01_datareader

September 29, 2021

1 Remote data access using pandas

The pandas library enables access to data displayed on websites using the `read_html()` function and access to the API endpoints of various data providers through the related `pandas-datareader` library.

[1]:
```
import os
import pandas_datareader.data as web
from datetime import datetime
import pandas as pd
```

1.1 Download html table with SP500 constituents

The download of the content of one or more html tables works as follows, for instance for the constituents of the S&P500 index from Wikipedia

[2]:
```
sp_url = 'https://en.wikipedia.org/wiki/List_of_S%26P_500_companies'
sp500_constituents = pd.read_html(sp_url, header=0)[0]
```

[3]:
```
sp500_constituents.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 505 entries, 0 to 504
Data columns (total 9 columns):
Security                 505 non-null object
Symbol                   505 non-null object
SEC filings              505 non-null object
GICS Sector              505 non-null object
GICS Sub Industry        505 non-null object
Headquarters Location    505 non-null object
Date first added         402 non-null object
CIK                      505 non-null int64
Founded                  172 non-null object
dtypes: int64(1), object(8)
memory usage: 35.6+ KB
```

[4]:
```
sp500_constituents.head()
```

[4]:
```
              Security Symbol SEC filings             GICS Sector  \
0           3M Company    MMM     reports             Industrials
1  Abbott Laboratories    ABT     reports             Health Care
2          AbbVie Inc.   ABBV     reports             Health Care
3          ABIOMED Inc   ABMD     reports             Health Care
4        Accenture plc    ACN     reports  Information Technology

                GICS Sub Industry    Headquarters Location Date first added  \
0        Industrial Conglomerates      St. Paul, Minnesota              NaN
1           Health Care Equipment  North Chicago, Illinois       1964-03-31
2                 Pharmaceuticals  North Chicago, Illinois       2012-12-31
3           Health Care Equipment   Danvers, Massachusetts       2018-05-31
4  IT Consulting & Other Services          Dublin, Ireland       2011-07-06

       CIK      Founded
0    66740         1902
1     1800         1888
2  1551152  2013 (1888)
3   815094         1981
4  1467373         1989
```

1.2 pandas-datareader for Market Data

`pandas` used to facilitate access to data providers' APIs directly, but this functionality has moved to the related pandas-datareader library. The stability of the APIs varies with provider policies, and as of June 2o18 at version 0.7, the following sources are available

See documentation; functionality frequently changes as underlying provider APIs evolve.

1.2.1 Yahoo Finance

[5]:
```
start = '2014'
end = datetime(2017, 5, 24)

yahoo= web.DataReader('FB', 'yahoo', start=start, end=end)
yahoo.info()
```

```
<class 'pandas.core.frame.DataFrame'>
DatetimeIndex: 856 entries, 2014-01-02 to 2017-05-25
Data columns (total 6 columns):
High         856 non-null float64
Low          856 non-null float64
Open         856 non-null float64
Close        856 non-null float64
Volume       856 non-null int64
Adj Close    856 non-null float64
dtypes: float64(5), int64(1)
memory usage: 46.8 KB
```

1.2.2 IEX

Note: IEX is transitioning to a new API that will require a (free) account; the datareader will be updated accordingly with the next release.

IEX is an alternative exchange started in response to the HFT controversy and portrayed in Michael Lewis' controversial Flash Boys. It aims to slow down the speed of trading to create a more level playing field and has been growing rapidly since launch in 2016 while still small with a market share of around 2.5% in June 2018.

[6]:
```
start = datetime(2015, 2, 9)
# end = datetime(2017, 5, 24)

iex = web.DataReader('FB', 'iex', start)
iex.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 1054 entries, 2015-02-09 to 2019-04-16
Data columns (total 5 columns):
open      1054 non-null float64
high      1054 non-null float64
low       1054 non-null float64
close     1054 non-null float64
volume    1054 non-null int64
dtypes: float64(4), int64(1)
memory usage: 49.4+ KB
```

[7]:
```
iex.tail()
```

[7]:

date	open	high	low	close	volume
2019-04-10	178.18	178.79	176.54	177.82	11701479
2019-04-11	178.24	178.40	177.00	177.51	8070967
2019-04-12	178.00	179.63	177.95	179.10	12329812
2019-04-15	178.50	180.50	176.87	179.65	10834762
2019-04-16	179.00	180.17	178.30	178.87	11215193

Book Data In addition to historical EOD price and volume data, IEX provides real-time depth of book quotations that offer an aggregated size of orders by price and side. This service also includes last trade price and size information.

DEEP is used to receive real-time depth of book quotations direct from IEX. The depth of book quotations received via DEEP provide an aggregated size of resting displayed orders at a price and side, and do not indicate the size or number of individual orders at any price level. Non-displayed orders and non-displayed portions of reserve orders are not represented in DEEP.

DEEP also provides last trade price and size information. Trades resulting from either displayed or non-displayed orders matching on IEX will be reported. Routed executions will not be reported.

Only works on trading days.

[8]:
```
book = web.get_iex_book('AAPL')
```

[9]:
```
list(book.keys())
```

[9]:
```
['symbol',
 'marketPercent',
 'volume',
 'lastSalePrice',
 'lastSaleSize',
 'lastSaleTime',
 'lastUpdated',
 'bids',
 'asks',
 'systemEvent',
 'tradingStatus',
 'opHaltStatus',
 'ssrStatus',
 'securityEvent',
 'trades',
 'tradeBreaks']
```

[10]:
```
orders = pd.concat([pd.DataFrame(book[side]).assign(side=side) for side in␣
↪['bids', 'asks']])
orders.head()
```

[10]:
```
Empty DataFrame
Columns: [side]
Index: []
```

[11]:
```
for key in book.keys():
    try:
        print(f'\n{key}')
        print(pd.DataFrame(book[key]))
    except:
        print(book[key])
```

```
symbol
AAPL

marketPercent
0.03324

volume
977659

lastSalePrice
```

```
203.19

lastSaleSize
3

lastSaleTime
1555531318248

lastUpdated
1555532174025

bids
Empty DataFrame
Columns: []
Index: []

asks
Empty DataFrame
Columns: []
Index: []

systemEvent
{'systemEvent': 'C', 'timestamp': 1555535400001}

tradingStatus
{'status': 'T', 'reason': '    ', 'timestamp': 1555500532036}

opHaltStatus
{'isHalted': False, 'timestamp': 1555500532036}

ssrStatus
{'isSSR': False, 'detail': ' ', 'timestamp': 1555500532036}

securityEvent
{'securityEvent': 'MarketClose', 'timestamp': 1555531200000}

trades
    isISO  isOddLot  isOutsideRegularHours  isSinglePriceCross  \
0    True      True                   True               False
1   False     False                  False               False
2    True     False                  False               False
3   False     False                  False               False
4    True     False                  False               False
5   False     False                  False               False
6   False     False                  False               False
7    True     False                  False               False
8    True     False                  False               False
9    True     False                  False               False
```

```
10  False      True                  False              False
11  False     False                  False              False
12   True      True                  False              False
13  False     False                  False              False
14  False     False                  False              False
15  False     False                  False              False
16   True      True                  False              False
17   True     False                  False              False
18   True      True                  False              False
19  False     False                  False              False

    isTradeThroughExempt    price  size      timestamp    tradeId
0                  False  203.190     3  1555531318248  891604355
1                  False  203.200   100  1555531197248  890272160
2                  False  203.140   100  1555531195857  889977488
3                  False  203.155   100  1555531195465  889859713
4                  False  203.140   100  1555531195292  889808657
5                  False  203.110   100  1555531194852  889619571
6                  False  203.100   100  1555531194822  889606541
7                  False  203.180   100  1555531194794  889585630
8                   True  203.180   100  1555531194794  889585556
9                  False  203.180   100  1555531191476  889002606
10                 False  203.190    25  1555531190110  888694678
11                 False  203.140   100  1555531187948  888262843
12                 False  203.160    64  1555531187684  888228377
13                 False  203.160   100  1555531187682  888227956
14                 False  203.160   100  1555531187681  888227469
15                 False  203.160   100  1555531187680  888227241
16                 False  203.160    36  1555531187680  888227079
17                 False  203.160   100  1555531187680  888227061
18                 False  203.170     9  1555531185364  887813837
19                 False  203.150   200  1555531184053  887531882

tradeBreaks
Empty DataFrame
Columns: []
Index: []
```

[12]:
```
pd.DataFrame(book['trades']).head()
```

[12]:
```
    isISO  isOddLot  isOutsideRegularHours  isSinglePriceCross  \
0   True      True                   True               False
1  False     False                  False               False
2   True     False                  False               False
3  False     False                  False               False
4   True     False                  False               False
```

	isTradeThroughExempt	price	size	timestamp	tradeId
0	False	203.190	3	1555531318248	891604355
1	False	203.200	100	1555531197248	890272160
2	False	203.140	100	1555531195857	889977488
3	False	203.155	100	1555531195465	889859713
4	False	203.140	100	1555531195292	889808657

1.2.3 Quandl

Obtain Quandl API Key and store in environment variable as `QUANDL_API_KEY`.

[1]:
```
symbol = 'FB.US'

quandl = web.DataReader(symbol, 'quandl', '2015-01-01')
quandl.info()
```

```
<class 'pandas.core.frame.DataFrame'>
DatetimeIndex: 813 entries, 2018-03-27 to 2015-01-02
Data columns (total 12 columns):
Open          813 non-null float64
High          813 non-null float64
Low           813 non-null float64
Close         813 non-null float64
Volume        813 non-null float64
ExDividend    813 non-null float64
SplitRatio    813 non-null float64
AdjOpen       813 non-null float64
AdjHigh       813 non-null float64
AdjLow        813 non-null float64
AdjClose      813 non-null float64
AdjVolume     813 non-null float64
dtypes: float64(12)
memory usage: 82.6 KB
```

1.2.4 FRED

[14]:
```
start = datetime(2010, 1, 1)

end = datetime(2013, 1, 27)

gdp = web.DataReader('GDP', 'fred', start, end)

gdp.info()
```

```
<class 'pandas.core.frame.DataFrame'>
DatetimeIndex: 13 entries, 2010-01-01 to 2013-01-01
Data columns (total 1 columns):
GDP    13 non-null float64
```

```
dtypes: float64(1)
memory usage: 208.0 bytes
```

[15]:
```
inflation = web.DataReader(['CPIAUCSL', 'CPILFESL'], 'fred', start, end)
inflation.info()
```

```
<class 'pandas.core.frame.DataFrame'>
DatetimeIndex: 37 entries, 2010-01-01 to 2013-01-01
Freq: MS
Data columns (total 2 columns):
CPIAUCSL    37 non-null float64
CPILFESL    37 non-null float64
dtypes: float64(2)
memory usage: 888.0 bytes
```

1.2.5 Fama/French

[16]:
```
from pandas_datareader.famafrench import get_available_datasets
get_available_datasets()
```

[16]:
```
['F-F_Research_Data_Factors',
 'F-F_Research_Data_Factors_weekly',
 'F-F_Research_Data_Factors_daily',
 'F-F_Research_Data_5_Factors_2x3',
 'F-F_Research_Data_5_Factors_2x3_daily',
 'Portfolios_Formed_on_ME',
 'Portfolios_Formed_on_ME_Wout_Div',
 'Portfolios_Formed_on_ME_Daily',
 'Portfolios_Formed_on_BE-ME',
 'Portfolios_Formed_on_BE-ME_Wout_Div',
 'Portfolios_Formed_on_BE-ME_Daily',
 'Portfolios_Formed_on_OP',
 'Portfolios_Formed_on_OP_Wout_Div',
 'Portfolios_Formed_on_INV',
 'Portfolios_Formed_on_INV_Wout_Div',
 '6_Portfolios_2x3',
 '6_Portfolios_2x3_Wout_Div',
 '6_Portfolios_2x3_weekly',
 '6_Portfolios_2x3_daily',
 '25_Portfolios_5x5',
 '25_Portfolios_5x5_Wout_Div',
 '25_Portfolios_5x5_Daily',
 '100_Portfolios_10x10',
 '100_Portfolios_10x10_Wout_Div',
 '100_Portfolios_10x10_Daily',
 '6_Portfolios_ME_OP_2x3',
 '6_Portfolios_ME_OP_2x3_Wout_Div',
```

```
'6_Portfolios_ME_OP_2x3_daily',
'25_Portfolios_ME_OP_5x5',
'25_Portfolios_ME_OP_5x5_Wout_Div',
'25_Portfolios_ME_OP_5x5_daily',
'100_Portfolios_ME_OP_10x10',
'100_Portfolios_10x10_ME_OP_Wout_Div',
'100_Portfolios_ME_OP_10x10_daily',
'6_Portfolios_ME_INV_2x3',
'6_Portfolios_ME_INV_2x3_Wout_Div',
'6_Portfolios_ME_INV_2x3_daily',
'25_Portfolios_ME_INV_5x5',
'25_Portfolios_ME_INV_5x5_Wout_Div',
'25_Portfolios_ME_INV_5x5_daily',
'100_Portfolios_ME_INV_10x10',
'100_Portfolios_10x10_ME_INV_Wout_Div',
'100_Portfolios_ME_INV_10x10_daily',
'25_Portfolios_BEME_OP_5x5',
'25_Portfolios_BEME_OP_5x5_Wout_Div',
'25_Portfolios_BEME_OP_5x5_daily',
'25_Portfolios_BEME_INV_5x5',
'25_Portfolios_BEME_INV_5x5_Wout_Div',
'25_Portfolios_BEME_INV_5x5_daily',
'25_Portfolios_OP_INV_5x5',
'25_Portfolios_OP_INV_5x5_Wout_Div',
'25_Portfolios_OP_INV_5x5_daily',
'32_Portfolios_ME_BEME_OP_2x4x4',
'32_Portfolios_ME_BEME_OP_2x4x4_Wout_Div',
'32_Portfolios_ME_BEME_INV_2x4x4',
'32_Portfolios_ME_BEME_INV_2x4x4_Wout_Div',
'32_Portfolios_ME_OP_INV_2x4x4',
'32_Portfolios_ME_OP_INV_2x4x4_Wout_Div',
'Portfolios_Formed_on_E-P',
'Portfolios_Formed_on_E-P_Wout_Div',
'Portfolios_Formed_on_CF-P',
'Portfolios_Formed_on_CF-P_Wout_Div',
'Portfolios_Formed_on_D-P',
'Portfolios_Formed_on_D-P_Wout_Div',
'6_Portfolios_ME_EP_2x3',
'6_Portfolios_ME_EP_2x3_Wout_Div',
'6_Portfolios_ME_CFP_2x3',
'6_Portfolios_ME_CFP_2x3_Wout_Div',
'6_Portfolios_ME_DP_2x3',
'6_Portfolios_ME_DP_2x3_Wout_Div',
'F-F_Momentum_Factor',
'F-F_Momentum_Factor_daily',
'6_Portfolios_ME_Prior_12_2',
'6_Portfolios_ME_Prior_12_2_Daily',
```

```
'25_Portfolios_ME_Prior_12_2',
'25_Portfolios_ME_Prior_12_2_Daily',
'10_Portfolios_Prior_12_2',
'10_Portfolios_Prior_12_2_Daily',
'F-F_ST_Reversal_Factor',
'F-F_ST_Reversal_Factor_daily',
'6_Portfolios_ME_Prior_1_0',
'6_Portfolios_ME_Prior_1_0_Daily',
'25_Portfolios_ME_Prior_1_0',
'25_Portfolios_ME_Prior_1_0_Daily',
'10_Portfolios_Prior_1_0',
'10_Portfolios_Prior_1_0_Daily',
'F-F_LT_Reversal_Factor',
'F-F_LT_Reversal_Factor_daily',
'6_Portfolios_ME_Prior_60_13',
'6_Portfolios_ME_Prior_60_13_Daily',
'25_Portfolios_ME_Prior_60_13',
'25_Portfolios_ME_Prior_60_13_Daily',
'10_Portfolios_Prior_60_13',
'10_Portfolios_Prior_60_13_Daily',
'Portfolios_Formed_on_AC',
'25_Portfolios_ME_AC_5x5',
'Portfolios_Formed_on_BETA',
'25_Portfolios_ME_BETA_5x5',
'Portfolios_Formed_on_NI',
'25_Portfolios_ME_NI_5x5',
'Portfolios_Formed_on_VAR',
'25_Portfolios_ME_VAR_5x5',
'Portfolios_Formed_on_RESVAR',
'25_Portfolios_ME_RESVAR_5x5',
'5_Industry_Portfolios',
'5_Industry_Portfolios_Wout_Div',
'5_Industry_Portfolios_daily',
'10_Industry_Portfolios',
'10_Industry_Portfolios_Wout_Div',
'10_Industry_Portfolios_daily',
'12_Industry_Portfolios',
'12_Industry_Portfolios_Wout_Div',
'12_Industry_Portfolios_daily',
'17_Industry_Portfolios',
'17_Industry_Portfolios_Wout_Div',
'17_Industry_Portfolios_daily',
'30_Industry_Portfolios',
'30_Industry_Portfolios_Wout_Div',
'30_Industry_Portfolios_daily',
'38_Industry_Portfolios',
'38_Industry_Portfolios_Wout_Div',
```

```
'38_Industry_Portfolios_daily',
'48_Industry_Portfolios',
'48_Industry_Portfolios_Wout_Div',
'48_Industry_Portfolios_daily',
'49_Industry_Portfolios',
'49_Industry_Portfolios_Wout_Div',
'49_Industry_Portfolios_daily',
'ME_Breakpoints',
'BE-ME_Breakpoints',
'OP_Breakpoints',
'INV_Breakpoints',
'E-P_Breakpoints',
'CF-P_Breakpoints',
'D-P_Breakpoints',
'Prior_2-12_Breakpoints',
'Global_3_Factors',
'Global_3_Factors_Daily',
'Global_ex_US_3_Factors',
'Global_ex_US_3_Factors_Daily',
'Europe_3_Factors',
'Europe_3_Factors_Daily',
'Japan_3_Factors',
'Japan_3_Factors_Daily',
'Asia_Pacific_ex_Japan_3_Factors',
'Asia_Pacific_ex_Japan_3_Factors_Daily',
'North_America_3_Factors',
'North_America_3_Factors_Daily',
'Global_5_Factors',
'Global_5_Factors_Daily',
'Global_ex_US_5_Factors',
'Global_ex_US_5_Factors_Daily',
'Europe_5_Factors',
'Europe_5_Factors_Daily',
'Japan_5_Factors',
'Japan_5_Factors_Daily',
'Asia_Pacific_ex_Japan_5_Factors',
'Asia_Pacific_ex_Japan_5_Factors_Daily',
'North_America_5_Factors',
'North_America_5_Factors_Daily',
'Global_Mom_Factor',
'Global_Mom_Factor_Daily',
'Global_ex_US_Mom_Factor',
'Global_ex_US_Mom_Factor_Daily',
'Europe_Mom_Factor',
'Europe_Mom_Factor_Daily',
'Japan_Mom_Factor',
'Japan_Mom_Factor_Daily',
```

```
'Asia_Pacific_ex_Japan_MOM_Factor',
'Asia_Pacific_ex_Japan_MOM_Factor_Daily',
'North_America_Mom_Factor',
'North_America_Mom_Factor_Daily',
'Global_6_Portfolios_ME_BE-ME',
'Global_6_Portfolios_ME_BE-ME_daily',
'Global_ex_US_6_Portfolios_ME_BE-ME',
'Global_ex_US_6_Portfolios_ME_BE-ME_daily',
'Europe_6_Portfolios_ME_BE-ME',
'Europe_6_Portfolios_ME_BE-ME_daily',
'Japan_6_Portfolios_ME_BE-ME',
'Japan_6_Portfolios_ME_BE-ME_daily',
'Asia_Pacific_ex_Japan_6_Portfolios_ME_BE-ME',
'Asia_Pacific_ex_Japan_6_Portfolios_ME_BE-ME_daily',
'North_America_6_Portfolios_ME_BE-ME',
'North_America_6_Portfolios_ME_BE-ME_daily',
'Global_25_Portfolios_ME_BE-ME',
'Global_25_Portfolios_ME_BE-ME_daily',
'Global_ex_US_25_Portfolios_ME_BE-ME',
'Global_ex_US_25_Portfolios_ME_BE-ME_daily',
'Europe_25_Portfolios_ME_BE-ME',
'Europe_25_Portfolios_ME_BE-ME_daily',
'Japan_25_Portfolios_ME_BE-ME',
'Japan_25_Portfolios_ME_BE-ME_daily',
'Asia_Pacific_ex_Japan_25_Portfolios_ME_BE-ME',
'Asia_Pacific_ex_Japan_25_Portfolios_ME_BE-ME_daily',
'North_America_25_Portfolios_ME_BE-ME',
'North_America_25_Portfolios_ME_BE-ME_daily',
'Global_6_Portfolios_ME_OP',
'Global_6_Portfolios_ME_OP_Daily',
'Global_ex_US_6_Portfolios_ME_OP',
'Global_ex_US_6_Portfolios_ME_OP_Daily',
'Europe_6_Portfolios_ME_OP',
'Europe_6_Portfolios_ME_OP_Daily',
'Japan_6_Portfolios_ME_OP',
'Japan_6_Portfolios_ME_OP_Daily',
'Asia_Pacific_ex_Japan_6_Portfolios_ME_OP',
'Asia_Pacific_ex_Japan_6_Portfolios_ME_OP_Daily',
'North_America_6_Portfolios_ME_OP',
'North_America_6_Portfolios_ME_OP_Daily',
'Global_25_Portfolios_ME_OP',
'Global_25_Portfolios_ME_OP_Daily',
'Global_ex_US_25_Portfolios_ME_OP',
'Global_ex_US_25_Portfolios_ME_OP_Daily',
'Europe_25_Portfolios_ME_OP',
'Europe_25_Portfolios_ME_OP_Daily',
'Japan_25_Portfolios_ME_OP',
```

```
'Japan_25_Portfolios_ME_OP_Daily',
'Asia_Pacific_ex_Japan_25_Portfolios_ME_OP',
'Asia_Pacific_ex_Japan_25_Portfolios_ME_OP_Daily',
'North_America_25_Portfolios_ME_OP',
'North_America_25_Portfolios_ME_OP_Daily',
'Global_6_Portfolios_ME_INV',
'Global_6_Portfolios_ME_INV_Daily',
'Global_ex_US_6_Portfolios_ME_INV',
'Global_ex_US_6_Portfolios_ME_INV_Daily',
'Europe_6_Portfolios_ME_INV',
'Europe_6_Portfolios_ME_INV_Daily',
'Japan_6_Portfolios_ME_INV',
'Japan_6_Portfolios_ME_INV_Daily',
'Asia_Pacific_ex_Japan_6_Portfolios_ME_INV',
'Asia_Pacific_ex_Japan_6_Portfolios_ME_INV_Daily',
'North_America_6_Portfolios_ME_INV',
'North_America_6_Portfolios_ME_INV_Daily',
'Global_25_Portfolios_ME_INV',
'Global_25_Portfolios_ME_INV_Daily',
'Global_ex_US_25_Portfolios_ME_INV',
'Global_ex_US_25_Portfolios_ME_INV_Daily',
'Europe_25_Portfolios_ME_INV',
'Europe_25_Portfolios_ME_INV_Daily',
'Japan_25_Portfolios_ME_INV',
'Japan_25_Portfolios_ME_INV_Daily',
'Asia_Pacific_ex_Japan_25_Portfolios_ME_INV',
'Asia_Pacific_ex_Japan_25_Portfolios_ME_INV_Daily',
'North_America_25_Portfolios_ME_INV',
'North_America_25_Portfolios_ME_INV_Daily',
'Global_6_Portfolios_ME_Prior_12_2',
'Global_6_Portfolios_ME_Prior_250_20_daily',
'Global_ex_US_6_Portfolios_ME_Prior_12_2',
'Global_ex_US_6_Portfolios_ME_Prior_250_20_daily',
'Europe_6_Portfolios_ME_Prior_12_2',
'Europe_6_Portfolios_ME_Prior_250_20_daily',
'Japan_6_Portfolios_ME_Prior_12_2',
'Japan_6_Portfolios_ME_Prior_250_20_daily',
'Asia_Pacific_ex_Japan_6_Portfolios_ME_Prior_12_2',
'Asia_Pacific_ex_Japan_6_Portfolios_ME_Prior_250_20_daily',
'North_America_6_Portfolios_ME_Prior_12_2',
'North_America_6_Portfolios_ME_Prior_250_20_daily',
'Global_25_Portfolios_ME_Prior_12_2',
'Global_25_Portfolios_ME_Prior_250_20_daily',
'Global_ex_US_25_Portfolios_ME_Prior_12_2',
'Global_ex_US_25_Portfolios_ME_Prior_250_20_daily',
'Europe_25_Portfolios_ME_Prior_12_2',
'Europe_25_Portfolios_ME_Prior_250_20_daily',
```

```
 'Japan_25_Portfolios_ME_Prior_12_2',
 'Japan_25_Portfolios_ME_Prior_250_20_daily',
 'Asia_Pacific_ex_Japan_25_Portfolios_ME_Prior_12_2',
 'Asia_Pacific_ex_Japan_25_Portfolios_ME_Prior_250_20_daily',
 'North_America_25_Portfolios_ME_Prior_12_2',
 'North_America_25_Portfolios_ME_Prior_250_20_daily',
 'Global_32_Portfolios_ME_BE-ME_OP_2x4x4',
 'Global_ex_US_32_Portfolios_ME_BE-ME_OP_2x4x4',
 'Europe_32_Portfolios_ME_BE-ME_OP_2x4x4',
 'Japan_32_Portfolios_ME_BE-ME_OP_2x4x4',
 'Asia_Pacific_ex_Japan_32_Portfolios_ME_BE-ME_OP_2x4x4',
 'North_America_32_Portfolios_ME_BE-ME_OP_2x4x4',
 'Global_32_Portfolios_ME_BE-ME_INV(TA)_2x4x4',
 'Global_ex_US_32_Portfolios_ME_BE-ME_INV(TA)_2x4x4',
 'Europe_32_Portfolios_ME_BE-ME_INV(TA)_2x4x4',
 'Japan_32_Portfolios_ME_BE-ME_INV(TA)_2x4x4',
 'Asia_Pacific_ex_Japan_32_Portfolios_ME_BE-ME_INV(TA)_2x4x4',
 'North_America_32_Portfolios_ME_BE-ME_INV(TA)_2x4x4',
 'Global_32_Portfolios_ME_INV(TA)_OP_2x4x4',
 'Global_ex_US_32_Portfolios_ME_INV(TA)_OP_2x4x4',
 'Europe_32_Portfolios_ME_INV(TA)_OP_2x4x4',
 'Japan_32_Portfolios_ME_INV(TA)_OP_2x4x4',
 'Asia_Pacific_ex_Japan_32_Portfolios_ME_INV(TA)_OP_2x4x4',
 'North_America_32_Portfolios_ME_INV(TA)_OP_2x4x4']
```

[17]:
```
ds = web.DataReader('5_Industry_Portfolios', 'famafrench')
print(ds['DESCR'])
```

```
5 Industry Portfolios
---------------------

This file was created by CMPT_IND_RETS using the 201901 CRSP database. It
contains value- and equal-weighted returns for 5 industry portfolios. The
portfolios are constructed at the end of June. The annual returns are from
January to December. Missing data are indicated by -99.99 or -999. Copyright
2019 Kenneth R. French

 0 : Average Value Weighted Returns -- Monthly (109 rows x 5 cols)
 1 : Average Equal Weighted Returns -- Monthly (109 rows x 5 cols)
 2 : Average Value Weighted Returns -- Annual (9 rows x 5 cols)
 3 : Average Equal Weighted Returns -- Annual (9 rows x 5 cols)
 4 : Number of Firms in Portfolios (109 rows x 5 cols)
 5 : Average Firm Size (109 rows x 5 cols)
 6 : Sum of BE / Sum of ME (9 rows x 5 cols)
 7 : Value-Weighted Average of BE/ME (9 rows x 5 cols)
```

1.2.6 World Bank

[16]:
```
from pandas_datareader import wb
gdp_variables = wb.search('gdp.*capita.*const')
gdp_variables.head()
```

[16]:
```
                        id                                                 name  \
646     6.0.GDPpc_constant  GDP per capita, PPP (constant 2011 internation…
9108        NY.GDP.PCAP.KD                 GDP per capita (constant 2010 US$)
9110        NY.GDP.PCAP.KN                      GDP per capita (constant LCU)
9112     NY.GDP.PCAP.PP.KD  GDP per capita, PPP (constant 2011 internation…
9113  NY.GDP.PCAP.PP.KD.87  GDP per capita, PPP (constant 1987 internation…

                            source  \
646                 LAC Equity Lab
9108  World Development Indicators
9110  World Development Indicators
9112  World Development Indicators
9113         WDI Database Archives

                                           sourceNote  \
646   GDP per capita based on purchasing power parit…
9108  GDP per capita is gross domestic product divid…
9110  GDP per capita is gross domestic product divid…
9112  GDP per capita based on purchasing power parit…
9113

                                    sourceOrganization            topics unit
646        b'World Development Indicators (World Bank)'  Economy & Growth
9108  b'World Bank national accounts data, and OECD …  Economy & Growth
9110  b'World Bank national accounts data, and OECD …  Economy & Growth
9112  b'World Bank, International Comparison Program…  Economy & Growth
9113                                              b''
```

[18]:
```
wb_data = wb.download(indicator='NY.GDP.PCAP.KD',
                      country=['US', 'CA', 'MX'],
                      start=1990,
                      end=2019)
wb_data.head()
```

[18]:
```
                NY.GDP.PCAP.KD
country year
Canada  2018             NaN
        2017    51315.888975
        2016    50407.341330
        2015    50303.836848
        2014    50221.841982
```

1.2.7 OECD

[20]:
```
df = web.DataReader('TUD', 'oecd', end='2015')
df[['Japan', 'United States']]
```

[20]:
```
Country            Japan                                                  \
Frequency         Annual
Source       Survey data
Series     Union members            Trade union  density            Employees
Measure        Thousands Percentage            Thousands Percentage Thousands
Year
2010-01-01           NaN        NaN                  NaN        NaN       NaN
2011-01-01           NaN        NaN                  NaN        NaN       NaN
2012-01-01           NaN        NaN                  NaN        NaN       NaN
2013-01-01           NaN        NaN                  NaN        NaN       NaN
2014-01-01           NaN        NaN                  NaN        NaN       NaN
2015-01-01           NaN        NaN                  NaN        NaN       NaN

Country                                                                    \
Frequency
Source                Administrative data
Series                      Union members            Trade union  density
Measure    Percentage           Thousands Percentage            Thousands
Year
2010-01-01        NaN           12417.527        NaN                  NaN
2011-01-01        NaN           12271.909        NaN                  NaN
2012-01-01        NaN           12227.073        NaN                  NaN
2013-01-01        NaN           12396.592        NaN                  NaN
2014-01-01        NaN            7661.568        NaN                  NaN
2015-01-01        NaN            8359.876        NaN                  NaN

Country               …        United States                       \
Frequency             …                Annual
Source                …          Survey data
Series                … Trade union  density            Employees
Measure    Percentage …            Thousands Percentage Thousands
Year                  …
2010-01-01  28.871256 …                  NaN  17.448617   97406.0
2011-01-01  27.589723 …                  NaN  16.516118  102403.0
2012-01-01  25.899329 …                  NaN  15.861734  106924.0
2013-01-01  24.489514 …                  NaN  15.510448  107102.0
2014-01-01  32.164433 …                  NaN        NaN       NaN
2015-01-01  34.516416 …                  NaN        NaN       NaN

Country                                                                    \
Frequency
Source                Administrative data
```

```
Series                      Union members            Trade union  density
Measure    Percentage           Thousands Percentage            Thousands
Year
2010-01-01        NaN                 NaN        NaN                  NaN
2011-01-01        NaN                 NaN        NaN                  NaN
2012-01-01        NaN                 NaN        NaN                  NaN
2013-01-01        NaN                 NaN        NaN                  NaN
2014-01-01        NaN             17049.0        NaN                  NaN
2015-01-01        NaN             16303.0        NaN                  NaN

Country
Frequency
Source
Series                 Employees
Measure    Percentage Thousands Percentage
Year
2010-01-01        NaN       NaN        NaN
2011-01-01        NaN       NaN        NaN
2012-01-01        NaN       NaN        NaN
2013-01-01        NaN       NaN        NaN
2014-01-01  30.897064   55180.0        NaN
2015-01-01  29.518378   55230.0        NaN

[6 rows x 24 columns]
```

1.2.8 EuroStat

[21]:
```
df = web.DataReader('tran_sf_railac', 'eurostat')
```

[22]:
```
df.head()
```

[22]:
```
ACCIDENT    Collisions of trains, including collisions with obstacles within the
clearance gauge  \
UNIT
Number
GEO
Austria
FREQ
Annual
TIME_PERIOD
2010-01-01                                                    3.0
2011-01-01                                                    2.0
2012-01-01                                                    1.0
2013-01-01                                                    4.0
2014-01-01                                                    1.0

ACCIDENT                                                        \
```

```
UNIT
GEO         Belgium Bulgaria Switzerland Channel Tunnel Czechia
FREQ         Annual   Annual      Annual         Annual  Annual
TIME_PERIOD
2010-01-01      5.0      2.0         5.0            0.0     3.0
2011-01-01      0.0      0.0         4.0            0.0     6.0
2012-01-01      3.0      3.0         4.0            0.0     6.0
2013-01-01      1.0      2.0         6.0            0.0     5.0
2014-01-01      3.0      4.0         0.0            0.0    13.0

ACCIDENT                                                                \
UNIT
GEO         Germany (until 1990 former territory of the FRG) Denmark Estonia
FREQ                                                  Annual  Annual  Annual
TIME_PERIOD
2010-01-01                                              13.0     0.0     1.0
2011-01-01                                              18.0     1.0     0.0
2012-01-01                                              23.0     1.0     3.0
2013-01-01                                              29.0     0.0     0.0
2014-01-01                                              32.0     0.0     0.0

ACCIDENT            …     Unknown                                       \
UNIT                …      Number
GEO         Greece  … Netherlands Norway Poland Portugal Romania Sweden
FREQ        Annual  …      Annual Annual Annual   Annual  Annual Annual
TIME_PERIOD         …
2010-01-01     4.0  …         NaN    NaN    NaN      NaN     NaN    NaN
2011-01-01     1.0  …         NaN    NaN    NaN      NaN     NaN    NaN
2012-01-01     2.0  …         NaN    NaN    NaN      NaN     NaN    NaN
2013-01-01     2.0  …         NaN    NaN    NaN      NaN     NaN    NaN
2014-01-01     1.0  …         NaN    NaN    NaN      NaN     NaN    NaN

ACCIDENT
UNIT
GEO         Slovenia Slovakia Turkey United Kingdom
FREQ          Annual   Annual Annual         Annual
TIME_PERIOD
2010-01-01       NaN      NaN    0.0            NaN
2011-01-01       NaN      NaN    0.0            NaN
2012-01-01       NaN      NaN    0.0            NaN
2013-01-01       NaN      NaN    0.0            NaN
2014-01-01       NaN      NaN    0.0            NaN

[5 rows x 264 columns]
```

1.2.9 Stooq

Google finance stopped providing common index data download. The Stooq site had this data for download for a while but is currently broken, awaiting release of fix

[13]:
```
index_url = 'https://stooq.com/t/'
ix = pd.read_html(index_url)
len(ix)
```

[13]: 46

[14]:
```
f = web.DataReader('^SPX', 'stooq', start='20000101')
f.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 0 entries
Empty DataFrame
```

[15]:
```
f.head()
```

[15]:
```
Empty DataFrame
Columns: []
Index: []
```

1.2.10 NASDAQ Symbols

[23]:
```
from pandas_datareader.nasdaq_trader import get_nasdaq_symbols
symbols = get_nasdaq_symbols()
symbols.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 8701 entries, A to ZYXI
Data columns (total 11 columns):
Nasdaq Traded       8701 non-null bool
Security Name       8701 non-null object
Listing Exchange    8701 non-null category
Market Category     8701 non-null object
ETF                 8701 non-null bool
Round Lot Size      8701 non-null float64
Test Issue          8701 non-null bool
Financial Status    3411 non-null category
CQS Symbol          5290 non-null object
NASDAQ Symbol       8701 non-null object
NextShares          8701 non-null bool
dtypes: bool(4), category(2), float64(1), object(4)
memory usage: 459.2+ KB
```

[24]:
```
url = 'https://www.nasdaq.com/screening/companies-by-industry.aspx?
 ↪exchange=NASDAQ'
res = pd.read_html(url)
len(res)
```

[24]: 4

[25]:
```
for r in res:
    print(r.info())
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1 entries, 0 to 0
Data columns (total 2 columns):
0    1 non-null object
1    1 non-null object
dtypes: object(2)
memory usage: 96.0+ bytes
None
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 101 entries, 0 to 100
Data columns (total 6 columns):
Name          101 non-null object
Symbol        51 non-null object
Market Cap    47 non-null object
Country       51 non-null object
IPO Year      28 non-null object
Subsector     51 non-null object
dtypes: object(6)
memory usage: 4.8+ KB
None
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1 entries, 0 to 0
Data columns (total 1 columns):
0    1 non-null object
dtypes: object(1)
memory usage: 88.0+ bytes
None
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1 entries, 0 to 0
Data columns (total 1 columns):
0    1 non-null object
dtypes: object(1)
memory usage: 88.0+ bytes
None
```

1.2.11 Tiingo

Requires signing up and storing API key in environment

[26]:
```
df = web.get_data_tiingo('GOOG', api_key=os.getenv('TIINGO_API_KEY'))
```

[27]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
MultiIndex: 1244 entries, (GOOG, 2014-03-27 00:00:00) to (GOOG, 2019-03-06
00:00:00)
Data columns (total 12 columns):
adjClose       1244 non-null float64
adjHigh        1244 non-null float64
adjLow         1244 non-null float64
adjOpen        1244 non-null float64
adjVolume      1244 non-null int64
close          1244 non-null float64
divCash        1244 non-null float64
high           1244 non-null float64
low            1244 non-null float64
open           1244 non-null float64
splitFactor    1244 non-null float64
volume         1244 non-null int64
dtypes: float64(10), int64(2)
memory usage: 130.1+ KB
```