모두를 위한 파이썬 프로그래밍

13주 판다스를 이용한 웹스크래핑
(정형 데이터의 웹스크래핑)

학습내용

빅데이터 분석과 시각화를 위한 판다스(Pandas)의 이해

- 엑셀과 같은 행열 자료구조의 빅데이터를 분석하고 시각화를 위해 필요합니다.

판다스를 활용한 웹 스크래핑

- 판다스 라이브러리를 활용하면 웹상에서 보여지는 표(table)형태의 데이터들을 쉽게 가져올 수 있습니다.

판다스(Pandas) 1

판다스(Pandas) 라이브러리

판다스(Pandas)의 이해

- 배열과 유사한 데이터분석 라이브러리로, 행과 열로 이루어진 데이터 객체를 만들어 다룰 수 있게 되며, 대용량의 데이터들을 처리하는데 편리한 도구
- 수학 라이브러리인 numpy(행렬처리)와 함께 빅데이터 분석에 유용
- 엑셀과 같은 행열처리 기능을 제공(csv 파일 읽고/쓰기 지원, xlsx)
- 웹사이트에서 정형화된 표의 내용을 크롤링하고 수집할 때 유용
- 시각화를 위한 자료구조로 활용됨

판다스(Pandas)의 데이터 형식

- 시리즈(Series)와 데이터프레임(Data Frame)을 사용
- Series는 1차원 데이터를 다루는 데 효과적인 자료구조이며, DataFrame은 행과 열로 구성된 2차원 데이터를 다루는 데 효과적인 자료구조

판다스(Pandas) 라이브러리

- Series 객체 활용
  - list로 시리즈 객체 생성하기

```
import pandas as pd
data = pd.Series([4, 5, -1, 2])
print(data) # index, 자료 출력
print(data.values)  # Series의 값만 확인하기 [ 4  5 -1  2]
print(data.index)
```

```
0  4
1  5
2 -1
3  2
dtype: int64
[ 4  5 -1  2]
RangeIndex(start=0, stop=4, step=1
```

판다스(Pandas) 라이브러리

- Series 객체 활용
  - 인덱스 지정하기
  - sorted 함수를 활용하여 인덱스나 값들로 정렬

```
import pandas as pd

list = [4, 5, 1, 2]
pd_data = pd.Series(list, index=['나', '라', '가', '다'])
print(pd_data)
print(sorted(pd_data))  # 데이터를 정렬

pd_data = pd_data.reindex(sorted(pd_data.index))
# 인덱스를 기준으로 데이터를 재정렬

print(pd_data)  # 정렬된 데이터 출력
```

판다스(Pandas) 라이브러리

- Series 객체 활용
  - 인덱스 별로 저장된 값들의 합 구하기
  - 두 변수 모두에 인덱스가 존재하는 값만 계산됨

```
import pandas as pd

x = pd.Series([1, 2, 3, 4], index=['서울', '대구', '대전', '부산'])
y = pd.Series([5, 6, 7, 8], index=['광주', '대전', '부산', '김포'])

print(x + y)
```

```
광주    NaN
김포    NaN
대구    NaN
대전    9.0
부산   11.0
서울    NaN
```

판다스(Pandas) 라이브러리

- Series 객체 활용
  - 시리즈 객체로 부터 유일한 값들만 반환하는 함수

```
import pandas as pd
weeks = ['월', '화', '일', '목', '토', '수', '금', '수', '일', '토']
x = pd.Series(weeks)
print(pd.unique(x))
```

['월''화''일''목''토''수''금']

  - 일반적인 배열의 인덱스 번호는 0부터 자동으로 index가 생성되지만, 판다스의 시리즈 객체는 고정된 인덱스 번호를 사용함.

```
import pandas as pd
weeks = ['월', '화', '일', '목', '토', '수', '금', '수', '일', '토']
x = pd.Series(weeks)
print(pd.unique(x))
a = x[2:]        # 인덱스 2 ~ 끝까지 복사해서 a에 저장
print(a)
print(a[0], a[1], a[2])  # 무엇이 출력될지 예상해 보자
```

판다스(Pandas) 라이브러리

- Series 객체 활용
  - 딕셔너리를 시리즈로 변환하기

```
import pandas as pd

age = {'홍길동': 23, '이민선': 32, '김동현': 56}
x = pd.Series(age)
print(x)
```

```
홍길동  23
이민선  32
김동현  56
```

판다스(Pandas) 라이브러리

- Data Frame 객체 (list로) 만들기
  - 데이터 프레임 객체는 2차원 배열의 개념을 가진 행과 열로 구성됨
  - DataFrame에 들어갈 데이터를 list로 생성하고 출력하기

```
import pandas as pd

data = [[1, 62], [3, 54], [5, 76], [7, 88]]
data_frame = pd.DataFrame(data, columns=['No', 'score'])
print(data_frame)
```

```
   No  score
0   1     62
1   3     54
2   5     76
3   7     88
```

판다스(Pandas) 라이브러리

- Data Frame 객체 (dictionary으로) 만들기
  - DataFrame에 들어갈 데이터를 dictionary로 생성하기

```
import pandas as pd
data = {'이름': ['홍길동', '이몽룡', '김철수', '장금이', '이순신'],
        '학번': [2017123, 2016312, 2014312, 2016332, 2019654],
        '점수': [77, 88, 99, 76, 78]}
data_frame = pd.DataFrame(data)
print(data_frame)
```

```
   이름      학번 점수
0 홍길동 2017123 77
1 이몽룡 2016312 88
2 김철수 2014312 99
3 장금이 2016332 76
4 이순신 2019654 78
```

판다스(Pandas) 라이브러리

- Data Frame 객체 만들기
  - 데이터 컬럼 순서를 변경하여 재구성하기

```
import pandas as pd
data = {'이름': ['홍길동', '이몽룡', '김철수', '장금이', '이순신'],
        '학번': [2017123, 2016312, 2014312, 2016332, 2019654],
        '점수': [77, 88, 99, 76, 78]}
data_frame = pd.DataFrame(data, columns=['학번', '이름', '점수'])
print(data_frame)
```

```
     학번  이름  점수
0 2017123 홍길동  77
1 2016312 이몽룡  88
2 2014312 김철수  99
3 2016332 장금이  76
4 2019654 이순신  78
```

판다스(Pandas) 라이브러리

- Pandas 데이터의 형변환
  - 문자/문자열, 정수형, 실수형이 일반적으로 사용된다.
  - astype() 함수를 사용하여, 정수형을 실수형으로 변환하기

```
import pandas as pd
data = [[1,62], [3,54], [5,76], [7,88]]
data_frame = pd.DataFrame(data, columns=['No', 'score'])
data_frame['score'] = data_frame['score'].astype('float')
print(data_frame)
```

```
   No  score
0   1   62.0
1   3   54.0
2   5   76.0
3   7   88.0
```

판다스(Pandas) 라이브러리

- Pandas에서 외부 파일 가져오기
  - 외부 파일(csv, xlsx 등)을 가져와서 판다스 자료형태로 저장하여 사용할 수 있다
  - 판다스 자료를 (csv, xlsx 등) 파일 포맷으로 다시 저장할 수 있다.
  - csv파일은 특성상 data type이 저장되지 않는다.

```
import pandas as pd

titanic = pd.read_csv('titanic.csv')
print(titanic)
```

```
import pandas as pd
import xlrd      # 엑셀파일 읽기 패키지

titanic = pd.read_excel('titanic.xlsx')
print(titanic)
```

```
      pclass  survived  ...   body                        home.dest
0        1.0       1.0  ...    NaN                     St Louis, MO
1        1.0       1.0  ...    NaN  Montreal, PQ / Chesterville, ON
2        1.0       0.0  ...    NaN  Montreal, PQ / Chesterville, ON
3        1.0       0.0  ...  135.0  Montreal, PQ / Chesterville, ON
4        1.0       0.0  ...    NaN  Montreal, PQ / Chesterville, ON
...      ...       ...  ...    ...                             ...
1305     3.0       0.0  ...    NaN                             NaN
1306     3.0       0.0  ...  304.0                             NaN
1307     3.0       0.0  ...    NaN                             NaN
1308     3.0       0.0  ...    NaN                             NaN
1309     NaN       NaN  ...    NaN                             NaN

[1310 rows x 14 columns]
```

판다스(Pandas) 라이브러리

- csv 파일을 xlsx 파일로 저장하기

```
import pandas as pd
import openpyxl        # 엑셀 파일 쓰기 패키지

titanic = pd.read_csv('titanic.csv')
titanic.to_excel('output.xlsx', encoding='utf-8-sig')
# encoding= 'utf-8-sig' → pandas에서 csv파일로 저장할 때 한글 깨짐 현상 해결
print(titanic)
```

```
      pclass  survived  ...   body                        home.dest
0        1.0       1.0  ...    NaN                     St Louis, MO
1        1.0       1.0  ...    NaN  Montreal, PQ / Chesterville, ON
2        1.0       0.0  ...    NaN  Montreal, PQ / Chesterville, ON
3        1.0       0.0  ...  135.0  Montreal, PQ / Chesterville, ON
4        1.0       0.0  ...    NaN  Montreal, PQ / Chesterville, ON
...      ...       ...  ...    ...                             ...
1305     3.0       0.0  ...    NaN                             NaN
1306     3.0       0.0  ...  304.0                             NaN
1307     3.0       0.0  ...    NaN                             NaN
1308     3.0       0.0  ...    NaN                             NaN
1309     NaN       NaN  ...    NaN                             NaN

[1310 rows x 14 columns]
```

판다스(Pandas) 라이브러리

- Pandas에 csv파일을 불러와서 필요한 열만 선택해서 xlsx파일로 재 저장하기

```
import pandas as pd
import openpyxl

titanic = pd.read_csv('titanic.csv')
titanic_new_list = pd.DataFrame(titanic, columns=['pclass', 'survived', 'age', 'sex'])
titanic_new_list.to_excel('output.xlsx', encoding='utf-8-sig')
print(titanic_new_list)
```

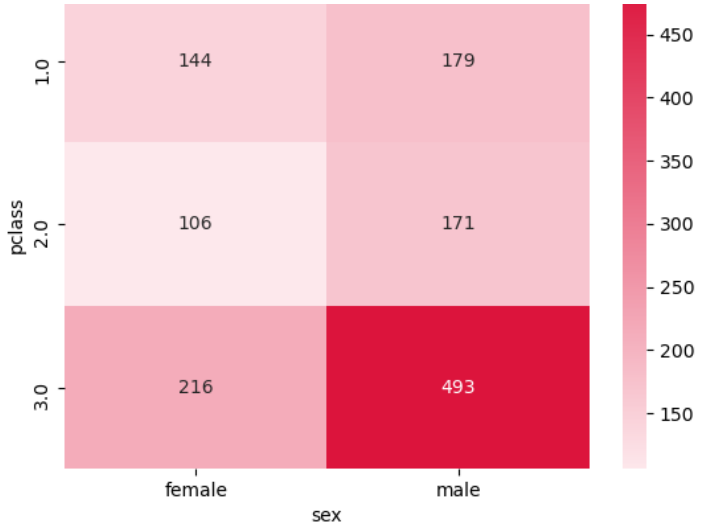
```
      pclass  survived      age     sex
0        1.0       1.0  29.0000  female
1        1.0       1.0   0.9167    male
2        1.0       0.0   2.0000  female
3        1.0       0.0  30.0000    male
4        1.0       0.0  25.0000  female
...      ...       ...      ...     ...
1305     3.0       0.0      NaN  female
1306     3.0       0.0  26.5000    male
1307     3.0       0.0  27.0000    male
1308     3.0       0.0  29.0000    male
1309     NaN       NaN      NaN     NaN

[1310 rows x 4 columns]
```

판다스(Pandas) 라이브러리

- 판다스 데이터는 빅데이터 시각화에 활용됩니다.
  - 예) 타이타닉 탑승자들의 승선권 등급에 따른 사망자 분석

```
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
titanic = pd.read_csv('titanic.csv')
titanic_new_list = titanic.pivot_table(index='pclass', columns='sex', aggfunc='size')
sns.heatmap(titanic_new_list, cmap=sns.light_palette('crimson', as_cmap=True), annot=True, fmt='d')
print(titanic_new_list)
plt.show()
```

sex	female	male
pclass		
1.0	144	179
2.0	106	171
3.0	216	493



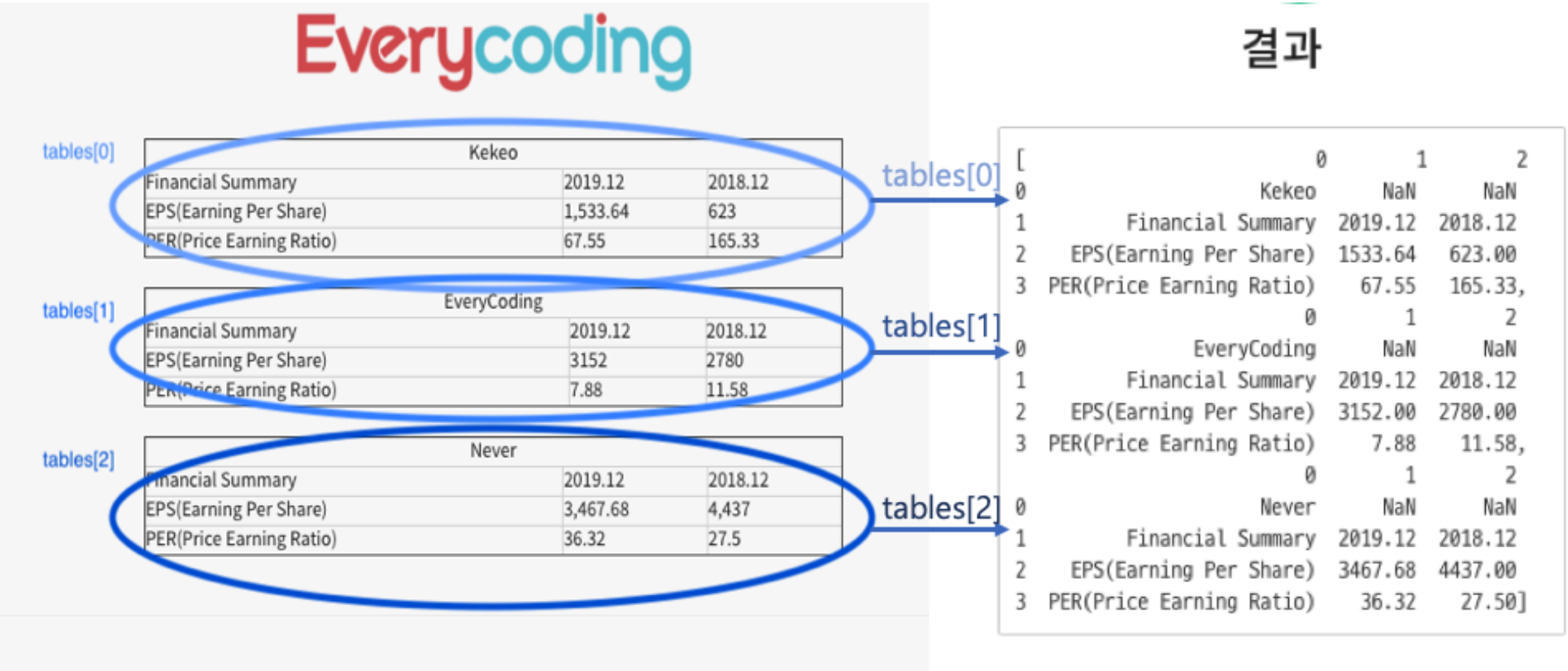
판다스를 이용한 Web Scraping

2

웹 스크래핑

1. 전문가들이 기업을 분석한 표인 '재무제표'를 인터넷에서 가져온다고 가정해 봅시다.

[웹 스크래핑]



*재무제표 = 기업의 재무현황을 분석한 표

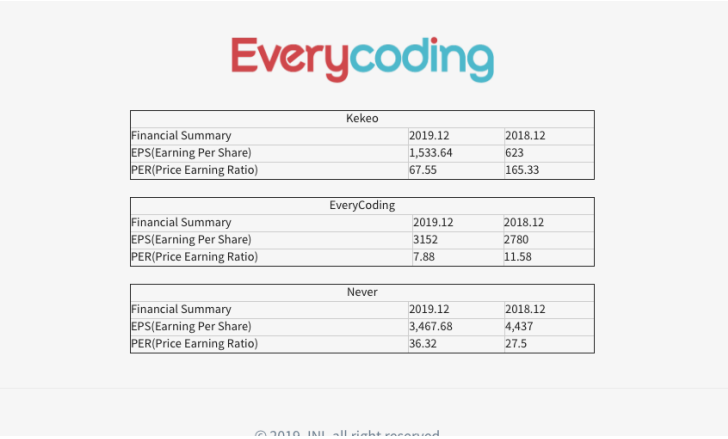
웹 스크래핑 (Scraping)

웹상에 있는 페이지(HTML로 작성된)들을 가져와서
우리에게 필요한 요소(정보)들만 추출하는 것
이것을 스크래핑이라고 합니다.

그럼 우리가 먼저 해야하는 작업은?

작업 1) 웹상에 있는 문서(HTML)를 가져와야 (Python의 데이터로) 합니다!

웹 스크래핑 (Scraping)




```
# 1. 웹페이지를 가져올 수 있도록 하는 라이브러리를 import 합니다.
import requests

# 2. 서버에 접근하여 사이트 응답을 가져옵니다.
site=requests.get("https://therk987.github.io/teachablemachine/sample_site
.html")

# 3. 접속한 사이트 페이지를 문자열 형태로 받아옵니다.
html_text = site.text
# html 문서 가져옴

# 여기까지 확인해 볼까요?
print(html_text)
```

https://therk987.github.io/teachablemachine/sample_site.html

사이트의 HTML 문서를 가져와 볼까요?

결과를 확인해 봅시다. 우리가 사용할 필요없는 내용도 많이 포함되어 있을거에요

웹 스크래핑 (Scraping)

그래서, 두번째! 작업은 불필요한 정보를 제거해서 필요한 정보만 추출 해 보겠습니다.

2) HTML 문서의 문자들을 파싱(PARSE)하여 원하는 정보(데이터 값)만 추출합니다!

Kekeo		
Financial Summary	2019.12	2018.12
EPS(Earning Per Share)	1,533.64	623
PER(Price Earning Ratio)	67.55	165.33

EveryCoding		
Financial Summary	2019.12	2018.12
EPS(Earning Per Share)	3152	2780
PER(Price Earning Ratio)	7.88	11.58

Never		
Financial Summary	2019.12	2018.12
EPS(Earning Per Share)	3,467.68	4,437
PER(Price Earning Ratio)	36.32	27.5

이 작업은 원래 '정규식' 이라는 것을 사용해서

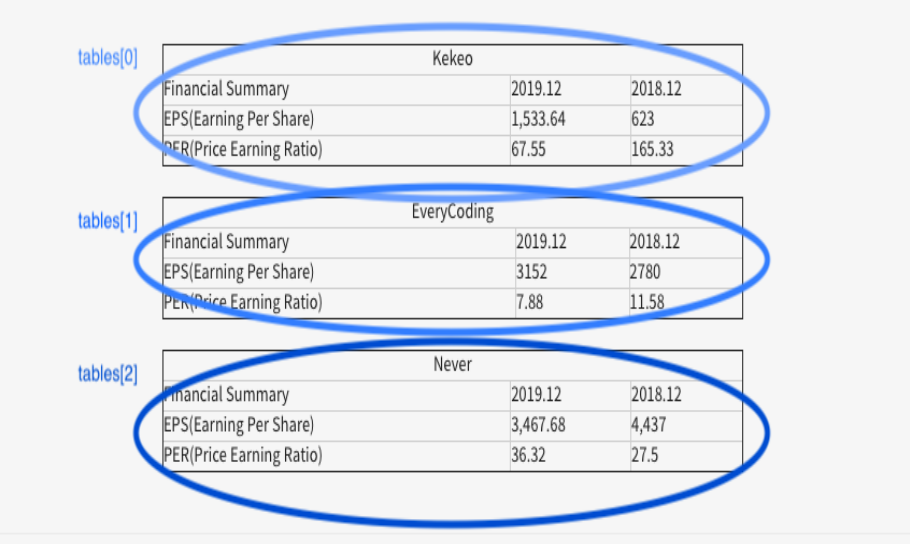
1. 문자에 규칙을 적용하고
2. 문자열을 분해하여
3. 원하는 값만 남겨두도록 해야 합니다.

할 게 많아 보이죠?

그러나, 이 정규식 분석을 판다스로 쉽게 처리 할 수 있습니다.

웹 스크래핑 (Scraping)

PANDAS 라이브러리를 사용하면 웹상의 표(TABLE 태그)들을 매우 간단하게 읽어들 일수 있습니다!



tables[0]

Kekeo		
Financial Summary	2019.12	2018.12
EPS(Earning Per Share)	1,533.64	623
PER(Price Earning Ratio)	67.55	165.33

tables[1]

EveryCoding		
Financial Summary	2019.12	2018.12
EPS(Earning Per Share)	3152	2780
PER(Price Earning Ratio)	7.88	11.58

tables[2]

Never		
Financial Summary	2019.12	2018.12
EPS(Earning Per Share)	3,467.68	4,437
PER(Price Earning Ratio)	36.32	27.5

우리가 가져오는 자료는 table 로
이루어진 '표' 라는 특징이 있습니다.

따라서!
pandas 라이브러리를 사용하여 표로 되어있는 값들을 추출하도록 합시다.

웹으로 부터 표로 이루어진 데이터를 가져오기 위해서는
pandas와 lxml 모듈을 설치해 주어야 합니다.

판다스 라이브러리로 웹 테이블 추출하기

HTML 문서로부터 표를 추출해 보는 코딩을 추가해 봅시다

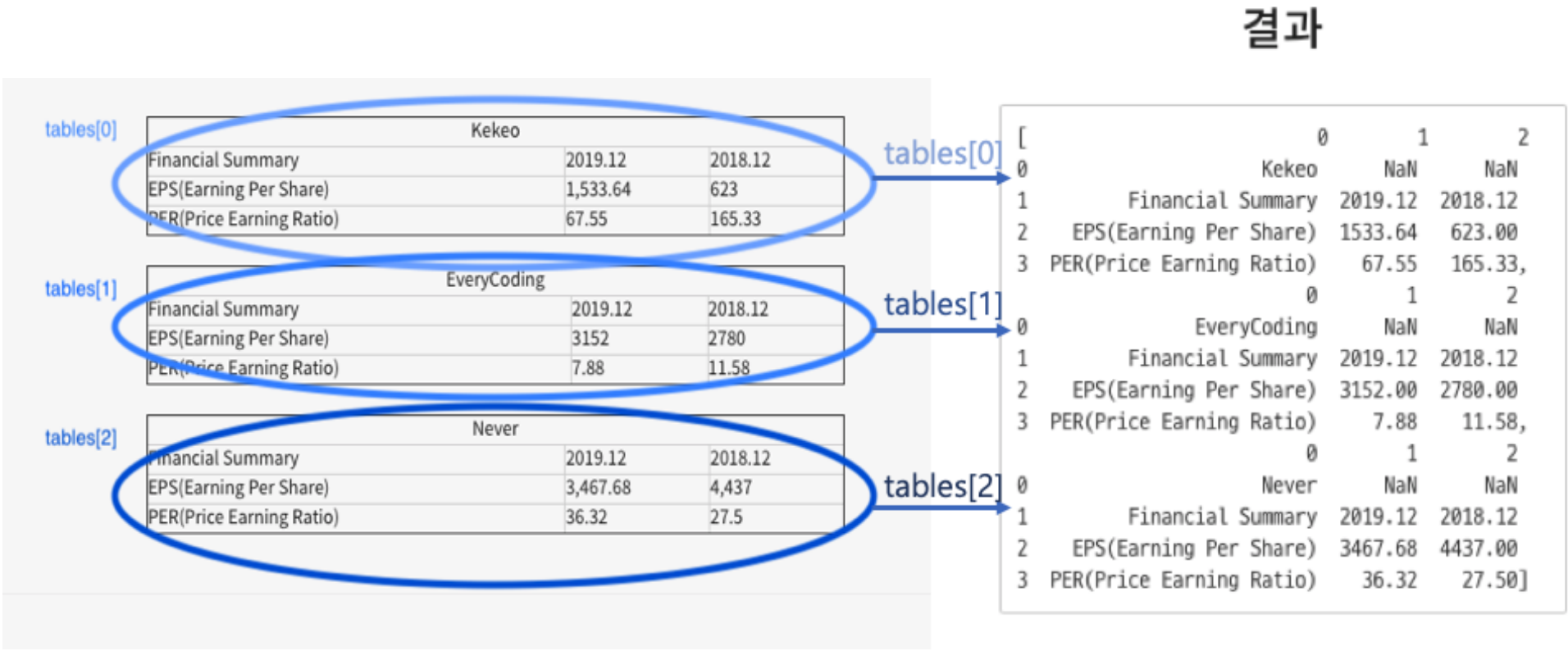
```
# 1. 라이브러리를 사용 할 수 있도록 import 합니다.
import requests
# 4. pandas 라이브러리를 사용 할 수 있도록 import 합니다.
import pandas

# 2. 서버에 접근하여 사이트 응답을 가져옵니다.
site=requests.get("https://therk987.github.io/teachablemachine/sample_site.html")
# 3. 접속한 사이트 페이지를 문자열 형태로 받아옵니다.
html_text = site.text
# (site.text 는 사이트의  html 문서 가져옴)

# 5. 사이트의 html 문서(#3)를 읽고 '표(table)' 형태를 모두 추출합니다.
tables = pandas.read_html(html_text)

# 여기까지 확인해 볼까요?
print(tables)
```

Kekeo		
Financial Summary	2019.12	2018.12
EPS(Earning Per Share)	1,533.64	623
PER(Price Earning Ratio)	67.55	165.33

EveryCoding		
Financial Summary	2019.12	2018.12
EPS(Earning Per Share)	3152	2780
PER(Price Earning Ratio)	7.88	11.58

Never		
Financial Summary	2019.12	2018.12
EPS(Earning Per Share)	3,467.68	4,437
PER(Price Earning Ratio)	36.32	27.5

판다스 라이브러리로 웹 테이블 추출하기

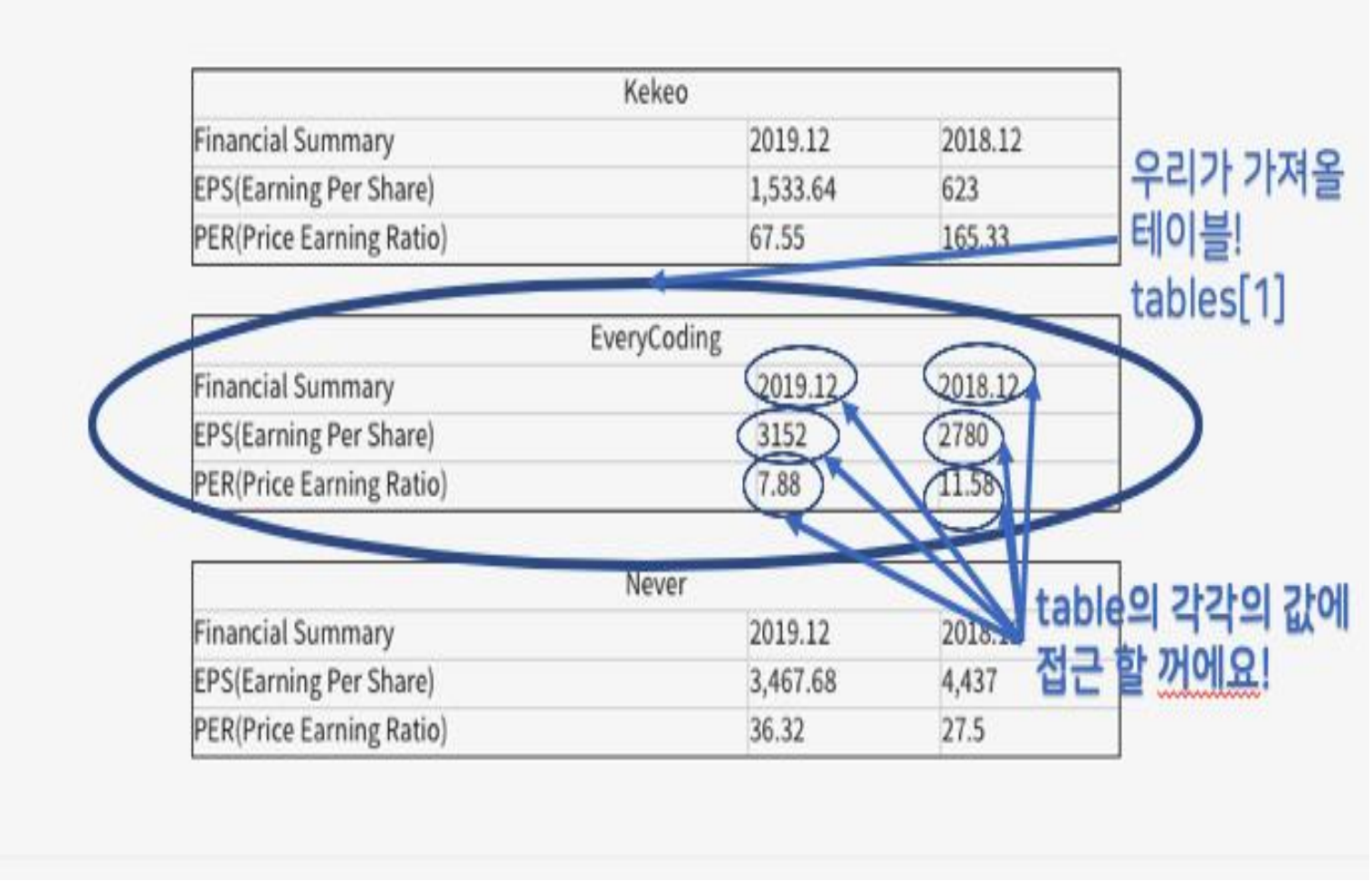


즉, PANDAS.READ_HTML() 을 이용하면
문자열을 해석하여 모든 table 태그를 찾아내고

각각의 table 요소들을 dataframe 이라는 객체로 변환해 리스트(list) 형태로 담아 반환합니다.

판다스 라이브러리로 웹 테이블 추출하기

에코 (EveryCoding) 재무제표의 값에 접근하고 적정주가를 계산한다고 가정해 봅시다!
에코 테이블의 각각의 값에 접근을 해야 합니다.



판다스 라이브러리로 웹 테이블 추출하기

판다스
.VALUES 함수를 활용 합니다

```
...
# 6. 가져온 표의 값을 가져옵니다!
# EveryCoding 재무제표는 두번째 이므로
# tables[1] !!
tables[1].values

# tables[1].values 를해주면
# 그 table 의 값들을 가져와줍니다.
# Type은 Numpy Array 형태
```

Kekeo

Financial Summary	2019.12	2018.12
EPS(Earning Per Share)	1,533.64	623
PER(Price Earning Ratio)	67.55	165.33

EveryCoding

Financial Summary	2019.12	2018.12
EPS(Earning Per Share)	3152	2780
PER(Price Earning Ratio)	7.88	11.58

Never

Financial Summary	2019.12	2018.12
EPS(Earning Per Share)	3,467.68	4,437
PER(Price Earning Ratio)	36.32	27.5

우리가 가져올 테이블! tables[1]

```
import requests
import pandas

site=requests.get("https://therk987.github.io/teachablemach
html_text = site.text
tables = pandas.read_html(html_text)
#-> 6. 가져온 표의 값을 가져옵니다!
# tables[1].values 는 에브리코딩(everycoding)의 재무제표를Array Type으로 반환 한 것
everycoding_stock = tables[1].values
#우리는 에코(EveryCoding) 주식을 분석 할 것입니다!
# tables[0].values 는 케케오(kekeo)의 재무제표
# tables[1].values 은 에브리코딩(everycoding)의 재무제표
# tables[2].values 은 네버(Never)의 재무제표
# 여기까지 확인해 볼까요?
print(everycoding_stock)
```

판다스 라이브러리로 웹 테이블 추출하기

이제 테이블 셀 하나하나의 값에 접근 해 보겠습니다.

```
# 6. 가져온 표의 값을 가져옵니다!
everycoding_stock = tables[1].values
#우리는 에코(EveryCoding) 주식을 분석 할 것입니다!
# tables[0].values 는 케케오(kekeo)의 재무제표
# tables[1].values 은 에브리코딩(everycoding)의 재무제표
# tables[2].values 은 네버(Never)의 재무제표
# 테이블의 셀 하나하나의 값에 접근해 봅시다

print(everycoding_stock[1][1]) # 2019.12 (tables[1].values[1][1] 과 같음)
print(everycoding_stock[1][2]) # 2018.12 (tables[1].values[1][2] 과 같음)
print(everycoding_stock[2][1]) # 3152 (tables[1].values[2][1] 과 같음)
print(everycoding_stock[2][2]) # 2780 (tables[1].values[2][2] 과 같음)
print(everycoding_stock[3][1]) # 7.88 (tables[1].values[3][1] 과 같음)
print(everycoding_stock[3][2]) # 11.58 (tables[1].values[3][2] 과 같음)
```

성공적으로 모두 가져왔는지 여기까지 결과를 확인해 봅시다
→ 이제 우리는 웹상에서 보여지는 표(table)형태의 데이터들을 판다스로 쉽게 가져올 수 있습니다.

[응용] 적정 주가 계산하기

이번에는

웹에서 가져온 정보에 사람의 지식을 적용하여

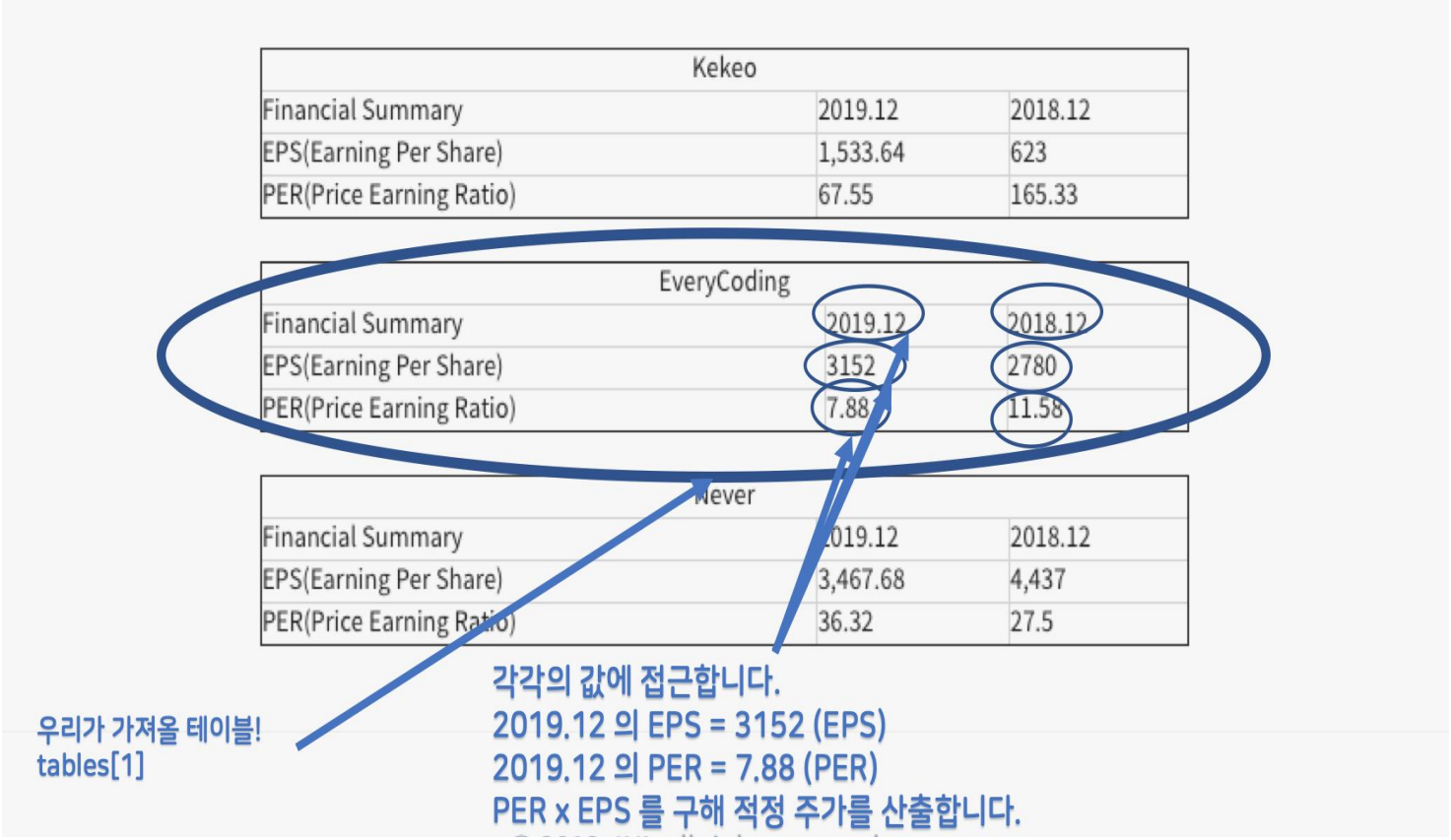
주식의 적정주가를 계산하는 프로그램을

작성해 보겠습니다.

**** 이 수식을 꼭 이해 할 필요는 없습니다 ****

PER x EPS = 적정주가
(적정 주식 가치 계산)

EPS = Earnings Per Share (1주당 벌어들인 돈의 순이익)
PER = Price Earnings Ratio (주가 수익률)



PER x EPS 를 계산하고 print() 해보는 작업입니다.

[응용] 적정 주가 계산하기

다음과 같이 일부분만 작성합니다.

```
import requests
import pandas
html = requests.get(' https://therk987.github.io/teachablemachine/sample_site.html ')
tables = pandas.read_html(html.text)
# 4. 사이트의 재무제표들중 Every 주식의 재무제표의 테이블들의 값을 가져옵니다.

everycoding_stock=tables[1].values
# 5. 재무제표 테이블에서 PER * EPS = 적정주가 공식을 이용하여 적정주가를 계산합니다.
stock_price = everycoding_stock[2][1] * everycoding_stock[3][1] // str에러가 날 경우 형변환 필요
# 6. 계산하여 나온 값을 print 합니다.

print(stock_price)
```

[응용] 함수로 작성하기

함수로 작성하면 중복된 작업을 줄일 수 있습니다

```
import requests
import pandas


def get_everycoding_stock_price():
    site = requests.get('https://everycoding.net/static/upload/tutor/image/value_investment/sample_site.html')
    html_text = site.text
    tables = pandas.read_html(html_text)
    kekeo_stock = tables[0].values
    everycoding_stock = tables[1].values
    never_stock = tables[2].values
    stock_price = float(everycoding_stock[2][1]) * float(everycoding_stock[3][1])
    return stock_price


print(get_everycoding_stock_price())
```

Thank you