Yahoo_fin패키지

Yahoo_fin은 시가총액, 배당수익률, 종목에 대한 최신 정보를 제공하는 Python 3 패키지다. 그외에 재무상태표, 손익계산서, 현금흐름표 등을 제공한된다. 패키지에는 현재 거래일에 실시간 주가, 암호화폐 데이터, 거래상위종목을 얻을 수 있는 기능이 포함되어 있다.

yahoo_fin은 stock_info module, options module 두 가지 모듈로 구성된다.

1.설치 및 업그레이드

- pip install yahoo_fin
- pip install yahoo_fin --upgrade

2.필요 패키지

Yahoo_fin을 사용하려면 다음 패키지가 미리 필요하다.

- ftplib
- io
- pandas
- requests
- requests_html

아나콘다를 설치하면 대부분 설치되지만 requests_html 는 빠지므로 다음과 같이 추가 설치한다.

pip install requests_html

3.제공함수

yahoo_fin 는 stock_info와 options 두 개의 모듈로 구성된다.
stock_info는 다음과 같은 함수를 제공한다.

- get_analysts_info
- get_balance_sheet
- get_cash_flow
- get_data
- get_day_gainers
- get_day_losers
- get_day_most_active
- get_holders
- get_income_statement

- get_live_price
- get_quote_table
- get_top_crypto
- get_stats
- get_stats_valuation
- tickers_dow
- tickers_nasdaq
- tickers_other
- tickers_sp500

options는 다음과 같은 함수를 제공한다.

- get_calls
- get_expiration_dates
- get_options_chain
- get_puts

4.stock_info 모듈(module)

4.01.패키지 임포트

패키지는 다음과 같이 전체를 임포트하거나

```
from yahoo_fin.stock_info import *
```

또는 다음과 같이 필요한 함수(예를 들어 다음과 같이get_analysts_info함수)만 임포트할 수 있다.

```
from yahoo_fin.stock_info import get_analysts_info
```

4.02.get_analysts_info(ticker)

get_analysts_info()함수는 Yahoo Finance의 Analysts 페이지를 가져온다(e.g. https://finance.yahoo.com/quote/NFLX/analysts?p=NFLX)

```
get_analysts_info('nflx')
```

4.03.get_balance_sheet(ticker)

get_balance_sheet()함수는 재무상태표(e.g. https://finance.yahoo.com/quote/NFLX/balance-sheet?p=NFLX.)를 가져온다.

```
get_balance_sheet('nflx')
```

4.04.get_cash_flow(ticker)

get_cash_flow()함수는 현금흐름표를 가져온다(e.g. https://finance.yahoo.com/quote/NFLX/cash-flow?p=NFLX.)

```
get_cash_flow('nflx')
```

4.05.get_data(ticker, start_date = None, end_date = None, index_as_date = True, interval = “1d”)

get_data()함수는 일간,주간,월간 과거 주가데이터를 가져온다

- ticker :
  종목티커
- start_date :
  시작일
- end_date :
  마지막일
- index_as_date :
  기본값은 True. index_as_date = True이면 날짜는 인덱스로 사용
- interval :
  기본값은 "1d"(일간), 주간("1wk"), 월간("1mo")

```
msft_data = get_data('msft')
from1999 = get_data('msft' , start_date = '01/01/1999')
few_days = get_data('msft' , start_date = '01/01/1999' , end_date = '01/10/1999')
weekly_data = get_data("msft", interval = "1wk")
monthly_data = get_data("msft", interval = "1mo")
```

4.06.get_day_gainers()

해당 매매일 상위 100개의 상승종목을 가져온다 (예, https://finance.yahoo.com/gainers).

```
get_day_gainers()
```

4.07.get_day_losers()

해당 매매일 상위 100개의 하락종목을 가져온다 (예, https://finance.yahoo.com/losers).

```
get_day_losers()
```

4.08.get_day_most_active()

해당 매매일 상위 100개의 거래종목을 가져온다 (예, https://finance.yahoo.com/most-active).

```
get_day_most_active()
```

4.09.get_holders(ticker)

Yahoo Finance의 Holders섹션을 가져온다 (e.g. https://finance.yahoo.com/quote/NFLX/holders?p=NFLX)

```
get_holders('nflx')
```

4.10.get_income_statement(ticker)

손익계산서(income statement)를 가져온다(e.g. https://finance.yahoo.com/quote/NFLX/financials?p=NFLX)

```
get_income_statement('nflx')
```

411.get_live_price(ticker)

종목의 실시간 호가가격을 가져온다.

```
get_live_price('nflx')
```

4.12.get_quote_table(ticker , dict_result = True)

Yahoo Finance의 호가정보 테이블을 가져온다 (e.g. https://finance.yahoo.com/quote/AAPL?p=AAPL)

- ticker :
  종목티커
- dict_result :
  기본값은 True. True이면 딕셔너리를 돌려주고, 아니면 데이터프레임을 돌려준다

돌려주는 값은 다음과 같다.

- 1y Target Est
- 52 Week Range
- Ask
- Volume
- Beta
- Bid
- Days Range
- Dividend & Yield
- EPS (TTM)
- Earnings Date
- Ex-Dividend Date
- Market Cap
- Open
- PE Ratio (TTM)
- Previous Close
- Quote Price
- Volume

```
get_quote_table('aapl')
```

4.13.get_top_crypto(ticker)

시총별 100개의 암호화 화폐 데이터를 가져온다(예, https://finance.yahoo.com/cryptocurrencies).

```
get_top_crypto('BTC-USD')
```

4.14.get_stats(ticker)

종목의 통계정보를 가져온(e.g. https://finance.yahoo.com/quote/NFLX/key-statistics?p=NFLX.)

```
get_stats('nflx')
```

4.15.get_stats_valuation(ticker)

종목의 통계정보중 “Valuation Measures”정보를 가져온다 (e.g. https://finance.yahoo.com/quote/NFLX/key-statistics?p=NFLX.

```
get_stats_valuation('nflx')
```

4.16.tickers_dow()

다우존스(Dow Jones) 구성종목 티커를 가져온다.(see https://finance.yahoo.com/quote/%5EDJI/components?p=%5EDJI.)

```
tickers = tickers_dow()
```

4.17.tickers_nasdaq()

나스닥(NASDAQ)구성종목티커를 가져온다

```
tickers = tickers_nasdaq()
```

4.18.tickers_other()

나스닥종목 설명

```
tickers = tickers_other()
```

4.19.tickers_sp500()

S&P 500 구성종목을 돌려준다(https://en.wikipedia.org/wiki/List_of_S%26P_500_companies)

```
tickers = tickers_sp500()
```

5.options 모듈(module)

options 모듈(module)의 제공 함수는 다음과 같다. 다음과 같이 options 모듈(module)을 임포트할 수 있다.

```
from yahoo_fin.options import *
from yahoo_fin.options import get_options_chain
```

5.01.get_calls(ticker, date = None)

Yahoo Finance 의 종목옵션 데이터를 가져온다(e.g. https://finance.yahoo.com/quote/NFLX/options?p=NFLX.)

- ticker :
  종목 티커
- date :
  옵션만기일(Expiration date) date매개변수는 선택사항이며 기본값은 None이다.

```
get_calls('nflx')
get_calls('nflx', '06/19/2020')
```

5.02.get_expiration_dates(ticker)

Yahoo Finance 옵션 만기일 정보를 가져온다(e.g. https://finance.yahoo.com/quote/NFLX/options?p=NFLX.)

```
get_expiration_dates('nflx')
get_expiration_dates('amzn')
```

5.03.get_options_chain(ticker, date)

Yahoo Finance의 콜옵션과 풋옵션 테이블을 가져온다(e.g. https://finance.yahoo.com/quote/NFLX/options?p=NFLX.)

- ticker :
  종목티커
- date :
  옵션만기일, 선택사항이면 기본값은 None이다. 생략하면 최근월물을 가져온다.

```
get_options_chain('nflx')
get_options_chain('amzn', '03/15/2019')
```

5.04.get_puts(ticker, date = None)

풋옵션 데이터를 가져온다 (e.g. https://finance.yahoo.com/quote/NFLX/options?p=NFLX.)

- ticker :
  종목티커
- date :
  만기일, 선택사항이며 기본값은 None이다. 생략하면 최근월물을 돌려준다

```
get_puts('nflx')
get_puts('nflx', '06/19/2020')
```

6.사용 예

파이썬으로 재무정보를 다운받는 방법을 소개하고자 한다. 재무데이터는 야후파이낸스의 자료이며 yhaoo_fin패키지를 통해 얻게 된다. yhaoo_fin패키지에 대한 자세한 내용은 YAHOO_FIN DOCUMENTATION(theautomatic.net/yahoo_fin-documentation/)에서 참조하길 바란다.

6.01.시작하기

가장 먼저 할 일은 yahoo_fin에서 stock_info모듈을 임포트하는 것이다. 이 모듈과 함께 같이 필요한 것이 pandas패키지이다.

```
import yahoo_fin.stock_info as si
import pandas as pd
```

6.02.주가수익률(Price-to-Earnings) 비율 구하기

PER을 구하는 방법은 몇 가지 있는 데, 다음은 get_quote_table 메서드를 이용한 방법이다.

```
quote = si.get_quote_table("aapl")
```

get_quote_table 메서드는 다음과 같이 여러 정보가 담긴 딕셔너리를 돌려준다.

```
{'1y Target Est': 303.56,
 '52 Week Range': '170.27 - 327.85',
 'Ask': '290.78 x 1400',
 'Avg. Volume': 50819157.0,
 'Beta (5Y Monthly)': 1.17,
 'Bid': '290.20 x 1200',
 "Day's Range": '285.85 - 299.00',
 'EPS (TTM)': 12.73,
 'Earnings Date': 'Jul 28, 2020 - Aug 03, 2020',
 'Ex-Dividend Date': 'Feb 07, 2020',
 'Forward Dividend & Yield': '3.08 (1.05%)',
 'Market Cap': '1.265T',
 'Open': 286.25,
 'PE Ratio (TTM)': 22.71,
 'Previous Close': 293.8,
 'Quote Price': 289.07000732421875,
 'Volume': 60154175.0}
```

따라서 다음과 같이 키 값(여기선 "PE Ratio (TTM)")을 사용하여 값을 얻을 수 있다.

```
quote["PE Ratio (TTM)"] # 22.71
```

PER을 구하는 또 다른 방법은 get_stats_valuation 메서드를 이용하는 것이다.
이것은 야후파이낸스의 종목 통계섹션의 내용이다. 다음은 애플의 경우이다.

```
val = si.get_stats_valuation("aapl")
val = val.iloc[:,:2]
val.columns = ["Attribute", "Recent"]
```

이제 P/E 비율을 추출하면 다음과 같다.

```
float(val[val.Attribute.str.contains("Trailing P/E")].iloc[0,1])
```

6.03.주가매출액(Price-to-Sales) 비율

인기있는 재무비율중 하나가 P/S 비율인데, 앞서와 마찬가지로 get_stats_valuation 메서드를 사용하여 얻을 수 있다.

```
float(val[val.Attribute.str.contains("Price/Sales")].iloc[0,1])
```

6.04.한번에 여러 재무 데이터 구하기

다우지수내 각 종목의 Price-to-Earnings 와 Price-to-Sales 비율을 구해보도록 하자. tickers_dow메서드는 지수를 구성하는 종목의 티커를 돌려준다.

```
# get list of Dow tickers
dow_list = si.tickers_dow()

# Get data in the current column for each stock's valuation table
dow_stats = {}
for ticker in dow_list:
    temp = si.get_stats_valuation(ticker)
    temp = temp.iloc[:,:2]
    temp.columns = ["Attribute", "Recent"]

    dow_stats[ticker] = temp

# combine all the stats valuation tables into a single data frame
combined_stats = pd.concat(dow_stats)
combined_stats = combined_stats.reset_index()

del combined_stats["level_1"]

# update column names
combined_stats.columns = ["Ticker", "Attribute", "Recent"]
```

6.04.1.각 종목의 주가수익률(P/E) 비율

```
# get P/E ratio for each stock
combined_stats[combined_stats.Attribute.str.contains("Trailing P/E")]
```

6.04.2.주가매출액비율(Price-to-Sales) 비율

```
# get P/S ratio for each stock
combined_stats[combined_stats.Attribute.str.contains("Price/Sales")]
```

6.04.3.주가순자산(Price / Book) 비율

```
# get Price-to-Book ratio for each stock
combined_stats[combined_stats.Attribute.str.contains("Price/Book")]
```

6.04.5.주가이익성장배율(Price / Earnings-to-Growth) 비율

```
# get PEG ratio for each stock
combined_stats[combined_stats.Attribute.str.contains("PEG")]
```

6.04.6.Forward P/E 비율

```
# get forward P/E ratio for each stock
combined_stats[combined_stats.Attribute.str.contains("Forward P/E")]
```

6.05.여러 종목의 기타 통계 구하기

야후파이낸스 종목 통계섹션에는“Valuation Measures” 테이블이 있는데 get_stats method를 통해 기타 통계 정보(Return on Equity (ROE), Return on Assets, profit margin)를 구할 수 있다.

```
dow_extra_stats = {}
for ticker in dow_list:
    dow_extra_stats[ticker] = si.get_stats(ticker)

combined_extra_stats = pd.concat(dow_extra_stats)

combined_extra_stats = combined_extra_stats.reset_index()

del combined_extra_stats["level_1"]

combined_extra_stats.columns = ["ticker", "Attribute", "Value"]
```

6.05.1.자기자본이익률(ROE)

```
combined_extra_stats[combined_extra_stats.Attribute.str.contains("Return
on Equity")]
```

6.05.2.총자산이익률(ROA)

```
combined_extra_stats[combined_extra_stats.Attribute.str.contains("Return
on Assets")]
```

6.05.3.이익률(profit margin)

```
combined_extra_stats[combined_extra_stats.Attribute.str.contains("Profit
Margin")]
```

6.06.재무상태표(balance sheets)

get_balance_sheet 메서드를 사용하여 재무상태표를 구할 수 있는 데, 유동현금, 자산, 부채등을 알 수 있다.

```
sheet = si.get_balance_sheet("aapl")
```

6.06.1.총현금(Total cash on hand)

```
sheet[sheet.Breakdown == "Total Cash"]
```

6.06.2.자본(Stockholders’ equity)

```
sheet[sheet.Breakdown == "Total stockholders' equity"]
```

6.06.3.총자산(Total Assets)

```
sheet[sheet.Breakdown == "Total Assets"]
```

6.07.여러 종목의 재무상태표 구하기

다우지수 구성종목들의 재무상태표를 다음과 같이 얻을 수 있다.

```
balance_sheets = {}
for ticker in dow_list:
    balance_sheets[ticker] = si.get_balance_sheet(ticker)
```

다음의 코드는 각 종목의 재무상태표를 묶어서 최근 데이터를 보여준다,

```
recent_sheets = {ticker : sheet.iloc[:,:2] for ticker,sheet in
balance_sheets.items()}

for ticker in recent_sheets.keys():
    recent_sheets[ticker].columns = ["Breakdown", "Recent"]

# combine all balance sheets together
combined_sheets = pd.concat(recent_sheets)

# reset index to pull in ticker
combined_sheets = combined_sheets.reset_index()

# get rid of numeric index field
del combined_sheets["level_1"]
```

```
# update column names
combined_sheets.columns = ["Ticker", "Breakdown", "Recent"]
```

Ticker	Breakdown	Recent
AAPL	Total Assets	338516000
AAPL	Total Liabilities Net Minority Interest	248028000
AAPL	Total Equity Gross Minority Interest	90488000
AAPL	Total Capitalization	182295000
AAPL	Common Stock Equity	90488000
...	...	...
XOM	Common Stock	15637000
XOM	Retained Earnings	421341000
XOM	Accumulated other comprehensive income	-19493000
XOM	Total stockholders' equity	191650000
XOM	Total liabilities and stockholders' equity	362597000

6.07.1.매출액(Total Assets)

```
combined_sheets[combined_sheets.Breakdown == "Total Assets"]
```

6.08.손익계산서(income statements)

손익계산서는 get_income_statement 메서드를 사용하여 얻을 수 있다

```
income=si.get_income_statement("aapl")
```

손익계산서의 매출액(total revenue), 매출총이익(gross profit), 총비용(total expenses) 같은 항목은 다음과 같이 얻는다.

6.08.1.매출액(total revenue)

```
income[income.Breakdown == "Total Revenue"]
```

6.08.2.매출총이익(gross profit)

```
income[income.Breakdown == "Gross Profit"]
```

6.09.여러 종목의 손익계산서 구하기

```
income_statements = {}
for ticker in dow_list:
    income_statements[ticker] = si.get_income_statement(ticker)
```

재무상태표의 경우처럼 각 종목의 손익계산서도 하나로 묶어 필요한 계정만 볼 수 있다.

```
recent_income_statements = {ticker : sheet.iloc[:,:2] for ticker,sheet in income_statements.items()}

for ticker in recent_income_statements.keys():
    recent_income_statements[ticker].columns = ["Breakdown", "Recent"]

combined_income = pd.concat(recent_income_statements)

combined_income = combined_income.reset_index()

del combined_income["level_1"]

combined_income.columns = ["Ticker", "Breakdown", "Recent"]
```

Ticker	Breakdown	Recent
AAPL	Total Revenue	267683000
AAPL	Cost of Revenue	166105000
AAPL	Gross Profit	101578000
AAPL	Research Development	16766000
AAPL	Selling General and Administrative	18659000
...	...	...
XOM	Net Income available to common shareholders	14340000
XOM	Basic EPS	-
XOM	Diluted EPS	-
XOM	Basic Average Shares	-
XOM	Diluted Average Shares	-

6.09.1.매출액(Total Revenue)

```
combined_income[combined_income.Breakdown == "Total Revenue"]
```

6.10.현금흐름표(cash flow statements)

현금흐름표(cash flow statements)는 get_cash_flow 메서드를 사용하여 구한다

```
flow = si.get_cash_flow("aapl")
```

	Breakdown	ttm	...	9/30/2017	9/30/2016
0	Net Income	57527000	...	48351000	45687000
1	Depreciation & amortization	11968000	...	10157000	10505000
2	Deferred income taxes	-742000	...	5966000	4938000
3	Stock based compensation	6219000	...	4840000	4210000
4	Change in working capital	-1015000	...	-5550000	484000

여러 종목의 경우 다음과 같이 구할 수 있다.

```
cash_flows = {}
for ticker in dow_list:
    cash_flows[ticker] = si.get_cash_flow(ticker)
```

그리고 여러 종목의 현금흐름표를 다음과 같이 묶을 수 있다.

```
recent_cash_flows = {ticker : flow.iloc[:,:2] for ticker,flow in
cash_flows.items()}

for ticker in recent_cash_flows.keys():
    recent_cash_flows[ticker].columns = ["Breakdown", "Recent"]

combined_cash_flows = pd.concat(recent_cash_flows)

combined_cash_flows = combined_cash_flows.reset_index()

del combined_cash_flows["level_1"]

combined_cash_flows.columns = ["Ticker", "Breakdown", "Recent"]
```

6.10.1.잉여현금흐름(free cash flow)

```
combined_cash_flows[combined_cash_flows.Breakdown == "Free Cash Flow"]
```

Ticker	Breakdown	Recent
AAPL	Free Cash Flow	63970000
AXP	Free Cash Flow	1234000
BA	Free Cash Flow	-11424000
CAT	Free Cash Flow	4251000
CSCO	Free Cash Flow	14831000
CVX	Free Cash Flow	13198000
DIS	Free Cash Flow	496000
DOW	Free Cash Flow	3969000
GS	Free Cash Flow	-
HD	Free Cash Flow	11045000
IBM	Free Cash Flow	11492000
INTC	Free Cash Flow	18184000
JNJ	Free Cash Flow	19764000
JPM	Free Cash Flow	-
KO	Free Cash Flow	8246000
MCD	Free Cash Flow	5728400
MMM	Free Cash Flow	5595000
MRK	Free Cash Flow	-
MSFT	Free Cash Flow	43362000
NKE	Free Cash Flow	3936000
PFE	Free Cash Flow	-

6.10.2.부채정보(debt information)

```
combined_cash_flows[combined_cash_flows.Breakdown == "Issuance of Debt"]
```