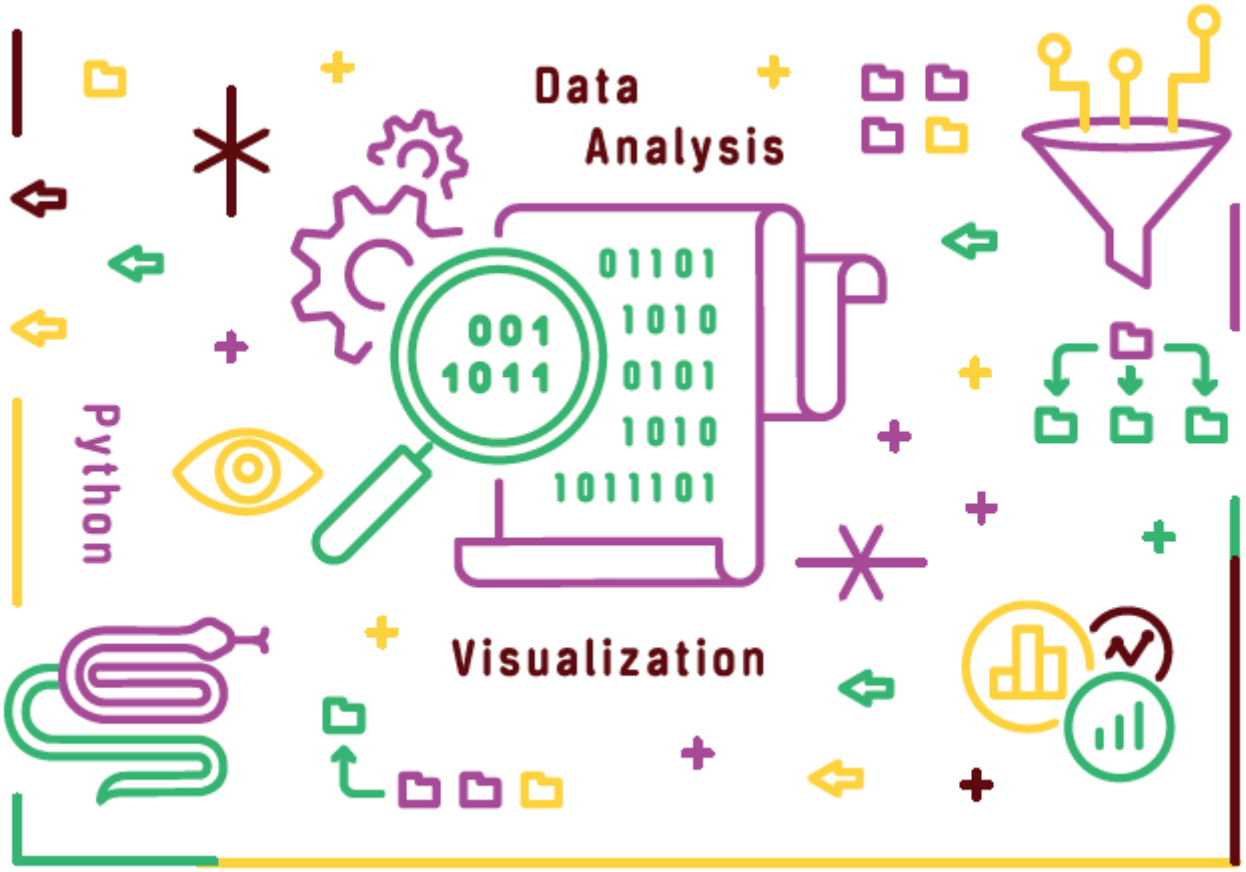


<8장> 환경 관련 데이터 분석

학습 목표

- 환경 관련 데이터 분석을 위한 분석 대상 데이터를 수집한다.
- 수집한 데이터를 목적에 따라 가공한다.
- 데이터를 분석하고 시각화하여 분석 결과를 해석한다.

목차

01 분석 대상 데이터 수집

02 데이터 확인

03 데이터 병합

04 데이터 분석 및 시각화

01

분석 대상 데이터 수집

1. 데이터 시각화 이해

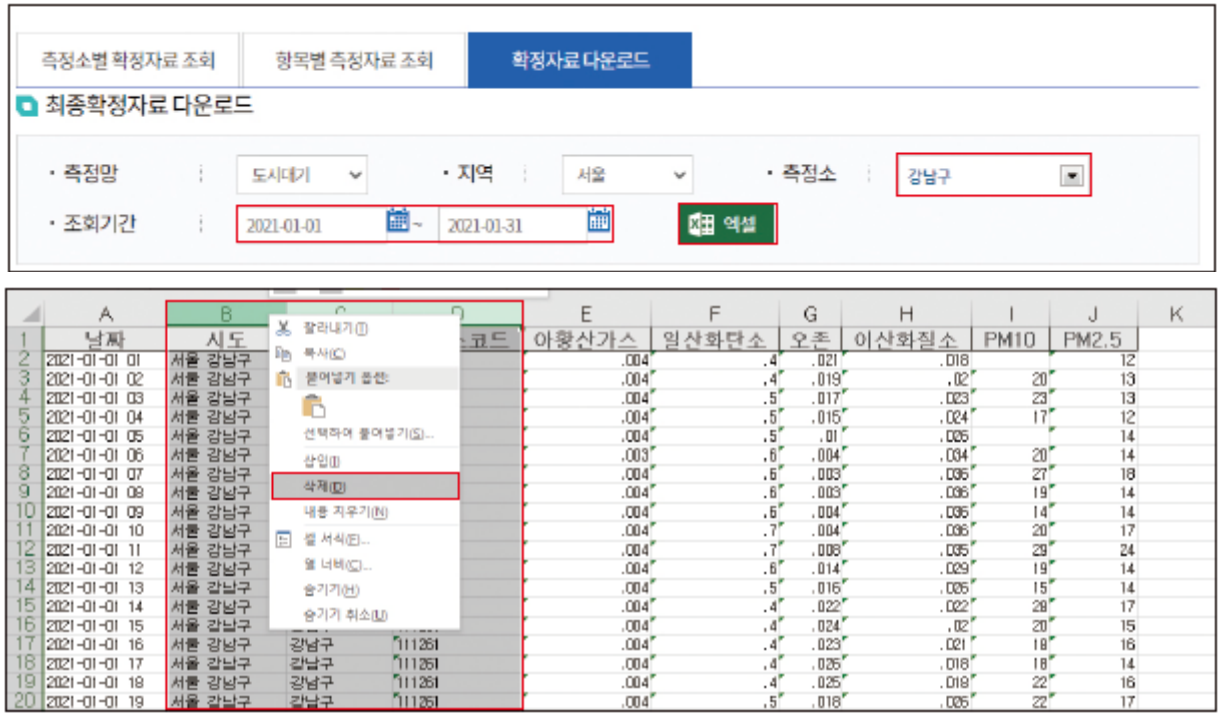
에어코리아 사이트에서 미세먼지 데이터 수집

- (1) 에어코리아(https://www.airkorea.or.kr/index)에 접속한다.
- 메뉴에서 '통계정보' → '최종확정자료다운로드'를 클릭한다.

1. 데이터 시각화 이해

에어코리아 사이트에서 미세먼지 데이터 수집

- (2) 다음과 같이 조회 기간과 측정소를 지정하고 <엑셀> 버튼을 클릭하여 파일을 다운로드한다.
- (3) 엑셀을 실행하고 다운로드한 파일을 연다. 데이터 분석에 필요 없는 컬럼(B~D)을 삭제한다.



1. 데이터 시각화 이해

■ 에어코리아 사이트에서 미세먼지 데이터 수집

- (4) 파일 형식을 'Excel 통합 문서(*.xlsx)'로 변경해서 저장한다.

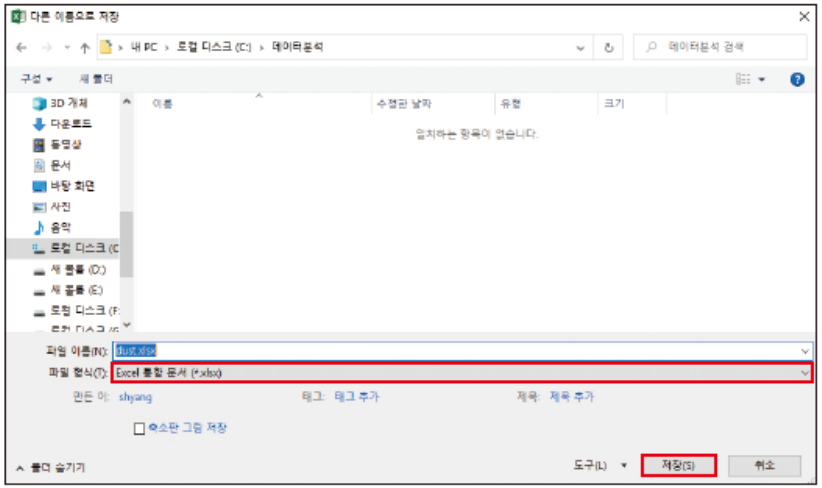


표 8-1. 미세먼지 데이터 정보(파일 : dust.xlsx)

변수명	변수 설명	단위
아황산가스(SO2)	대기오염물질, 아황산가스의 공기 중 농도	ppm
일산화탄소(CO)	대기오염물질, 일산화탄소의 공기 중 농도	ppm
오존(O3)	대기오염물질, 오존의 공기 중 농도	ppm
이산화질소(NO2)	대기오염물질, 이산화질소의 공기 중 농도	ppm
PM10	1000분의 10mm보다 작은 먼지의 공기 중 농도 (미세먼지)	microgram/cubicmeter
PM2.5	1000분의 2.5mm보다 작은 먼지의 공기 중 농도 (초미세먼지)	microgram/cubicmeter

1. 데이터 시각화 이해

■ 기상청 사이트에서 날씨 데이터 수집

- (1) 기상청(https://data.kma.go.kr/cmmn/main.do)에 접속한다.
- 메뉴에서 '데이터'를 클릭한다.

1. 데이터 시각화 이해

■ 기상청 사이트에서 날씨 데이터 수집

- (2) 왼쪽 메뉴에서 '기상관측' → '지상' → '방재기상관측(AWS)'를 선택한다.
- 자료 화면에서 지점을 선택하고 데이터 분석에 필요한 자료를 선택한 한다.
- <조회> 버튼을 클릭 후 <엑셀>버튼을 클릭하여 다운로드한다.

1. 데이터 시각화 이해

■ 기상청 사이트에서 날씨 데이터 수집

- (3) 엑셀을 실행하고 다운로드한 파일을 열어 데이터를 확인한다.

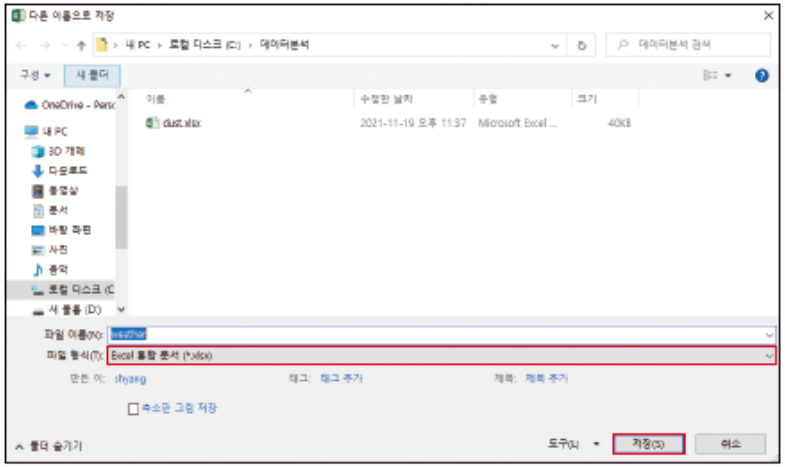


표 8-2. 날씨 정보에 관한 데이터 정보(파일 : weather.xlsx)

변수명	변수 설명	단위
기온	공기의 온도	℃
풍속	바람의 속도	m/s
강수량	강수량 혹은 강우량은 어떤 곳에 일정 기간 동안 내린 물의 총량	mm
습도	공기 중에 포함되어 있는 수증기의 양 또는 비율을 나타내는 단위	%

1. 데이터 시각화 이해

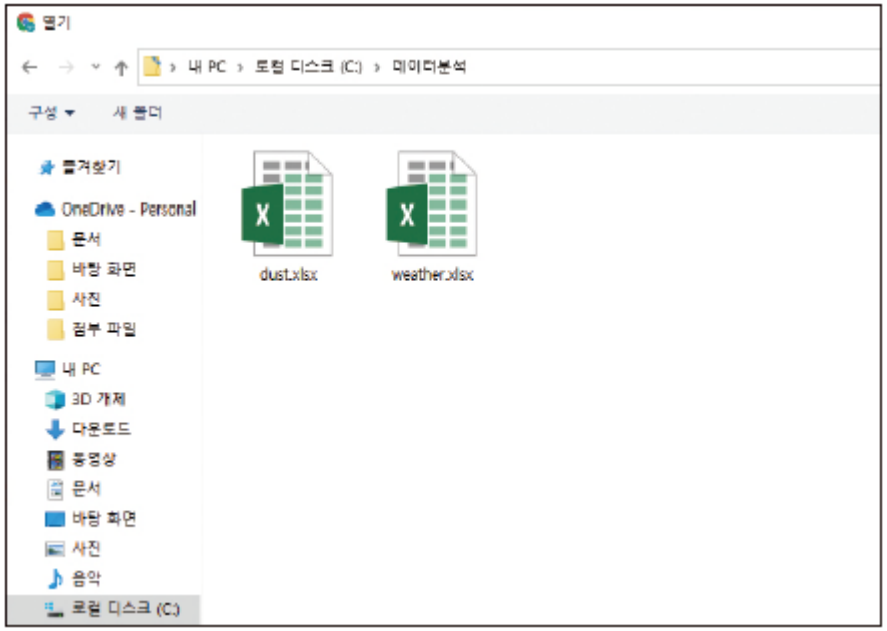
구글 코랩에 업로드

- (1) 왼쪽 메뉴에서 '파일' 아이콘을 클릭하고 파일 화면이 열리면 '드라이브 마운트' 아이콘을 클릭한다.

1. 데이터 시각화 이해

구글 코랩에 업로드

- (2) ‘MyDrive’ 폴더에서 마우스 오른쪽 버튼을 클릭하여 ‘업로드’를 선택한다.
- (3) 열기 대화상자가 표시되면 수집한 2개의 엑셀 파일을선택하여 구글 코랩에 업로드 한다.



02

데이터 확인

2. 데이터 확인

■ 미세먼지 데이터

▪ 데이터 읽어서 확인하기

'미세먼지' 엑셀 파일 읽어오기

```
1  import pandas as pd
2  # dust.xlsx 불러오기
3  file_path='/content/drive/MyDrive/dust.xlsx'
4  dust=pd.read_excel(file_path)
5  dust.head()
```

〈실행결과〉

	날짜	아황산가스	일산화탄소	오존	이산화질소	PM10	PM2.5
0	2021-01-01 01	0.004	0.4	0.021	0.018	NaN	12.0
1	2021-01-01 02	0.004	0.4	0.019	0.020	20.0	13.0
2	2021-01-01 03	0.004	0.5	0.017	0.023	23.0	13.0
3	2021-01-01 04	0.004	0.5	0.015	0.024	17.0	12.0
4	2021-01-01 05	0.004	0.5	0.010	0.026	NaN	14.0

데이터의 기본 정보 출력하기

〈코드〉

```
1  dust.info()
```

〈실행결과〉

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 744 entries, 0 to 743
Data columns (total 7 columns):
 #   Column  Non-Null Count  Dtype
---  ------  --------------  -----
 0   날짜      744 non-null    object
 1   아황산가스   740 non-null    float64
 2   일산화탄소   740 non-null    float64
 3   오존      740 non-null    float64
 4   이산화질소   740 non-null    float64
 5   PM10    725 non-null    float64
 6   PM2.5   739 non-null    float64
dtypes: float64(6), object(1)
memory usage: 40.8+ KB
```

데이터의 기초 통계량 출력

```
1  dust.describe()
```

〈실행결과〉

	아황산가스	일산화탄소	오존	이산화질소	PM10	PM2.5
count	740.000000	740.000000	740.000000	740.000000	725.000000	739.000000
mean	0.003654	0.563243	0.014154	0.030422	33.325517	21.833559
std	0.000628	0.164593	0.010689	0.014664	19.930029	12.222892
min	0.002000	0.300000	0.001000	0.006000	3.000000	3.000000
25%	0.003000	0.400000	0.003000	0.017000	20.000000	13.000000
50%	0.004000	0.500000	0.014000	0.030000	29.000000	19.000000
75%	0.004000	0.700000	0.024000	0.043000	43.000000	29.000000
max	0.006000	1.200000	0.037000	0.063000	163.000000	72.000000

2. 데이터 확인

■ 미세먼지 데이터

▪ 데이터 가공하기

한글 컬럼명을 영문명으로 변경

```
1  dust.rename(columns={'날짜':'date','아황산가스':'so2',
2                       '일산화탄소':'co','오존':'o3',
3                       '이산화질소':'no2'},inplace=True)
4  dust.head()
```

〈실행결과〉

	date	so2	co	o3	no2	PM10	PM2.5
0	2021-01-01 01	0.004	0.4	0.021	0.018	NaN	12.0
1	2021-01-01 02	0.004	0.4	0.019	0.020	20.0	13.0
2	2021-01-01 03	0.004	0.5	0.017	0.023	23.0	13.0
3	2021-01-01 04	0.004	0.5	0.015	0.024	17.0	12.0
4	2021-01-01 05	0.004	0.5	0.010	0.026	NaN	14.0

특정 문자열 추출

```
1  dust['date']=dust['date'].str[:11]
2  dust.head()
```

〈실행결과〉

	date	so2	co	o3	no2	PM10	PM2.5
0	2021-01-01	0.004	0.4	0.021	0.018	NaN	12.0
1	2021-01-01	0.004	0.4	0.019	0.020	20.0	13.0
2	2021-01-01	0.004	0.5	0.017	0.023	23.0	13.0
3	2021-01-01	0.004	0.5	0.015	0.024	17.0	12.0
4	2021-01-01	0.004	0.5	0.010	0.026	NaN	14.0

2. 데이터 확인

■ 미세먼지 데이터

▪ 데이터 가공하기

데이터형 변경

```
1  dust['date']=pd.to_datetime(dust['date'])
2  dust.dtypes
```

〈실행결과〉

```
date    datetime64[ns]
so2            float64
co             float64
o3             float64
no2            float64
PM10           float64
PM2.5          float64
dtype: object
```

새로운 컬럼 생성

```
1  dust['year']=dust['date'].dt.year
2  dust['month']=dust['date'].dt.month
3  dust['day']=dust['date'].dt.day
4  dust.columns
```

〈실행결과〉

```
Index(['date', 'so2', 'co', 'o3', 'no2', 'PM10', 'PM2.5', 'year',
'month', 'day'],dtype='object')
```

컬럼 순서 재정렬

```
1  dust=dust[['date','year','month','day','so2','co','o3','no2','PM10',
   'PM2.5']]
2  dust.head()
```

〈실행결과〉

```
Index(['date','year','month','day','so2','co','o3','no2','PM10','PM2.5'],
dtype='object')
```

2. 데이터 확인

■ 미세먼지 데이터

▪ 데이터 전처리

결측치 확인하기

〈코드〉

```
1  dust.isnull().sum()
```

〈실행결과〉

```
date      0
year      0
month     0
day       0
so2       4
co        4
o3        4
no2       4
PM10     19
PM2.5     5
dtype: int64
```

결측치 처리 : 이전 데이터로 채우기

```
1  dust=dust.fillna(method='pad')
```

	date	year	month	day	so2	co	o3	no2	PM10	PM2.5
0	2021-01-01	2021	1	1	0.004	0.4	0.021	0.018	NaN	12.0
1	2021-01-01	2021	1	1	0.004	0.4	0.019	0.020	20.0	13.0
2	2021-01-01	2021	1	1	0.004	0.5	0.017	0.023	23.0	13.0
3	2021-01-01	2021	1	1	0.004	0.5	0.015	0.024	17.0	12.0
4	2021-01-01	2021	1	1	0.004	0.5	0.010	0.026	NaN	14.0

〈실행결과〉

	date	year	month	day	so2	co	o3	no2	PM10	PM2.5
0	2021-01-01	2021	1	1	0.004	0.4	0.021	0.018	NaN	12.0
1	2021-01-01	2021	1	1	0.004	0.4	0.019	0.020	20.0	13.0
2	2021-01-01	2021	1	1	0.004	0.5	0.017	0.023	23.0	13.0
3	2021-01-01	2021	1	1	0.004	0.5	0.015	0.024	17.0	12.0
4	2021-01-01	2021	1	1	0.004	0.5	0.010	0.026	17.0	14.0

2. 데이터 확인

미세먼지 데이터

- 데이터 전처리

결측치 처리 : 이전 값이 없는 경우 특정 값으로 채우기

```
1  dust.fillna(20,inplace=True)
2  dust.head()
```

〈실행결과〉

	date	year	month	day	so2	co	o3	no2	PM10	PM2.5
0	2021-01-01	2021	1	1	0.004	0.4	0.021	0.018	20.0	12.0
1	2021-01-01	2021	1	1	0.004	0.4	0.019	0.020	20.0	13.0
2	2021-01-01	2021	1	1	0.004	0.5	0.017	0.023	23.0	13.0
3	2021-01-01	2021	1	1	0.004	0.5	0.015	0.024	17.0	12.0
4	2021-01-01	2021	1	1	0.004	0.5	0.010	0.026	17.0	14.0

결측치 확인

〈코드〉

```
1  dust.isnull().sum()
```

〈실행결과〉

```
date      0
year      0
month     0
day       0
so2       0
co        0
o3        0
no2       0
PM10      0
PM2.5     0
dtype:int64
```

2. 데이터 확인

날씨 데이터

- 데이터 읽어와서 확인하기

'날씨데이터' 엑셀 파일 읽어오기

```
1  file_path='/content/drive/MyDrive/weather.xlsx'
2  weather=pd.read_excel(file_path)
3  weather.head()
```

〈실행결과〉

	지점	지점명	일시	기온(°C)	풍속(m/s)	강수량(mm)	습도(%)
0	400	강남	2021-01-01 01:00:00	-7.2	0.6	0.0	57.5
1	400	강남	2021-01-01 02:00:00	-7.6	0.7	0.0	57.5
2	400	강남	2021-01-01 03:00:00	-8.2	0.6	0.0	62.0
3	400	강남	2021-01-01 04:00:00	-8.1	0.5	0.0	60.5
4	400	강남	2021-01-01 05:00:00	-8.7	1.3	0.0	66.4

데이터의 기본 정보 출력

```
1  weather.info()
```

〈실행결과〉

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 743 entries, 0 to 742
Data columns (total 7 columns):
 #   Column   Non-Null Count  Dtype
---  ------   --------------  -----
 0   지점       743 non-null    int64
 1   지점명      743 non-null    object
 2   일시       743 non-null    datetime64[ns]
 3   기온(°C)   743 non-null    float64
 4   풍속(m/s)  743 non-null    float64
 5   강수량(mm)  743 non-null    float64
 6   습도(%)    743 non-null    float64
dtypes: datetime64[ns](1), float64(4), int64(1), object(1)
memory usage: 40.8+ KB
```

2. 데이터 확인

날씨 데이터

- 데이터 가공하기

컬럼 삭제 : 지점, 지점명 삭제

```
1  weather.drop('지점',axis=1,inplace=True)
2  weather.drop('지점명',axis=1,inplace=True)
3  weather.head()
```

〈실행결과〉

	일시	기온(°C)	풍속(m/s)	강수량(mm)	습도(%)
0	2021-01-01 01:00:00	-7.2	0.6	0.0	57.5
1	2021-01-01 02:00:00	-7.6	0.7	0.0	57.5
2	2021-01-01 03:00:00	-8.2	0.6	0.0	62.0
3	2021-01-01 04:00:00	-8.1	0.5	0.0	60.5
4	2021-01-01 05:00:00	-8.7	1.3	0.0	66.4

컬럼명 변경 : columns 사용

```
1  weather.columns=['date','temp','wind','rain','humid']
2  weather.info()
```

〈실행결과〉

	date	temp	wind	rain	humid
0	2021-01-01 01:00:00	-7.2	0.6	0.0	57.5
1	2021-01-01 02:00:00	-7.6	0.7	0.0	57.5
2	2021-01-01 03:00:00	-8.2	0.6	0.0	62.0
3	2021-01-01 04:00:00	-8.1	0.5	0.0	60.5
4	2021-01-01 05:00:00	-8.7	1.3	0.0	66.4

2. 데이터 확인

■ 날씨 데이터

▪ 데이터 가공하기

날짜 컬럼 시간 데이터 제거 : date

```
1  weather['date']=pd.to_datetime(weather['date']).dt.date
2  weather['date']=weather.astype('datetime64[ns]')
3  weather.info()
```

〈실행결과〉

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 743 entries, 0 to 742
Data columns (total 5 columns):
 #   Column  Non-Null Count  Dtype
---  ------  --------------  -----
 0   date    743 non-null    datetime64[ns]
 1   temp    743 non-null    float64
 2   wind    743 non-null    float64
 3   rain    743 non-null    float64
 4   humid   743 non-null    float64
dtypes: datetime64[ns](1), float64(4)
memory usage: 29.1 KB
```

데이터 변경

```
1  weather['rain']=weather['rain'].replace([0], 0.01)
2  weather['rain'].value_counts()
```

〈실행결과〉

```
0.01  720
0.50    9
1.00    7
1.50    3
2.00    2
2.50    2
Name: rain, dtype: int64
```

03

데이터 병합

3. 데이터 병합

■ 미세먼지와 날씨 데이터 병합

- (1) 데이터를 병합하기 전 미세먼지 데이터(dust)와 날씨 데이터(weather)의 행, 열 크기를 확인한다.

미세먼지 데이터 행, 열 크기 확인

```
1  dust.shape
```

〈실행결과〉

```
(744, 10)
```

날씨 데이터 행, 열 크기 확인

```
1  weather.shape
```

〈실행결과〉

```
(743, 5)
```

3. 데이터 병합

■ 미세먼지와 날씨 데이터 병합

- (2) 미세먼지 데이터에서 날씨데이터와 공통적인 내용이 아닌 행을 찾아서 삭제한다.
- (3) dust와 weather 데이터프레임이 동일하게 가진 date 컬럼을 기준으로 병합해서 df 프레임을 생성한 후 확인한다.

미세먼지 데이터 특정 행 삭제

```
1  dust.drop(index=743,inplace=True)
```

〈실행결과〉

```
(743, 10)
```

미세먼지 데이터와 날씨 데이터 병합

```
1  df=pd.merge(dust,weather,on='date')
2  df.head()
```

〈실행결과〉

	date	year	month	day	so2	co	o3	no2	PM10	PM2.5	temp	wind	rain	humid
0	2021-01-01	2021	1	1	0.004	0.4	0.021	0.018	20.0	12.0	-7.2	0.6	0.01	57.5
1	2021-01-01	2021	1	1	0.004	0.4	0.021	0.018	20.0	12.0	-7.6	0.7	0.01	57.5
2	2021-01-01	2021	1	1	0.004	0.4	0.021	0.018	20.0	12.0	-8.2	0.6	0.01	62.0
3	2021-01-01	2021	1	1	0.004	0.4	0.021	0.018	20.0	12.0	-8.1	0.5	0.01	60.5
4	2021-01-01	2021	1	1	0.004	0.4	0.021	0.018	20.0	12.0	-8.7	1.3	0.01	66.4

04

데이터 분석 및 시각화

4. 데이터 분석 및 시각화

데이터 분석

- (1) 상관 계수를 계산하는 corr( ) 함수를 이용하여 미세먼지 데이터와 날씨 데이터의 모든 요소별 상관관계를 확인한다.
- (2) 미세먼지(PM10)를 기준으로 각 변수와의 상관관계를 알아본다.

모든 요소별 상관관계 확인

```
1  df.corr()
```

〈실행결과〉

	year	month	day	so2	co	o3	no2	PM10	PM2.5	temp	wind	rain	humid
year	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
month	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
day	NaN	NaN	1.000000	-0.319732	0.227379	-0.119688	0.209155	0.016818	0.050328	0.491673	-0.075695	0.026279	0.176865
so2	NaN	NaN	-0.319732	1.000000	0.139818	-0.067423	0.083555	0.160285	0.147822	-0.375592	0.032008	-0.019673	-0.096696
co	NaN	NaN	0.227379	0.139818	1.000000	-0.756933	0.841533	0.529804	0.692035	0.318469	-0.322632	0.077316	0.337989
o3	NaN	NaN	-0.119688	-0.067423	-0.756933	1.000000	-0.924152	-0.348411	-0.524511	-0.204392	0.355292	-0.096531	-0.288304
no2	NaN	NaN	0.209155	0.083555	0.841533	-0.924152	1.000000	0.420515	0.564851	0.314004	-0.403820	0.109401	0.315366
PM10	NaN	NaN	0.016818	0.160285	0.529804	-0.348411	0.420515	1.000000	0.825227	0.175605	-0.108610	0.026195	0.216703
PM2.5	NaN	NaN	0.050328	0.147822	0.692035	-0.524511	0.564851	0.825227	1.000000	0.190270	-0.201781	0.069162	0.354332
temp	NaN	NaN	0.491673	-0.375592	0.318469	-0.204392	0.314004	0.175605	0.190270	1.000000	-0.210847	0.077425	0.212419
wind	NaN	NaN	-0.075695	0.032008	-0.322632	0.355292	-0.403820	-0.108610	-0.201781	-0.210847	1.000000	-0.077878	-0.462202
rain	NaN	NaN	0.026279	-0.019673	0.077316	-0.096531	0.109401	0.026195	0.069162	0.077425	-0.077878	1.000000	0.283847
humid	NaN	NaN	0.176865	-0.096696	0.337989	-0.288304	0.315366	0.216703	0.354332	0.212419	-0.462202	0.283847	1.000000

미세먼지와 다른 요소와의 상관관계

〈코드〉

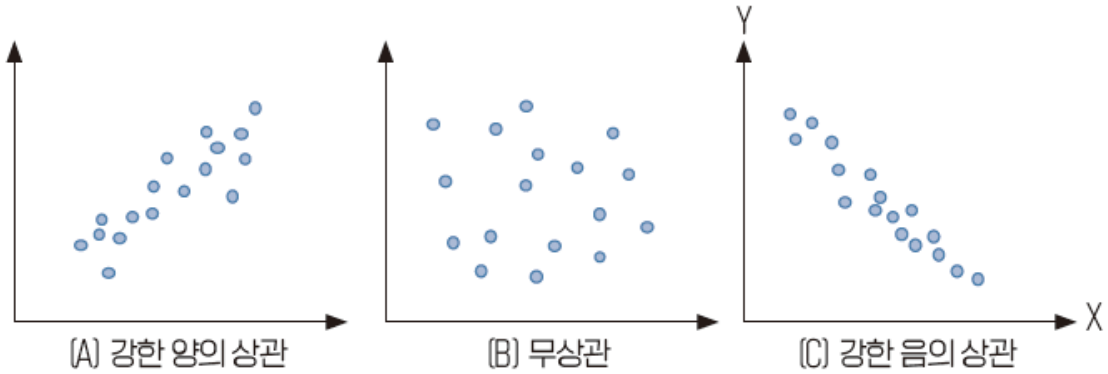
```
1  corr=df.corr()
2  corr
3  corr['PM10'].sort_values(ascending=False)
```

〈실행결과〉

```
PM10        1.000000
PM2.5       0.825227
co          0.529804
no2         0.420515
humid       0.216703
temp        0.175605
so2         0.160285
rain        0.026195
day         0.016818
wind       -0.108610
o3         -0.348411
year             NaN
month            NaN
Name: PM10, dtype: float64
```

4. 데이터 분석 및 시각화

■ 데이터 분석



상관	상관 계수
양의 상관	+0.7 ~ +1.0이면, 강한 양의 상관관계 +0.3 ~ +0.7이면, 뚜렷한 양의 상관관계 +0.1 ~ +0.3이면, 약한 양의 상관관계
무상관	−0.1 ~ +0.1이면, 관계가 없음
음의 상관	−1.0 ~ −0.7이면, 강한 음의 상관관계 −0.7 ~ −0.3이면, 뚜렷한 음의 상관관계 −0.3 ~ −0.1이면, 약한 음의 상관관계

4. 데이터 분석 및 시각화

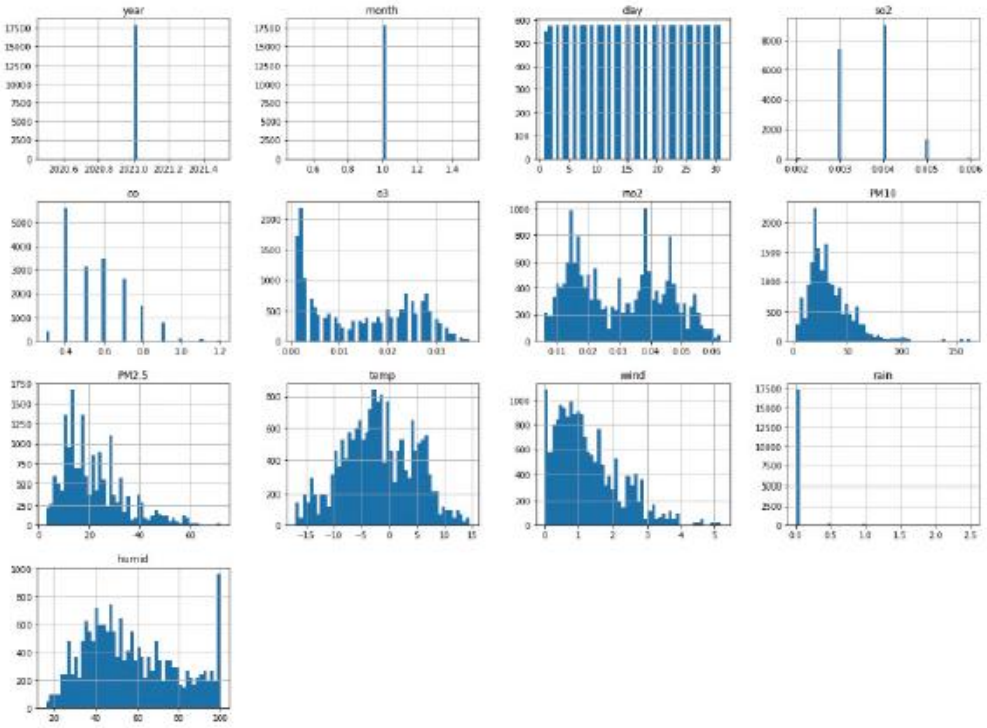
데이터 시각화

- 히스토그램 그래프로 시각화

히스토그램 그래프로 시각화

```
1   df.hist(bins=50,figsize=(20,15))
```

〈실행결과〉



4. 데이터 분석 및 시각화

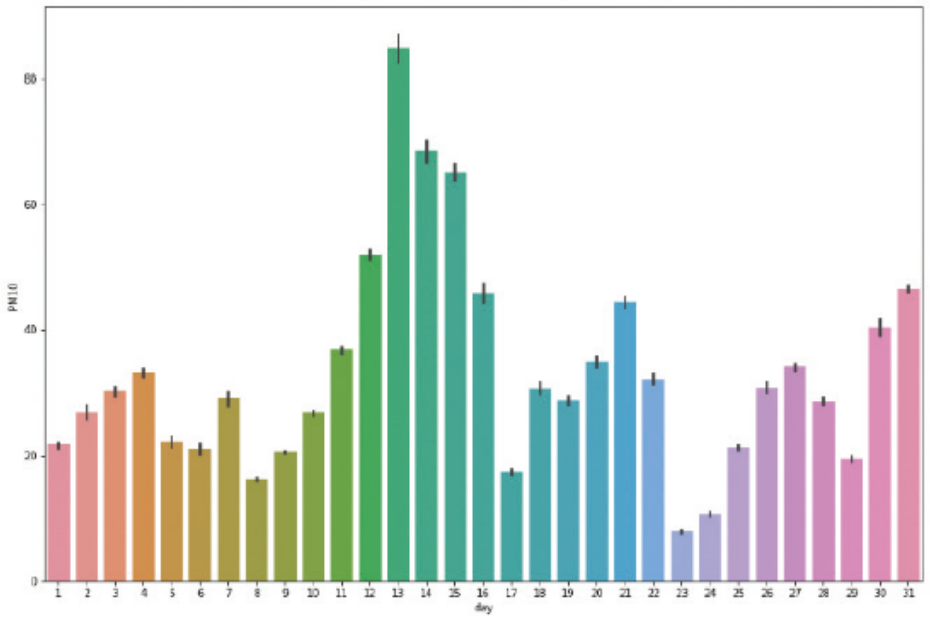
데이터 시각화

- 막대 그래프로 시각화

막대 그래프로 시각화

```
1  import matplotlib.pyplot as plt
2  import seaborn as sns
3  plt.figure(figsize=(15,10))
4  dayGraph=sns.barplot(x='day',y='PM10',data=df)
5  plt.xticks(rotation=0)
6  plt.show()
```

〈실행결과〉



4. 데이터 분석 및 시각화

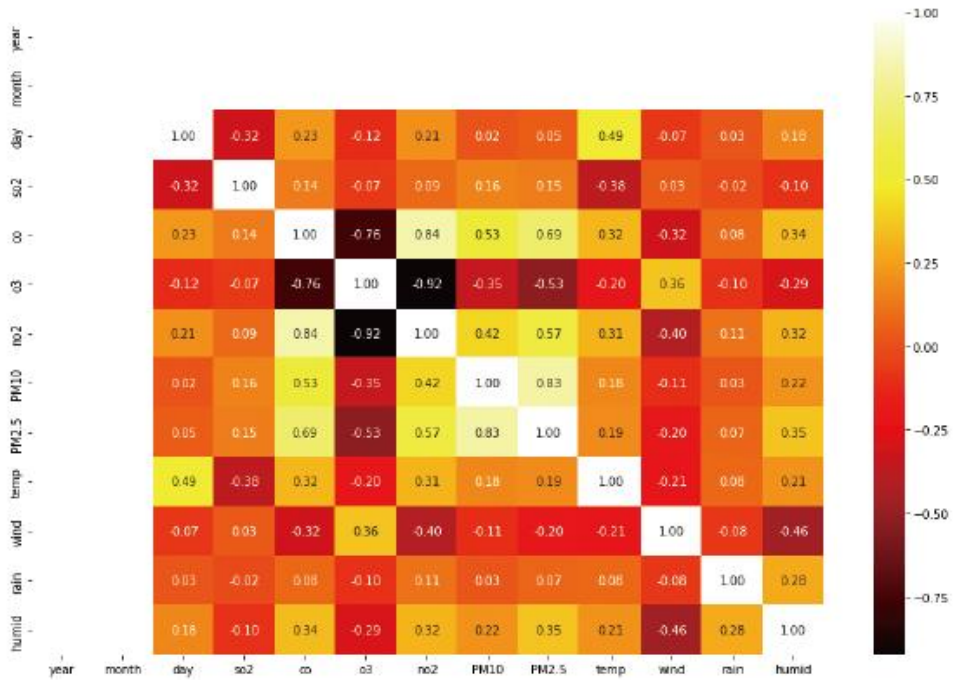
■ 데이터 시각화

- 히트맵 그래프로 시각화

히트맵 그래프로 시각화

```
1  plt.figure(figsize=(15,10))
2  sns.heatmap(data=corr, annot=True, fmt='.2f',cmap='hot')
```

〈실행결과〉



4. 데이터 분석 및 시각화

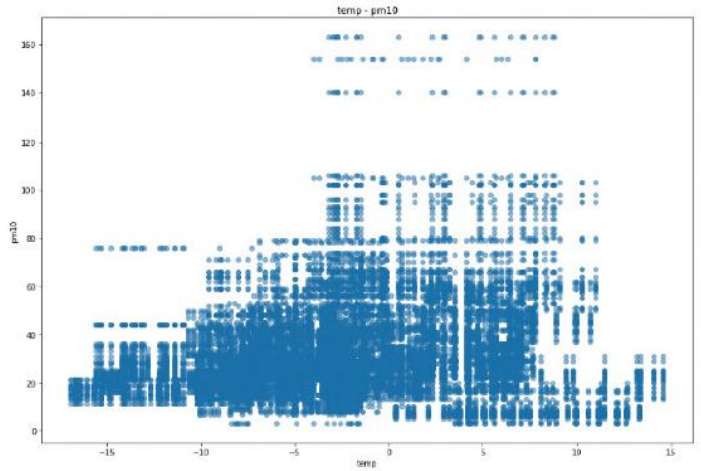
■ 데이터 시각화

▪ 산점도 그래프로 시각화

산점도 그래프로 시각화 (1)

```
1  plt.figure(figsize=(15,10))
2  x=df['temp']  # 온도
3  y=df['PM10'] # 미세먼지
4  plt.plot(x,y,marker='o',linestyle='none',alpha=0.5)
5  plt.title('temp - pm10')
6  plt.xlabel('temp')
7  plt.ylabel('pm10')
8  plt.show()
```

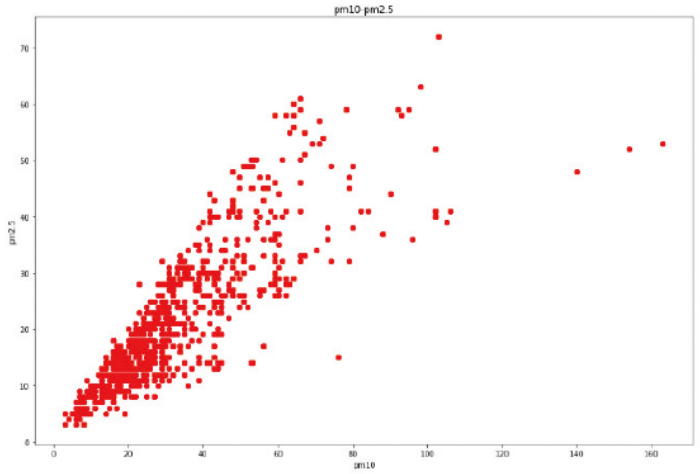
〈실행결과〉



산점도 그래프로 시각화 (2)

```
1  plt.figure(figsize=(15,10))
2  x=df['PM10']  # 미세먼지
3  y=df['PM2.5']  # 초미세먼지
4  plt.plot(x,y,marker='o',linestyle='none',color='red',alpha=0.5)
5  plt.title('pm10-pm2.5')
6  plt.xlabel('pm10')
7  plt.ylabel('pm2.5')
8  plt.show()
```

〈실행결과〉



4. 데이터 분석 및 시각화

■ 데이터 시각화

분석 요약

- 미세먼지(pm10)와 초미세먼지(pm2.5)는 강한 관계성이 있다.
- 미세먼지 변수 중 대기오염과 관련된 변수들은 관련성이 있다.
- 일산화탄소(co)와 이산화질소(no2)는 강한 관계성이 있다.
- 오존(o3)과 바람(wind)은 약한 관계성이 있다.
- 기온(temp)과 미세먼지는 무관하다.