysis/TechnicalIndicators/AverageTrueRange.php5

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

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

get_value_df(*df: pandas.core.frame.DataFrame*, *time_period: int = 14*)

Get The expected indicator in a pandas dataframe.

Args: df(pandas.DataFrame): pandas Dataframe with high, low and close values

time_period(int): look back period to calculate ATR

Returns: pandas.DataFrame: new pandas dataframe adding ATR as a new column, preserving the columns which already exists

get_value_list(*high_values: pandas.core.series.Series*, *low_values: pandas.core.series.Series*, *close_values: pandas.core.series.Series*, *time_period: int = 14*)

Get The expected indicator in a pandas series.

Args: high_values(pandas.Series): 'High' values.

low_values: 'Low' values.

close_values: 'Close' values.

time_period: Look back time period

Returns: pandas.Series: A pandas Series of ATR values

info()

Provides basic information about the indicator

class technical_indicators_lib.indicators.**BollingerBands**

Bases: object

Bollinger Bands Indicator

Bollinger Bands are the type of indicators which use mean and standard deviation of the movement of the stock to estimate the volatility of the stock

Links: http://www.ta-guru.com/Book/TechnicalAnalysis/TechnicalIndicators/BollingerBands.php5

https://www.investopedia.com/terms/b/bollingerbands.asp

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

get_value_df(*df: pandas.core.frame.DataFrame*, *time_period: int = 20*, *std_dev_multiplier: int = 2*)

Get The expected indicator in a pandas dataframe.

Args: df(pandas.DataFrame): pandas Dataframe with high, low, and close values

time_period(int): look back time period to calculate moving average std_dev_multiplier(int): constant value which will be multiplied by standard deviation

Returns: pandas.DataFrame: new pandas dataframe adding bb_upper and bb_lower as two columns, preserving the columns which already exists

get_value_list(*high_values: pandas.core.series.Series*, *low_values: pandas.core.series.Series*, *close_values: pandas.core.series.Series*, *time_period: int = 20*, *std_dev_multiplier: int = 2*)

Get The expected indicator in a pandas series.

Args: high_values(pandas.Series): 'High' values

low_values(pandas.Series): 'Low' values

close_values(pandas.Series): 'Close' values

time_period(int): look back time period to calculate moving average std_dev_multiplier(int): constant value which will be multiplied by standard deviation

Returns: pandas.Series: A tuple containing bb_upper and bb_lower values

info()

Provides basic information about the indicator

class technical_indicators_lib.indicators.**CCI**

Bases: object

CCI -> Commodity Channel Index

Main intention behind using the commodity channel index is to identify cyclical trends.

Links: http://www.ta-guru.com/Book/TechnicalAnalysis/TechnicalIndicators/CommodityChannelIndex.php5

https://www.investopedia.com/terms/c/commoditychannelindex.asp

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

get_value_df(*df: pandas.core.frame.DataFrame*, *time_period: int = 20*, *sd_multiplier: float = 0.015*)

Get The expected indicator in a pandas dataframe.

Args: df(pandas.DataFrame): pandas Dataframe with high, low, and close values

time_period(int): look back time period

sd_multiplier(float): constant value to be multiplied by standard deviation

Returns: pandas.DataFrame: new pandas dataframe adding CCI as a new column, preserving the columns which already exists

get_value_list(*high_values: pandas.core.series.Series*, *low_values: pandas.core.series.Series*, *close_values: pandas.core.series.Series*, *time_period: int = 20*, *sd_multiplier: float = 0.015*)

Get The expected indicator in a pandas series.

Args: high_values(pandas.Series): 'High' values

low_values(pandas.Series): 'Low' values

close_values(pandas.Series): 'Close' values

time_period(int): look back time period

sd_multiplier(float): constant value to be multiplied by standard deviation

Returns: pandas.Series: pandas series of CCI values

info()

Provides basic information about the indicator

class technical_indicators_lib.indicators.**CHO**
Bases: object

CHO -> Chaikin Oscillators

Chaikin oscillator is designed to anticipate the directional changes in Accumulation distributin line by measuring the momentum behind the movements.

Links: http://www.ta-guru.com/Book/TechnicalAnalysis/TechnicalIndicators/ChaikinOscillator.php5

https://en.wikipedia.org/wiki/Chaikin_Analytics#Chaikin_oscillator

get_value_df(*df: pandas.core.frame.DataFrame*, *short_time_period: int = 3*, *long_time_period: int = 10*)
Get The expected indicator in a pandas dataframe.

Args: df(pandas.DataFrame): pandas Dataframe with high, low, close and volume values

short_time_period(int): look back period to calculate short term moving average

long_time_period(int): look back period to calculate long term moving average

Returns: pandas.DataFrame: new pandas dataframe adding CHO as a new column, preserving the columns which already exists

get_value_list(*high_values: pandas.core.series.Series*, *low_values: pandas.core.series.Series*, *close_values: pandas.core.series.Series*, *volume_values: pandas.core.series.Series*, *short_time_period: int = 3*, *long_time_period: int = 10*)
Get The expected indicator in a pandas series.

Args: high_values(pandas.Series): 'High' values.

low_values: 'Low' values.

close_values: 'Close' values.

volume_levels: 'Volume' values

short_time_period(int): look back period to calculate short term moving average

long_time_period(int): look back period to calculate long term moving average

Returns: pandas.Series: A pandas Series of CHO values

info()
Provides basic information about the indicator

class technical_indicators_lib.indicators.**CHV**
Bases: object

CHV -> Chaikin Volatility

Chaikin Volatility determines the volatility of instrument using percentage change in a moving average of difference between high price and the low price over a specific period of time.

Links: http://www.ta-guru.com/Book/TechnicalAnalysis/TechnicalIndicators/ChaikinVolatility.php5

get_value_df(*df: pandas.core.frame.DataFrame*, *time_period: int = 10*)
Get The expected indicator in a pandas dataframe.

Args: df(pandas.DataFrame): pandas Dataframe with high and low values

time_period(int): look back period to calculate moving average

Returns: pandas.DataFrame: new pandas dataframe adding CHV as a new column, preserving existing columns

get_value_list(*high_values: pandas.core.series.Series*, *low_values: pandas.core.series.Series*, *time_period: int = 10*)
Get The expected indicator in a pandas series.

Args: high_values(pandas.Series): 'High' values.

low_values: 'Low' values.

time_period(int): look back period to calculate moving average

Returns: pandas.Series: A pandas Series of CHV values

info()
Provides basic information about the indicator

class technical_indicators_lib.indicators.**CMF**
Bases: object

CMF -> Chaikin Money Flow

Chaikin Money flow is used to measure money flow volume over a certain time periods.

Links: http://www.ta-guru.com/Book/TechnicalAnalysis/TechnicalIndicators/ChaikinMoneyFlow.php5

https://en.wikipedia.org/wiki/Chaikin_Analytics#Chaikin_Money_Flow

get_value_df(*df: pandas.core.frame.DataFrame*, *time_period: int = 20*)
Get The expected indicator in a pandas dataframe.

Args: df(pandas.DataFrame): pandas Dataframe with high, low, close and volume values

time_period(int): look back period to calculate CMF

Returns: pandas.DataFrame: new pandas dataframe adding CMF as a new column, preserving the columns which already exists

get_value_list(*high_values: pandas.core.series.Series*, *low_values: pandas.core.series.Series*, *close_values: pandas.core.series.Series*, *volume_values: pandas.core.series.Series*, *time_period: int = 20*)
Get The expected indicator in a pandas series.

Args: high_values(pandas.Series): 'High' values.

low_values: 'Low' values.

close_values: 'Close' values.

volume_levels: 'Volume' values

time_period: Look back time period

Returns: pandas.Series: A pandas Series of CMF values

info()
Provides basic information about the indicator

class technical_indicators_lib.indicators.**DC**
Bases: object

DC -> Donchian Channel

Donchian Channel is used as a volatility indicator by providing a channel of three lines whose contraction and expansion acting as signals.

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

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

get_value_df(*df: pandas.core.frame.DataFrame*, *time_period: int = 14*)
Get The expected indicator in a pandas dataframe.

Args: df(pandas.DataFrame): pandas Dataframe with high, low values

time_period(int): look back time period

Returns: pandas.DataFrame: new pandas dataframe adding dc_upper, dc_middle and dc_lower as three columns, preserving the columns which already exists

get_value_list(*high_values: pandas.core.series.Series*, *low_values: pandas.core.series.Series*, *time_period: int = 14*)
Get The expected indicator in a pandas series.

Args: high_values(pandas.Series): 'High' values

low_values(pandas.Series): 'Low' values

time_period(int): look back time period

Returns: pandas.Series: A tuple containing dc_upper, dc_middle and dc_lower values

info()
Provides basic information about the indicator

class technical_indicators_lib.indicators.**DPO**
Bases: object

DPO -> Detrended Price Oscillator

Detrend Price Oscillator tries to eliminates long term trends in order to easily identify small term trends.

Links: http://www.ta-guru.com/Book/TechnicalAnalysis/TechnicalIndicators/DetrendedPriceOscillator.php5

https://www.investopedia.com/terms/d/detrended-price-oscillator-dpo.asp

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

get_value_df(*df: pandas.core.frame.DataFrame*, *time_period: int = 20*)
Get The expected indicator in a pandas dataframe.

Args: df(pandas.DataFrame): pandas Dataframe with close values

time_period(int): look back time period

Returns: pandas.DataFrame: new pandas dataframe adding DPO as a new column, preserving the columns which already exists

get_value_list(*close_values: pandas.core.series.Series*, *time_period: int = 20*)
Get The expected indicator in a pandas series.

Args: close_values(pandas.Series): 'Close' values.

time_period(int): look back period to calculate moving average

Returns: pandas.Series: A pandas Series of DPO values

info()
Provides basic information about the indicator

class technical_indicators_lib.indicators.**EMA**
Bases: object

EMA -> Exponential Moving Average

Exponential Moving Average is a type of moving average which puts more weightage to the recent points, where as moving average puts same weightage all the points in consideration.

Links: http://www.ta-guru.com/Book/TechnicalAnalysis/TechnicalIndicators/MovingAverage.php5

https://www.investopedia.com/terms/e/ema.asp

https://en.wikipedia.org/wiki/Moving_average#Exponential_moving_average

get_value_df(*df: pandas.core.frame.DataFrame*, *time_period: int = 21*)
Get The expected indicator in a pandas dataframe.

Args: df(pandas.DataFrame): pandas Dataframe with close values

time_period(int): look back time period

Returns: pandas.DataFrame: new pandas dataframe adding EMA as a new column, preserving the columns which already exists

get_value_list(*close_values: pandas.core.series.Series*, *time_period: int = 21*)
Get The expected indicator in a pandas series.

Args: close_values(pandas.Series): 'Close' values.

time_period(int): look back time period

Returns: pandas.Series: A pandas Series of EMA values

info()
Provides basic information about the indicator

class technical_indicators_lib.indicators.**EMV**
Bases: object

EMV -> Ease of Movement

Ease of movement tries to identify amount of volume needed to move prices.

Links: http://www.ta-guru.com/Book/TechnicalAnalysis/TechnicalIndicators/EaseOfMovement.php5

https://www.investopedia.com/terms/e/easeofmovement.asp

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

get_value_df(*df: pandas.core.frame.DataFrame*, *volume_divisor: int = 1000000*, *need_moving_average: bool = True*, *time_period: int = 14*)
Get The expected indicator in a pandas dataframe.

Args: df(pandas.DataFrame): pandas Dataframe with high, low and volume values

volume_divisor(int): arbitrary divisor value required in the calculation of EMV

need_moving_average(bool): if True the moving avearge of the calculated values are returned time_period(int): look back time period

Returns: pandas.DataFrame: new pandas dataframe adding EMV as a new column, preserving the columns which already exists

get_value_list(*high_values: pandas.core.series.Series*, *low_values: pandas.core.series.Series*, *volume_values: pandas.core.series.Series*, *volume_divisor: int = 1000000*, *need_moving_average: bool = True*, *time_period: int = 14*)

Get The expected indicator in a pandas series.

Args: high_values(pandas.Series): 'High' values.

low_values(pandas.Series): 'Low' values.

volume_values(pandas.Series): 'Volume' values.

volume_divisor(int): arbitrary divisor value required in the calculation of EMV

need_moving_average(bool): if True the moving avearge of the calculated values are returned

time_period(int): look back time period

Returns: pandas.Series: A pandas Series of EMV values

info()

Provides basic information about the indicator

class technical_indicators_lib.indicators.**FI**

Bases: object

FI -> Force Index

Force index tries to determine the amount of power used to move the price of an asset.

Links: http://www.ta-guru.com/Book/TechnicalAnalysis/TechnicalIndicators/ForceIndex.php5

https://www.investopedia.com/terms/f/force-index.asp

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

get_value_df(*df: pandas.core.frame.DataFrame*, *time_period: int = 14*)

Get The expected indicator in a pandas dataframe.

Args: df(pandas.DataFrame): pandas Dataframe with close and volume values

time_period(int): look back time period

Returns: pandas.DataFrame: new pandas dataframe adding FI as a new column, preserving the columns which already exists

get_value_list(*close_values: pandas.core.series.Series*, *volume_values: pandas.core.series.Series*, *time_period: int = 14*)

Get The expected indicator in a pandas series.

Args: close_values(pandas.Series): 'Close' values.

volume_values(pandas.Series): 'Volume' values.

time_period(int): look back time period

Returns: pandas.Series: A pandas Series of FI values

info()

Provides basic information about the indicator

class technical_indicators_lib.indicators.**KC**

Bases: object

KC -> Keltner Channel

Keltner Channel is a volatility indicator which provides a channel of 3 lines whose contraction and expansion are used as signals.

Links: https://www.investopedia.com/terms/k/keltnerchannel.asp

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

get_value_df(*df: pandas.core.frame.DataFrame*, *time_period: int = 20*, *atr_time_period: int = 14*, *atr_multiplier: int = 2*)

Get The expected indicator in a pandas dataframe.

Args: df(pandas.DataFrame): pandas Dataframe with close values

time_period(int): look back time period to calculate moving average atr_time_period(int): time period to calculate average true range atr_multiplier(int): constant value which will be multiplied by average true range

Returns: pandas.DataFrame: new pandas dataframe adding kc_upper, kc_middle and kc_lower as three columns, preserving the columns which already exists

get_value_list(*close_values: pandas.core.series.Series*, *time_period: int = 20*, *atr_time_period: int = 14*, *atr_multiplier: int = 2*)

Get The expected indicator in a pandas series.

Args: close_values(pandas.Series): 'Close' values

time_period(int): look back time period to calculate moving average atr_time_period(int): time period to calculate average true range atr_multiplier(int): constant value which will be multiplied by average true range

Returns: pandas.Series: A tuple containing kc_upper, kc_middle and kc_lower values

info()

Provides basic information about the indicator

class technical_indicators_lib.indicators.**KST**

Bases: object

KST -> KST Oscillator

KST oscillator is a momentum indicator which makes it easy to interpret rate of change indicator.

Links: https://www.investopedia.com/terms/k/know-sure-thing-kst.asp

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

get_value_df(*df: pandas.core.frame.DataFrame*, *roc1: int = 10*, *roc1_ma_period: int = 10*, *roc2: int = 15*, *roc2_ma_period: int = 10*, *roc3: int = 20*, *roc3_ma_period: int = 10*, *roc4: int = 30*, *roc4_ma_period: int = 15*)

Get The expected indicator in a pandas dataframe.

Args: df(pandas.DataFrame): pandas Dataframe with close values

roc1(int): ROC Calculator - 1

roc1_ma_period(int): Smoothing constant for roc1

roc2(int): ROC Calculator - 2

roc2_ma_period(int): Smoothing constant for roc2

roc3(int): ROC Calculator - 3

roc3_ma_period(int): Smoothing constant for roc3

roc4(int): ROC Calculator - 4

roc4_ma_period(int): Smoothing constant for roc4

Returns: pandas.DataFrame: new pandas dataframe adding KST as a new column, preserving the columns which already exists

get_value_list(*close_values: pandas.core.series.Series*, *roc1: int = 10*, *roc1_ma_period: int = 10*, *roc2: int = 15*, *roc2_ma_period: int = 10*, *roc3: int = 20*, *roc3_ma_period: int = 10*, *roc4: int = 30*, *roc4_ma_period: int = 15*)

Get The expected indicator in a pandas series.

Args: close_values(pandas.Series): 'Close' values

roc1(int): ROC Calculator - 1

roc1_ma_period(int): Smoothing constant for roc1

roc2(int): ROC Calculator - 2

roc2_ma_period(int): Smoothing constant for roc2

roc3(int): ROC Calculator - 3

roc3_ma_period(int): Smoothing constant for roc3

roc4(int): ROC Calculator - 4

roc4_ma_period(int): Smoothing constant for roc4

Returns: pandas.Series: A pandas Series of KST values

info()

Provides basic information about the indicator

class technical_indicators_lib.indicators.**MACD**

Bases: object

MACD -> Moving Average Convergence Divergence

Moving Average Convergence is a trend following momentum indicator that shows a relationship between two moving averages of an asset.

Links: http://www.ta-guru.com/Book/TechnicalAnalysis/TechnicalIndicators/MovingAverageConvergenceDivergence.php5

https://www.investopedia.com/terms/m/macd.asp

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

get_value_df(*df: pandas.core.frame.DataFrame*, *short_time_period: int = 12*, *long_time_period: int = 26*, *need_signal: bool = True*, *signal_time_period: int = 9*)

Get The expected indicator in a pandas dataframe.

Args: df(pandas.DataFrame): pandas Dataframe with close values

short_time_period(int): short term look back time period.

long_time_period(int): long term look back time period.

need_signal(bool): if True MACD signal line is added as a new column to the returning pandas dataframe.

signal_time_period(int): look back period to calculate signal line

Returns: pandas.DataFrame: new pandas dataframe adding MACD and MACD_signal_line(if required) as new column/s, preserving the columns which already exists

get_value_list(*close_values: pandas.core.series.Series*, *short_time_period: int = 12*, *long_time_period: int = 26*, *need_signal: bool = True*, *signal_time_period: int = 9*)

Get The expected indicator in a pandas series.

Args: close_values(pandas.Series): 'Close' values

short_time_period(int): short term look back time period.

long_time_period(int): long term look back time period.

need_signal(bool): if True MACD signal line is also returned along with MACD line

signal_time_period(int): look back period to calculate signal line

Returns: pandas.Series: A tuple containing MACD and MACD_signal_line(if required)

info()

Provides basic information about the indicator

class technical_indicators_lib.indicators.**MED**

Bases: object

MED -> Median Price

Median Price calculates a mid point of a price range.

Links: http://www.ta-guru.com/Book/TechnicalAnalysis/TechnicalIndicators/MedianPrice.php5

get_value_df(*df: pandas.core.frame.DataFrame*)

Get The expected indicator in a pandas dataframe.

Args: df(pandas.DataFrame): pandas Dataframe with high and low values

Returns: pandas.DataFrame: new pandas dataframe adding MED as a new column, preserving the columns which already exists

get_value_list(*high_values: pandas.core.series.Series*, *low_values: pandas.core.series.Series*)

Get The expected indicator in a pandas series.

Args: high_values(pandas.Series): 'High' values.

low_values(pandas.Series): 'Low' values.

Returns: pandas.Series: A pandas Series of MED values

info()

Provides basic information about the indicator

class technical_indicators_lib.indicators.**MFI**

Bases: object

MFI -> Money Flow Index

Money flow index uses price and volume data to for identifying overbought and oversold signals of an asset

Links: http://www.ta-guru.com/Book/TechnicalAnalysis/TechnicalIndicators/MoneyFlowIndex.php5

https://www.investopedia.com/terms/m/mfi.asp https://en.wikipedia.org/wiki/Money_flow_index

get_value_df(*df: pandas.core.frame.DataFrame*, *time_period: int = 14*)

Get The expected indicator in a pandas dataframe.

Args: df(pandas.DataFrame): pandas Dataframe with high, low, close and volume values

time_period(int): look back time period.

Returns: pandas.DataFrame: new pandas dataframe adding MFI as a new column, preserving the columns which already exists

get_value_list(*high_values: pandas.core.series.Series*, *low_values: pandas.core.series.Series*, *close_values: pandas.core.series.Series*, *volume_values: pandas.core.series.Series*, *time_period: int = 14*)

Get The expected indicator in a pandas series.

Args: high_values(pandas.Series): 'High' values

low_values(pandas.Series): 'Low' values

close_values(pandas.Series): 'Close' values

volume_values(pandas.Series): 'Volume' values

time_period(int): look back time period

Returns: pandas.Series: A pandas Series of MFI values

info()

Provides basic information about the indicator

class technical_indicators_lib.indicators.**MI**

Bases: object

MI -> Mass Index

Mass index tries to determine the range of high and low values over a specified period of time

Links: http://www.ta-guru.com/Book/TechnicalAnalysis/TechnicalIndicators/MassIndex.php5

get_value_df(*df: pandas.core.frame.DataFrame*, *time_period: int = 25*, *ema_time_period: int = 9*)

Get The expected indicator in a pandas dataframe.

Args: df(pandas.DataFrame): pandas Dataframe with high and low values

time_period(int): look back time period to calculate the sum

ema_time_period(int): look back time period to calculate the exponential moving average

Returns: pandas.DataFrame: new pandas dataframe adding MI as a new column, preserving the columns which already exists

get_value_list(*high_values: pandas.core.series.Series*, *low_values: pandas.core.series.Series*, *time_period: int = 25*, *ema_time_period: int = 9*)

Get The expected indicator in a pandas series.

Args: high_values(pandas.Series): 'High' values.

low_values(pandas.Series): 'Low' values.

time_period(int): look back time period to calculate the sum

ema_time_period(int): look back time period to calculate the exponential moving average

Returns: pandas.Series: A pandas Series of MI values

info()

Provides basic information about the indicator

class technical_indicators_lib.indicators.**MOM**

Bases: object

MOM -> Momentum

Momentum helps to determine the price changes from one period to another.

Links: http://www.ta-guru.com/Book/TechnicalAnalysis/TechnicalIndicators/Momentum.php5

get_value_df(*df: pandas.core.frame.DataFrame*, *time_period: int = 1*)
Get The expected indicator in a pandas dataframe.

Args: df(pandas.DataFrame): pandas Dataframe with close values

time_period(int): look back time period.

Returns: pandas.DataFrame: new pandas dataframe adding MOM as a new column, preserving the columns which already exists

get_value_list(*close_values: pandas.core.series.Series*, *time_period: int = 1*)
Get The expected indicator in a pandas series.

Args: close_values(pandas.Series): 'Close' values

time_period(int): look back time period

Returns: pandas.Series: A pandas Series of MOM values

info()
Provides basic information about the indicator

class technical_indicators_lib.indicators.**NVI**
Bases: object

NVI -> Negative Volume Index

Negative Volume Index helps in identifying trends and reversals.

Links: http://www.ta-guru.com/Book/TechnicalAnalysis/TechnicalIndicators/NegativeVolumeIndex.php5

https://www.investopedia.com/terms/n/nvi.asp

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

get_value_df(*df: pandas.core.frame.DataFrame*, *start_value: int = 1000*)
Get The expected indicator in a pandas dataframe.

Args: df(pandas.DataFrame): pandas Dataframe with close and volume values

start_value(int): arbitrary starting value to calculate NVI

Returns: pandas.DataFrame: new pandas dataframe adding NVI as new column, preserving the columns which already exists

get_value_list(*close_values: pandas.core.series.Series*, *volume_values: pandas.core.series.Series*, *start_value: int = 1000*)
Get The expected indicator in a pandas series.

Args: close_values(pandas.Series): 'Close' values

volume_values(pands.Series): 'Volume' values

start_value(int): arbitrary starting value to calculate NVI

Returns: pandas.Series: A pandas Series of NVI values

info()
Provides basic information about the indicator

class technical_indicators_lib.indicators.**NegativeDirectionIndicator**
Bases: object

get_value_df(*high_values*, *low_values*, *time_period=14*)

info()
Provides basic information about the indicator

class technical_indicators_lib.indicators.**OBV**
Bases: object

OBV -> On Balance Volume

On Balance Volume provides the signal whether the volume is flowing in or out of a given security.

Links: http://www.ta-guru.com/Book/TechnicalAnalysis/TechnicalIndicators/OnBalanceVolume.php5

https://www.investopedia.com/terms/o/onbalancevolume.asp

https://en.wikipedia.org/wiki/On-balance_volume

get_value_df(*df: pandas.core.frame.DataFrame*)
Get The expected indicator in a pandas dataframe.

Args: df(pandas.DataFrame): pandas Dataframe with close and volume values

Returns: pandas.DataFrame: new pandas dataframe adding OBV as new column, preserving the columns which already exists

get_value_list(*close_values: pandas.core.series.Series*, *volume_values: pandas.core.series.Series*)
Get The expected indicator in a pandas series.

Args: close_values(pandas.Series): 'Close' values

volume_values(pands.Series): 'Volume' values

Returns: pandas.Series: A pandas Series of OBV values

info()
Provides basic information about the indicator

class technical_indicators_lib.indicators.**PC**
Bases: object

PC -> Price Channels

Price channels forms a boundary and between them the close price of an asset is oscillating.

Links: http://www.ta-guru.com/Book/TechnicalAnalysis/TechnicalIndicators/PriceChannels.php5

https://www.investopedia.com/terms/p/price-channel.asp

get_value_df(*df: pandas.core.frame.DataFrame*, *percent_value: int = 6*, *time_period: int = 21*)
Get The expected indicator in a pandas dataframe.

Args: df(pandas.DataFrame): pandas Dataframe with close values

percent_value(int): value to calculate the percentage of close value to create the boundary

time_period(int): look back time period to calculate moving average

Returns: pandas.DataFrame: new pandas dataframe adding PC as new column, preserving the columns which already exists

get_value_list(*close_values: pandas.core.series.Series*, *percent_value: int = 6*, *ema_period: int = 21*)

Get The expected indicator in a pandas series.

Args: close_values(pandas.Series): 'Close' values

percent_value(int): value to calculate the percentage of close value to create the boundary

time_period(int): look back time period to calculate moving average

Returns: pandas.Series: A tuple containing PC_upper and PC_lower values

info()

Provides basic information about the indicator

class technical_indicators_lib.indicators.**PO**

Bases: object

PO -> Price Oscillator

Price oscillator is a momentum osciallator which shows a difference between two moving averages.

Links: http://www.ta-guru.com/Book/TechnicalAnalysis/TechnicalIndicators/PriceOscillator.php5

get_value_df(*df: pandas.core.frame.DataFrame*, *short_time_period: int = 9*, *long_time_period: int = 26*)

Get The expected indicator in a pandas dataframe.

Args: df(pandas.DataFrame): pandas Dataframe with close values

short_time_period(int): look back time period to calculate short term moving average

long_time_period(int): look back time period to calculate long term moving average

Returns: pandas.DataFrame: new pandas dataframe adding PO as new column, preserving the columns which already exists

get_value_list(*close_values: pandas.core.series.Series*, *short_ema_period: int = 9*, *long_ema_period: int = 26*)

Get The expected indicator in a pandas series.

Args: close_values(pandas.Series): 'Close' values

short_time_period(int): look back time period to calculate short term moving average

long_time_period(int): look back time period to calculate long term moving average

Returns: pandas.Series: A pandas Series of PO values

info()

Provides basic information about the indicator

class technical_indicators_lib.indicators.**PVT**

Bases: object

PVT -> Price Volume Trend

Price Volume Trend helps in identifying trend by using cumulative volume adjusted by change in price

Links: http://www.ta-guru.com/Book/TechnicalAnalysis/TechnicalIndicators/PriceAndVolumeTrend.php5

https://www.investopedia.com/terms/v/vptindicator.asp

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

get_value_df(*df: pandas.core.frame.DataFrame*)
Get The expected indicator in a pandas dataframe.

Args: df(pandas.DataFrame): pandas Dataframe with close and volume values

Returns: pandas.DataFrame: new pandas dataframe adding PVT as new column, preserving the columns which already exists

get_value_list(*close_values: pandas.core.series.Series*, *volume_values: pandas.core.series.Series*)
Get The expected indicator in a pandas series.

Args: close_values(pandas.Series): 'Close' values

volume_values(pands.Series): 'Volume' values

Returns: pandas.Series: A pandas Series of PVT values

info()
Provides basic information about the indicator

class technical_indicators_lib.indicators.**PositiveDirectionIndicator**
Bases: object

MOM -> Momentum

get_value_df(*df*, *time_period=14*)

get_value_list(*high_values: pandas.core.series.Series*, *low_values: pandas.core.series.Series*, *time_period: int = 14*)

info()
Provides basic information about the indicator

class technical_indicators_lib.indicators.**PositiveVolumeIndex**
Bases: object

get_value_df(*df*, *starting_value=100*)

get_value_list(*close_values: pandas.core.series.Series*, *volume_values: pandas.core.series.Series*, *starting_value: int = 100*)

info()
Provides basic information about the indicator

class technical_indicators_lib.indicators.**ROC**
Bases: object

ROC -> Rate Of Change

Rate of change helps in calculation of speed of ascent or descent.

Links: http://www.ta-guru.com/Book/TechnicalAnalysis/TechnicalIndicators/RateOfChange.php5

https://www.investopedia.com/terms/p/pricerateofchange.asp

get_value_df(*df: pandas.core.frame.DataFrame*, *time_period: int = 12*)
Get The expected indicator in a pandas dataframe.

Args: df(pandas.DataFrame): pandas Dataframe with close values

time_period(int): look back time period to calculate previous close

Returns: pandas.DataFrame: new pandas dataframe adding ROC as new column, preserving the columns which already exists

get_value_list(*close_values: pandas.core.series.Series*, *time_period: int = 12*)
Get The expected indicator in a pandas series.

Args: close_values(pandas.Series): 'Close' values

time_period(int): look back time period

Returns: pandas.Series: A pandas Series of ROC values

info()
Provides basic information about the indicator

class technical_indicators_lib.indicators.**ROCV**
Bases: object

ROCV -> Rate of Change Volume

ROCV indicator is used to identify whether the price movement is confirmed by trading volume.

Links: http://www.ta-guru.com/Book/TechnicalAnalysis/TechnicalIndicators/VolumeRateOfChange.php5

https://www.investopedia.com/articles/technical/02/091002.asp

get_value_df(*df: pandas.core.frame.DataFrame*, *time_period: int = 12*)
Get The expected indicator in a pandas dataframe.

Args: df(pandas.DataFrame): pandas Dataframe with volume values

time_period(int): look back time period

Returns: pandas.DataFrame: new pandas dataframe adding ROCV as new column, preserving the columns which already exists

get_value_list(*volume_values: pandas.core.series.Series*, *time_period: int = 12*)
Get The expected indicator in a pandas series.

Args: volume_values(pandas.Series): 'Volume' values

time_period(int): look back time period

Returns: pandas.Series: A pandas Series of ROCV values

info()
Provides basic information about the indicator

class technical_indicators_lib.indicators.**RSI**
Bases: object

RSI -> Relative Strength Index

Relative Strength Index is used to generate oversold and overbought signals.

Links: http://www.ta-guru.com/Book/TechnicalAnalysis/TechnicalIndicators/RelativeStrengthIndex.php5

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

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

get_value_df(*df*, *time_period=14*)
Get The expected indicator in a pandas dataframe.

Args: df(pandas.DataFrame): pandas Dataframe with close values

time_period(int): look back time period to calculate moving average

Returns: pandas.DataFrame: new pandas dataframe adding RSI as new column, preserving the columns which already exists

get_value_list(*close_values: pandas.core.series.Series*, *time_period: int = 14*)
Get The expected indicator in a pandas series.

Args: close_values(pandas.Series): 'Close' values

time_period(int): look back time period

Returns: pandas.Series: A pandas Series of RSI values

info()
Provides basic information about the indicator

class technical_indicators_lib.indicators.**SMA**
Bases: object

SMA -> Simple Moving Avearge

Simple Moving Average is an arithmetic moving average which is calculated by taking the sum of values from recent time periods and then divided by number of time periods.

Links: http://www.ta-guru.com/Book/TechnicalAnalysis/TechnicalIndicators/MovingAverage.php5

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

https://en.wikipedia.org/wiki/Moving_average#Simple_moving_average

get_value_df(*df: pandas.core.frame.DataFrame*, *time_period: int = 21*)
Get The expected indicator in a pandas dataframe.

Args: df(pandas.DataFrame): pandas Dataframe with close values

time_period(int): look back time period to calculate moving average

Returns: pandas.DataFrame: new pandas dataframe adding SMA as new column, preserving the columns which already exists

get_value_list(*close_values: pandas.core.series.Series*, *time_period: int = 21*)
Get The expected indicator in a pandas series.

Args: close_values(pandas.Series): 'Close' values

time_period(int): look back time period

Returns: pandas.Series: A pandas Series of SMA values

info()
Provides basic information about the indicator

class technical_indicators_lib.indicators.**StochasticKAndD**
Bases: object

StochasticKAndD -> Stochastic K and StochasticD

Stochastic Oscillator is a momentum indicator comparing a particular price to a range of prices over specific period of time.

Links: http://www.ta-guru.com/Book/TechnicalAnalysis/TechnicalIndicators/Stochastic.php5

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

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

get_value_df(*df: pandas.core.frame.DataFrame*, *time_period: int = 14*)
Get The expected indicator in a pandas dataframe.

Args: df(pandas.DataFrame): pandas Dataframe with high, low and close values

time_period(int): look back time period

Returns: pandas.DataFrame: new pandas dataframe adding stoc_d and stoc_k as new columns, preserving the columns which already exists

get_value_list(*high_values: pandas.core.series.Series*, *low_values: pandas.core.series.Series*, *close_values: pandas.core.series.Series*, *time_period: int = 14*)
Get The expected indicator in a pandas series.

Args: high_values(pandas.Series): 'High' values

low_values(pandas.Series): 'Low' values

close_values(pandas.Series): 'Close' values

time_period(int): look back time period

Returns: pandas.Series:A tuple containing stoch_k and stoc_d values

info()
Provides basic information about the indicator

class technical_indicators_lib.indicators.**TR**
Bases: object

TR -> True Range

True range is an essential component of determination of average true range.

Links: http://www.ta-guru.com/Book/TechnicalAnalysis/TechnicalIndicators/TrueRange.php5

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

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

get_value_df(*df: pandas.core.frame.DataFrame*)
Get The expected indicator in a pandas dataframe.

Args: df(pandas.DataFrame): pandas Dataframe with high, low, and close values

Returns: pandas.DataFrame: new pandas dataframe adding TR as new column, preserving the columns which already exists

get_value_list(*high_values: pandas.core.series.Series*, *low_values: pandas.core.series.Series*, *close_values: pandas.core.series.Series*)
Get The expected indicator in a pandas series.

Args: high_values(pandas.Series): 'High' values

low_values(pandas.Series): 'Low' values

close_values(pandas.Series): 'Close' values

Returns: pandas.Series: A pandas Series of TR values

info()
Provides basic information about the indicator

class technical_indicators_lib.indicators.**TSI**
Bases: object

TSI -> True Strength Index

True Strength Index is a momentum indicator which is useful in identifying oversold and overbough conditions.

Links: https://www.investopedia.com/terms/t/tsi.asp

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

get_value_df(*df: pandas.core.frame.DataFrame*, *time_period1: int = 25*, *time_period2: int = 13*)
Get The expected indicator in a pandas dataframe.

Args: df(pandas.DataFrame): pandas Dataframe with close values

time_period1(int): time period to calculate moving average of price change time_period2(int): time period to calcualte moving average of prior moving average

Returns: pandas.DataFrame: new pandas dataframe adding TSI as a new column, preserving the columns which already exists

get_value_list(*close_values: pandas.core.series.Series*, *time_period1: int = 25*, *time_period2: int = 13*)
Get The expected indicator in a pandas series.

Args: close_values(pandas.Series): 'Close' values

time_period1(int): time period to calculate moving average of price change time_period2(int): time period to calcualte moving average of prior moving average

Returns: pandas.Series: A pandas Series of TSI values

info()
Provides basic information about the indicator

class technical_indicators_lib.indicators.**TYP**
Bases: object

TYP -> Typical Price

Typical Price is an average of low, high and close. It is used as an alternative to close price.

Links: http://www.ta-guru.com/Book/TechnicalAnalysis/TechnicalIndicators/TypicalPrice.php5

get_value_df(*df: pandas.core.frame.DataFrame*)
Get The expected indicator in a pandas dataframe.

Args: df(pandas.DataFrame): pandas Dataframe with high, low, and close values

Returns: pandas.DataFrame: new pandas dataframe adding TYP as new column, preserving the columns which already exists

get_value_list(*high_values: pandas.core.series.Series*, *low_values: pandas.core.series.Series*, *close_values: pandas.core.series.Series*)
Get The expected indicator in a pandas series.

Args: high_values(pandas.Series): 'High' values

low_values(pandas.Series): 'Low' values

close_values(pandas.Series): 'Close' values

Returns: pandas.Series: A pandas Series of TYP values

info()
Provides basic information about the indicator

class technical_indicators_lib.indicators.**Trix**
Bases: object

Trix -> Triple exponential moving average

Trix is triple exponential moving average, can be used as both oscillator and momentum indicator.

Links: http://www.ta-guru.com/Book/TechnicalAnalysis/TechnicalIndicators/TRIX.php5

https://www.investopedia.com/terms/t/trix.asp

https://en.wikipedia.org/wiki/Trix_(technical_analysis)

get_value_df(*df: pandas.core.frame.DataFrame*, *time_period: int = 14*)
Get The expected indicator in a pandas dataframe.

Args: df(pandas.DataFrame): pandas Dataframe with close values

time_period(int): look back time period

Returns: pandas.DataFrame: new pandas dataframe adding Trix as new column, preserving the columns which already exists

get_value_list(*close_values: pandas.core.series.Series*, *time_period: int = 14*)
Get The expected indicator in a pandas series.

Args: close_values(pandas.Series): 'Close' values

time_period(int): look back time period

Returns: pandas.Series: A pandas Series of Trix values

info()
Provides basic information about the indicator

class technical_indicators_lib.indicators.**VHF**
Bases: object

VHF -> Vertical Horizontal Filter

VHF is an indicator which is used in identifying trend activity.

Links: http://www.ta-guru.com/Book/TechnicalAnalysis/TechnicalIndicators/TypicalPrice.php5

get_value_df(*df: pandas.core.frame.DataFrame*, *time_period: int = 28*)
Get The expected indicator in a pandas dataframe.

Args: df(pandas.DataFrame): pandas Dataframe with close values

time_period(int): look back time period

Returns: pandas.DataFrame: new pandas dataframe adding VHF as new column, preserving the columns which already exists

get_value_list(*close_values: pandas.core.series.Series*, *time_period: int = 28*)
Get The expected indicator in a pandas series.

Args: close_values(pandas.Series): 'Close' values

time_period(int): look back time period

Returns: pandas.Series: A pandas Series of VHF values

info()
Provides basic information about the indicator

class technical_indicators_lib.indicators.**VI**
Bases: object

VI -> Vortex Indicator

Vortex indicator is used to identify the start of the new trend or the continuation of existing trends.

Links: https://www.investopedia.com/terms/v/vortex-indicator-vi.asp

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

get_value_df(*df: pandas.core.frame.DataFrame*, *time_period: int = 25*)
Get The expected indicator in a pandas dataframe.

Args: df(pandas.DataFrame): pandas Dataframe with high, low, and close values

time_period(int): look back time period

Returns: pandas.DataFrame: new pandas dataframe adding VI+ and VI- as two columns, preserving the columns which already exists

get_value_list(*high_values: pandas.core.series.Series*, *low_values: pandas.core.series.Series*, *close_values: pandas.core.series.Series*, *time_period: int = 14*)
Get The expected indicator in a pandas series.

Args: high_values(pandas.Series): 'High' values

low_values(pandas.Series): 'Low' values

close_values(pandas.Series): 'Close' values

time_period(int): look back time period

Returns: pandas.Series: A tuple containing vi_plus and vi_minus values

info()
Provides basic information about the indicator

class technical_indicators_lib.indicators.**VLT**
Bases: object

VLT -> Volatility

Standard Deviation, variance and volatility are used to evaluate the volatility in the movement of the stock.

Links: http://www.ta-guru.com/Book/TechnicalAnalysis/TechnicalIndicators/Volatility.php5

get_value_df(*df: pandas.core.frame.DataFrame*, *time_period: int = 21*, *need_variance: bool = True*, *need_deviation: bool = True*)
Get The expected indicator in a pandas dataframe.

Args: df(pandas.DataFrame): pandas Dataframe with close values

time_period(int): look back time period to calculate moving average

need_variance(bool): if True variance will be added as a new column to the returning dataframe

need_deviation(bool): if True deviation will be added as a new column to the returning dataframe

Returns: pandas.DataFrame: new pandas dataframe adding VLT, SV(if required), SD(if required) as new column/s, preserving the columns which already exists

get_value_list(*close_values: pandas.core.series.Series*, *time_period: int = 21*, *need_variance: bool = True*, *need_deviation: bool = True*)

Returns a series of SMA values

Get The expected indicator in a pandas series.

Args: close_values(pandas.Series): 'Close' values

time_period(int): look back time period to calculate moving average

need_variance(bool): if True variance will be added as a new column to the returning dataframe

need_deviation(bool): if True deviation will be added as a new column to the returning dataframe

Returns: pandas.Series: A tuple containing Volatility, variance(if required), deviation(if required) values

info()

Provides basic information about the indicator

class technical_indicators_lib.indicators.**VO**

Bases: object

VO -> Volume Oscillator

Volume Oscillator is used to identify the expansion or the contraction of volumes.

Links: http://www.ta-guru.com/Book/TechnicalAnalysis/TechnicalIndicators/VolumeOscillator.php5

get_value_df(*df: pandas.core.frame.DataFrame*, *short_time_period: int = 9*, *long_time_period: int = 26*)

Get The expected indicator in a pandas dataframe.

Args: df(pandas.DataFrame): pandas Dataframe with volume values

short_time_period(int): look back time period for short term moving average

long_time_period(int): look back time period for long term moving average

Returns: pandas.DataFrame: new pandas dataframe adding VO as new column, preserving the columns which already exists

get_value_list(*volume_values: pandas.core.series.Series*, *short_ema: int = 9*, *long_ema: int = 26*)

Get The expected indicator in a pandas series.

Args: volume_values(pandas.Series): 'Volume' values

short_time_period(int): look back time period for short term moving average

long_time_period(int): look back time period for long term moving average

Returns: pandas.Series: A pandas Series of VO values

info()

Provides basic information about the indicator

class technical_indicators_lib.indicators.**WCL**

Bases: object

WCL -> Weighted Close

Weighted Close is a type of technical indicator which averages price of each period.

Links: http://www.ta-guru.com/Book/TechnicalAnalysis/TechnicalIndicators/WeightedClose.php5

get_value_df(*df: pandas.core.frame.DataFrame*)
Get The expected indicator in a pandas dataframe.

Args: df(pandas.DataFrame): pandas Dataframe with high, low, and close values

Returns: pandas.DataFrame: new pandas dataframe adding WCL as new column, preserving the columns which already exists

get_value_list(*high_values: pandas.core.series.Series*, *low_values: pandas.core.series.Series*, *close_values: pandas.core.series.Series*)
Get The expected indicator in a pandas series.

Args: high_values(pandas.Series): 'High' values

low_values(pandas.Series): 'Low' values

close_values(pandas.Series): 'Close' values

Returns: pandas.Series: A pandas Series of WCL values

info()
Provides basic information about the indicator

class technical_indicators_lib.indicators.**WilliamsR**
Bases: object

WilliamsR -> Williams R indicator

Williams R is tries to determine overbought and oversold levels of an asset.

Links: http://www.ta-guru.com/Book/TechnicalAnalysis/TechnicalIndicators/WilliamsAccumulationDistribution.php5

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

https://en.wikipedia.org/wiki/Williams_%25R

get_value_df(*df: pandas.core.frame.DataFrame*, *time_period: int = 14*)
Get The expected indicator in a pandas dataframe.

Args: df(pandas.DataFrame): pandas Dataframe with high, low, and close values

time_period: look back time period

Returns: pandas.DataFrame: new pandas dataframe adding WilliamsR as new column, preserving the columns which already exists

get_value_list(*high_values: pandas.core.series.Series*, *low_values: pandas.core.series.Series*, *close_values: pandas.core.series.Series*, *time_period: int = 14*)
Get The expected indicator in a pandas series.

Args: high_values(pandas.Series): 'High' values

low_values(pandas.Series): 'Low' values

close_values(pandas.Series): 'Close' values

time_period: look back time period

Returns: pandas.Series: A pandas Series of Williams R values

info()
Provides basic information about the indicator

PYTHON MODULE INDEX

i

t

INDEX

K

M

N

O

P

R

S

T

V

W