

파이썬 머신러닝 판다스 데이터분석

Lecture (9)



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 1. 판다스 입문

- 1. 데이터과학자가 판다스를 배우는 이유
- 2. 판다스 자료구조
 - 2-1. 시리즈
 - 2-2. 데이터프레임
- 3. 인덱스 활용
- 4. 산술 연산
 - 4-1. 시리즈 연산
 - 4-2. 데이터프레임 연산



Part 1. 판다스 입문

4-1. 시리즈 연산

• 연산 메소드 활용

- 객체 사이에 공통 인덱스가 없는 경우에 NaN으로 반환.
- 이런 상황을 피하려면 연산 메소드에 fill_value 옵션을 설정.

※ 다음 예제는 누락 데이터 NaN 대신 숫자 0을 입력하는 것을 예시로 설명.

연산 메소드 사용(시리즈와 시리즈 덧셈): `Series1.add(Series2, fill_value=0)`

예제 1-24

연산 메소드 사용 - 시리즈 연산

(File: example/part1/1.24_series_to_series3.py)

```
1 # -*- coding: utf-8 -*-
2
3 # 라이브러리 불러오기
4 import pandas as pd
5 import numpy as np
6
7 # 딕셔너리 데이터로 판다스 시리즈 만들기
8 student1 = pd.Series({'국어':np.nan, '영어':80, '수학':90})
9 student2 = pd.Series({'수학':80, '국어':90})
10
11 print(student1)
12 print('\n')
13 print(student2)
14 print('\n')
15
16 # 두 학생의 과목별 점수로 사칙연산 수행 (연산 메소드 사용)
17 sr_add = student1.add(student2, fill_value=0) # 덧셈
18 sr_sub = student1.sub(student2, fill_value=0) # 뺄셈
19 sr_mul = student1.mul(student2, fill_value=0) # 곱셈
20 sr_div = student1.div(student2, fill_value=0) # 나눗셈
21
22 # 사칙연산 결과를 데이터프레임으로 합치기 (시리즈 -> 데이터프레임)
23 result = pd.DataFrame([sr_add, sr_sub, sr_mul, sr_div],
24                        index=['덧셈', '뺄셈', '곱셈', '나눗셈'])
25 print(result)
```

〈실행 결과〉 코드 전부 실행

```
국어      NaN
영어      80.0
수학      90.0
dtype: float64

수학      80
국어      90
dtype: int64
```

	국어	수학	영어
덧셈	90.0	170.000	80.000000
뺄셈	-90.0	10.000	80.000000
곱셈	0.0	7200.000	0.000000
나눗셈	0.0	1.125	inf

연산 메소드에 `fill_value=0` 옵션을 설정하여 student1의 국어 점수와 student2의 영어 점수는 NaN 대신 0으로 입력된다. 한편 result의 '영어' 열의 나눗셈 값은 무한대(inf)이다. student1의 영어 점수 80을 student2의 영어 점수 0으로 나눈 값이기 때문이다.



Spyder (Python 3.7)

File Edit Search Source Run Debug Consoles Projects Tools View Help

untitled0.py 1.24_series_to_series3.py

```

1 # -*- coding: utf-8 -*-
2
3 # 라이브러리 불러오기
4 import pandas as pd
5 import numpy as np
6
7 # 딕셔너리 데이터로 판다스 시리즈 만들기
8 student1 = pd.Series({'국어':np.nan, '영어':80, '수학':90})
9 student2 = pd.Series({'수학':80, '국어':90})
10
11 print(student1)
12 print('\n')
13 print(student2)
14 print('\n')
15
16 # 두 학생의 과목별 점수로 사칙연산 수행 (연산 메소드 사용)
17 sr_add = student1.add(student2, fill_value=0) #덧셈
18 sr_sub = student1.sub(student2, fill_value=0) #뺄셈
19 sr_mul = student1.mul(student2, fill_value=0) #곱셈
20 sr_div = student1.div(student2, fill_value=0) #나눗셈
21
22 # 사칙연산 결과를 데이터프레임으로 합치기 (시리즈 -> 데이터프레임)
23 result = pd.DataFrame([sr_add, sr_sub, sr_mul, sr_div],
24                        index=['덧셈', '뺄셈', '곱셈', '나눗셈'])
25 print(result)
26

```

Variable explorer

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

IPython console

Console 1/A

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: 89 %

Part 1. 판다스 입문

4-2. 데이터프레임 연산

데이터프레임은 여러 시리지가 한데 모인 것이므로, 시리즈 연산을 확장하는 개념으로 이해한다. 먼저 행/열 인덱스를 기준으로 정렬하고, 일대일 대응되는 원소끼리 연산한다.

• 데이터프레임 vs. 숫자

- 데이터프레임에 어떤 숫자를 더하면, 모든 원소에 숫자를 더한다. 덧셈, 뺄셈, 곱셈, 나눗셈 모두 가능하다.
- 기존 데이터프레임의 형태를 그대로 유지한 채, 원소 값만 새로운 값으로 바뀐다.

데이터프레임과 숫자 연산: DataFrame 객체 + 연산자(+, -, *, /) + 숫자

예제 1-25

데이터프레임에 숫자 더하기

(File: example/part1/1.25_df_to_number.py)

```
1 # -*- coding: utf-8 -*-
2
3 # 라이브러리 불러오기
4 import pandas as pd
5 import seaborn as sns
6
7 # titanic 데이터셋에서 age, fare 2개 열을 선택하여 데이터프레임 만들기
8 titanic = sns.load_dataset('titanic')
9 df = titanic.loc[:, ['age', 'fare']]
10 print(df.head())          # 첫 5행만 표시
11 print('\n')
12 print(type(df))
13 print('\n')
14
15 # 데이터프레임에 숫자 10 더하기
16 addition = df + 10
17 print(addition.head())    # 첫 5행만 표시
18 print('\n')
19 print(type(addition))
```

<실행 결과> 코드 전부 실행

```
   age  fare
0  22.0   7.2500
1  38.0  71.2833
2  26.0   7.9250
3  35.0  53.1000
4  35.0   8.0500
```

```
<class 'pandas.core.frame.DataFrame'>
```

```
   age  fare
0  32.0  17.2500
1  48.0  81.2833
2  36.0  17.9250
3  45.0  63.1000
4  45.0  18.0500
```

```
class 'pandas.core.frame.DataFrame'>
```

※ 타이타닉('titanic') 데이터셋

Seaborn 라이브러리에서 제공하는 데이터셋으로, 타이타닉호 탑승자에 대한 인적사항과 구조 여부 등을 정리한 자료이다.

load_dataset() 함수로 불러온다.

[Seaborn 내장 데이터셋의 종류]

'anscombe', 'attention', 'brain_networks', 'car_crashes',
'diamonds', 'dots', 'exercise', 'flights', 'fmri', 'gammas', 'iris',
'mpg', 'planets', 'tips', 'titanic'

Anaconda Navigator

File Help

ANACONDA NAVIGATOR [Sign in to Anaconda Cloud](#)

Home

Environments

Learning

Community

Documentation

Developer Blog

Twitter YouTube GitHub

Create Clone Import Remove

Search Environments

base (root) ai anaconda3 py35

All Channels Update index... seaborn

Name	Description	Version
seaborn	Statistical data visualization	0.9.0

1 package available matching "seaborn"



The screenshot shows the Spyder Python IDE interface. The main editor displays a Python script with the following code:

```

1 # -*- coding: utf-8 -*-
2
3 # 라이브러리 불러오기
4 import pandas as pd
5 import seaborn as sns
6
7 # titanic 데이터셋에서 age, fare 2개 열을 선택하여 데이터프레임 만들기
8 titanic = sns.load_dataset('titanic')
9 df = titanic.loc[:, ['age', 'fare']]
10 print(df.head()) #첫 5행만 표시
11 print('\n')
12 print(type(df))
13 print('\n')
14
15 # 데이터프레임에 숫자 10 더하기
16 addition = df + 10
17 print(addition.head()) #첫 5행만 표시
18 print('\n')
19 print(type(addition))
20

```

The Variable explorer on the right is empty. The IPython console at the bottom shows the following output:

```

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]:

```

The status bar at the bottom indicates: Permissions: RW, End-of-lines: CRLF, Encoding: UTF-8, Line: 8, Column: 38, Memory: 89 %.

Part 1. 판다스 입문

4-2. 데이터프레임 연산

• 데이터프레임 vs. 데이터프레임

- 각 데이터프레임의 같은 행, 같은 열 위치에 있는 원소끼리 계산한다. 이처럼 동일한 위치의 원소끼리 계산한 결과값을 원래 위치에 다시 입력하여 데이터프레임을 만든다.

데이터프레임의 연산자 활용: `DataFrame1 + 연산자(+, -, *, /) + DataFrame2`

df1					df2				df1+df2			
	열 A	열 B	열 C			열 A	열 B			열 A	열 B	열 C
행 0	15	100	1	+	행 0	10	200	=	행 0	25	300	NaN
행 1	20	500	2		행 1	10	100		행 1	30	600	NaN
행 2	50	200	3		행 2	10	200		행 2	60	400	NaN
행 3	NaN	500	2		행 3	20	100		행 3	NaN	600	NaN
행 4	NaN	200	3		행 4	30	100		행 4	NaN	300	NaN

[그림 1-19] 데이터프레임의 산술연산(덧셈)

예제 1-26

데이터프레임끼리 더하기

(File: example/part1/1.26_df_to_df.py)

```
1 # -*- coding: utf-8 -*-
2
3 # 라이브러리 불러오기
4 import pandas as pd
5 import seaborn as sns
6
7 # titanic 데이터셋에서 age, fare 2개 열을 선택하여 데이터프레임 만들기
8 titanic = sns.load_dataset('titanic')
9 df = titanic.loc[:, ['age', 'fare']]
10 print(df.tail())      # 마지막 5행 표시
11 print('\n')
12 print(type(df))
```

```
15 # 데이터프레임에 숫자 10 더하기
16 addition = df + 10
17 print(addition.tail())      # 마지막 5행 표시
18 print('\n')
19 print(type(addition))
20 print('\n')
21
22 # 데이터프레임끼리 연산하기 (addition - df)
23 subtraction = addition - df
24 print(subtraction.tail())   # 마지막 5행 표시
25 print('\n')
26 print(type(subtraction))
```

<실행 결과> 코드 전부 실행

```
      age  fare
886  27.0  13.00
887  19.0  30.00
888   NaN  23.45
889  26.0  30.00
890  32.0   7.75
```

```
<class 'pandas.core.frame.DataFrame'>
```

```
      age  fare
886  37.0  23.00
887  29.0  40.00
888   NaN  33.45
889  36.0  40.00
890  42.0  17.75
```

```
<class 'pandas.core.frame.DataFrame'>
```

```
      age  fare
886  10.0  10.0
887  10.0  10.0
888   NaN  10.0
889  10.0  10.0
890  10.0  10.0
```

```
<class 'pandas.core.frame.DataFrame'>
```

Spyder (Python 3.7)

File Edit Search Source Run Debug Consoles Projects Tools View Help

untitled0.py 1.24_series_to_series3.py 1.25_df_to_number.py 1.26_df_to_df.py

```

1 #-*- coding: utf-8 -*-
2
3 # 라이브러리 불러오기
4 import pandas as pd
5 import seaborn as sns
6
7 # titanic 데이터셋에서 age, fare 2개 열을 선택하여 데이터프레임 만들기
8 titanic = sns.load_dataset('titanic')
9 df = titanic.loc[:, ['age', 'fare']]
10 print(df.tail())      #마지막 5행을 표시
11 print('\n')
12 print(type(df))
13 print('\n')
14
15 # 데이터프레임에 숫자 10 더하기
16 addition = df + 10
17 print(addition.tail()) #마지막 5행을 표시
18 print('\n')
19 print(type(addition))
20 print('\n')
21
22 # 데이터프레임끼리 연산하기 (addition - df)
23 subtraction = addition - df
24 print(subtraction.tail()) #마지막 5행을 표시
25 print('\n')
26 print(type(subtraction))
    
```

Variable explorer

Name	type	Size	Value
addition	DataFrame	(891, 2)	Column names: age, fare
df	DataFrame	(891, 2)	Column names: age, fare
titanic	DataFrame	(891, 15)	Column names: survived, pclass, sex, age, sibsp, parch, fare, embarked ...

IPython console

Console 5/A

```

1 48.0 81.2833
2 36.0 17.9250
3 45.0 63.1000
4 45.0 18.0500

In [7]: print('\n')
...: print(type(addition))
...:

<class 'pandas.core.frame.DataFrame'>

In [8]:
    
```

IPython console History log

Permissions: RW End-of-lines: CRLF Encoding: UTF-8 Line: 1 Column: 1 Memory: 87 %

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



Any Question?

Thank you.

