PyReuters

Overview

pyreuters provides an API to access reuters market data stored on a remote server.

Version Info

Release v1.0.0: v1.0.0. would be a major release.

It would be one and only feature release. No new features will be added afterwards. Later versions will only be bug fixes.

Features

- [x] Command line tools to download data, convert to hdf5 and search remote server for symbols
- [x] Functions to read raw market data file, quotes and trades
- [x] Functions to clean quotes and trades data
- [x] `Symbol` API to load market data for a particular symbol, and merge quotes and trades data
- [x] Comprehensive documentation

Installation

Navigate to the root directory of the package - the directory that has `setup.py`

```
$ python setup.py install
```

Command Line Tools

reuters_download

```
$ reuters_download --help
```

```
usage: reuters_download [-h] [-v] [-n NETWORK_IP] [-u USERNAME] [-p PASSWORD]
                        [-i INSTRUMENTS] [-d DIR] [-s START_DATE]
                        [-e END_DATE]

Download Reuters data from the configured server

optional arguments:
  -h, --help            show this help message and exit
  -v, --verbose         Verbose output
```

```
  -n NETWORK_IP, --network_ip NETWORK_IP
                        IP address of the server
  -u USERNAME, --username USERNAME
                        Username to connect to reuters data server
  -p PASSWORD, --password PASSWORD
                        Password to connect to reuters data server
  -i INSTRUMENTS, --instruments INSTRUMENTS
                        Instruments for which data is needed. Separate
                        multiple instruments by ,
  -d DIR, --dir DIR     Directory to save data
  -s START_DATE, --start START_DATE
                        Start date for data in format YYYYMMDD
  -e END_DATE, --end END_DATE
                        End date for data in format YYYYMMDD

Example : reuters_download -i ED -s 20160101 -e 20160104 -v -u ksharma -p
*******
```

reuters_convert

```
$ reuters_convert --help
```

```
usage: reuters_convert [-h] [-v] [-i INSTRUMENTS] [-k] [-s SYMBOLS]
                       [-e EXCHANGE] [-c] [-r DATA_PATH] [-d DEST_PATH]

Convert the raw data files into hdf5 format

optional arguments:
  -h, --help            show this help message and exit
  -v, --verbose         Verbose output for the conversion
  -i INSTRUMENTS, --instruments INSTRUMENTS
                        Instruments to be converted to hdf5separate
                        instruments by ,
  -k, --keep_ric        Keep the RIC symbol as hdf5 filename
  -s SYMBOLS, --symbols SYMBOLS
                        json config file for symbols. Overrides the package
                        symbols config
  -e EXCHANGE, --exchange EXCHANGE
                        Add exchange acronym in hdf5 filename
  -c, --clean           Clean the market data before saving
  -r DATA_PATH, --raw_path DATA_PATH
                        Path with dated folders for tick data
  -d DEST_PATH, --destination DEST_PATH
                        Destination directory

Example : reuters_convert -i ED
```

reuters_search

```
$ reuters_search --help
```

```
usage: reuters_search [-h] [-v] [-n NETWORK_IP] -u USERNAME -p PASSWORD
                      [-d DATE] [-g GREP]

Download Reuters data from the configured server

optional arguments:
  -h, --help            show this help message and exit
  -v, --verbose         Verbose output
  -n NETWORK_IP, --network_ip NETWORK_IP
                        IP address of the server
  -u USERNAME, --username USERNAME
                        Username to connect to reuters data server
  -p PASSWORD, --password PASSWORD
                        Password to connect to reuters data server
  -d DATE, --date DATE  Date to check in format YYYYMMDD
  -g GREP, --grep GREP  Search for a particular word

Example : reuters_search -d 20160104 -v -u ksharma -p ******* -g NG
```

Reading Raw Data

`pyreuters.data` module provides functions to read the raw reuters tick data files and filter out quotes or trades

- `read_raw`
- `quotes_data`
- `trades_data`

```
In[1]: import pyreuters.data as reuters

In[2]: reuters.read_raw("NGQ6", "2016-01-03")[:1]
Out[2]:
                            #RIC      Date[G]          Time[G]  GMT Offset  \
DateTime
2016-01-03 17:00:04.259805  NGQ6  03-JAN-2016  17:00:04.259805          -5

                                  Type  Price  Volume  Bid Price  Bid Size  \
DateTime
2016-01-03 17:00:04.259805  Correction    0.0     NaN        NaN       NaN

                            Ask Price  Ask Size  \
DateTime
2016-01-03 17:00:04.259805        NaN       NaN

                                                                 Qualifiers  \
DateTime
2016-01-03 17:00:04.259805    [CLSRNGTP];   [IRGCOND];   [MKT_ST_IND];  [O...

                            New Price  New Vol
```

```
DateTime
2016-01-03 17:00:04.259805        NaN       NaN
```

```
In[3]: reuters.quotes_data(symbol="NGQ6", date="2016-01-03")[:1]
Out[3]:
                            Bid  BidSize    Ask  AskSize
DateTime
2016-01-03 20:00:31.929364  NaN      NaN  2.594      1.0
```

```
In[4]: reuters.trades_data(symbol="NGQ6", date="2016-01-03")[:1]
Out[4]:
                            Price  Volume
DateTime
2016-01-03 23:08:03.323453    NaN     1.0
```

Configuration

Default configuration is provided with package distribution. `json` config can be found in `pyreuters/resources`. These config files can be changed to have user's own settings before/after install.

server_config.json

Change settings for server and local machine.

- Used by `reuters_download` to point to a particular network ip.
- Used by `reuters_convert` to access files for hdf5 conversion.
- Provides the default directory for functions that read raw files

```
{
  "local_machine": {
    "reuters_data_dir": "~/dev/reuters/data",
    "hdf5_dir": "~/dev/reuters"
  },
  "server": {
    "server_ip": "10.10.100.222",
    "server_dir": "/home/storage/csv/"
  }
}
```

symbols.json

Allows user to save symbol specific market data files with actual exchange symbols and not the `RIC` code

```
{
  "NG": "NG",
```

```
    "CL": "CL",
    "HO": "HO",
    "NTG": "NN",
    "BZZ": "BZ",
    "ED": "GE",
    "GE": "GE"
}
```

Cleaning

`pyreuters.clean` provides functions to clean market data using some helper functions. Cleaning can be done using wrapper functions `clean_quotes` and `clean_trades` or individual functions can be called separately.

clean_quotes

`clean_quotes` calls below functions from within:

- `pyreuters.clean.rm_erroneous_quotes`
- `pyreuters.clean.rm_large_spreads`
- `pyreuters.clean.rm_quote_outliers` - with `filter_type` = `standard` or `advanced`
- `pyreuters.clean.no_zero_quotes`

These functions are wrapped in a `Python dictionary` and all functions in the dictionary are called by default in `clean_quotes`

```
{
    'error_quotes': <function pyreuters.clean.rm_erroneous_quotes>,
    'large_spreads': <function pyreuters.clean.rm_large_spreads>,
    'outliers': <function pyreuters.clean.rm_quote_outliers>,
    'zero_quotes': <function pyreuters.clean.no_zero_quotes>
}
```

clean_trades

`clean_trades` calls below functions from within:

- `pyreuters.clean.no_zero_prices`

Similar to `clean_quotes`, these functions are wrapped in a `Python dictionary` and all functions in the dictionary are called by default in `clean_trades`

```
{'zero_prices': <function pyreuters.clean.no_zero_prices>}
```

Examples

```
In[1]: import pyreuters.data as reuters
      import pyreuters.clean as clean

In[2]: quotes = reuters.quotes_data(symbol="NGQ6", date="2016-01-03")

In[3] quotes = clean.clean_quotes(quotes)
Removed 0 zero quotes
Removed 1 erroneous quotes
Removed 18 outliers
Removed 804 large spread quotes

In[4]: trades = reuters.trades_data(symbol="NGQ6", date="2016-01-03")

In[5]: trades = clean.clean_trades(trades)
Removed 5 zero priced trades
```

Symbol API

`pyreuters.symbol` API provides a `class Symbol` that loads market data from known hdf5 data files.

`Symbol` class takes a `symbol` such as `NG` as an argument. `exchange` can also be provided in case data is saved as `CME_NG.h5`. This links directly to the `exchange` argument to `reuters_convert`

Market data is saved in `dict quotes` and `trades` where keys are different contracts

- `pyreuters.symbol.Symbol.load(start_time, end_time)` : Loads data between `start_time` and `end_time` with `start_time` inclusive
- `pyreuters.symbol.Symbol.load_contract(contract, start_time, end_time)` : Loads data between `start_time` and `end_time` for a specific contract
- `pyreuters.symbol.Symbol.loaded_contracts(data_type='Quote')` : All the contracts that have been loaded in `quotes` and `trades`
- `pyreuters.symbol.Symbol.merge_qt()` : Merges `quotes` and `trades` and save it in `quotes` dictionary
- `pyreuters.symbol.Symbol.get_quotes(contract)` : Helper function to get `quotes` for a particular contract
- `pyreuters.symbol.Symbol.get_trades(contract)` : Helper function to get `trades` for a particular contract
- `pyreuters.symbol.Symbol.available(hdf_file)` : Static function that gives all available contracts in a particular hdf5 file

Example

For the following example, I will assume that I have saved `NGZ6` data in `CME_NG.h5`

```
In[1]: from pyreuters.symbol import Symbol
```

```
In[3]: ng = Symbol("NG", exchange="CME")

In[3]: ng.load(start_time="2016-01-03 17:00:00", end_time="2016-01-04 07:00:00")

In[4]: ng.quotes["NGZ6"].head(10)
Out[4]:
                                       bid  bid_size    ask  ask_size
2016-01-03 17:00:01.435565056-06:00  2.698       1.0    NaN       NaN
2016-01-03 17:00:01.443333888-06:00  2.711       1.0    NaN       NaN
2016-01-03 17:00:01.466745088-06:00  2.698       1.0    NaN       NaN
2016-01-03 17:00:01.466745088-06:00  2.709       1.0    NaN       NaN
2016-01-03 17:00:01.466757120-06:00    NaN       NaN  2.829       1.0
2016-01-03 17:00:01.481402112-06:00    NaN       NaN  2.813       1.0
2016-01-03 17:00:01.487341056-06:00    NaN       NaN  2.813       2.0
2016-01-03 17:00:01.499045888-06:00  2.711       1.0    NaN       NaN
2016-01-03 17:00:01.508812032-06:00  2.698       1.0    NaN       NaN
2016-01-03 17:00:01.508812032-06:00  2.709       1.0    NaN       NaN

In[5]: ng.quotes["NGZ6"].tail(10)
Out[5]:
                                     bid  bid_size    ask  ask_size
2016-01-04 06:59:59.263430912-06:00  NaN       NaN  2.761       2.0
2016-01-04 06:59:59.263460096-06:00  NaN       NaN  2.760       1.0
2016-01-04 06:59:59.263460096-06:00  NaN       NaN  2.760       2.0
2016-01-04 06:59:59.273178112-06:00  NaN       NaN  2.760       1.0
2016-01-04 06:59:59.273178112-06:00  NaN       NaN  2.761       4.0
2016-01-04 06:59:59.273178112-06:00  NaN       NaN  2.761       5.0
2016-01-04 06:59:59.273178112-06:00  NaN       NaN  2.761       4.0
2016-01-04 06:59:59.692164096-06:00  NaN       NaN  2.760       1.0
2016-01-04 06:59:59.705784064-06:00  NaN       NaN  2.760       2.0
2016-01-04 06:59:59.705784064-06:00  NaN       NaN  2.760       3.0

In[6]: ng.trades["NGZ6"].head(10)
Out[6]:
                                     price  volume
2016-01-03 17:07:07.609774080-06:00  2.775     1.0
2016-01-03 17:07:11.469128960-06:00  2.775     1.0
2016-01-03 17:07:11.469140992-06:00  2.775     3.0
2016-01-03 17:07:16.374342912-06:00  2.775     1.0
2016-01-03 17:07:18.888011008-06:00  2.775     1.0
2016-01-03 17:07:30.113586944-06:00  2.775     2.0
2016-01-03 17:07:33.810831104-06:00  2.775     2.0
2016-01-03 17:07:35.088183040-06:00  2.775     4.0
2016-01-03 17:08:36.860453120-06:00  2.775     1.0
2016-01-03 17:08:40.223708928-06:00  2.775     5.0

In[7]: ng.merge_qt()
Out[7]: <pyreuters.symbol.Symbol at 0x111a2fe50>

In[8]: ng.quotes["NGZ6"].head(10)
Out[8]:
                                       ask  ask_size    bid  bid_size  price
volume
2016-01-03 17:00:01.435565056-06:00    NaN       NaN  2.698       1.0    NaN
NaN
2016-01-03 17:00:01.443333888-06:00    NaN       NaN  2.711       1.0    NaN
NaN
```

```
2016-01-03 17:00:01.466745088-06:00    NaN       NaN  2.698       1.0    NaN
NaN
2016-01-03 17:00:01.466745088-06:00    NaN       NaN  2.709       1.0    NaN
NaN
2016-01-03 17:00:01.466757120-06:00  2.829       1.0    NaN       NaN    NaN
NaN
2016-01-03 17:00:01.481402112-06:00  2.813       1.0    NaN       NaN    NaN
NaN
2016-01-03 17:00:01.487341056-06:00  2.813       2.0    NaN       NaN    NaN
NaN
2016-01-03 17:00:01.499045888-06:00    NaN       NaN  2.711       1.0    NaN
NaN
2016-01-03 17:00:01.508812032-06:00    NaN       NaN  2.698       1.0    NaN
NaN
2016-01-03 17:00:01.508812032-06:00    NaN       NaN  2.709       1.0    NaN
NaN

In[9]: ngz6 = ng.quotes["NGZ6"]

In[10]: ngz6[ngz6["price"].notnull()].head(10)
Out[10]:
                                       ask  ask_size    bid  bid_size  price
volume
2016-01-03 17:07:07.609774080-06:00    NaN       NaN  2.775      14.0  2.775
1.0
2016-01-03 17:07:11.469128960-06:00    NaN       NaN  2.775      13.0  2.775
1.0
2016-01-03 17:07:11.469140992-06:00    NaN       NaN  2.775      10.0  2.775
3.0
2016-01-03 17:07:11.469140992-06:00  2.787       5.0    NaN       NaN  2.775
3.0
2016-01-03 17:07:11.469140992-06:00  2.787       6.0    NaN       NaN  2.775
3.0
2016-01-03 17:07:16.374342912-06:00    NaN       NaN  2.775       9.0  2.775
1.0
2016-01-03 17:07:18.888011008-06:00    NaN       NaN  2.775       8.0  2.775
1.0
2016-01-03 17:07:30.113586944-06:00    NaN       NaN  2.775       6.0  2.775
2.0
2016-01-03 17:07:33.810831104-06:00    NaN       NaN  2.775       4.0  2.775
2.0
2016-01-03 17:07:35.088183040-06:00    NaN       NaN  2.770       6.0  2.775
4.0

In[11]: h5_file = ng.hdf5_file

In[12]: Symbol.available(h5_file)
Out[12]:
{
    'Quote': ['NGZ6'],
    'Trade': ['NGZ6']
}

In[12]: ng.loaded_contracts()
Out[12]: ['NGZ6']

In[13]: ng.get_quotes("NGZ6")[:10]
```

```
Out[13]:
                                        ask  ask_size    bid  bid_size  price
volume
2016-01-03 17:00:01.435565056-06:00    NaN       NaN  2.698       1.0    NaN
NaN
2016-01-03 17:00:01.443333888-06:00    NaN       NaN  2.711       1.0    NaN
NaN
2016-01-03 17:00:01.466745088-06:00    NaN       NaN  2.698       1.0    NaN
NaN
2016-01-03 17:00:01.466745088-06:00    NaN       NaN  2.709       1.0    NaN
NaN
2016-01-03 17:00:01.466757120-06:00  2.829       1.0    NaN       NaN    NaN
NaN
2016-01-03 17:00:01.481402112-06:00  2.813       1.0    NaN       NaN    NaN
NaN
2016-01-03 17:00:01.487341056-06:00  2.813       2.0    NaN       NaN    NaN
NaN
2016-01-03 17:00:01.499045888-06:00    NaN       NaN  2.711       1.0    NaN
NaN
2016-01-03 17:00:01.508812032-06:00    NaN       NaN  2.698       1.0    NaN
NaN
2016-01-03 17:00:01.508812032-06:00    NaN       NaN  2.709       1.0    NaN
NaN

In[14]: ng.get_trades("NGZ6")[:10]
Out[14]:
                                     price  volume
2016-01-03 17:07:07.609774080-06:00  2.775     1.0
2016-01-03 17:07:11.469128960-06:00  2.775     1.0
2016-01-03 17:07:11.469140992-06:00  2.775     3.0
2016-01-03 17:07:16.374342912-06:00  2.775     1.0
2016-01-03 17:07:18.888011008-06:00  2.775     1.0
2016-01-03 17:07:30.113586944-06:00  2.775     2.0
2016-01-03 17:07:33.810831104-06:00  2.775     2.0
2016-01-03 17:07:35.088183040-06:00  2.775     4.0
2016-01-03 17:08:36.860453120-06:00  2.775     1.0
2016-01-03 17:08:40.223708928-06:00  2.775     5.0
```