

판다스 데이터 분석

Pandas

파이썬을 통해 데이터 분석을 할 때, Pandas를 빼놓고 이야기할 수 없다. 온전히 통계 분석을 위해 고안된 R과는 다르게 python은 일반적인 프로그래밍 언어(general purpose programming language)이며, 데이터 분석을 하기 위해서는 여러가지 라이브러리를 사용할 수 밖에 없다. 이 패키지들 중 R의 dataframe 데이터 타입을 참고하여 만든 것이 바로 pandas dataframe이다. pandas는 dataframe을 주로 다루기 위한 라이브러리이며, dataframe을 자유롭게 가공하는 것은 데이터 과학자들에게 중요하다.

공식사이트 :

<http://pandas.pydata.org>

Pandas

Pandas의 주요 기능을 설명한다.

- 쉬운 결손 값(missing data) 처리
- 레이블 위치를 자동적/명시적으로 정리한 데이터 작성
- 데이터 집약
- 고도의 레이블 베이스의 슬라이싱, 추출, 큰 데이터세트의 서브세트화
- 직감적인 데이터세트 결합
- 축의 계층적 레이블 붙임
- 여러가지 데이터 형식에 대응한 강력한 I/O
- 시계열 데이터 고유의 처리

Pandas

```
!pip install pandas
```

판다스 설치

Series

파이썬이 인기 있는 이유 중 하나는 파이썬의 기본 자료 구조인 리스트, 튜플, 딕셔너리가 사용하기 편리하고 데이터를 다루는 데 효과적이기 때문입니다.

pandas 역시 효과적인 데이터 분석을 위한 고수준의 자료구조와 데이터 분석 도구를 제공합니다. pandas의 Series는 1차원 데이터를 다루는 데 효과적인 자료구조이며, DataFrame은 행과 열로 구성된 2차원 데이터를 다루는 데 효과적인 자료구조입니다.

pandas를 이해하려면 가장 먼저 pandas의 핵심 자료구조인 Series와 DataFrame을 알아야 합니다.

이번 절에서는 먼저 Series에 대해 간단히 살펴봅니다.

- 인덱스라(레이블)를 가지는 1차원 데이터
- 인덱스는 중복 가능
- 레이블 또는 데이터의 위치를 지정한 추출 기능, 인덱스에 대한 슬라이스가 가능
- 산술 연산이 가능, 통계량을 산출하는 메리트를 가지고 있음

Series

Series 작성하기

Series의 작성에는 pandas.series 클래스를 사용 한다. 첫 번째 인수에는 다음과 같은 데이터를 넘겨준다.

- 리스트
- 튜플
- 사전
- numpy.ndarray

```
import pandas as pd
ser = pd.Series([1, 2, 3], index=['a', 'b', 'c'])
ser
```

```
a    1
b    2
c    3
```

키워드 인수 index에 레이블이 되는 값을 넘기는 것으로 데이터를 표시한다.

Series

Index를 생략한 경우

Index를 생략한 경우에는 0부터 차례대로 정수가 할당된다.

```
pd.Series([1, 2, 3])
```

0	1
1	2
2	3

레이블을 사용해서 데이터를 선택하기

Series.loc를 사용해서 레이블에서 데이터를 선택한다.

```
ser.loc['b']
```

2

Series

레이블의 범위 지정

레이블의 범위를 지정해서 슬라이스할 수 있다.

```
ser.loc['b':'c']
```

b	2
c	3

복수의 요소 지정

복수의 요소를 리스트로 지정할 수 있다.

```
ser.loc[['a', 'c']]
```

a	1
c	3

Series

위치를 지정해서 데이터 선택하기

Series.iloc를 사용해서 데이터의 위치를 정수값으로 지정하고 선택할 수 있다.

```
ser.iloc[1]
```

```
2
```

위치를 슬라이스로 지정

```
ser.iloc[1:3]
```

```
b    2  
c    3
```

Series

논리값을 사용해서 데이터 선택하기

loc와 iloc에는 논리값의 리스트를 넘길 수 있다.

```
ser.loc[[True, False, True]]
```

```
a    1  
c    3
```

Series에 대한 비교 연산

```
ser != 2
```

```
a    True  
b   False  
c    True
```

Series

비교 연산을 이용한 데이터 추출

```
ser.loc[ser != 2]
```


```
a    1  
c    3
```

DataFrame

pandas의 Series가 1차원 형태의 자료 구조라면 DataFrame은 여러 개의 칼럼 (Column) 으로 구성된 2차원 형태의 자료 구조 입니다.

DataFrame 객체를 생성하는 가장 쉬운 방법은 파이썬의 딕셔너리를 사용하는 것이다. 딕셔너리를 통해 각 칼럼에 대한 데이터를 저장한 후 딕셔너리를 DataFrame 클래스의 생성자 인자로 넘겨주면 DataFrame 객체가 생성됩니다.

- 행과 열에 레이블을 가진 2차원 데이터
- 열마다 다른 형태를 가질 수 있음
- 테이블형 데이터에 대해 불러오기, 데이터 쓰기가 가능
- DataFrame끼리 여러 가지 조건을 사용한 결합 처리가 가능
- 크로스 집계 가능



DataFrame			
	Series ('col0')	Series ('col1')	Series ('col2')
Index	Value	value	Value
0	1	10	100
1	2	20	200
2	3	30	300
3	4	40	400

DataFrame

DataFrame 작성하기

DataFrame의 작성에는 pandas.DataFrame 클래스를 사용한다. 첫 인수에는 1차원 또는 2차원 데이터를 넘긴다.

키워드 인수 index(행) 및 columns(열)에 레이블이 되는 값을 넘기는 것으로 데이터를 표시한다.

```
import pandas as pd
df = pd.DataFrame(
    [[1, 10, 100], [2, 20, 200], [3, 30, 300]],
    index = ['r1', 'r2', 'r3'],
    columns = ['c1', 'c2', 'c3'])
df
```

	c1	c2	c3
r1	1	10	100
r2	2	20	200
r3	3	30	300

DataFrame

레이블을 사용해서 데이터 선택하기

Series와 같이 DataFrame.loc를 사용해서 데이터를 추출한다.

```
df.loc['r2', 'c2']
```

```
20
```

모든 행(열)을 지정하는 경우

모든 열을 지정하는 경우 요소에 [:]를 넘긴다.

```
df.loc['r2', :]
```

```
c1    2
```

```
c2   20
```

```
c3  200
```

```
Name: r2, dtype: int64
```

DataFrame

모든 행에 레이블 지정

```
df.loc[:, 'c2']
```

```
r1  10  
r2  20  
r3  30  
Name: c2, dtype: int64
```

행의 데이터 수가 1이고 열의 데이터 수가 복수 또는 행의 데이터 수가 복수이고 열의 데이터 수가 1의 경우, 되돌아오는 데이터형은 Series가 된다.

2. DataFrame

슬라이스나 리스트를 넘겨주는 방법

슬라이스나 리스트를 넘기는 방법은 Series와 같다.

```
# 행 레이블을 리스트로 지정, 열 레이블을 슬라이스로 지정  
df.loc[['r1', 'r3'], 'c2':'c3']
```

	c2	c3
r1	10	100
r3	30	300

행의 데이터 수와 열의 데이터 수가 복수가 될 경우, 되돌아오는 데이터형은 DataFrame이다.

DataFrame

iloc를 사용해서 데이터 선택하기

Series와 같이 DataFrame.iloc를 사용해서 데이터의 위치에 의한 데이터 추출이 가능하다.

DataFrame의 경우에는 행과 열의 위치를 각각 지정한다.

```
df.iloc[1:3, [0, 2]]
```

	c1	c3
r2	2	200
r3	3	300

DataFrame

열 이름을 지정해서 데이터 선택하기

loc나 iloc를 지정하지 않고 DataFrame에 대해 다음과 같이 지정한 경우는 열 지정이 되고 되돌아 오는 데이터형은 Series가 된다.

```
df['c2']
```

```
r1    10  
r2    20  
r3    30  
Name: c2, dtype: int64
```

DataFrame

논리값을 사용해서 데이터 선택하기

Series와 같이 비교 연산을 하면 논리값이 되 돌아온다.

```
df > 10
```

	c1	c2	c3
r1	False	False	True
r2	False	True	True
r3	False	True	True

DataFrame

DataFrame에서 Series를 사용해 비교 연산을 하여 데이터를 추출할 수 있다.

```
# c2열의 값이 10보다 큰 데이터  
df.loc[df['c2'] > 10]
```

	c1	c2	c3
r2	2	20	200
r3	3	30	300

DataFrame

2개 이상의 조건을 조합하는 경우
다음의 연산자를 사용한다.

- `&` : and 조건
- `|` : or 조건

2개 이상의 조건을 조합하는 경우, 괄호()로 묶어 조건을 분류한다.

```
# c1열이 1보다 큰 동시에 c3열이 300보다 작은 데이터  
df.loc[(df['c1'] > 1) & (df['c3'] < 300)]
```

	c1	c2	c3
r2	2	20	200

DataFrame

DataFrame 생성자에서 입력 가능한 데이터

Type	Contents
2차원 ndarray	데이터를 담고 있는 행렬. 행과 열을 선택해서 데이터프레임으로 만들 수 있음
배열, 리스트, 사전	사전의 모든 항목은 같은 길이를 가져야 함. key 값이 컬럼이 됨
NumPy 배열	배열의 사전과 같은 방식으로 됨
Series 사전	Series의 각 값이 컬럼이 됨. 명시적으로 색인을 넘겨줬 않으면 각 Series의 색인이 하나로 합쳐져 행이 됨
사전의 사전	가장 외부에 있는 key가 컬럼이 되고 내부 사전의 key가 색인이 됨
사전이나 Series 리스트	리스트의 각 항목이 데이터프레임의 로우가 됨.
리스트나 튜플의 리스트	2차원 ndarray와 동일
다른 DataFrame	색인을 따로 지정하지 않으면 이전 데이터프레임 색인과 동일
NumPy MaskedArray	'2차원 ndarray'와 같은 방식이지만 마스크 값은 반환되는 DataFrame에서 NA 값이 됨

다양한 데이터 불러오기

pandas는 다음과 같이 다양한 형식의 데이터를 불러올 수 있다.

- CSV
- Excel
- 데이터베이스
- JSON
- MessagePack
- HTML
- Google BigQuery
- 클립보드
- Pickle

* 이외의 데이터 형식은 pandas documentation 참조

CSV 파일 불러오기

pandas.read_csv() 함수를 사용한다. 첫번째 인수에 파일 경로를 넘겨주면 DataFrame형 오브젝트를 되돌려준다. 파일 경로 또는 URL 형식으로 지정할 수 있다.

* 실습에 앞서 제공된 sample.zip 파일을 C:/python/sample/ 폴더에 압축을 푼다.

```
import os
import pandas as pd
csv = os.path.join('./sample/anime.csv')
df = pd.read_csv(csv)
df.head()
```

DataFrame.head() 메서드는 선두에서 5행분의 DataFrame을 되돌려준다. 인수에 정수값을 넘기는 방식으로 행수를 지정할 수 있다.

	anime_id	name	genre	type	episodes	rating	members
0	32281	Kimi no Na wa.	Drama, Romance, School, Supernatural	Movie	1	9.37	200630
1	5114	Fullmetal Alchemist: Brotherhood	Action, Adventure, Drama, Fantasy, Magic, Mili...	TV	64	9.26	793665
2	28977	Gintama°	Action, Comedy, Historical, Parody, Samurai, S...	TV	51	9.25	114262
3	9253	Steins;Gate	Sci-Fi, Thriller	TV	24	9.17	673572
4	9969	Gintama'	Action, Comedy, Historical, Parody, Samurai, S...	TV	51	9.16	151266

head()는 화면에 출력하는 기능이다.

CSV 파일 불러오기

지정한 열을 DataFrame의 인덱스로 하기

키워드 인수 index_col에 수치 또는 열 이름을 지정하여 지정한 열을 DataFrame의 인덱스로 한다.

```
# 인덱스로 할 열을 번호로 지정
df = pd.read_csv(csv, index_col=0)
df.head()
```

```
# 인덱스로 할 열을 이름으로 지정
df = pd.read_csv(csv, index_col='anime_id')
df.head()
```

	name	genre	type	episodes	rating	members
anime_id						
32281	Kimi no Na wa.	Drama, Romance, School, Supernatural	Movie	1	9.37	200630
5114	Fullmetal Alchemist: Brotherhood	Action, Adventure, Drama, Fantasy, Magic, Mili...	TV	64	9.26	793665
28977	Gintama°	Action, Comedy, Historical, Parody, Samurai, S...	TV	51	9.25	114262
9253	Steins;Gate	Sci-Fi, Thriller	TV	24	9.17	673572
9969	Gintama'	Action, Comedy, Historical, Parody, Samurai, S...	TV	51	9.16	151266

CSV 파일 불러오기

지정한 열을 지정한 형으로 불러오기

키워드 인수 dtype에 열 이름(키)과 형(값)을 사전형으로 지정하여 지정한 열을 지정한 형으로 불러올 수 있다.

```
df = pd.read_csv(csv, dtype={'members': float})  
df.head()
```

	anime_id	name	genre	type	episodes	rating	members
0	32281	Kimi no Na wa.	Drama, Romance, School, Supernatural	Movie	1	9.37	200630.0
1	5114	Fullmetal Alchemist: Brotherhood	Action, Adventure, Drama, Fantasy, Magic, Mili...	TV	64	9.26	793665.0
2	28977	Gintama°	Action, Comedy, Historical, Parody, Samurai, S...	TV	51	9.25	114262.0
3	9253	Steins;Gate	Sci-Fi, Thriller	TV	24	9.17	673572.0
4	9969	Gintama'	Action, Comedy, Historical, Parody, Samurai, S...	TV	51	9.16	151266.0

CSV 파일 불러오기

datetime 형의 열이 포함되어 있는 경우

datetime 형의 열이 포함되어 있는 경우, 키워드 인수 `parse_dates`에 열 이름을 리스트로 지정하여 불러올 때 형을 변환할 수 있다.

```
csv = os.path.join('./sample/anime_stock_price.csv')  
df = pd.read_csv(csv, parse_dates=['Date'])  
df.dtypes
```

```
Date                datetime64[ns]  
TOEI ANIMATION      float64  
IG Port             float64  
dtype: object
```