2019/09/21

데이터 전처리 (NUMPY/PANDAS)

5기 최수연

Growth Hackers

CONTENTS

2019/09/21

1. BASIC NUMPY

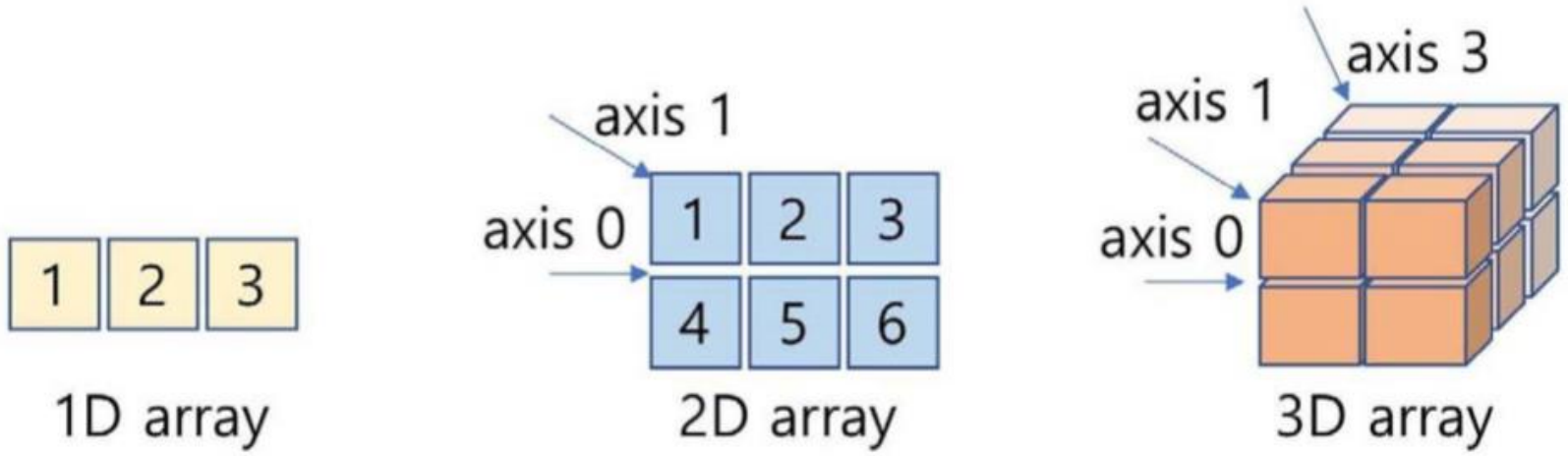
Growth Hackers

Numpy

- 행렬 연산을 위한 핵심 라이브러리로, ‘Numerical Python’의 약자
- **대규모 다차원 배열과 행렬 연산**에 필요한 함수를 제공
- 파이썬 list 객체를 개선한 Numpy의 ndarray 객체를 사용하면 더 많은 데이터를 더 빠르게 처리할 수 있음!

Numpy

- 그림과 같이, 다차원 배열을 지원함.
- 배열의 구조는 'shape'로 표현
- 데이터 방향은 'axis'로 표현됨.(행방향=0, 열방향=1, 채널방향=2)



Growth Hackers

Numpy

<데이터 타입 이해하기>
: dtype으로 데이터 타입을 확인하고, 타입 변경 시엔 astype 사용

자주 나오는 데이터 타입
- np.int32 : 32비트 정수 타입
- np.float64 : 64비트 소수 타입
- np.string : 문자 타입
- np.object : 파이썬 객체 타입

Numpy

np.array 쉽게 생성하기 – 자주 쓰는 3가지!

- np.arange([start],stop,[step]) ex) np.arrange(0,10,2)
- np.linspace(start,stop,number) ex) np.linspace(0,10,5)
- np.random.normal(mean, sd, size) ex) np.random.normal(0,1,(3,3))

Numpy

Array 메서드

- reshape: array의 형태를 반환, 형태 변경 가능(단, 변경할 수 있는 사이즈만 가능)
***실전에선 array.reshape(-1,1)**을 많이 써요! (원본 ndarray가 어떤 형태라도 2차원으로 변환되기 때문에)
- ndim, shape, size: array의 차원, 형태, 원소 개수 반환

Numpy

<배열 연산>

- 항상 Axis를 기준으로 계산!
- np.sum(), np.min(), np.max(), np.mean(), np.dot()등 여러 배열 연산을 할 수 있음. 지정된 axis를 기준으로 계산됨.

(특히 np.sum()에서 많이 헷갈리니 파일을 참고하세요)

* 자세한 것은 'Numpy_Basic.ipynb'로!

2019/09/21

2. BASIC PANDAS

Growth Hackers

Pandas

- 데이터 분석 및 가공에 사용되는 가장 인기 있는 파이썬 라이브러리
- DataFrame: 가로축과 세로축이 있는, 엑셀과 유사한 2차원의 데이터 구조를 가짐
- Series: 데이터프레임의 컬럼
- 지금부터 소개할 판다스의 기능들은 정말 정말 많이 쓰이니 꼭!! 알아두도록 해요☺

Pandas

csv 파일 입출력

- 입력 : pd.read_csv('파일 경로', engine= , index=)
- 출력: pd.to_csv('파일 경로', encoding=)

DataFrame의 기본 정보 조회

- df.shape
- df.info
- df.describe

Pandas

인덱스 설정, 초기화하기

df.index : 인덱스 확인
df.reset_index() :인덱스 초기화
df.set_index() : 인덱스 설정

인덱싱, 슬라이싱, 불린 인덱싱

- 파일로 확인!

Growth Hackers

Pandas

판다스의 주요 메서드

- sorting: 오름차순/내림차순으로 정렬(옵션: ascending=True)
- Groupby: 지정한 칼럼을 기준으로 그룹을 묶어 줌, 항상 aggregation 함수와 함께 사용
- Aggregation 함수: 기본적인 함수들. 뒤에 붙이기만 하면 돼서 사용법이 아주 간단하다.(min, max, sum, mean() 등..)

* 자세한 것은 'Pandas_Basic.ipynb'로!

2019/09/21

3. EXERCISE

미세먼지 데이터 이용 – '실습+Quest.ipynb' 파일로 ☺

Growth Hackers

2019/09/21

4. QUEST

Growth Hackers

Quest

1. 결측치 제거하기!

로드한 'df2'데이터프레임에는 결측치가 상당수 포함되어 있습니다.
결측치를 확인하는 코드를 이용하여, 어떤 칼럼에 결측치가 얼마나 있는지 파악해보세요.
'SO2','CO' 컬럼의 결측치는 그것의 평균값으로 채우고,
'PM10','PM25' 컬럼의 결측치는 모두 삭제하는 코드를 작성해주세요.

```
df2.isna().sum()
```

```
SGG     0
id      0
SO2     0
CO      0
PM10    0
PM25    0
SIDO    0
dtype: int64
```

Quest

2. 기존 칼럼을 응용해, 새로운 칼럼 생성하기!

df2의 'date' 칼럼은 '2018030509'와 같이 '연+월+일+시'로 되어 있습니다.
우리는 여기서 연+월+일만 슬라이싱해 datetime형태로 DATE를 다시 정의했었죠?(ex. 2018-03-05)
마찬가지로, 이번엔 '시'만 슬라이싱해 df2의 새로운 칼럼인 'hour' 칼럼을 만들어주세요.(ex. 09)

	SGG	id	date	SO2	CO	PM10	PM25	DATE	hour
0	서울 중구	111121	2019040101	0.002	0.4	27.0	16.0	2019-04-01	01
1	서울 중구	111121	2019040102	0.003	0.4	25.0	15.0	2019-04-01	02
2	서울 중구	111121	2019040103	0.003	0.4	23.0	13.0	2019-04-01	03
3	서울 중구	111121	2019040104	0.002	0.5	22.0	12.0	2019-04-01	04
4	서울 중구	111121	2019040105	0.002	0.4	23.0	12.0	2019-04-01	05

Quest

3. 종합 문제!

미세먼지가 나쁜 기준은 초미세먼지(' PM25 ')가 35보다 크면 ' 나쁨 ' 으로 분류한다고 합니다.
불린 인덱싱과, 위의 2번에서 만든 ' hour ' 칼럼을 이용하여, 시군구(SGG)별로 PM25가 35보다 큰 시간들을 카운트하고, 초미세먼지가 35보다 큰 개수가 많은 시군구와 그 시간대가 언제인지 sorting하여 구해주세요.

SGG	hour		
경기 부천시	09	69	
	10	69	
	08	62	
	11	56	
	07	55	
충북 청주시	10	55	
경기 부천시	06	52	
충북 청주시	09	51	
경기 부천시	05	51	
경기 화성시	09	51	
경기 부천시	04	50	
충남 아산시	10	50	
경기 부천시	03	49	
	02	48	

정답으로는 이렇게 '경기도 부천시', '오전 9시',' 오전 10시'가 69개씩 카운트되어,
오전 9시&10시의 경기도 부천시에서 가장 미세먼지가 나빴음을 알 수 있네요!

Growth Hackers