
babao Documentation

Release 0.2

JeanMax

Sep 26, 2019

Contents:

1	babao package	1
1.1	Subpackages	1
1.1.1	babao.inputs package	1
1.1.1.1	Subpackages	1
1.1.1.2	Submodules	13
1.1.1.3	babao.inputs.inputBase module	13
1.1.1.4	babao.inputs.inputManager module	14
1.1.1.5	babao.inputs.krakenInputBase module	15
1.1.1.6	Module contents	15
1.1.2	babao.models package	15
1.1.2.1	Subpackages	15
1.1.2.2	Submodules	16
1.1.2.3	babao.models.modelBase module	16
1.1.2.4	babao.models.modelManager module	17
1.1.2.5	babao.models.rootModel module	17
1.1.2.6	Module contents	18
1.1.3	babao.utils package	18
1.1.3.1	Submodules	18
1.1.3.2	babao.utils.date module	18
1.1.3.3	babao.utils.enum module	19
1.1.3.4	babao.utils.file module	21
1.1.3.5	babao.utils.indicators module	21
1.1.3.6	babao.utils.lock module	22
1.1.3.7	babao.utils.log module	22
1.1.3.8	babao.utils.scale module	22
1.1.3.9	babao.utils.signal module	23
1.1.3.10	Module contents	23
1.2	Submodules	23
1.3	babao.arg module	23
1.4	babao.babao module	23
1.5	babao.commands module	23
1.6	babao.config module	24
1.7	babao.graph module	24
1.8	Module contents	24
2	Indices and tables	25

Python Module Index	27
Index	29

1.1 Subpackages

1.1.1 babao.inputs package

1.1.1.1 Subpackages

babao.inputs.ledger package

Submodules

babao.inputs.ledger.fakeLedgerInput module

Handle logging in database all our fake transactions (dry-run)

```
class babao.inputs.ledger.fakeLedgerInput.ABCFakeLedgerInput (log_to_file=True,  
                                                         temp=False)
```

Bases: *babao.inputs.ledger.ledgerInputBase.ABCLedgerInput*

Base class for any fake ledger

asset

Override this method with the desired CryptoEnum / QuoteEnum ex: self.asset = CryptoEnum.XBT

buy (*ledger, volume_spent, price, timestamp=None*)

Buy with the current ledger asset the asset of the given 'ledger'

(If the current ledger is a quote, this is a buy) 'volume_spent' quantity spent (including fees)

deposit (*ledger, volume, timestamp=None*)

Deposit from the current ledger to the given 'ledger'

fetch ()

Return a time-serie DataFrame fetched from the internets

This data will be stored on database for later use (and eventual resampling). Data can be continuous. Index must be nanosecond timestamps.

logTransaction (*typ, volume, refid, fee=0, product=0, timestamp=None*)

Log transaction in database if 'timestamp' is not given, the current time will be used

This should'nt be used outside of this class

sell (*ledger, volume_spent, price, timestamp=None*)

Buy with the asset of the given 'ledger' the current ledger asset

(If the current ledger is a quote, this is a sell) 'volume_spent' quantity spent (including fees)

withdraw (*ledger, volume, timestamp=None*)

Withdraw from the current ledger to the given 'ledger'

babao.inputs.ledger.krakenLedgerInput module

Handle logging in database all our real transactions on kraken api (wet-run) Also fetch from kraken api the history of your past transactions

```
class babao.inputs.ledger.krakenLedgerInput.ABCKrakenLedgerInput (log_to_file=True,  
                                                                un-  
                                                                used_temp=False)  
  
Bases:      babao.inputs.ledger.ledgerInputBase.ABCLedgerInput,  babao.inputs.  
krakenInputBase.ABCKrakenInput
```

Base class for any kraken ledger

buy (*ledger, volume_spent, price, timestamp=None*)

Buy with the current ledger asset the asset of the given 'ledger'

(If the current ledger is a quote, this is a buy) 'volume_spent' quantity spent (including fees)

deposit (*ledger, volume, timestamp=None*)

Deposit from the current ledger to the given 'ledger'

fetch ()

Return a time-serie DataFrame fetched from the internets

This data will be stored on database for later use (and eventual resampling). Data can be continuous. Index must be nanosecond timestamps.

sell (*ledger, volume_spent, price, timestamp=None*)

Buy with the asset of the given 'ledger' the current ledger asset

(If the current ledger is a quote, this is a sell) 'volume_spent' quantity spent (including fees)

withdraw (*ledger, volume, timestamp=None*)

Withdraw from the current ledger to the given 'ledger'

babao.inputs.ledger.ledgerInputBase module

Handle logging in database all our transactions

TODO: It is really not obvious how you're gonna link the transaction across various ledgers... kraken doesn't give you anything else than an "order id", so you still have to iterate over all entries in all ledgers to find the matching one :/ The good new is, I don't know when we'll need that! Anyway, I'll leave an empty column "product", which reference another ledger; this could be used for later indexing?

```

class babao.inputs.ledger.ledgerInputBase.ABCLedgerInput
    Bases: babao.inputs.inputBase.ABCInput

    Base class for any ledger

    asset
        Override this method with the desired CryptoEnum / QuoteEnum ex: self.asset = CryptoEnum.XBT

    buy (ledger, volume_spent, price, timestamp=None)
        Buy with the current ledger asset the asset of the given 'ledger'

        (If the current ledger is a quote, this is a buy) 'volume_spent' quantity spent (including fees)

    deposit (ledger, volume, timestamp=None)
        Deposit from the current ledger to the given 'ledger'

    fillMissing (resampled_data)
        Fill missing values (np.nan/np.inf) in 'resampled_data'

    raw_columns = ['volume', 'balance', 'fee', 'refid', 'type', 'product']

    resampled_columns = ['balance']

    sell (ledger, volume_spent, price, timestamp=None)
        Buy with the asset of the given 'ledger' the current ledger asset

        (If the current ledger is a quote, this is a sell) 'volume_spent' quantity spent (including fees)

    withdraw (ledger, volume, timestamp=None)
        Withdraw from the current ledger to the given 'ledger'

```

babao.inputs.ledger.ledgerManager module

Manage all the ledgers and give some utils functions to check balance or buy/sell

```

babao.inputs.ledger.ledgerManager.buy (crypto_enum, volume)
    Buy the given 'volume' of 'crypto_enum'

babao.inputs.ledger.ledgerManager.buyOrSell (action_enum, crypto_enum, volume=None)
    Decide wether to buy or sell (or not) 'volume' the 'crypto_enum' based on the given 'action_enum'

    It will consider the current 'ledger.BALANCE', and eventually update it.

babao.inputs.ledger.ledgerManager.gameOver ()
    Check if you're broke

babao.inputs.ledger.ledgerManager.getBalanceInQuote (crypto_enum)
    Convert to quote the balance of the LEDGERS[crypto_enum]

babao.inputs.ledger.ledgerManager.getGlobalBalanceInQuote ()
    Sum and convert to quote the balance of all your LEDGERS

babao.inputs.ledger.ledgerManager.getLastTx ()
    Return the timestamp of the last transaction in all LEDGERS

babao.inputs.ledger.ledgerManager.initLedgers ()
    Instantiate all the ledgers and corresponding trade-inputs needed by conf.CRYPTOS / conf.QUOTE

babao.inputs.ledger.ledgerManager.sell (crypto_enum, volume)
    Sell the given 'volume' of 'crypto_enum'

```

Module contents

babao.inputs.trades package

Submodules

babao.inputs.trades.krakenTradesInput module

Concrete class for kraken trade inputs

We could have defined all these with the following commentend out snippet, but for explicitness reasons we'll keep them this way. This also allows linter to understand what's going on.

```
class babao.inputs.trades.krakenTradesInput.KrakenTradesBCHEURInput
    Bases: babao.inputs.trades.krakenTradesInputBase.ABCKrakenTradesInput

    Kraken trade input for CHE crypto vs R quote

    crypto = -1
    pair = 'BCHEUR'
    quote = 2

class babao.inputs.trades.krakenTradesInput.KrakenTradesBCHUSDInput
    Bases: babao.inputs.trades.krakenTradesInputBase.ABCKrakenTradesInput

    Kraken trade input for CHU crypto vs D quote

    crypto = -1
    pair = 'BCHUSD'
    quote = 5

class babao.inputs.trades.krakenTradesInput.KrakenTradesDASHEURInput
    Bases: babao.inputs.trades.krakenTradesInputBase.ABCKrakenTradesInput

    Kraken trade input for ASH crypto vs UR quote

    crypto = -2
    pair = 'DASHEUR'
    quote = 2

class babao.inputs.trades.krakenTradesInput.KrakenTradesDASHUSDInput
    Bases: babao.inputs.trades.krakenTradesInputBase.ABCKrakenTradesInput

    Kraken trade input for ASH crypto vs SD quote

    crypto = -2
    pair = 'DASHUSD'
    quote = 5

class babao.inputs.trades.krakenTradesInput.KrakenTradesEOSEURInput
    Bases: babao.inputs.trades.krakenTradesInputBase.ABCKrakenTradesInput

    Kraken trade input for OSE crypto vs R quote

    crypto = -3
    pair = 'EOSEUR'
```

```
quote = 2
class babao.inputs.trades.krakenTradesInput.KrakenTradesEOSUSDInput
    Bases: babao.inputs.trades.krakenTradesInputBase.ABCKrakenTradesInput
    Kraken trade input for OSU crypto vs D quote
    crypto = -3
    pair = 'EOSUSD'
    quote = 5
class babao.inputs.trades.krakenTradesInput.KrakenTradesGNOEURInput
    Bases: babao.inputs.trades.krakenTradesInputBase.ABCKrakenTradesInput
    Kraken trade input for NOE crypto vs R quote
    crypto = -4
    pair = 'GNOEUR'
    quote = 2
class babao.inputs.trades.krakenTradesInput.KrakenTradesGNOUSDInput
    Bases: babao.inputs.trades.krakenTradesInputBase.ABCKrakenTradesInput
    Kraken trade input for NOU crypto vs D quote
    crypto = -4
    pair = 'GNOUSD'
    quote = 5
class babao.inputs.trades.krakenTradesInput.KrakenTradesXETCZCADInput
    Bases: babao.inputs.trades.krakenTradesInputBase.ABCKrakenTradesInput
    Kraken trade input for ETC crypto vs CAD quote
    crypto = -5
    pair = 'XETCZCAD'
    quote = 1
class babao.inputs.trades.krakenTradesInput.KrakenTradesXETCZEURInput
    Bases: babao.inputs.trades.krakenTradesInputBase.ABCKrakenTradesInput
    Kraken trade input for ETC crypto vs EUR quote
    crypto = -5
    pair = 'XETCZEUR'
    quote = 2
class babao.inputs.trades.krakenTradesInput.KrakenTradesXETCZGBPInput
    Bases: babao.inputs.trades.krakenTradesInputBase.ABCKrakenTradesInput
    Kraken trade input for ETC crypto vs GBP quote
    crypto = -5
    pair = 'XETCZGBP'
    quote = 3
```

```
class babao.inputs.trades.krakenTradesInput.KrakenTradesXETCZJPYInput
    Bases: babao.inputs.trades.krakenTradesInputBase.ABCKrakenTradesInput
    Kraken trade input for ETC crypto vs JPY quote
    crypto = -5
    pair = 'XETCZJPY'
    quote = 4

class babao.inputs.trades.krakenTradesInput.KrakenTradesXETCZUSDInput
    Bases: babao.inputs.trades.krakenTradesInputBase.ABCKrakenTradesInput
    Kraken trade input for ETC crypto vs USD quote
    crypto = -5
    pair = 'XETCZUSD'
    quote = 5

class babao.inputs.trades.krakenTradesInput.KrakenTradesXETHZCADInput
    Bases: babao.inputs.trades.krakenTradesInputBase.ABCKrakenTradesInput
    Kraken trade input for ETH crypto vs CAD quote
    crypto = -6
    pair = 'XETHZCAD'
    quote = 1

class babao.inputs.trades.krakenTradesInput.KrakenTradesXETHZEURInput
    Bases: babao.inputs.trades.krakenTradesInputBase.ABCKrakenTradesInput
    Kraken trade input for ETH crypto vs EUR quote
    crypto = -6
    pair = 'XETHZEUR'
    quote = 2

class babao.inputs.trades.krakenTradesInput.KrakenTradesXETHZGBPInput
    Bases: babao.inputs.trades.krakenTradesInputBase.ABCKrakenTradesInput
    Kraken trade input for ETH crypto vs GBP quote
    crypto = -6
    pair = 'XETHZGBP'
    quote = 3

class babao.inputs.trades.krakenTradesInput.KrakenTradesXETHZJPYInput
    Bases: babao.inputs.trades.krakenTradesInputBase.ABCKrakenTradesInput
    Kraken trade input for ETH crypto vs JPY quote
    crypto = -6
    pair = 'XETHZJPY'
    quote = 4
```

```
class babao.inputs.trades.krakenTradesInput.KrakenTradesXETHZUSDInput
    Bases: babao.inputs.trades.krakenTradesInputBase.ABCKrakenTradesInput
    Kraken trade input for ETH crypto vs USD quote
    crypto = -6
    pair = 'XETHZUSD'
    quote = 5

class babao.inputs.trades.krakenTradesInput.KrakenTradesXLTCZCADInput
    Bases: babao.inputs.trades.krakenTradesInputBase.ABCKrakenTradesInput
    Kraken trade input for LTC crypto vs CAD quote
    crypto = -7
    pair = 'XLTCZCAD'
    quote = 1

class babao.inputs.trades.krakenTradesInput.KrakenTradesXLTCZEURInput
    Bases: babao.inputs.trades.krakenTradesInputBase.ABCKrakenTradesInput
    Kraken trade input for LTC crypto vs EUR quote
    crypto = -7
    pair = 'XLTCZEUR'
    quote = 2

class babao.inputs.trades.krakenTradesInput.KrakenTradesXLTCZGBPInput
    Bases: babao.inputs.trades.krakenTradesInputBase.ABCKrakenTradesInput
    Kraken trade input for LTC crypto vs GBP quote
    crypto = -7
    pair = 'XLTCZGBP'
    quote = 3

class babao.inputs.trades.krakenTradesInput.KrakenTradesXLTCZJPYInput
    Bases: babao.inputs.trades.krakenTradesInputBase.ABCKrakenTradesInput
    Kraken trade input for LTC crypto vs JPY quote
    crypto = -7
    pair = 'XLTCZJPY'
    quote = 4

class babao.inputs.trades.krakenTradesInput.KrakenTradesXLTCZUSDInput
    Bases: babao.inputs.trades.krakenTradesInputBase.ABCKrakenTradesInput
    Kraken trade input for LTC crypto vs USD quote
    crypto = -7
    pair = 'XLTCZUSD'
    quote = 5
```

```
class babao.inputs.trades.krakenTradesInput.KrakenTradesXREPZCADInput
    Bases: babao.inputs.trades.krakenTradesInputBase.ABCKrakenTradesInput
    Kraken trade input for REP crypto vs CAD quote
    crypto = -8
    pair = 'XREPZCAD'
    quote = 1

class babao.inputs.trades.krakenTradesInput.KrakenTradesXREPZEURInput
    Bases: babao.inputs.trades.krakenTradesInputBase.ABCKrakenTradesInput
    Kraken trade input for REP crypto vs EUR quote
    crypto = -8
    pair = 'XREPZEUR'
    quote = 2

class babao.inputs.trades.krakenTradesInput.KrakenTradesXREPZGBPInput
    Bases: babao.inputs.trades.krakenTradesInputBase.ABCKrakenTradesInput
    Kraken trade input for REP crypto vs GBP quote
    crypto = -8
    pair = 'XREPZGBP'
    quote = 3

class babao.inputs.trades.krakenTradesInput.KrakenTradesXREPZJPYInput
    Bases: babao.inputs.trades.krakenTradesInputBase.ABCKrakenTradesInput
    Kraken trade input for REP crypto vs JPY quote
    crypto = -8
    pair = 'XREPZJPY'
    quote = 4

class babao.inputs.trades.krakenTradesInput.KrakenTradesXREPZUSDInput
    Bases: babao.inputs.trades.krakenTradesInputBase.ABCKrakenTradesInput
    Kraken trade input for REP crypto vs USD quote
    crypto = -8
    pair = 'XREPZUSD'
    quote = 5

class babao.inputs.trades.krakenTradesInput.KrakenTradesXXBTZCADInput
    Bases: babao.inputs.trades.krakenTradesInputBase.ABCKrakenTradesInput
    Kraken trade input for XBT crypto vs CAD quote
    crypto = -9
    pair = 'XXBTZCAD'
    quote = 1
```

```
class babao.inputs.trades.krakenTradesInput.KrakenTradesXXBTZEURInput
    Bases: babao.inputs.trades.krakenTradesInputBase.ABCKrakenTradesInput
    Kraken trade input for XBT crypto vs EUR quote
    crypto = -9
    pair = 'XXBTZEUR'
    quote = 2

class babao.inputs.trades.krakenTradesInput.KrakenTradesXXBTZGBPInput
    Bases: babao.inputs.trades.krakenTradesInputBase.ABCKrakenTradesInput
    Kraken trade input for XBT crypto vs GBP quote
    crypto = -9
    pair = 'XXBTZGBP'
    quote = 3

class babao.inputs.trades.krakenTradesInput.KrakenTradesXXBTZJPYInput
    Bases: babao.inputs.trades.krakenTradesInputBase.ABCKrakenTradesInput
    Kraken trade input for XBT crypto vs JPY quote
    crypto = -9
    pair = 'XXBTZJPY'
    quote = 4

class babao.inputs.trades.krakenTradesInput.KrakenTradesXXBTZUSDInput
    Bases: babao.inputs.trades.krakenTradesInputBase.ABCKrakenTradesInput
    Kraken trade input for XBT crypto vs USD quote
    crypto = -9
    pair = 'XXBTZUSD'
    quote = 5

class babao.inputs.trades.krakenTradesInput.KrakenTradesXXLMZCADInput
    Bases: babao.inputs.trades.krakenTradesInputBase.ABCKrakenTradesInput
    Kraken trade input for XLM crypto vs CAD quote
    crypto = -10
    pair = 'XXLMZCAD'
    quote = 1

class babao.inputs.trades.krakenTradesInput.KrakenTradesXXLMZEURInput
    Bases: babao.inputs.trades.krakenTradesInputBase.ABCKrakenTradesInput
    Kraken trade input for XLM crypto vs EUR quote
    crypto = -10
    pair = 'XXLMZEUR'
    quote = 2
```

```
class babao.inputs.trades.krakenTradesInput.KrakenTradesXXLMZGBPInput
    Bases: babao.inputs.trades.krakenTradesInputBase.ABCKrakenTradesInput
    Kraken trade input for XLM crypto vs GBP quote
    crypto = -10
    pair = 'XXLMZGBP'
    quote = 3

class babao.inputs.trades.krakenTradesInput.KrakenTradesXXLMZJPYInput
    Bases: babao.inputs.trades.krakenTradesInputBase.ABCKrakenTradesInput
    Kraken trade input for XLM crypto vs JPY quote
    crypto = -10
    pair = 'XXLMZJPY'
    quote = 4

class babao.inputs.trades.krakenTradesInput.KrakenTradesXXLMZUSDInput
    Bases: babao.inputs.trades.krakenTradesInputBase.ABCKrakenTradesInput
    Kraken trade input for XLM crypto vs USD quote
    crypto = -10
    pair = 'XXLMZUSD'
    quote = 5

class babao.inputs.trades.krakenTradesInput.KrakenTradesXXMRZCADInput
    Bases: babao.inputs.trades.krakenTradesInputBase.ABCKrakenTradesInput
    Kraken trade input for XMR crypto vs CAD quote
    crypto = -11
    pair = 'XXMRZCAD'
    quote = 1

class babao.inputs.trades.krakenTradesInput.KrakenTradesXXMRZEURInput
    Bases: babao.inputs.trades.krakenTradesInputBase.ABCKrakenTradesInput
    Kraken trade input for XMR crypto vs EUR quote
    crypto = -11
    pair = 'XXMRZEUR'
    quote = 2

class babao.inputs.trades.krakenTradesInput.KrakenTradesXXMRZGBPInput
    Bases: babao.inputs.trades.krakenTradesInputBase.ABCKrakenTradesInput
    Kraken trade input for XMR crypto vs GBP quote
    crypto = -11
    pair = 'XXMRZGBP'
    quote = 3
```

```
class babao.inputs.trades.krakenTradesInput.KrakenTradesXXMRZJPYInput
    Bases: babao.inputs.trades.krakenTradesInputBase.ABCKrakenTradesInput
    Kraken trade input for XMR crypto vs JPY quote
    crypto = -11
    pair = 'XXMRZJPY'
    quote = 4

class babao.inputs.trades.krakenTradesInput.KrakenTradesXXMRZUSDInput
    Bases: babao.inputs.trades.krakenTradesInputBase.ABCKrakenTradesInput
    Kraken trade input for XMR crypto vs USD quote
    crypto = -11
    pair = 'XXMRZUSD'
    quote = 5

class babao.inputs.trades.krakenTradesInput.KrakenTradesXXRPZCADInput
    Bases: babao.inputs.trades.krakenTradesInputBase.ABCKrakenTradesInput
    Kraken trade input for XRP crypto vs CAD quote
    crypto = -12
    pair = 'XXRPZCAD'
    quote = 1

class babao.inputs.trades.krakenTradesInput.KrakenTradesXXRPZEURInput
    Bases: babao.inputs.trades.krakenTradesInputBase.ABCKrakenTradesInput
    Kraken trade input for XRP crypto vs EUR quote
    crypto = -12
    pair = 'XXRPZEUR'
    quote = 2

class babao.inputs.trades.krakenTradesInput.KrakenTradesXXRPZGBPInput
    Bases: babao.inputs.trades.krakenTradesInputBase.ABCKrakenTradesInput
    Kraken trade input for XRP crypto vs GBP quote
    crypto = -12
    pair = 'XXRPZGBP'
    quote = 3

class babao.inputs.trades.krakenTradesInput.KrakenTradesXXRPZJPYInput
    Bases: babao.inputs.trades.krakenTradesInputBase.ABCKrakenTradesInput
    Kraken trade input for XRP crypto vs JPY quote
    crypto = -12
    pair = 'XXRPZJPY'
    quote = 4
```

```
class babao.inputs.trades.krakenTradesInput.KrakenTradesXXRPZUSDInput
    Bases: babao.inputs.trades.krakenTradesInputBase.ABCKrakenTradesInput
    Kraken trade input for XRP crypto vs USD quote
    crypto = -12
    pair = 'XXRPZUSD'
    quote = 5

class babao.inputs.trades.krakenTradesInput.KrakenTradesXZECZCADInput
    Bases: babao.inputs.trades.krakenTradesInputBase.ABCKrakenTradesInput
    Kraken trade input for ZEC crypto vs CAD quote
    crypto = -13
    pair = 'XZECZCAD'
    quote = 1

class babao.inputs.trades.krakenTradesInput.KrakenTradesXZECZEURInput
    Bases: babao.inputs.trades.krakenTradesInputBase.ABCKrakenTradesInput
    Kraken trade input for ZEC crypto vs EUR quote
    crypto = -13
    pair = 'XZECZEUR'
    quote = 2

class babao.inputs.trades.krakenTradesInput.KrakenTradesXZECZGBPInput
    Bases: babao.inputs.trades.krakenTradesInputBase.ABCKrakenTradesInput
    Kraken trade input for ZEC crypto vs GBP quote
    crypto = -13
    pair = 'XZECZGBP'
    quote = 3

class babao.inputs.trades.krakenTradesInput.KrakenTradesXZECZJPYInput
    Bases: babao.inputs.trades.krakenTradesInputBase.ABCKrakenTradesInput
    Kraken trade input for ZEC crypto vs JPY quote
    crypto = -13
    pair = 'XZECZJPY'
    quote = 4

class babao.inputs.trades.krakenTradesInput.KrakenTradesXZECZUSDInput
    Bases: babao.inputs.trades.krakenTradesInputBase.ABCKrakenTradesInput
    Kraken trade input for ZEC crypto vs USD quote
    crypto = -13
    pair = 'XZECZUSD'
    quote = 5
```

`babao.inputs.trades.krakenTradesInputBase` module

Module containing base class for any kraken trades input

```
class babao.inputs.trades.krakenTradesInputBase.ABCKrakenTradesInput
    Bases: babao.inputs.trades.tradesInputBase.ABCTradesInput, babao.inputs.krakenInputBase.ABCKrakenInput

    Base class for any kraken trades input

    fetch ()
        Return a time-serie DataFrame fetched from the internets

        This data will be stored on database for later use (and eventual resampling). Data can be continuous. Index must be nanosecond timestamps.

    pair
        Override this method with the desired asset pair as string ex: self.pair = "XXBTZEUR"
```

`babao.inputs.trades.tradesInputBase` module

Module containing base class for any trades input

```
class babao.inputs.trades.tradesInputBase.ABCTradesInput
    Bases: babao.inputs.inputBase.ABCInput

    Base class for any trades input

    crypto
        Override this method with the desired CryptoEnum ex: self.crypto = CryptoEnum.XBT

    fetch ()
        Return a time-serie DataFrame fetched from the internets

        This data will be stored on database for later use (and eventual resampling). Data can be continuous. Index must be nanosecond timestamps.

    fillMissing (resampled_data)
        Fill missing values (np.nan/np.inf) in 'resampled_data'

    quote
        Override this method with the desired QuoteEnum ex: self.quote = QuoteEnum.EUR

    raw_columns = ['price', 'volume']

    resampled_columns = ['open', 'high', 'low', 'close', 'vwap', 'volume', 'count']
```

Module contents

1.1.1.2 Submodules

1.1.1.3 `babao.inputs.inputBase` module

Base class for any input

```
class babao.inputs.inputBase.ABCInput
    Bases: abc.ABC

    Base class for any input
```

Your subclass should at least implement: * `fetch` : `self -> DataFrame` * `raw_columns` : `List[str]`

And eventually: (if you want `self.resample` to works) * `_resample` : `self -> DataFrame -> DataFrame` * `fillMissing` : `self -> DataFrame -> DataFrame` * `resampled_columns` : `List[str]`

(cf. specific method doc-string in this class)

cache (*fresh_data=None, since=None, till=None*)

Save some data to cache

If 'fresh_data' is given, append it to cache, otherwise read in database from 'since' to 'till' and cache it

fetch ()

Return a time-serie DataFrame fetched from the internets

This data will be stored on database for later use (and eventual resampling). Data can be continuous. Index must be nanosecond timestamps.

fillMissing (*resampled_data*)

Fill missing values (`np.nan/np.inf`) in 'resampled_data'

raw_columns

The columns names of your raw data (as fetched and stored in database)

read (*since=None, till=None*)

Read data in database or cache from 'since' to 'till'

refreshCache ()

Make sure the cache is up to date

resample (*raw_data*)

Return the DataFrame 'raw_data' as a continuous time-serie

This is a wrapper around `_resample` and `fillMissing`

resampled_columns

The columns names of your resampled data (from raw data)

updateCurrentRow (*current_row=None, timestamp=None*)

Update the property `self.current_row`, useful for time travel

write (*raw_data*)

Write the given `raw_data` to the database, and cache it if needed

`babao.inputs.inputBase.resampleSerie` (*s*)

Call `Serie.resample` on `s` with preset parameters (the serie's index must be datetime)

1.1.1.4 babao.inputs.inputManager module

Common interface to inputs to call methods on all of them at once

`babao.inputs.inputManager.fetchInputs` ()

Fetch all the INPUTS in a pool thread

The raw data resulting is then wrote to database.

`babao.inputs.inputManager.readInputs` (*input_list: Optional[List[babao.inputs.inputBase.ABCInput]] = None, since=None*)

Read all INPUTS from 'since' and resample them with matching time base

The return is one dataframe containing all concatenated columns (so they will be renamed with the input name as prefix)

`babao.inputs.inputManager.timeTravel(timestamp)`
Travel to the specified timestamp, for simulation purposes

This module define the base class and methods for kraken inputs

1.1.1.6 Module contents

1.1.2.1 Subpackages

Submodules

The idea of that model is to find local extrema, then classify them as minimum/nop/maximum (-1/0/1) using a knn classifier (sklearn)

1.1. Subpackages

train (*since*)
Train the model with data starting from 'since' timestamp
Return the score of model.

`babao.models.tree.macdModel` module

Simple macd based model, with a very elegant algorithm (aka: brute-force)

```
class babao.models.tree.macdModel.MacdModel
    Bases: babao.models.modelBase.ABCModel

    Simple macd based model

    dependencies_class = [<class 'babao.inputs.trades.krakenTradesInput.KrakenTradesXXBTZE
```

load ()
Load the model from self.model_file

need_training = **True**

plot (*since*)
Plot the model predictions from 'since' timestamp

predict (*since*)
Return a dataframe of prediction starting from 'since' timestamp

save ()
Save the model to self.model_file

train (*since*)
Train the model with data starting from 'since' timestamp
Return the score of model.

`babao.models.tree.tendencyModel` module

Module contents

1.1.2.2 Submodules

1.1.2.3 `babao.models.modelBase` module

Base class for any model

```
class babao.models.modelBase.ABCModel
    Bases: abc.ABC

    Base class for any model

    dependencies_class
        List of models or inputs needed by the current model
        These should be class, not instances!

    load ()
        Load the model from self.model_file

    need_training
        Specify if the current model need to be trained
```

plot (*since*)
 Plot the model predictions from 'since' timestamp

predict (*since*)
 Return a dataframe of prediction starting from 'since' timestamp

save ()
 Save the model to self.model_file

train (*since*)
 Train the model with data starting from 'since' timestamp
 Return the score of model.

`babao.models.modelBase.addLookbacks(df, look_back)`
 Add lookback(s) (shifted columns) to each df columns

`babao.models.modelBase.addLookbacks3d(arr, look_back)`
 Add lookback(s) (shifted columns) to each df columns Reshape the features to be keras-proof (3d)

`babao.models.modelBase.getVerbose()` → int
 Transform our verbose level to match keras one

`babao.models.modelBase.reshape(arr)`
 Reshape the features to be keras-proof

1.1.2.4 babao.models.modelManager module

The idea here is to give a common interface to all the models so you can use these wrappers to call all of them at once.

`babao.models.modelManager.plotModels(since)`
 Plot all models

`babao.models.modelManager.predictModelsMaybeTrade(since)`
 Call predict on the root model, then eventually trade based on the production

`babao.models.modelManager.trainModels(since)`
 Train all models and save the awesome result

1.1.2.5 babao.models.rootModel module

Root Model, base of the models tree

```
class babao.models.rootModel.RootModel
    Bases: babao.models.modelBase.ABCModel
    Root Model, base of the models tree
    Not modeling much, but handle the call of the dependencies predictions
    dependencies_class = [<class 'babao.models.tree.extremaModel.ExtremaModel'>]
    load ()
        Load the model from self.model_file
    need_training = False
    plot (since)
        Plot the model predictions from 'since' timestamp
```

predict (*since*)

Call predict on the dependencies, then somehow merge the results

save ()

Save the model to self.model_file

train (*since*)

Train the model with data starting from 'since' timestamp

Return the score of model.

1.1.2.6 Module contents

1.1.3 babao.utils package

1.1.3.1 Submodules

1.1.3.2 babao.utils.date module

Some utils functions for date handling and time traveling

class babao.utils.date.**TimeTraveler**

Bases: object

Class handling time travel tricks

getTime (*force=False*)

Return the current time in nanoseconds

Used for time traveling purpose, so this might be a date in the past matching the current simulation state, unless 'force' is set to True.

nowMinus (*years=0, weeks=0, days=0, hours=0, minutes=0*)

Return the current timestamp (nanoseconds) minus the given parameters

This will take into account time traveling tricks.

setTime (*now*)

Set time to the given 'now' nanoseconds

Used for time traveling purpose

babao.utils.date.nanoToSec (*nano*)

Convert nanoseconds to seconds

babao.utils.date.secToNano (*sec*)

Convert seconds to nanoseconds Just trying to avoid float rounding...

babao.utils.date.toDatetime (*df*)

Convert the index of the given dataframe to datetime

Also works directly on a dataframe index.

babao.utils.date.toStr (*t*)

Return the string representation of timestamp

't' can be a nanoseconds timestamp, or a panda datetime object.

babao.utils.date.toTimestamp (*df*)

Convert the index of the given dataframe to nanoseconds

Also works directly on a dataframe index.

1.1.3.3 babao.utils.enum module

Enums describing positions and assets.

```
class babao.utils.enum.ActionEnum
```

Bases: enum.Enum

Enum describing a transaction action

BUY = 1

DEPOSIT = 2

FEE = -3

HODL = 0

SELL = -1

WITHDRAW = -2

```
class babao.utils.enum.CryptoEnum
```

Bases: enum.Enum

Enum describing a crypto asset

BCH = -1

DASH = -2

EOS = -3

ETC = -5

ETH = -6

GNO = -4

LTC = -7

REP = -8

XBT = -9

XTM = -10

XMR = -11

XRP = -12

ZEC = -13

```
class babao.utils.enum.QuoteEnum
```

Bases: enum.Enum

Enum describing a quote asset

CAD = 1

EUR = 2

GBP = 3

JPY = 4

USD = 5

```
class babao.utils.enum.TradeEnum
```

```
    Bases: enum.Enum
```

```
    Enum describing a transaction on a given asset
```

```
    BUY_BCH = 1
```

```
    BUY_DASH = 2
```

```
    BUY_EOS = 3
```

```
    BUY_ETC = 5
```

```
    BUY_ETH = 6
```

```
    BUY_GNO = 4
```

```
    BUY_LTC = 7
```

```
    BUY_REP = 8
```

```
    BUY_XBT = 9
```

```
    BUY_XLM = 10
```

```
    BUY_XMR = 11
```

```
    BUY_XRP = 12
```

```
    BUY_ZEC = 13
```

```
    HODL = 0
```

```
    SELL_BCH = -1
```

```
    SELL_DASH = -2
```

```
    SELL_EOS = -3
```

```
    SELL_ETC = -5
```

```
    SELL_ETH = -6
```

```
    SELL_GNO = -4
```

```
    SELL_LTC = -7
```

```
    SELL_REP = -8
```

```
    SELL_XBT = -9
```

```
    SELL_XLM = -10
```

```
    SELL_XMR = -11
```

```
    SELL_XRP = -12
```

```
    SELL_ZEC = -13
```

```
babao.utils.enum.cryptoAndActionToTrade (crypto_enum_val, action_enum_val)
```

```
    Convert a crypto enum and an action enum to the matching trade enum
```

```
babao.utils.enum.floatToTrade (f)
```

```
    Round a float value to the nearest trade enum
```

```
babao.utils.enum.tradeToAction (trade_enum_val)
```

```
    Extract an action enum from a trade enum
```

```
babao.utils.enum.tradeToCrypto (trade_enum_val)
```

```
    Extract a crypto enum from a trade enum
```

1.1.3.4 babao.utils.file module

Some utils functions for hdf handling

```
babao.utils.file.closeStore()
    Close the hdf database

babao.utils.file.delete(frame)
    Remove the given 'frame' entry (key) from the hdf database

    Thread Safe!

babao.utils.file.getLastRows(frame, nrows)
    Return 'nrows' rows from a 'frame' (key) in the hdf database as a DataFrame

babao.utils.file.initStore(filename)
    Open the hdf database from 'filename'

babao.utils.file.maintenance()
    Maintenance routine on the hdf database

    Create table index for each table, and make sure everything is sorted.

babao.utils.file.read(frame, where=None)
    Read a 'frame' (key) from the hdf database

    'where' can be use to specify search criteria. Thread Safe!

babao.utils.file.setLock(lock)
    Store the given 'lock' object for later use in database processing

babao.utils.file.write(frame, df)
    Append the given 'df' dataframe to the 'frame' entry (key) in the hdf database

    Thread Safe!
```

1.1.3.5 babao.utils.indicators module

Various indicators which can be added to any serie

[//www.quantinsti.com/blog/build-technical-indicators-in-python](http://www.quantinsti.com/blog/build-technical-indicators-in-python)

```
babao.utils.indicators.ewma(serie, look_back_delay)
    Exponentially-weighted Moving Average

babao.utils.indicators.get(df, columns)
    Add indicators specified by columns to the given df

    Expected 'columns' format: ["sma_vwap_42", "ewma_volume_12"]

babao.utils.indicators.macd(serie, fast_delay, slow_delay, signal_delay, full=False)
    Moving Average Convergence/Divergence Oscillator

babao.utils.indicators.ppo(serie, fast_delay, slow_delay, signal_delay, full=False)
    Percentage Price Oscillator

    Same as macd, but we do (a-b)/b instead of a-b, so the final value does not depend on input scale (it's a percentage!)

babao.utils.indicators.sma(serie, look_back_delay)
    Simple Moving Average
```

1.1.3.6 babao.utils.lock module

Some utils functions for lock file handling

`babao.utils.lock.isLocked(lockfile)`
Check if the 'lockfile' exists

`babao.utils.lock.tryLock(lockfile)`
Create the given 'lockfile'
Return false if it already exists

`babao.utils.lock.tryUnlock(lockfile)`
Remove the given 'lockfile'
Return false if it doesn't exist

1.1.3.7 babao.utils.log module

Some utils functions for logging

`babao.utils.log.debug(*args)`
Log a debug (magenta)

`babao.utils.log.error(*args)`
Log an error (red)

`babao.utils.log.info(*args)`
Log a simple message (blue)

`babao.utils.log.initLogLevel(verbose, quiet)`
Initialize log level based on verbose flag

`babao.utils.log.setLock(lock)`
Store the given 'lock' object for later use in logging

`babao.utils.log.warning(*args)`
Log a warning (yellow)

1.1.3.8 babao.utils.scale module

Scaler

class `babao.utils.scale.Scaler`

Bases: `object`

Basic min/max scaler

fit (*arr*)
Init scaler

scale (*arr*)
Scale features before train/predict

scaleFit (*arr*)
Scale n Fit

unscale (*arr*)
Unscale features after train/predict

1.1.3.9 babao.utils.signal module

Signal handling

```
babao.utils.signal.catchSignal()  
    Catch signal INT/TERM, so we won't exit while playing with data files
```

```
babao.utils.signal.restoreSignal()  
    Restore the previous signal handler
```

1.1.3.10 Module contents

1.2 Submodules

1.3 babao.arg module

Argv parsing

```
babao.arg.parseArgv(args)  
    Parse argv 'args'
```

1.4 babao.babao module

Module that contains the command line app.

Why does this file exist, and why not put this in `__main__`?

You might be tempted to import things from `__main__` later, but that will cause problems: the code will get executed twice:

- When you run `python -mbabao` python will execute `__main__.py` as a script. That means there won't be any `babao.__main__` in `sys.modules`.
- When you import `__main__` it will get executed again (as a module) because there's no `babao.__main__` in `sys.modules`.

Also see (1) from <http://click.pocoo.org/5/setuptools/#setuptools-integration>

```
babao.babao.main(args=None)  
    Babao entry point
```

1.5 babao.commands module

Commands launched by `parseArgv`

```
babao.commands.backtest(args)  
    Just a naive backtester
```

It will call the trained strategies on each test data point

```
babao.commands.dryRun(UNUSED_ARGS)  
    Real-time bot simulation
```

```
babao.commands.fetch(UNUSED_ARGS)  
    Fetch raw trade data since the beginning of times
```

`babao.commands.train(args)`

Train the various (awesome) algorithms

`babao.commands.wetRun(UNUSED_ARGS)`

Dummy

1.6 babao.config module

Here we'll handle the config file and the various file/dir paths

`babao.config.readConfigFile(cmd_name='dry-run')`

Read config file and initialize global config variables

1.7 babao.graph module

1.8 Module contents

CHAPTER 2

Indices and tables

- `genindex`
- `modindex`
- `search`

b

- [babao](#), 24
- [babao.arg](#), 23
- [babao.babao](#), 23
- [babao.commands](#), 23
- [babao.config](#), 24
- [babao.inputs](#), 15
 - [babao.inputs.inputBase](#), 13
 - [babao.inputs.inputManager](#), 14
 - [babao.inputs.krakenInputBase](#), 15
 - [babao.inputs.ledger](#), 4
 - [babao.inputs.ledger.fakeLedgerInput](#), 1
 - [babao.inputs.ledger.krakenLedgerInput](#), 2
 - [babao.inputs.ledger.ledgerInputBase](#), 2
 - [babao.inputs.ledger.ledgerManager](#), 3
- [babao.inputs.trades](#), 13
 - [babao.inputs.trades.krakenTradesInput](#), 4
 - [babao.inputs.trades.krakenTradesInputBase](#), 13
- [babao.inputs.trades.tradesInputBase](#), 13
- [babao.models](#), 18
 - [babao.models.modelBase](#), 16
 - [babao.models.modelManager](#), 17
 - [babao.models.rootModel](#), 17
 - [babao.models.tree](#), 16
 - [babao.models.tree.extremaModel](#), 15
 - [babao.models.tree.macdModel](#), 16
- [babao.utils](#), 23
 - [babao.utils.date](#), 18
 - [babao.utils.enum](#), 19
 - [babao.utils.file](#), 21
 - [babao.utils.indicators](#), 21
 - [babao.utils.lock](#), 22
 - [babao.utils.log](#), 22
 - [babao.utils.scale](#), 22
 - [babao.utils.signal](#), 23

A

- ABCFakeLedgerInput (class *babao.inputs.ledger.fakeLedgerInput*), 1
 - ABCInput (class in *babao.inputs.inputBase*), 13
 - ABCKrakenInput (class *babao.inputs.krakenInputBase*), 15
 - ABCKrakenLedgerInput (class *babao.inputs.ledger.krakenLedgerInput*), 2
 - ABCKrakenTradesInput (class *babao.inputs.trades.krakenTradesInputBase*), 13
 - ABCLedgerInput (class *babao.inputs.ledger.ledgerInputBase*), 2
 - ABCModel (class in *babao.models.modelBase*), 16
 - ABCTradesInput (class *babao.inputs.trades.tradesInputBase*), 13
 - ActionEnum (class in *babao.utils.enum*), 19
 - addLookbacks () (in *babao.models.modelBase*), 17
 - addLookbacks3d () (in *babao.models.modelBase*), 17
 - asset (*babao.inputs.ledger.fakeLedgerInput.ABCFakeLedgerInput* attribute), 1
 - asset (*babao.inputs.ledger.ledgerInputBase.ABCLedgerInput* attribute), 3
- B**
- babao* (module), 24
 - babao.arg* (module), 23
 - babao.babao* (module), 23
 - babao.commands* (module), 23
 - babao.config* (module), 24
 - babao.inputs* (module), 15
 - babao.inputs.inputBase* (module), 13
 - babao.inputs.inputManager* (module), 14
 - babao.inputs.krakenInputBase* (module), 15
 - babao.inputs.ledger* (module), 4
 - babao.inputs.ledger.fakeLedgerInput* (module), 1
 - babao.inputs.ledger.krakenLedgerInput* (module), 2
 - babao.inputs.ledger.ledgerInputBase* (module), 13
 - babao.inputs.ledger.ledgerManager* (module), 3
 - babao.inputs.trades* (module), 13
 - babao.inputs.trades.krakenTradesInput* (module), 4
 - babao.inputs.trades.krakenTradesInputBase* (module), 13
 - babao.inputs.trades.tradesInputBase* (module), 13
 - babao.models* (module), 18
 - babao.models.modelBase* (module), 16
 - babao.models.modelManager* (module), 17
 - babao.models.rootModel* (module), 17
 - babao.models.tree* (module), 16
 - babao.models.tree.extremaModel* (module), 15
 - babao.models.tree.macdModel* (module), 16
 - babao.utils* (module), 23
 - babao.utils.date* (module), 18
 - babao.utils.enum* (module), 19
 - babao.utils.file* (module), 21
 - babao.utils.indicators* (module), 21
 - babao.utils.lock* (module), 22
 - babao.utils.log* (module), 22
 - babao.utils.scale* (module), 22
 - babao.utils.signal* (module), 23
 - backtest* () (in module *babao.commands*), 23
 - BCH (*babao.utils.enum.CryptoEnum* attribute), 19
 - BUY (*babao.utils.enum.ActionEnum* attribute), 19
 - buy () (*babao.inputs.ledger.fakeLedgerInput.ABCFakeLedgerInput* method), 1
 - buy () (*babao.inputs.ledger.krakenLedgerInput.ABCKrakenLedgerInput* method), 2
 - buy () (*babao.inputs.ledger.ledgerInputBase.ABCLedgerInput* method), 3

```
buy () (in module babao.inputs.ledger.ledgerManager), 3
BUY_BCH (babao.utils.enum.TradeEnum attribute), 20
BUY_DASH (babao.utils.enum.TradeEnum attribute), 20
BUY_EOS (babao.utils.enum.TradeEnum attribute), 20
BUY_ETC (babao.utils.enum.TradeEnum attribute), 20
BUY_ETH (babao.utils.enum.TradeEnum attribute), 20
BUY_GNO (babao.utils.enum.TradeEnum attribute), 20
BUY_LTC (babao.utils.enum.TradeEnum attribute), 20
BUY_REP (babao.utils.enum.TradeEnum attribute), 20
BUY_XBT (babao.utils.enum.TradeEnum attribute), 20
BUY_XLM (babao.utils.enum.TradeEnum attribute), 20
BUY_XMR (babao.utils.enum.TradeEnum attribute), 20
BUY_XRP (babao.utils.enum.TradeEnum attribute), 20
BUY_ZEC (babao.utils.enum.TradeEnum attribute), 20
buyOrSell () (in module babao.inputs.ledger.ledgerManager), 3
```

C

cache () (*babao.inputs.inputBase.ABCInput method*), 14

CAD (*babao.utils.enum.QuoteEnum attribute*), 19

catchSignal () (*in module babao.utils.signal*), 23

closeStore () (*in module babao.utils.file*), 21

crypto (*babao.inputs.trades.krakenTradesInput.Kraken attribute*), 4

crypto (*babao.inputs.trades.krakenTradesInput.Kraken attribute*), 4

crypto (*babao.inputs.trades.krakenTradesInput.Kraken attribute*), 4

crypto (*babao.inputs.trades.krakenTradesInput.Kraken attribute*), 4

crypto (*babao.inputs.trades.krakenTradesInput.Kraken attribute*), 4

crypto (*babao.inputs.trades.krakenTradesInput.Kraken attribute*), 5

crypto (*babao.inputs.trades.krakenTradesInput.Kraken attribute*), 5

crypto (*babao.inputs.trades.krakenTradesInput.Kraken attribute*), 5

crypto (*babao.inputs.trades.krakenTradesInput.Kraken attribute*), 5

crypto (*babao.inputs.trades.krakenTradesInput.Kraken attribute*), 5

crypto (*babao.inputs.trades.krakenTradesInput.Kraken attribute*), 5

crypto (*babao.inputs.trades.krakenTradesInput.Kraken attribute*), 6

crypto (*babao.inputs.trades.krakenTradesInput.Kraken attribute*), 6

crypto (*babao.inputs.trades.krakenTradesInput.Kraken attribute*), 6

crypto (*babao.inputs.trades.krakenTradesInput.Kraken attribute*), 6

```
crypto (babao.inputs.trades.krakenTradesInput.KrakenTradesXETHZGBI
attribute), 6
crypto (babao.inputs.trades.krakenTradesInput.KrakenTradesXETHZJPY
attribute), 6
crypto (babao.inputs.trades.krakenTradesInput.KrakenTradesXETHZUSD
attribute), 7
crypto (babao.inputs.trades.krakenTradesInput.KrakenTradesXLTCZCAD
attribute), 7
crypto (babao.inputs.trades.krakenTradesInput.KrakenTradesXLTCZEUR
attribute), 7
crypto (babao.inputs.trades.krakenTradesInput.KrakenTradesXLTCZGBP
attribute), 7
crypto (babao.inputs.trades.krakenTradesInput.KrakenTradesXLTCZJPY
attribute), 7
crypto (babao.inputs.trades.krakenTradesInput.KrakenTradesXLTCZUSD
attribute), 7
crypto (babao.inputs.trades.krakenTradesInput.KrakenTradesXREPZCAD
attribute), 8
crypto (babao.inputs.trades.krakenTradesInput.KrakenTradesXREPZEUR
attribute), 8
crypto (babao.inputs.trades.krakenTradesInput.KrakenTradesXREPZGBP
attribute), 8
crypto (babao.inputs.trades.krakenTradesInput.KrakenTradesXREPZJPY
attribute), 8
crypto (babao.inputs.trades.krakenTradesInput.KrakenTradesXREPZUSD
attribute), 8
crypto (babao.inputs.trades.krakenTradesInput.KrakenTradesXXBTZCAD
attribute), 8
crypto (babao.inputs.trades.krakenTradesInput.KrakenTradesXXBTZEUR
attribute), 9
crypto (babao.inputs.trades.krakenTradesInput.KrakenTradesXXBTZGBP
attribute), 9
crypto (babao.inputs.trades.krakenTradesInput.KrakenTradesXXBTZJPY
attribute), 9
crypto (babao.inputs.trades.krakenTradesInput.KrakenTradesXXBTZUSD
attribute), 9
crypto (babao.inputs.trades.krakenTradesInput.KrakenTradesXXLMZCAD
attribute), 9
crypto (babao.inputs.trades.krakenTradesInput.KrakenTradesXXLMZEUR
attribute), 9
crypto (babao.inputs.trades.krakenTradesInput.KrakenTradesXXLMZGBP
attribute), 10
crypto (babao.inputs.trades.krakenTradesInput.KrakenTradesXXLMZJPY
attribute), 10
crypto (babao.inputs.trades.krakenTradesInput.KrakenTradesXXLMZUSD
attribute), 10
crypto (babao.inputs.trades.krakenTradesInput.KrakenTradesXXMRZCAD
attribute), 10
crypto (babao.inputs.trades.krakenTradesInput.KrakenTradesXXMRZEUR
attribute), 10
crypto (babao.inputs.trades.krakenTradesInput.KrakenTradesXXMRZGBP
attribute), 10
crypto (babao.inputs.trades.krakenTradesInput.KrakenTradesXXMRZJPY
attribute), 11
```

crypto (babao.inputs.trades.krakenTradesInput.KrakenTradesXXMRZUSDInput attribute), 11
 crypto (babao.inputs.trades.krakenTradesInput.KrakenTradesXXRPZCADInput attribute), 11
 crypto (babao.inputs.trades.krakenTradesInput.KrakenTradesXXRPZEURInput attribute), 11
 crypto (babao.inputs.trades.krakenTradesInput.KrakenTradesXXRPZGBPInput attribute), 11
 crypto (babao.inputs.trades.krakenTradesInput.KrakenTradesXXRPZJPYInput attribute), 11
 crypto (babao.inputs.trades.krakenTradesInput.KrakenTradesXXRPZUSDInput attribute), 12
 crypto (babao.inputs.trades.krakenTradesInput.KrakenTradesXZECZCADInput attribute), 12
 crypto (babao.inputs.trades.krakenTradesInput.KrakenTradesXZECZEURInput attribute), 12
 crypto (babao.inputs.trades.krakenTradesInput.KrakenTradesXZECZGBPInput attribute), 12
 crypto (babao.inputs.trades.krakenTradesInput.KrakenTradesXZECZJPYInput attribute), 12
 crypto (babao.inputs.trades.krakenTradesInput.KrakenTradesXZECZUSDInput attribute), 12
 crypto (babao.inputs.trades.tradesInputBase.ABCTradesInput attribute), 13
 cryptoAndActionToTrade () (in module babao.utils.enum), 20
 CryptoEnum (class in babao.utils.enum), 19

D

DASH (babao.utils.enum.CryptoEnum attribute), 19
 debug () (in module babao.utils.log), 22
 delete () (in module babao.utils.file), 21
 dependencies_class (babao.models.modelBase.ABCModel attribute), 16
 dependencies_class (babao.models.rootModel.RootModel attribute), 17
 dependencies_class (babao.models.tree.extremaModel.ExtremaModel attribute), 15
 dependencies_class (babao.models.tree.macdModel.MacdModel attribute), 16
 DEPOSIT (babao.utils.enum.ActionEnum attribute), 19
 deposit () (babao.inputs.ledger.fakeLedgerInput.ABCFakeLedgerInput method), 1
 deposit () (babao.inputs.ledger.krakenLedgerInput.ABCKrakenLedgerInput method), 2
 deposit () (babao.inputs.ledger.ledgerInputBase.ABCLedgerInput method), 3
 dryRun () (in module babao.commands), 23

EOS (babao.utils.enum.CryptoEnum attribute), 19
 error () (in module babao.utils.log), 22
 ETC (babao.utils.enum.CryptoEnum attribute), 19
 ETH (babao.utils.enum.CryptoEnum attribute), 19
 EUR (babao.utils.enum.QuoteEnum attribute), 19
 ewma () (in module babao.utils.indicators), 21
 ExtremaModel (class in babao.models.tree.extremaModel), 15

F

FEE (babao.utils.enum.ActionEnum attribute), 19
 fetch () (babao.inputs.inputBase.ABCInput method), 14
 fetch () (babao.inputs.krakenInputBase.ABCKrakenInput method), 15
 fetch () (babao.inputs.ledger.fakeLedgerInput.ABCFakeLedgerInput method), 1
 fetch () (babao.inputs.ledger.krakenLedgerInput.ABCKrakenLedgerInput method), 2
 fetch () (babao.inputs.trades.krakenTradesInputBase.ABCKrakenTradesInput method), 13
 fetch () (babao.inputs.trades.tradesInputBase.ABCTradesInput method), 13
 fetch () (in module babao.commands), 23
 fetchInputs () (in module babao.inputs.inputManager), 14
 fillMissing () (babao.inputs.inputBase.ABCInput method), 14
 fillMissing () (babao.inputs.ledger.ledgerInputBase.ABCLedgerInput method), 3
 fillMissing () (babao.inputs.trades.tradesInputBase.ABCTradesInput method), 13
 fit () (babao.utils.scale.Scaler method), 22
 floatToTrade () (in module babao.utils.enum), 20

G

gameOver () (in module babao.inputs.ledger.ledgerManager), 3
 GBP (babao.utils.enum.QuoteEnum attribute), 19
 get () (in module babao.utils.indicators), 21
 getBalanceInQuote () (in module babao.inputs.ledger.ledgerManager), 3
 getGlobalBalanceInQuote () (in module babao.inputs.ledger.ledgerManager), 3
 getLastRows () (in module babao.utils.file), 21
 getLastTx () (in module babao.inputs.ledger.ledgerManager), 3
 getTime () (babao.utils.date.TimeTraveler method), 18
 getVerbose () (in module babao.models.modelBase), 17
 GNO (babao.utils.enum.CryptoEnum attribute), 19

H

HODL (*babao.utils.enum.ActionEnum* attribute), 19HODL (*babao.utils.enum.TradeEnum* attribute), 20

I

info() (*in module babao.utils.log*), 22initLedgers() (*in module babao.inputs.ledger.ledgerManager*), 3initLogLevel() (*in module babao.utils.log*), 22initStore() (*in module babao.utils.file*), 21isLocked() (*in module babao.utils.lock*), 22

J

JPY (*babao.utils.enum.QuoteEnum* attribute), 19

K

KrakenTradesBCHEURInput (*class in babao.inputs.trades.krakenTradesInput*), 4KrakenTradesBCHUSDInput (*class in babao.inputs.trades.krakenTradesInput*), 4KrakenTradesDASHEURInput (*class in babao.inputs.trades.krakenTradesInput*), 4KrakenTradesDASHUSDInput (*class in babao.inputs.trades.krakenTradesInput*), 4KrakenTradesEOSEURInput (*class in babao.inputs.trades.krakenTradesInput*), 4KrakenTradesEOSUSDInput (*class in babao.inputs.trades.krakenTradesInput*), 5KrakenTradesGNOEURInput (*class in babao.inputs.trades.krakenTradesInput*), 5KrakenTradesGNOUSDInput (*class in babao.inputs.trades.krakenTradesInput*), 5KrakenTradesXETCZCADInput (*class in babao.inputs.trades.krakenTradesInput*), 5KrakenTradesXETCZEURInput (*class in babao.inputs.trades.krakenTradesInput*), 5KrakenTradesXETCZGBPInput (*class in babao.inputs.trades.krakenTradesInput*), 5KrakenTradesXETCZJPYInput (*class in babao.inputs.trades.krakenTradesInput*), 5KrakenTradesXETCZUSDInput (*class in babao.inputs.trades.krakenTradesInput*), 6KrakenTradesXETHZCADInput (*class in babao.inputs.trades.krakenTradesInput*), 6KrakenTradesXETHZEURInput (*class in babao.inputs.trades.krakenTradesInput*), 6KrakenTradesXETHZGBPInput (*class in babao.inputs.trades.krakenTradesInput*), 6KrakenTradesXETHZJPYInput (*class in babao.inputs.trades.krakenTradesInput*), 6KrakenTradesXETHZUSDInput (*class in babao.inputs.trades.krakenTradesInput*), 6KrakenTradesXLTCZCADInput (*class in babao.inputs.trades.krakenTradesInput*), 7KrakenTradesXLTCZEURInput (*class in babao.inputs.trades.krakenTradesInput*), 7KrakenTradesXLTCZGBPInput (*class in babao.inputs.trades.krakenTradesInput*), 7KrakenTradesXLTCZJPYInput (*class in babao.inputs.trades.krakenTradesInput*), 7KrakenTradesXLTCZUSDInput (*class in babao.inputs.trades.krakenTradesInput*), 7KrakenTradesXREPZCADInput (*class in babao.inputs.trades.krakenTradesInput*), 7KrakenTradesXREPZEURInput (*class in babao.inputs.trades.krakenTradesInput*), 8KrakenTradesXREPZGBPInput (*class in babao.inputs.trades.krakenTradesInput*), 8KrakenTradesXREPZJPYInput (*class in babao.inputs.trades.krakenTradesInput*), 8KrakenTradesXREPZUSDInput (*class in babao.inputs.trades.krakenTradesInput*), 8KrakenTradesXXBTZCADInput (*class in babao.inputs.trades.krakenTradesInput*), 8KrakenTradesXXBTZEURInput (*class in babao.inputs.trades.krakenTradesInput*), 8

[KrakenTradesXXBTZGBPInput](#) (class [babao.inputs.trades.krakenTradesInput](#)),
[9](#)
[KrakenTradesXXBTZJPYInput](#) (class [babao.inputs.trades.krakenTradesInput](#)),
[9](#)
[KrakenTradesXXBTZUSDInput](#) (class [babao.inputs.trades.krakenTradesInput](#)),
[9](#)
[KrakenTradesXXLMZCADInput](#) (class [babao.inputs.trades.krakenTradesInput](#)),
[9](#)
[KrakenTradesXXLMZEURInput](#) (class [babao.inputs.trades.krakenTradesInput](#)),
[9](#)
[KrakenTradesXXLMZGBPInput](#) (class [babao.inputs.trades.krakenTradesInput](#)),
[9](#)
[KrakenTradesXXLMZJPYInput](#) (class [babao.inputs.trades.krakenTradesInput](#)),
[10](#)
[KrakenTradesXXLMZUSDInput](#) (class [babao.inputs.trades.krakenTradesInput](#)),
[10](#)
[KrakenTradesXXMRZCADInput](#) (class [babao.inputs.trades.krakenTradesInput](#)),
[10](#)
[KrakenTradesXXMRZEURInput](#) (class [babao.inputs.trades.krakenTradesInput](#)),
[10](#)
[KrakenTradesXXMRZGBPInput](#) (class [babao.inputs.trades.krakenTradesInput](#)),
[10](#)
[KrakenTradesXXMRZJPYInput](#) (class [babao.inputs.trades.krakenTradesInput](#)),
[10](#)
[KrakenTradesXXMRZUSDInput](#) (class [babao.inputs.trades.krakenTradesInput](#)),
[11](#)
[KrakenTradesXXRPZCADInput](#) (class [babao.inputs.trades.krakenTradesInput](#)),
[11](#)
[KrakenTradesXXRPZEURInput](#) (class [babao.inputs.trades.krakenTradesInput](#)),
[11](#)
[KrakenTradesXXRPZGBPInput](#) (class [babao.inputs.trades.krakenTradesInput](#)),
[11](#)
[KrakenTradesXXRPZJPYInput](#) (class [babao.inputs.trades.krakenTradesInput](#)),
[11](#)
[KrakenTradesXXRPZUSDInput](#) (class [babao.inputs.trades.krakenTradesInput](#)),
[11](#)

[in KrakenTradesXZECZCADInput](#) (class [babao.inputs.trades.krakenTradesInput](#)),
[12](#)
[in KrakenTradesXZECZEURInput](#) (class [babao.inputs.trades.krakenTradesInput](#)),
[12](#)
[in KrakenTradesXZECZGBPInput](#) (class [babao.inputs.trades.krakenTradesInput](#)),
[12](#)
[in KrakenTradesXZECZJPYInput](#) (class [babao.inputs.trades.krakenTradesInput](#)),
[12](#)
[in KrakenTradesXZECZUSDInput](#) (class [babao.inputs.trades.krakenTradesInput](#)),
[12](#)

L

[load\(\)](#) ([babao.models.modelBase.ABCModel](#) method),
[16](#)
[in load\(\)](#) ([babao.models.rootModel.RootModel](#) method),
[17](#)
[in load\(\)](#) ([babao.models.tree.extremaModel.ExtremaModel](#) method),
[15](#)
[load\(\)](#) ([babao.models.tree.macdModel.MacdModel](#) method),
[16](#)
[in logTransaction\(\)](#) ([babao.inputs.ledger.fakeLedgerInput.ABCFakeLedgerInput](#) method),
[2](#)
[in LTC](#) ([babao.utils.enum.CryptoEnum](#) attribute),
[19](#)

M

[in macd\(\)](#) (in module [babao.utils.indicators](#)),
[21](#)
[MacdModel](#) (class in [babao.models.tree.macdModel](#)),
[16](#)
[in main\(\)](#) (in module [babao.babao](#)),
[23](#)
[maintenance\(\)](#) (in module [babao.utils.file](#)),
[21](#)

N

[in nanoToSec\(\)](#) (in module [babao.utils.date](#)),
[18](#)
[in need_training](#) ([babao.models.modelBase.ABCModel](#) attribute),
[16](#)
[need_training](#) ([babao.models.rootModel.RootModel](#) attribute),
[17](#)
[in need_training](#) ([babao.models.tree.extremaModel.ExtremaModel](#) attribute),
[15](#)
[in need_training](#) ([babao.models.tree.macdModel.MacdModel](#) attribute),
[16](#)
[nowMinus\(\)](#) ([babao.utils.date.TimeTraveler](#) method),
[18](#)

P

[in pair](#) ([babao.inputs.trades.krakenTradesInput.KrakenTradesBCHEURInput](#) attribute),
[4](#)
[pair](#) ([babao.inputs.trades.krakenTradesInput.KrakenTradesBCHUSDInput](#) attribute),
[4](#)

```

pair (babao.inputs.trades.krakenTradesInput.KrakenTradesDASHEURInput, KrakenTradesXXBTZEURIn
    attribute), 4
pair (babao.inputs.trades.krakenTradesInput.KrakenTradesDASHUSDInput, KrakenTradesXXBTZGBPIn
    attribute), 4
pair (babao.inputs.trades.krakenTradesInput.KrakenTradesEOSGBPInput, KrakenTradesXXBTZJPYIn
    attribute), 4
pair (babao.inputs.trades.krakenTradesInput.KrakenTradesEOSUSDInput, KrakenTradesXXBTZUSDIn
    attribute), 5
pair (babao.inputs.trades.krakenTradesInput.KrakenTradesGNOEURInput, KrakenTradesXXLMZCADIn
    attribute), 5
pair (babao.inputs.trades.krakenTradesInput.KrakenTradesGNOUSDInput, KrakenTradesXXLMZEURIn
    attribute), 5
pair (babao.inputs.trades.krakenTradesInput.KrakenTradesXEFZCADInput, KrakenTradesXXLMZGBPIn
    attribute), 5
pair (babao.inputs.trades.krakenTradesInput.KrakenTradesXEFZEURInput, KrakenTradesXXLMZJPYIn
    attribute), 5
pair (babao.inputs.trades.krakenTradesInput.KrakenTradesXEFZGBPInput, KrakenTradesXXLMZUSDIn
    attribute), 5
pair (babao.inputs.trades.krakenTradesInput.KrakenTradesXEFZJPYInput, KrakenTradesXXMRZCADIn
    attribute), 6
pair (babao.inputs.trades.krakenTradesInput.KrakenTradesXEFZUSDInput, KrakenTradesXXMRZEURIn
    attribute), 6
pair (babao.inputs.trades.krakenTradesInput.KrakenTradesXEFZCADInput, KrakenTradesXXMRZGBPIn
    attribute), 6
pair (babao.inputs.trades.krakenTradesInput.KrakenTradesXEFZEURInput, KrakenTradesXXMRZJPYIn
    attribute), 6
pair (babao.inputs.trades.krakenTradesInput.KrakenTradesXEFZGBPInput, KrakenTradesXXMRZUSDIn
    attribute), 6
pair (babao.inputs.trades.krakenTradesInput.KrakenTradesXEFZJPYInput, KrakenTradesXXRPZCADIn
    attribute), 6
pair (babao.inputs.trades.krakenTradesInput.KrakenTradesXEFZUSDInput, KrakenTradesXXRPZEURIn
    attribute), 7
pair (babao.inputs.trades.krakenTradesInput.KrakenTradesXEFZCADInput, KrakenTradesXXRPZGBPIn
    attribute), 7
pair (babao.inputs.trades.krakenTradesInput.KrakenTradesXEFZEURInput, KrakenTradesXXRPZJPYIn
    attribute), 7
pair (babao.inputs.trades.krakenTradesInput.KrakenTradesXEFZGBPInput, KrakenTradesXXRPZUSDIn
    attribute), 7
pair (babao.inputs.trades.krakenTradesInput.KrakenTradesXEFZJPYInput, KrakenTradesXZECZCADIn
    attribute), 7
pair (babao.inputs.trades.krakenTradesInput.KrakenTradesXEFZUSDInput, KrakenTradesXZECZEURIn
    attribute), 7
pair (babao.inputs.trades.krakenTradesInput.KrakenTradesXZECZCADInput, KrakenTradesXZECZGBPIn
    attribute), 8
pair (babao.inputs.trades.krakenTradesInput.KrakenTradesXZECZEURInput, KrakenTradesXZECZJPYIn
    attribute), 8
pair (babao.inputs.trades.krakenTradesInput.KrakenTradesXZECZGBPInput, KrakenTradesXZECZUSDIn
    attribute), 8
pair (babao.inputs.trades.krakenTradesInput.KrakenTradesXZECZJPYInput, KrakenTradesInputBase.ABCKrakenTradesInput
    attribute), 8
pair (babao.inputs.trades.krakenTradesInput.KrakenTradesXZECZUSDInput, KrakenTradesInputBase.ABCKrakenTradesInput
    attribute), 8
pair (babao.inputs.trades.krakenTradesInput.KrakenTradesXXBTZCADInput, KrakenTradesInputBase.ABCKrakenTradesInput
    attribute), 8

```

```

17
plot () (babao.models.tree.extremaModel.ExtremaModel
method), 15
plot () (babao.models.tree.macdModel.MacdModel
method), 16
plotModels () (in module
babao.models.modelManager), 17
ppo () (in module babao.utils.indicators), 21
predict () (babao.models.modelBase.ABCModel
method), 17
predict () (babao.models.rootModel.RootModel
method), 17
predict () (babao.models.tree.extremaModel.ExtremaModel
method), 15
predict () (babao.models.tree.macdModel.MacdModel
method), 16
predictModelsMaybeTrade () (in module
babao.models.modelManager), 17
Q
quote (babao.inputs.trades.krakenTradesInput.KrakenTradesXETHZUSD
attribute), 4
quote (babao.inputs.trades.krakenTradesInput.KrakenTradesXLTZCAD
attribute), 4
quote (babao.inputs.trades.krakenTradesInput.KrakenTradesXLTZEUR
attribute), 4
quote (babao.inputs.trades.krakenTradesInput.KrakenTradesXBTZCAD
attribute), 4
quote (babao.inputs.trades.krakenTradesInput.KrakenTradesXBTZEUR
attribute), 9
quote (babao.inputs.trades.krakenTradesInput.KrakenTradesXBTZGBP
attribute), 9
quote (babao.inputs.trades.krakenTradesInput.KrakenTradesXBTZJPY
attribute), 9
quote (babao.inputs.trades.krakenTradesInput.KrakenTradesXBTZUSD
attribute), 9
quote (babao.inputs.trades.krakenTradesInput.KrakenTradesXXLMZCAD
attribute), 9
quote (babao.inputs.trades.krakenTradesInput.KrakenTradesXXLMZEUR
attribute), 9
quote (babao.inputs.trades.krakenTradesInput.KrakenTradesXXLMZGBP
attribute), 10
quote (babao.inputs.trades.krakenTradesInput.KrakenTradesXXLMZJPY
attribute), 10
quote (babao.inputs.trades.krakenTradesInput.KrakenTradesXXLMZUSD
attribute), 10
quote (babao.inputs.trades.krakenTradesInput.KrakenTradesXXMRZCAD
attribute), 10
quote (babao.inputs.trades.krakenTradesInput.KrakenTradesXXMRZEUR
attribute), 10
quote (babao.inputs.trades.krakenTradesInput.KrakenTradesXXMRZGBP
attribute), 10
quote (babao.inputs.trades.krakenTradesInput.KrakenTradesXXMRZJPY
attribute), 11
quote (babao.inputs.trades.krakenTradesInput.KrakenTradesXXMRZUSD
attribute), 11
quote (babao.inputs.trades.krakenTradesInput.KrakenTradesXXRPZCAD
attribute), 11

```

`quote (babao.inputs.trades.krakenTradesInput.KrakenTradesXXRPZEURInput in babao.models.rootModel), 17`
`attribute), 11`
`quote (babao.inputs.trades.krakenTradesInput.KrakenTradesXXRPZGBPInput`
`attribute), 11`
`save () (babao.models.modelBase.ABCModel method),`
`quote (babao.inputs.trades.krakenTradesInput.KrakenTradesXXRPZJPYInput`
`attribute), 11`
`save () (babao.models.rootModel.RootModel method),`
`quote (babao.inputs.trades.krakenTradesInput.KrakenTradesXXRPZUSDInput`
`attribute), 12`
`save () (babao.models.tree.extremaModel.ExtremaModel`
`method), 15`
`quote (babao.inputs.trades.krakenTradesInput.KrakenTradesXZECZCADInput,`
`attribute), 12`
`save () (babao.models.tree.macdModel.MacdModel`
`method), 16`
`quote (babao.inputs.trades.krakenTradesInput.KrakenTradesXZECZEURInput,`
`attribute), 12`
`scale () (babao.utils.scale.Scaler method), 22`
`quote (babao.inputs.trades.krakenTradesInput.KrakenTradesXZECZGBPInput`
`attribute), 12`
`Scaler (class in babao.utils.scale), 22`
`quote (babao.inputs.trades.krakenTradesInput.KrakenTradesXZECZJPYInput`
`attribute), 12`
`sectionName () (in module babao.utils.date), 18`
`quote (babao.inputs.trades.krakenTradesInput.KrakenTradesXZECZUSDInput`
`attribute), 12`
`SELL (babao.inputs.ledger.fakeLedgerInput.ABCFakeLedgerInput`
`method), 2`
`quote (babao.inputs.trades.tradesInputBase.ABCTradesInput`
`attribute), 13`
`SELL () (babao.inputs.ledger.krakenLedgerInput.ABCKrakenLedgerInput`
`method), 2`
`QuoteEnum (class in babao.utils.enum), 19`
`sell () (babao.inputs.ledger.ledgerInputBase.ABCLedgerInput`
`method), 3`
`sell () (in module babao.inputs.ledger.ledgerManager),`
`3`
R
`raw_columns (babao.inputs.inputBase.ABCInput at-`
`tribute), 14`
`raw_columns (babao.inputs.ledger.ledgerInputBase.ABCLedgerInput`
`attribute), 3`
`raw_columns (babao.inputs.trades.tradesInputBase.ABCTradesInput`
`attribute), 13`
`read () (babao.inputs.inputBase.ABCInput method), 14`
`read () (in module babao.utils.file), 21`
`readConfigFile () (in module babao.config), 24`
`readInputs () (in module`
`babao.inputs.inputManager), 14`
`refreshCache () (babao.inputs.inputBase.ABCInput`
`method), 14`
`refreshInputs () (in module`
`babao.inputs.inputManager), 14`
`REP (babao.utils.enum.CryptoEnum attribute), 19`
`resample () (babao.inputs.inputBase.ABCInput`
`method), 14`
`resampled_columns`
`(babao.inputs.inputBase.ABCInput attribute),`
`14`
`resampled_columns`
`(babao.inputs.ledger.ledgerInputBase.ABCLedgerInput`
`attribute), 3`
`resampled_columns`
`(babao.inputs.trades.tradesInputBase.ABCTradesInput`
`attribute), 13`
`resampleSerie () (in module`
`babao.inputs.inputBase), 14`
`reshape () (in module babao.models.modelBase), 17`
`restoreSignal () (in module babao.utils.signal), 23`

`SELL_BCH (babao.utils.enum.TradeEnum attribute), 20`
`SELL_BTC (babao.utils.enum.TradeEnum attribute), 20`
`SELL_ETH (babao.utils.enum.TradeEnum attribute), 20`
`SELL_GNO (babao.utils.enum.TradeEnum attribute), 20`
`SELL_LTC (babao.utils.enum.TradeEnum attribute), 20`
`SELL_REP (babao.utils.enum.TradeEnum attribute), 20`
`SELL_XBT (babao.utils.enum.TradeEnum attribute), 20`
`SELL_XLM (babao.utils.enum.TradeEnum attribute), 20`
`SELL_XMR (babao.utils.enum.TradeEnum attribute), 20`
`SELL_XRP (babao.utils.enum.TradeEnum attribute), 20`
`SELL_ZEC (babao.utils.enum.TradeEnum attribute), 20`
`setLock () (in module babao.utils.file), 21`
`setLock () (in module babao.utils.log), 22`
`setTime () (babao.utils.date.TimeTraveler method), 18`
`sma () (in module babao.utils.indicators), 21`

T
`timeTravel () (in module`
`babao.inputs.inputManager), 15`
`TimeTraveler (class in babao.utils.date), 18`
`toDatetime () (in module babao.utils.date), 18`
`toEnter () (in module babao.utils.date), 18`
`toTimestamp () (in module babao.utils.date), 18`
`TradeEnum (class in babao.utils.enum), 19`
`tradeToAction () (in module babao.utils.enum), 20`
`tradeToCrypto () (in module babao.utils.enum), 20`

[train\(\)](#) (*`babao.models.modelBase.ABCModel` method*), [17](#)
[train\(\)](#) (*`babao.models.rootModel.RootModel` method*), [18](#)
[train\(\)](#) (*`babao.models.tree.extremaModel.ExtremaModel` method*), [15](#)
[train\(\)](#) (*`babao.models.tree.macdModel.MacdModel` method*), [16](#)
[train\(\)](#) (*in module `babao.commands`*), [24](#)
[trainModels\(\)](#) (*in module `babao.models.modelManager`*), [17](#)
[tryLock\(\)](#) (*in module `babao.utils.lock`*), [22](#)
[tryUnlock\(\)](#) (*in module `babao.utils.lock`*), [22](#)

U

[unscale\(\)](#) (*`babao.utils.scaleScaler` method*), [22](#)
[updateCurrentRow\(\)](#) (*`babao.inputs.inputBase.ABCInput` method*), [14](#)
[USD](#) (*`babao.utils.enum.QuoteEnum` attribute*), [19](#)

W

[warning\(\)](#) (*in module `babao.utils.log`*), [22](#)
[wetRun\(\)](#) (*in module `babao.commands`*), [24](#)
[WITHDRAW](#) (*`babao.utils.enum.ActionEnum` attribute*), [19](#)
[withdraw\(\)](#) (*`babao.inputs.ledger.fakeLedgerInput.ABCFakeLedgerInput` method*), [2](#)
[withdraw\(\)](#) (*`babao.inputs.ledger.krakenLedgerInput.ABCKrakenLedgerInput` method*), [2](#)
[withdraw\(\)](#) (*`babao.inputs.ledger.ledgerInputBase.ABCLedgerInput` method*), [3](#)
[write\(\)](#) (*`babao.inputs.inputBase.ABCInput` method*), [14](#)
[write\(\)](#) (*in module `babao.utils.file`*), [21](#)

X

[XBT](#) (*`babao.utils.enum.CryptoEnum` attribute*), [19](#)
[XLM](#) (*`babao.utils.enum.CryptoEnum` attribute*), [19](#)
[XMR](#) (*`babao.utils.enum.CryptoEnum` attribute*), [19](#)
[XRP](#) (*`babao.utils.enum.CryptoEnum` attribute*), [19](#)

Z

[ZEC](#) (*`babao.utils.enum.CryptoEnum` attribute*), [19](#)