일봉 데이터 가져오기

FinanceDataReader 로 일봉 데이터 가져오기

가설 분석과 수익율 예측 모델링은 변동성이 큰 코스닥 종목만을 대상으로 하겠습니다. 가설검정을 위하여 과거 수 개월치의 일봉데이터가 필요합니다. 우선 데이터를 종목별로 가져오기 위해서 FinanceDataReader 의 Stocklisting 메소드에서 코스닥의 종목 코드와 정보를 불러옵니다.

**import** **FinanceDataReader** **as** **fdr**

**import** **matplotlib.pyplot** **as** **plt**

%**matplotlib** inline

**import** **os**

**import** **FinanceDataReader** **as** **fdr**

**import** **pandas** **as** **pd**

**import** **numpy** **as** **np**

**import** **requests**

**import** **bs4**

pd.options.display.float\_format = '*{:,.3f}*'.format

kosdaq\_df = fdr.StockListing('KOSDAQ')

kosdaq\_df.head().style.set\_table\_attributes('style="font-size: 12px"')

|  | **Symbol** | **Market** | **Name** | **Sector** | **Industry** | **ListingDate** | **SettleMonth** | **Representative** | **HomePage** | **Region** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **0** | 060310 | KOSDAQ | 3S | 전자부품 제조업 | 반도체 웨이퍼 캐리어 | 2002-04-23 00:00:00 | 03월 | 김세완 | http://www.3sref.com | 서울특별시 |
| **3** | 054620 | KOSDAQ | APS홀딩스 | 기타 금융업 | 인터넷 트래픽 솔루션 | 2001-12-04 00:00:00 | 12월 | 정기로 | http://www.apsholdings.co.kr | 경기도 |
| **4** | 265520 | KOSDAQ | AP시스템 | 특수 목적용 기계 제조업 | 디스플레이 제조 장비 | 2017-04-07 00:00:00 | 12월 | 김영주 | http://www.apsystems.co.kr | 경기도 |
| **5** | 211270 | KOSDAQ | AP위성 | 통신 및 방송 장비 제조업 | 위성통신 단말기 | 2016-03-04 00:00:00 | 12월 | 류장수 | http://www.apsi.co.kr | 서울특별시 |
| **60** | 032790 | KOSDAQ | BNGT | 기계장비 및 관련 물품 도매업 | Bio 이종장기 사업, ICT 프린터 현상기 | 1997-06-26 00:00:00 | 12월 | 조상환 | http://www.mgenplus.com | 서울특별시 |

섹터가 정의되지 않은 종목과 2021년 1월 1일 이후 상장된 종목은 제외하겠습니다. 종 1422 개의 종목이 있습니다. 독자분이 책을 보시는 시점에는 종목 수가 바뀌어 있을 것입니다.  
kosdaq\_df 에서 필요한 컬럼 ‘Symbol’ 과 ‘Name’ 두 개만 kosdaq\_list 에 저장합니다. 그리고 종목코드 ‘Symbol’ 과 ‘Name’ 을 각 각 ‘code’ 외 ‘name’ 으로 바꿔줍니다. 그리고 나중을 위해서 결과물을 pickle 파일로 저장도 합니다.

print(kosdaq\_df['Symbol'].nunique())

c1 = (kosdaq\_df['ListingDate']>'2021-01-01') *# 2021년 1월 1일 이후 상장된 종목*

c2 = (kosdaq\_df['Sector'].isnull()) *# 섹터 값이 비어있음*

print(kosdaq\_df[~c1 & ~c2]['Symbol'].nunique()) *# c1 이 아니고 c2 가 아닌 종목의 갯 수*

kosdaq\_list = kosdaq\_df[~c1 & ~c2][['Symbol','Name','Sector']].rename(columns={'Symbol':'code','Name':'name','Sector':'sector'})

kosdaq\_list.to\_pickle('kosdaq\_list.pkl')

1582

1416

저장한 pickle 파일을 읽고, sector 가 몇개나 있는 지 세어봅니다.

kosdaq\_list = pd.read\_pickle('kosdaq\_list.pkl')

kosdaq\_list['sector'].nunique()

132

For Loop 에서 kosdaq\_list 의 종목코드와 종목이름을 하나씩 불러서 DataReader 로 2021년 1월 3일부터 2022년 3월 31일까지 일봉데이터를 수집합니다.

price\_data = pd.DataFrame()

**for** code, name **in** zip(kosdaq\_list['code'], kosdaq\_list['name']): *# 코스닥 모든 종목에서 대하여 반복*

daily\_price = fdr.DataReader(code, start='2021-01-03', end='2022-03-31') *# 종목, 일봉, 데이터 갯수*

daily\_price['code'] = code

daily\_price['name'] = name

price\_data = pd.concat([price\_data, daily\_price], axis=0)

price\_data.index.name = 'date'

price\_data.columns= price\_data.columns.str.lower() *# 컬럼 이름 소문자로 변경*

price\_data.to\_pickle('stock\_data\_from\_fdr.pkl')

저장한 pickle 파일을 다시 읽어 첫 5 라인을 head 메소드로 찍어보면 아래와 같습니다. 여기서 date가 인덱스로 처리되어 있다는 것을 기억해주시면 좋습니다. 타이핑 편의를 위해 컬럼이름을 소문자료 변경하겠습니다.

price\_data = pd.read\_pickle('stock\_data\_from\_fdr.pkl')

price\_data.head().style.set\_table\_attributes('style="font-size: 12px"')

|  | **open** | **high** | **low** | **close** | **volume** | **change** | **code** | **name** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **date** |  |  |  |  |  |  |  |  |
| **2021-01-04 00:00:00** | 2185 | 2320 | 2135 | 2260 | 588133 | 0.043880 | 060310 | 3S |
| **2021-01-05 00:00:00** | 2270 | 2285 | 2200 | 2250 | 410263 | -0.004425 | 060310 | 3S |
| **2021-01-06 00:00:00** | 2225 | 2310 | 2215 | 2290 | 570349 | 0.017778 | 060310 | 3S |
| **2021-01-07 00:00:00** | 2290 | 2340 | 2240 | 2290 | 519777 | 0.000000 | 060310 | 3S |
| **2021-01-08 00:00:00** | 2300 | 2315 | 2225 | 2245 | 462568 | -0.019651 | 060310 | 3S |

몇 개의 종목이 있고, 각 종목별 일봉의 갯 수 가 몇 개인지 확인해 보겠습니다. 종목 수는 1417 개, 307 개의 일봉이 있습니다.

print(price\_data['code'].nunique())

print(price\_data.groupby('code')['close'].count().agg(['min','max']))

1417

min 307

max 307

Name: close, dtype: int64

네이버 증권 웹크롤링으로 일봉 데이터 가져오기

이 번에는 네이버 증권 차트 *(네이버 차트 예시 필요)* 에서 데이터를 가져오는 방법도 시도해 보겠습니다. 웹 크롤링은 코드가 복잡합니다. 첫 번째 방법인 FinanceDataReader 로 추출하는 방법을 추천드립니다.  
다시 pickle 파일을 읽습니다. make\_price\_data 함수는 ‘종목’, ‘추출단위’, ‘데이터 건수’ 를 인자로 네이버증권에서 데이터를 가져오는 함수입니다. 인자는 작은 따옴표에 넣어야 합니다. 셀트리온 헬스케어(091990) 의 일봉 데이터를 최근 300 일 가져오고 싶다면 make\_price\_data(‘091990’, ‘day’, ‘300’) 와 같이 호출합니다. 이 함수를 for 문을 이용해 모든 코스닥 종목에서 대하여 호출하고, 각 결과를 price\_data 라는 데이터프레임에 담습니다. for 문을 돌리고 결과를 concat 함수로 연속으로 저장하는 방법은 자주 활용되는 기법입니다.

*# 네이버 증권 차트에서 데이터 크롤링*

kosdaq\_list = pd.read\_pickle('kosdaq\_list.pkl')

**def** make\_price\_data(code, name, timeframe, count):

url = 'https://fchart.stock.naver.com/sise.nhn?symbol=' + code + '&timeframe=' + timeframe + '&count=' + count + '&requestType=0'

price\_data = requests.get(url)

price\_data\_bs = bs4.BeautifulSoup(price\_data.text, 'lxml')

item\_list = price\_data\_bs.find\_all('item')

date\_list = []

open\_list = []

high\_list = []

low\_list = []

close\_list = []

trade\_list = []

**for** item **in** item\_list:

data = item['data'].split('|')

date\_list.append(data[0])

open\_list.append(data[1])

high\_list.append(data[2])

low\_list.append(data[3])

close\_list.append(data[4])

trade\_list.append(data[5])

price\_df = pd.DataFrame({'open': open\_list, 'high': high\_list, 'low': low\_list, 'close': close\_list, 'volume': trade\_list}, index=date\_list)

price\_df['code'] = code

price\_df['name'] = name

num\_vars = ['open','high','low','close','volume']

char\_vars = ['code','name']

price\_df = price\_df.reindex(columns = char\_vars + num\_vars)

**for** var **in** num\_vars:

price\_df[var] = pd.to\_numeric(price\_df[var], errors='coerce')

price\_df.index = pd.to\_datetime(price\_df.index, errors='coerce')

**return** price\_df

price\_data = pd.DataFrame()

**for** code, name **in** zip(kosdaq\_list['code'], kosdaq\_list['name']): *# 코스닥 모든 종목에서 대하여 반복*

daily\_price = make\_price\_data(code, name, 'day', '307') *# 종목, 일봉, 데이터 갯수*

price\_data = pd.concat([price\_data, daily\_price], axis=0)

price\_data.index.name = 'date'

price\_data.to\_pickle('stock\_data\_from\_naver.pkl')

저장한 pickle 파일을 다시 읽어 첫 5 라인을 head 메소드로 찍어보면 아래와 같습니다. 여기서 date가 인덱스로 처리되어 있다는 것을 기억해주시면 좋습니다.

price\_data = pd.read\_pickle('stock\_data\_from\_naver.pkl')

price\_data.head().style.set\_table\_attributes('style="font-size: 12px"')

|  | **code** | **name** | **open** | **high** | **low** | **close** | **volume** |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **date** |  |  |  |  |  |  |  |
| **2021-03-23 00:00:00** | 060310 | 3S | 2525 | 2525 | 2390 | 2410 | 245741 |
| **2021-03-24 00:00:00** | 060310 | 3S | 2410 | 2420 | 2350 | 2410 | 156213 |
| **2021-03-25 00:00:00** | 060310 | 3S | 2410 | 2510 | 2400 | 2465 | 288725 |
| **2021-03-26 00:00:00** | 060310 | 3S | 2480 | 2480 | 2400 | 2410 | 195825 |
| **2021-03-29 00:00:00** | 060310 | 3S | 2410 | 2435 | 2350 | 2385 | 194419 |

몇 개의 종목이 있고, 각 종목별 일봉의 갯 수 가 몇 개인지 확인해 보겠습니다. 종목 수는 1422 개, 307 개의 일봉이 있습니다.

print(price\_data['code'].nunique())

print(price\_data.groupby('code')['close'].count().agg(['min','max']))

1417

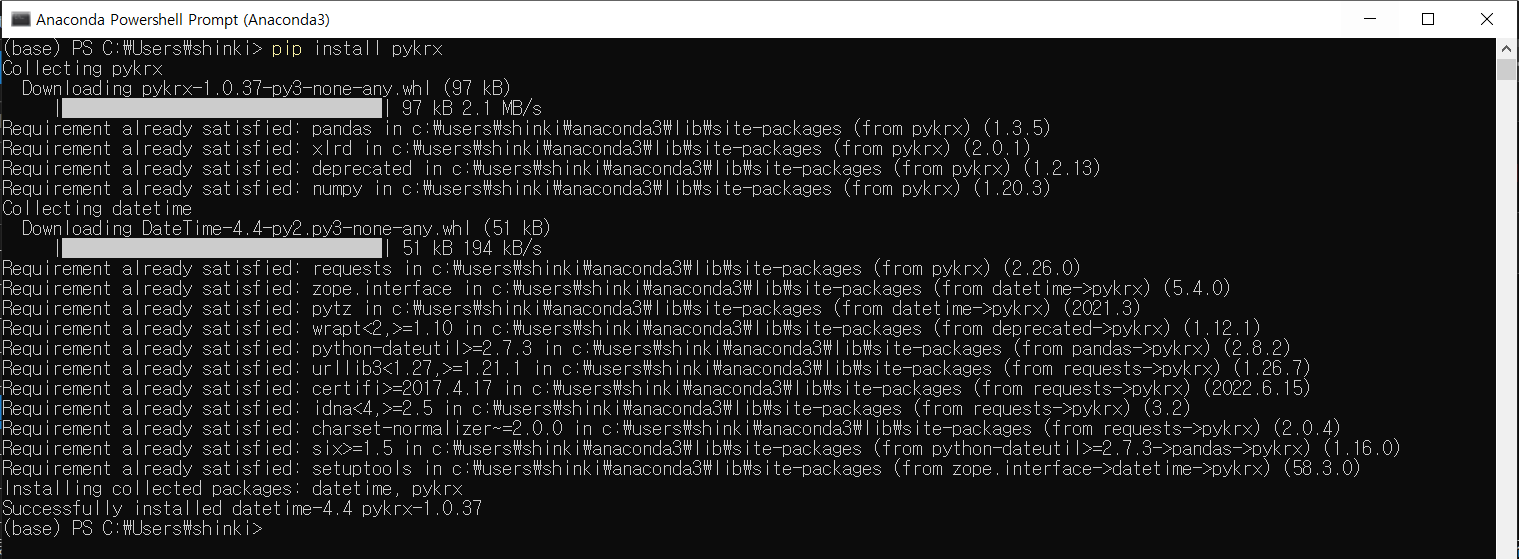
min 307

max 307

Name: close, dtype: int64

Pykrx 로 일봉 데이터 가져오기

일봉을 가져올 수 있는 또 다른 라이브러러는 pykrx 입니다. 주피터노트북 상에서 설치할때는 !pip install pykrx 과 같이 앞이 ‘!’ 느낌표 후에 명령어를 타이핑합니다. 셀을 실행하면 주피터노트북 상에서 설치가 진행됩니다. 저는 아나콘다 프롬프트에서 설치하는 것을 선호합니다. 왜냐하면 설치 과정을 볼 수 있기 때문입니다. 아나콘다 프롬프트에서 아래와 같이 설치를 합니다. 잘 작동하는 지 삼성전자 일봉을 몇 개만 호출해 봅니다. 컬럼이 한글로 되어 있는 것이 이전 패키지와 다른 점입니다.



**from** **pykrx** **import** stock

df = stock.get\_market\_ohlcv('20220104','20220108','005930') *# 메소드 작동을 확인*

df.style.set\_table\_attributes('style="font-size: 12px"')

|  | **시가** | **고가** | **저가** | **종가** | **거래량** |
| --- | --- | --- | --- | --- | --- |
| **날짜** |  |  |  |  |  |
| **2022-01-04 00:00:00** | 78800 | 79200 | 78300 | 78700 | 12427416 |
| **2022-01-05 00:00:00** | 78800 | 79000 | 76400 | 77400 | 25470640 |
| **2022-01-06 00:00:00** | 76700 | 77600 | 76600 | 76900 | 12931954 |
| **2022-01-07 00:00:00** | 78100 | 78400 | 77400 | 78300 | 15163757 |

kosdaq\_list = pd.read\_pickle('kosdaq\_list.pkl')

price\_data = pd.DataFrame()

**for** code, name **in** zip(kosdaq\_list['code'], kosdaq\_list['name']): *# 코스닥 모든 종목에서 대하여 반복*

daily\_price = stock.get\_market\_ohlcv(fromdate='2021-01-03', todate='2022-03-31', ticker=code) *# 종목, 일봉, 데이터 갯수*

daily\_price['code'] = code

daily\_price['name'] = name

price\_data = pd.concat([price\_data, daily\_price], axis=0)

price\_data.index.name = 'date'

price\_data.columns= ['open','high','low','close','volume','code','name'] *# 컬럼 이름 영문자로 변경*

price\_data.to\_pickle('stock\_data\_from\_pykrx.pkl')

price\_data = pd.read\_pickle('stock\_data\_from\_pykrx.pkl')

price\_data.head().style.set\_table\_attributes('style="font-size: 12px"')

|  | **open** | **high** | **low** | **close** | **volume** | **code** | **name** |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **date** |  |  |  |  |  |  |  |
| **2021-01-04 00:00:00** | 2185 | 2320 | 2135 | 2260 | 588133 | 060310 | 3S |
| **2021-01-05 00:00:00** | 2270 | 2285 | 2200 | 2250 | 410263 | 060310 | 3S |
| **2021-01-06 00:00:00** | 2225 | 2310 | 2215 | 2290 | 570349 | 060310 | 3S |
| **2021-01-07 00:00:00** | 2290 | 2340 | 2240 | 2290 | 519777 | 060310 | 3S |
| **2021-01-08 00:00:00** | 2300 | 2315 | 2225 | 2245 | 462568 | 060310 | 3S |