

파이썬 머신러닝 판다스 데이터분석

Lecture (12)



Dr. Heesuk Kim

Part 0. 개발환경 준비

Part 1. 판다스 입문

Part 2. 데이터 입출력

Part 3. 데이터 살펴보기

Part 4. 시각화 도구

Part 5. 데이터 사전처리

Part 6. 데이터프레임의 다양한 응용

Part 7. 머신러닝 데이터 분석



Part 2. 데이터 입출력

1. 외부파일 읽기

1-1. CSV 파일

1-2. Excel 파일

1-3. JSON 파일

2. 웹(web)에서 가져오기

2-1. HTML 웹 페이지에서 표 속성 가져오기

2-2. 웹 스크래핑

3. API 활용하여 데이터 수집하기

4. 데이터 저장하기

4-1. CSV 파일로 저장

4-2. JSON 파일로 저장

4-3. Excel 파일로 저장

4-4. 여러 개의 데이터프레임을 하나의 Excel 파일로 저장



Part 2. 데이터 입출력

2. 웹(web)에서 가져오기

2-2. 웹 스크래핑

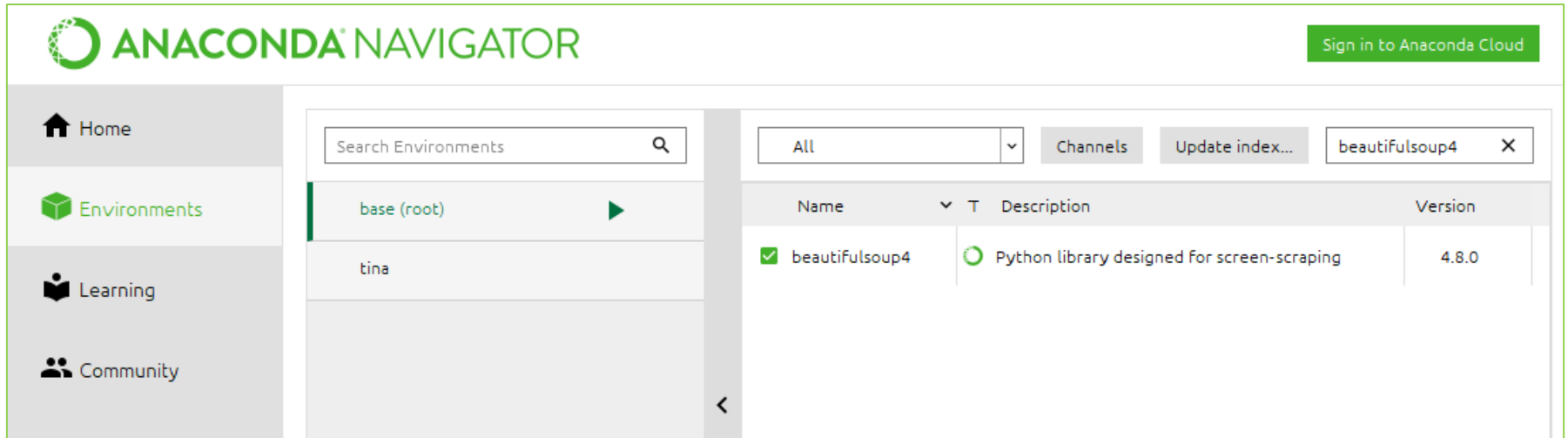
BeautifulSoup 등 웹 스크래핑(scraping) 도구로 수집한 내용을 파이썬 리스트, 딕셔너리 등으로 정리하고, DataFrame() 함수에 리스트/딕셔너리 형태로 전달하여 데이터프레임으로 변환한다.

다음은 위키피디아에서 미국 ETF 리스트 데이터를 가져와서 데이터프레임으로 변환하는 예제이다.



Part 2. 데이터 입출력

2. 웹(web)에서 가져오기



The screenshot shows the Anaconda Navigator application interface. On the left is a sidebar with navigation links: Home, Environments, Learning, and Community. The main area is divided into two panes. The left pane shows a search bar labeled 'Search Environments' and a list of environments: 'base (root)' (highlighted with a green play button) and 'tina'. The right pane shows a search results table for the 'beautifulsoup4' package.

ANACONDA NAVIGATOR Sign in to Anaconda Cloud

Search Environments

base (root) ▶

tina

All Channels Update index... beautifulsoup4 X

Name	Description	Version
✓ beautifulsoup4	Python library designed for screen-scraping	4.8.0



Part 2. 데이터 입출력

2. 웹(web)에서 가져오기

또는
pip install BeautifulSoup4

```
(base) C:\WINDOWS\system32> pip install bs4
Collecting bs4
  Using cached
  https://files.pythonhosted.org/packages/10/ed/7e8b97591f6f456174139ec089c769f89a94a1a4025fe9676
  91de971f314/bs4-0.0.1.tar.gz
  Requirement already satisfied: beautifulsoup4 in c:\programdata\anaconda3\lib\site-packages (from
  bs4) (4.8.0)
  Requirement already satisfied: soupsieve>=1.2 in c:\programdata\anaconda3\lib\site-packages
  (from beautifulsoup4->bs4) (1.9.2)
  Building wheels for collected packages: bs4
    Building wheel for bs4 (setup.py) ... done
    Created wheel for bs4: filename=bs4-0.0.1-cp37-none-any.whl size=1279
    sha256=be09ed88dea2b27b77487475a7fd7f99d19cedf13c24f7433870ced232bf6bc0
    Stored in directory:
    C:\Users\whskim\AppData\Local\pip\Cache\wheels\wa0\wb0\wb2\W4f80b9456b87abedbc0bf2d522354
    14c3467d8889be38dd472
    Successfully built bs4
  Installing collected packages: bs4
  Successfully installed bs4-0.0.1

(base) C:\WINDOWS\system32>
```

가장 정확

Anaconda Prompt (anaconda3)

앱

명령 프롬프트

웹 검색

cmd - 웹 결과 보기

폴더 (2+)

문서 (5+)

설정 (1)

cmd

```
(base) C:\WINDOWS\system32> conda install BeautifulSoup4
```



Part 2. 데이터 입출력 [화면 녹화]

Spyder (Python 3.7)

File Edit Search Source Run Debug Consoles Projects Tools View Help

2,4_read_html.py

```
1 # -*- coding: utf-8 -*-
2
3 import pandas as pd
4
5 # HTML 파일 경로 or 웹 페이지 주소를 url 변수에 저장
6 url = './sample.html'
7
8 # HTML 웹페이지의 표(table)를 가져와서 데이터프레임으로 변환
9 tables = pd.read_html(url)
10
11 # 표(table)의 개수 확인
12 print(len(tables))
13 print('\n')
14
15 # tables 리스트의 원소를 iteration하면서 각각 화면 출력
16 for i in range(len(tables)):
17     print("tables[%s]" % i)
18     print(tables[i])
19     print('\n')
20
21 # 파이썬 패키지 정보가 들어 있는 두 번째 데이터프레임을 선택하여 df 변수에 저장
22 df = tables[1]
23
24 # 'name' 열을 인덱스로 지정
25 df.set_index(['name'], inplace=True)
26 print(df)
```

Variable explorer

Name	Type	Size	Value
ex1	str	1	<html>

IPython console

Console 4/A

```
In [4]: from bs4 import BeautifulSoup
...:
...: ex1 = ''
...: <html>
...:   <head>
...:     <title> HTML 연습 </title>
...:   </head>
...:   <body>
...:     <p align="center"> text 1 </p>
...:   </body>
...: </html> ''
...:
...: soup = BeautifulSoup(ex1 , 'html.parser')
...:
In [5]:
```

IPython console History log

Permissions: RW End-of-lines: CRLF Encoding: UTF-8 Line: 9 Column: 1 Memory: 73%

Part 2. 데이터 입출력

2. 웹(web)에서 가져오기

1. find() : 조건을 만족하는 태그를 하나만 가져올 때 사용

```
from bs4 import BeautifulSoup
ex1 = '''
<html>
  <head>
    <title> HTML 연습 </title>
  </head>
  <body>
    <p align="center"> text 1 </p>

  </body>
</html> '''
```



Part 2. 데이터 입출력

2. 웹(web)에서 가져오기

```
from bs4 import BeautifulSoup
ex1 = '''
<html>
  <head>
    <title> HTML 연습 </title>
  </head>
  <body>
    <p align="center"> text 1 </p>
    <p align="righth"> text 2 </p>
    <p align="left"> text 3 </p>
  </body>
</html> '''
```



Part 2. 데이터 입출력 [화면 녹화]

Spyder (Python 3.7)

File Edit Search Source Run Debug Consoles Projects Tools View Help

2_4_read_html.py x untitle3.py+ x

```
1 from bs4 import BeautifulSoup
2 ex1 = '''
3 <html>
4     <head>
5         <title> HTML 연습 </title>
6     </head>
7     <body>
8         <p align="center"> text 1 </p>
9     </body>
10 </html> '''
11
12 soup = BeautifulSoup(ex1, 'html.parser')
13 print(soup)
14
15 a = soup.find('title')
16 print(a)
17
18 soup.find('p')
19 b = soup.find('p')
20 print(b)
21
```

Variable explorer

Name	Type	Size	Value
ex1	str	1	<html>

IPython console

Console 5/A x

```
In [5]: soup.find('title')
Out[5]: <title> HTML 연습 </title>

In [6]: a = soup.find('title')

In [7]: print(a)
<title> HTML 연습 </title>

In [8]: soup.find('p')
Out[8]: <p align="center"> text 1 </p>

In [9]: b = soup.find('p')

In [10]: print(b)
<p align="center"> text 1 </p>

In [11]:
```

IPython console History log

Permissions: RW End-of-lines: CRLF Encoding: UTF-8 Line: 21 Column: 1 Memory: 78 %

Part 2. 데이터 입출력 [화면 녹화]

Spyder (Python 3.7)

File Edit Search Source Run Debug Consoles Projects Tools View Help

2,4_read_html.py x untitle3.py* x untitle4.py* x 2,5_us_etf_list.py x

```
1 #-*- coding: utf-8 -*-
2
3 # 라이브러리 불러오기
4 from bs4 import BeautifulSoup
5 import requests
6 import re
7 import pandas as pd
8
9 # 위키피디아 미국 ETF 웹 페이지에서 필요한 정보를 스크래핑하여 딕셔너리 형태로 변수 etfs에 저장
10 url = "https://en.wikipedia.org/wiki/List_of_American_exchange-traded_funds"
11 resp = requests.get(url)
12
13 #html 문서 내 요소를 탐색할 수 있도록 문서 계층구조로 생성
14 soup = BeautifulSoup(resp.text, 'lxml')
15
16 rows = soup.select('div > ul > li')
17
18 etfs = {}
19 for row in rows:
20
21     try:
22         etf_name = re.findall('^.* \\\(NYSE', row.text)
23         etf_market = re.findall '\\((.*)\\)', row.text)
24         etf_ticker = re.findall('NYSE Arca\\(.*\\)', row.text)
25
26         if (len(etf_ticker) > 0) & (len(etf_market) > 0):
27             etfs[etf_ticker[0]] = [etf_market[0], etf_name[0]]
28
29     except AttributeError as err:
30         pass
31
32 # etfs 딕셔너리 출력
33 print(etfs)
34 print('\\n')
35
36 # etfs 딕셔너리를 데이터프레임으로 변환
37 df = pd.DataFrame(etfs)
38 print(df)
39
```

Variable explorer

Name	Type	Size	Value
------	------	------	-------

IPython console

Console 7/A x

Python 3.7.3 (default, Apr 24 2019, 15:29:51) [MSC v.1915 64 bit (AMD64)]
Type "copyright", "credits" or "license" for more information.

IPython 7.6.1 -- An enhanced Interactive Python.

In [1]: |

IPython console History log

Permissions: RW End-of-lines: CRLF Encoding: UTF-8 Line: 1 Column: 1 Memory: 76 %

Part 2. 데이터 입출력 [화면 녹화]

Spyder (Python 3.7)

File Edit Search Source Run Debug Consoles Projects Tools View Help

2,4_read_html.py x untitle3.py* x untitle4.py* x 2,5_us_etf_list.py* x

```
1 # -*- coding: utf-8 -*-
2
3 # 라이브러리 불러오기
4 from bs4 import BeautifulSoup
5 import requests
6 import re
7 import pandas as pd
8
9 # 위키피디아 미국 ETF 웹 페이지에서 필요한 정보를 스크래핑하여 딕셔너리 형태로
10 url = "https://en.wikipedia.org/wiki/List_of_American_exchange-traded_funds"
11 resp = requests.get(url)
12
13 #html 문서 내 요소를 탐색할 수 있도록 문서 계층구조로 생성
14 soup = BeautifulSoup(resp.text, 'lxml')
15
16 rows = soup.select('div > ul > li')
17
18 etfs = {}
19 for row in rows:
20     try:
21         etf_name = re.findall('^(.*) \((NYSE', row.text)
22         etf_market = re.findall('\((.*)\)', row.text)
23         etf_ticker = re.findall('NYSE Arca\((.*)\)', row.text)
24
25         if (len(etf_ticker) > 0) & (len(etf_market) > 0):
26             etfs[etf_ticker[0]] = [etf_market[0], etf_name[0]]
27
28     except AttributeError as err:
29         pass
30
31
```

Variable explorer

Name	Type	Size	Value
df	DataFrame	(2, 126)	Column names: ITOT, IWV, SCHB, FNDB, VT, VTI, VXUS, VTHR, DIA, RSP, IO ...
etf_market	list	1	['NERD:NYSE Arca']
etf_name	list	0	[]
etf_ticker	list	1	['NERD']
etfs	dict	126	{ 'ITOT': ['NYSE Arca', 'iShares Core S&P Total US Stock Mkt'], 'IWV': [' ...
rows	list	477	[Tag, Tag, Tag, Tag, Tag, Tag, Tag, Tag, Tag, Tag, ...]
url	str	1	https://en.wikipedia.org/wiki/List_of_American_exchange-traded_funds

IPython console

Console 7/A x

```
Arca', 'ALPS Alerian MLP ETF'], 'XLE': ['NYSE Arca', 'Energy Select Sector SPDR'], 'IYE': ['NYSE Arca', 'iShares Dow Jones US Energy'], 'IGE': ['NYSE Arca', 'iShares North American Natural Resources'], 'OIH': ['NYSE Arca', 'Market Vectors Oil Services ETF'], 'XOP': ['NYSE Arca', 'SPDR S&P Oil & Gas Explor & Prod ETF'], 'VDE': ['NYSE Arca', 'Vanguard Energy']}]

In [9]: df = pd.DataFrame(etfs)

In [10]: print(df)
           ITOT  ...  VDE
0           NYSE Arca  ...  NYSE Arca
1  iShares Core S&P Total US Stock Mkt  ...  Vanguard Energy

[2 rows x 126 columns]

In [11]:
```

IPython console History log

Permissions: RW End-of-lines: CRLF Encoding: UTF-8 Line: 13 Column: 13 Memory: 72%

Part 2. 데이터 입출력

2. 웹(web)에서 가져오기

2-2. 웹 스크래핑

〈예제 2-5〉 미국 ETF 리스트 가져오기

(File: example/part2/2.5_us_etf_list.py)

```
1 # -*- coding: utf-8 -*-
2
3 # 라이브러리 불러오기
4 from bs4 import BeautifulSoup
5 import requests
6 import re
7 import pandas as pd
8
9 # 위키피디아 미국 ETF 웹 페이지에서 필요한 정보를 스크래핑하여 딕셔너리 형태로 변수 etfs에 저장
10 url = "https://en.wikipedia.org/wiki/List_of_American_exchange-traded_funds"
11 resp = requests.get(url)
12 soup = BeautifulSoup(resp.text, 'lxml')
13 rows = soup.select('div > ul > li')
14
15 etfs = {}
16 for row in rows:
17
18     try:
19         etf_name = re.findall('^(.*) \((NYSE', row.text)
20         etf_market = re.findall('^\((.*)\)', row.text)
21         etf_ticker = re.findall('NYSE Arca\((.*)\)', row.text)
22
23         if (len(etf_ticker) > 0) & (len(etf_market) > 0):
24             etfs[etf_ticker[0]] = [etf_market[0], etf_name[0]]
25
26     except AttributeError as err:
27         pass
```

```
29 # etfs 딕셔너리 출력
30 print(etfs)
31 print('\n')
32
33 # etfs 딕셔너리를 데이터프레임으로 변환
34 df = pd.DataFrame(etfs)
35 print(df)
```

〈실행 결과〉 코드 전부 실행

```
{'ITOT': ['NYSE Arca', 'iShares Core S&P Total US Stock Mkt'], 'IWV': ['NYSE Arca', 'iS-
hares Russell 3000 Index'], 'SCHB': ['NYSE Arca', 'Schwab US Broad Market ETF'], 'FNDB':
['NYSE Arca', 'Schwab Fundamental U.S. Broad Market Index ETF'], 'VT': ['NYSE Arca',
'Vanguard Total World Stock'], 'VTI': ['NYSE Arca', 'Vanguard Total Stock Market'],
'VXUS': ['NYSE Arca', 'Vanguard Total International Stock'], 'VTHR': ['NYSE Arca', 'Van-
guard Russell 3000'], 'DIA': ['NYSE Arca', 'DIAMONDS Trust, Series 1'], 'RSP': ['NYSE
Arca', 'Guggenheim S&P 500 Equal Weight'], 'IOO': ['NYSE Arca', 'iShares S&P Global 100
Index'],
...
, 'EMCB': ['NYSE Arca', 'WisdomTree Emerging Markets Corporate Bond Fund'], 'EU': ['NYSE
Arca', 'WisdomTree Euro Debt Fund'], 'ICB': ['NYSE Arca', 'WisdomTree Dreyfus Indian Ru-
pee'], 'RRF': ['NYSE Arca', 'WisdomTree Global Real Return'], 'USDU': ['NYSE Arca', 'Wis-
domTree Bloomberg U.S. Dollar Bullish Fund'], 'WDTI': ['NYSE Arca', 'WisdomTree Managed
Futures Strategy Fund']}
```

	ITOT	...	WDTI
0	NYSE Arca	...	NYSE Arca
1	iShares Core S&P Total US Stock Mkt	...	WisdomTree Managed Futures Strategy Fund

[2 rows x 378 columns]

24번 라인의 `etfs[etf_ticker[0]] = [etf_market[0], etf_name[0]]`와 같이 리스트를 원소로 갖는 딕셔너리를 정의하는 방법을 반드시 기억한다. 왼쪽의 딕셔너리 키는 열 이름이 되고, 오른쪽 리스트는 열 데이터가 된다. 예제에서는, ETF 거래코드(`etf_ticker`)가 데이터프레임의 열 이름이 된다.

Any Question?

Thank you.

