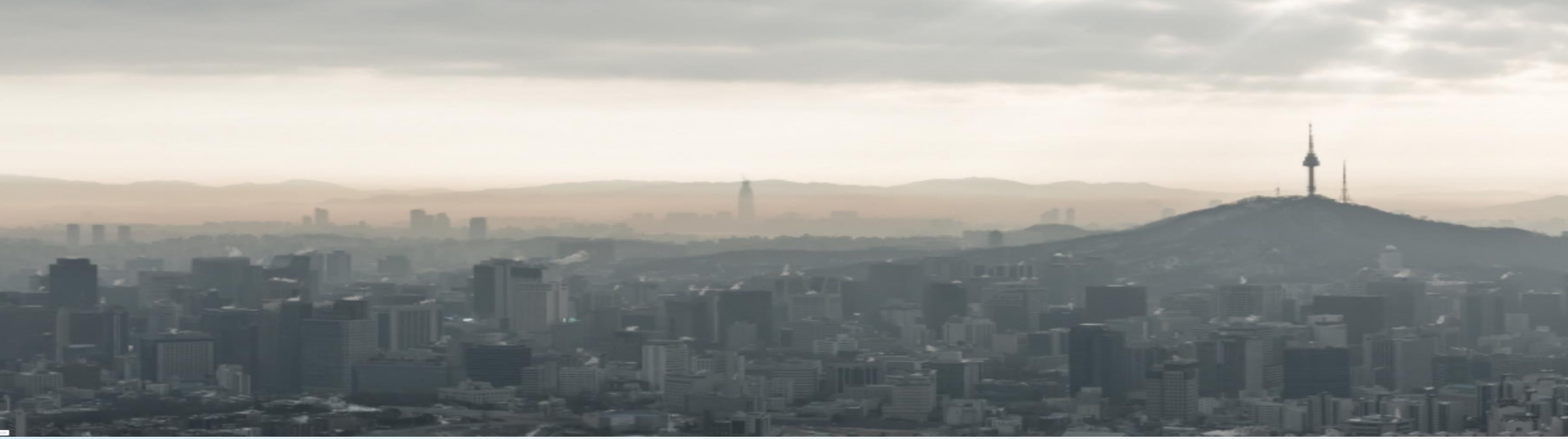


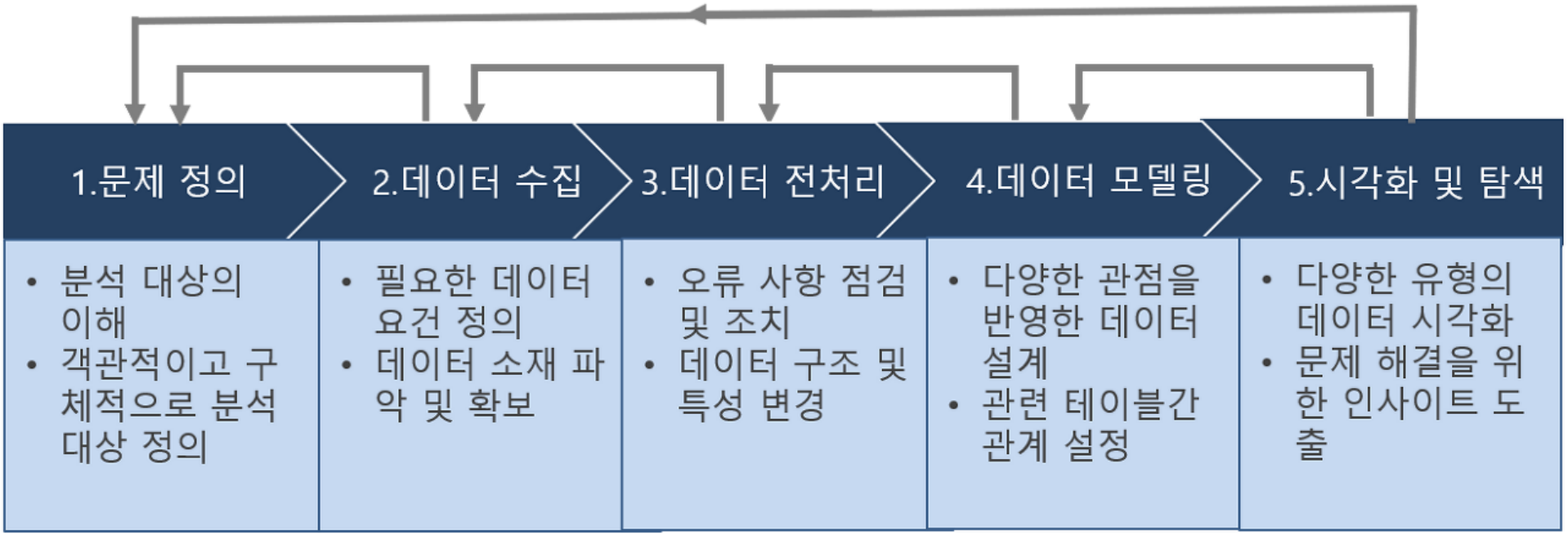
미세먼지 요인 분석 및 개선안 도출
POSCO AI BIGDATA 아카데미 종합실습 1

A반 이무동

과제 정의

한국, 화석연료 미세먼지로 연간 8만명 사망 – 사망률 세계 4위
이에 맞추어 미세먼지 요인 분석 및 개선안 도출이 시급함
'AIR-POLLUTION'과 같은 데이터를 통해 미세먼지와 관련된 데이터 분석 진행

분석 계획



데이터 현황

데이터 구성 및 전처리

```
[3]  # 데이터 구성하기
     df_raw = pd.read_csv('/content/sample_data/AIR_POLLUTION.csv', encoding='euc-kr')
     df_raw
```

'AIR-POLLUTION.CSV'를 READ 하여 데이터 구성하기

```
[7]  # 데이터 구성하기 - 결측치 처리
     df_raw.isnull().sum()
```

```
MeasDate      0
PM10          1
O3            1
NO2           1
CO           55
SO2           1
TEMP          0
RAIN          0
WIND          0
WIND_DIR      0
HUMIDITY      0
ATM_PRESS     0
SNOW          0
CLOUD         0
dtype: int64
```

PM10, O3, NO2, CO, SO2 결측치 처리 필요

```
[10] #PM10, O3 , NO2, SO2 결측치는 같은 인덱스에 있어 제거함.
     df_raw[df_raw['PM10'].isnull()]
```

	MeasDate	PM10	O3	NO2	CO	SO2	TEMP	RAIN	WIND	WIND_DIR	HUMIDITY	ATM_PRESS	CLOUD
328	2020-05-24	NaN	NaN	NaN	NaN	NaN	17.48	1.45	2.85	257	83.3	999.4	8.21

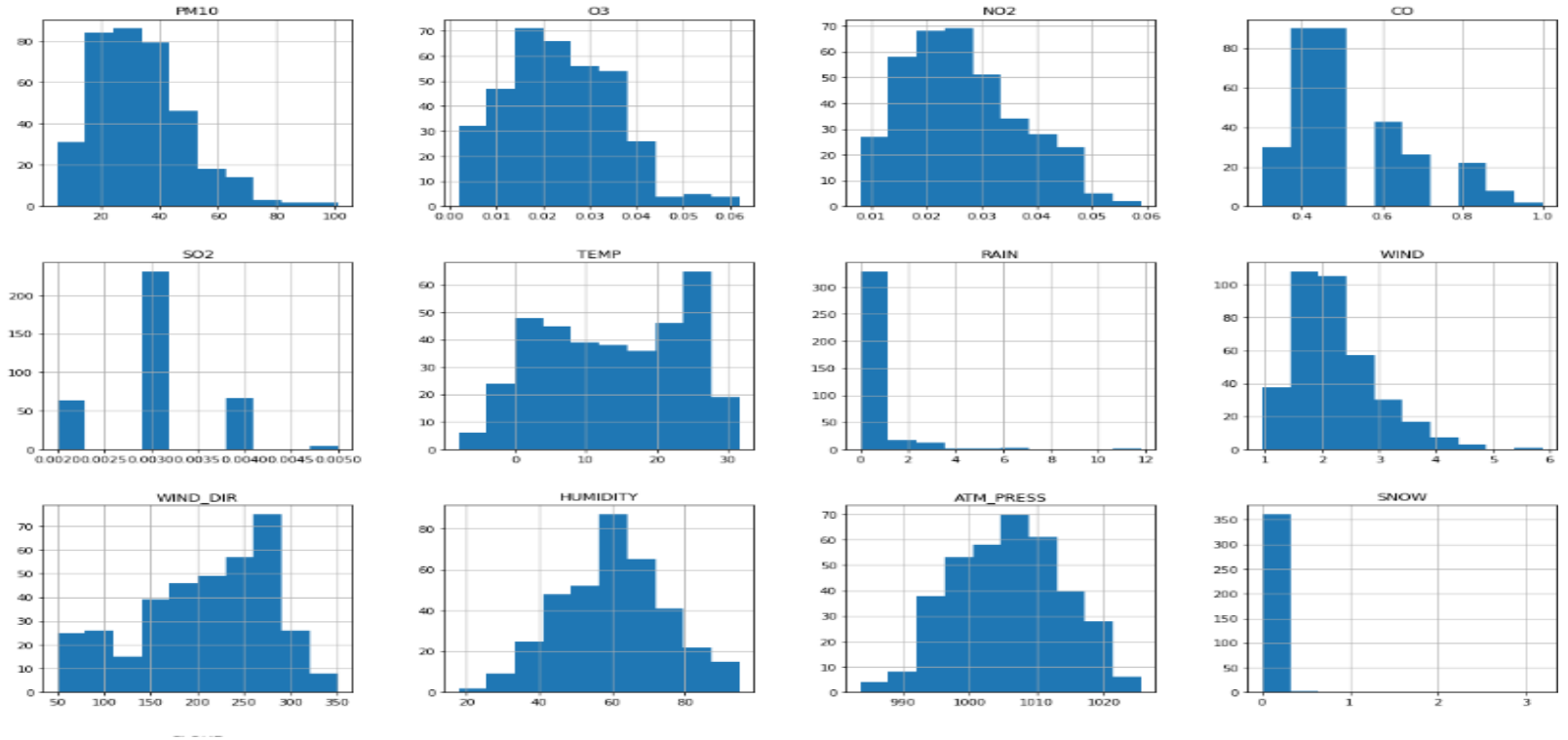
PM10, O3, NO2, SO2 결측치는 같은 인덱스에 있어 1개씩 있어 제거

```
# CO의 결측치 값이 55개이므로, 결측치 값을 평균값으로 대체
means= df_raw["CO"].mean().round(1)
df_raw["CO"] = df_raw["CO"].fillna(means)
```

CO의 결측치 값이 55개이므로, 결측치 값을 평균값으로 대체

히스토그램을 통하여 'SNOW' 변수 삭제

```
# 전체적인 데이터의 히스토그램으로 제거해야할 변수 선택
df_raw.hist(figsize = (20,20))
```




```
# 히스토그램의 관찰 결과, 측정 기간 동안 'SNOW'는 거의 없었다고 볼 수 있으므로 변수 삭제
df_raw = df_raw.loc[:, ['MeasDate','PM10','O3','NO2','CO','SO2','TEMP','RAIN','WIND','WIND_DIR','HUMIDITY','ATM_PRESS','CLOUD']]
df_raw
```

최종 DATA -> 목표변수 : PM10(미세먼지 지수)

```
df_raw = df_raw.dropna()
df_raw
```

	MeasDate	PM10	O3	NO2	CO	SO2	TEMP	RAIN	WIND	WIND_DIR	HUMIDITY	ATM_PRESS	CLOUD
0	2019-07-01	29.0	0.054	0.021	0.5	0.003	24.03	0.00	2.30	249	63.2	995.1	5.70
1	2019-07-02	26.0	0.053	0.020	0.5	0.003	24.29	0.00	2.26	265	63.2	998.6	3.83
2	2019-07-03	30.0	0.042	0.023	0.4	0.003	24.18	0.00	1.79	280	65.3	998.3	6.29
3	2019-07-04	28.0	0.034	0.026	0.4	0.003	25.35	0.00	2.04	263	58.6	996.6	2.54
4	2019-07-05	29.0	0.045	0.035	0.5	0.003	27.30	0.00	1.45	175	45.5	993.5	3.92
...	...	...	...	...	...	...	...	...	...	...	...	...	...
361	2020-06-26	19.0	0.039	0.016	0.4	0.003	21.66	0.41	3.12	228	84.0	996.0	8.73
362	2020-06-27	22.0	0.044	0.017	0.4	0.004	23.94	0.00	1.93	217	69.8	995.8	6.21
363	2020-06-28	27.0	0.044	0.009	0.4	0.003	25.03	0.00	2.35	283	71.3	994.7	2.63
364	2020-06-29	36.0	0.026	0.028	0.6	0.003	24.06	1.26	2.48	103	75.5	992.9	7.58
365	2020-06-30	6.0	0.039	0.009	0.3	0.002	20.60	5.60	4.00	50	92.0	983.8	10.00

Normalizer 적용을 통한 Heatmap 출력

```
# Normalizer 적용
from sklearn.preprocessing import Normalizer
```

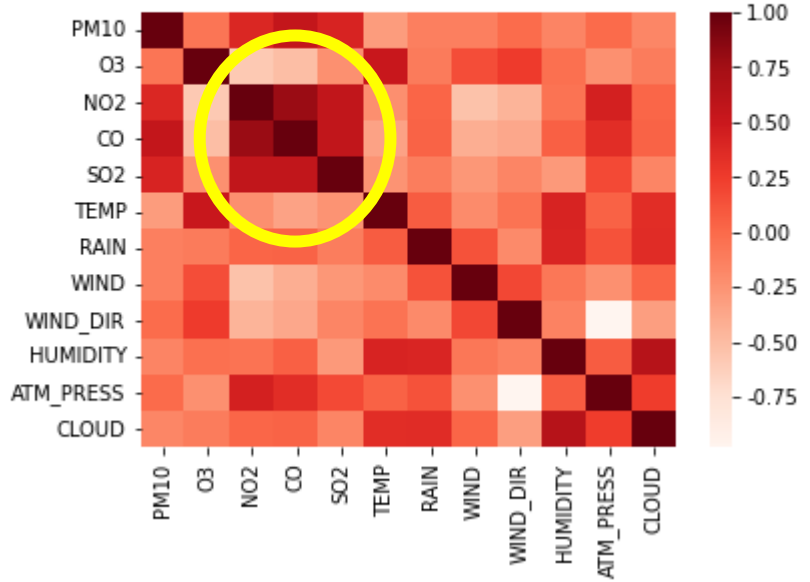
```
# Scale 변환 : Normalizer scaler (평균, 표준편차 적용)
df_scale_normal = Normalizer()
df_scale_normal = df_scale_normal.fit_transform(df_raw_dummy)
df_scale_normal
```

```
# Scale 변환 결과값의 전체 상관관계 분석
df_scale_normal.corr().round(3)
```

	PM10	O3	NO2	CO	SO2	TEMP	RAIN	WIND	WIND_DIR	HUMIDITY	ATM_PRESS	CLOUD
PM10	1.000	-0.054	0.400	0.557	0.423	-0.299	-0.118	-0.116	-0.005	-0.149	0.006	-0.165
O3	-0.054	1.000	-0.578	-0.503	-0.228	0.528	-0.096	0.159	0.258	-0.023	-0.228	-0.104
NO2	0.400	-0.578	1.000	0.796	0.573	-0.222	0.040	-0.532	-0.444	-0.048	0.447	0.032
CO	0.557	-0.503	0.796	1.000	0.570	-0.336	0.043	-0.406	-0.369	0.060	0.351	0.047
SO2	0.423	-0.228	0.573	0.570	1.000	-0.240	-0.110	-0.267	-0.157	-0.278	0.183	-0.160
TEMP	-0.299	0.528	-0.222	-0.336	-0.240	1.000	0.083	-0.198	-0.048	0.423	0.052	0.351
RAIN	-0.118	-0.096	0.040	0.043	-0.110	0.083	1.000	0.139	-0.188	0.405	0.139	0.362
WIND	-0.116	0.159	-0.532	-0.406	-0.267	-0.198	0.139	1.000	0.189	-0.074	-0.229	0.037
WIND_DIR	-0.005	0.258	-0.444	-0.369	-0.157	-0.048	-0.188	0.189	1.000	-0.139	-0.981	-0.312
HUMIDITY	-0.149	-0.023	-0.048	0.060	-0.278	0.423	0.405	-0.074	-0.139	1.000	0.082	0.643
ATM_PRESS	0.006	-0.228	0.447	0.351	0.183	0.052	0.139	-0.229	-0.981	0.082	1.000	0.251
CLOUD	-0.165	-0.104	0.032	0.047	-0.160	0.351	0.362	0.037	-0.312	0.643	0.251	1.000

```
#  Scale 변환 결과값의 Heatmap 출력
sns.heatmap(df_scale_normal.corr(), cmap="Reds")
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7fd50d152e50>
```



Scale 변환 결과값의 Heatmap 출력 결과, NO2, CO, SO2 와 PM10이 연관이 있다는걸 알 수 있다

다중선형회귀분석

```
#상관관계 분석
df_raw.corr().round(3)
```

	PM10	O3	NO2	CO	SO2	TEMP	RAIN	WIND	WIND_DIR	HUMIDITY	ATM_PRESS	CLOUD
PM10	1.000	-0.052	0.396	0.561	0.429	-0.310	-0.121	-0.100	0.020	-0.150	0.253	-0.172
O3	-0.052	1.000	-0.592	-0.513	-0.234	0.516	-0.104	0.165	0.269	-0.038	-0.534	-0.119
NO2	0.396	-0.592	1.000	0.791	0.563	-0.237	0.029	-0.537	-0.408	-0.066	0.420	0.017
CO	0.561	-0.513	0.791	1.000	0.567	-0.362	0.030	-0.403	-0.319	0.044	0.401	0.026
SO2	0.429	-0.234	0.563	0.567	1.000	-0.274	-0.129	-0.253	-0.093	-0.302	0.334	-0.191
TEMP	-0.310	0.516	-0.237	-0.362	-0.274	1.000	0.077	-0.216	-0.050	0.404	-0.792	0.342
RAIN	-0.121	-0.104	0.029	0.030	-0.129	0.077	1.000	0.126	-0.183	0.397	-0.236	0.358
WIND	-0.100	0.165	-0.537	-0.403	-0.253	-0.216	0.126	1.000	0.235	-0.084	-0.054	0.017
WIND_DIR	0.020	0.269	-0.408	-0.319	-0.093	-0.050	-0.183	0.235	1.000	-0.099	0.068	-0.297
HUMIDITY	-0.150	-0.038	-0.066	0.044	-0.302	0.404	0.397	-0.084	-0.099	1.000	-0.510	0.628
ATM_PRESS	0.253	-0.534	0.420	0.401	0.334	-0.792	-0.236	-0.054	0.068	-0.510	1.000	-0.430
CLOUD	-0.172	-0.119	0.017	0.026	-0.191	0.342	0.358	0.017	-0.297	0.628	-0.430	1.000

다중선형회귀분석을 위하여, 상관관계 분석 DATA SETTING

```
#train/test data 분리 (test_size=0.3)
df_train, df_test = train_test_split(df_raw, test_size=0.3, random_state=1234)
print("train data size : {}".format(df_train.shape))
print("test data size : {}".format(df_test.shape))
```

```
train data size : (255, 13)
test data size : (110, 13)
```

7 : 3 비율로 X, Y를 train, test로 분리

다중선형회귀분석 모델

```
#회귀 모델 생성
rfe_reg_model = smf.ols(formula = "PM10 ~ O3 + NO2 + CO + SO2 + WIND", data = df_train)
#적합
rfe_reg_result = rfe_reg_model.fit()
print(rfe_reg_result.summary())
```

OLS Regression Results

Dep. Variable:	PM10	R-squared:	0.426
Model:	OLS	Adj. R-squared:	0.415
Method:	Least Squares	F-statistic:	37.02
Date:	Mon, 15 Aug 2022	Prob (F-statistic):	2.66e-28
Time:	14:46:13	Log-Likelihood:	-994.84
No. Observations:	255	AIC:	2002.
Df Residuals:	249	BIC:	2023.
Df Model:	5		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
Intercept	-43.9160	6.793	-6.465	0.000	-57.294	-30.538
O3	529.0278	80.308	6.587	0.000	370.858	687.197
NO2	245.8473	153.239	1.604	0.110	-55.962	547.657
CO	72.0276	9.544	7.547	0.000	53.230	90.825
SO2	3166.2315	1528.441	2.072	0.039	155.911	6176.552
WIND	5.1414	1.289	3.990	0.000	2.603	7.680

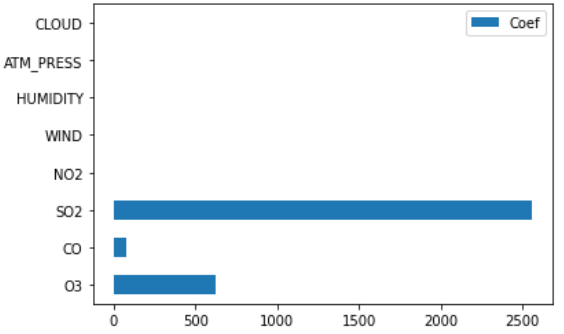
Omnibus:	70.262	Durbin-Watson:	1.997
Prob(Omnibus):	0.000	Jarque-Bera (JB):	241.635
Skew:	1.132	Prob(JB):	3.39e-53
Kurtosis:	7.197	Cond. No.	5.18e+03

Warnings:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[2] The condition number is large, 5.18e+03. This might indicate that there are
strong multicollinearity or other numerical problems.

1) No.Observations (분석 자료 수) : 255
2) Df Residuals (잔차 자유도) : 249
3) Df Model (모델 자유도) : 5
4) 분산분석결과 : p값이 '2.66e-28'으로 유의수준 0.05보다 작으므로 회귀모델로서 적합
5) 설명력 : '0.415' 으로 모델을 통하여 41.5%를 설명할 수 있음

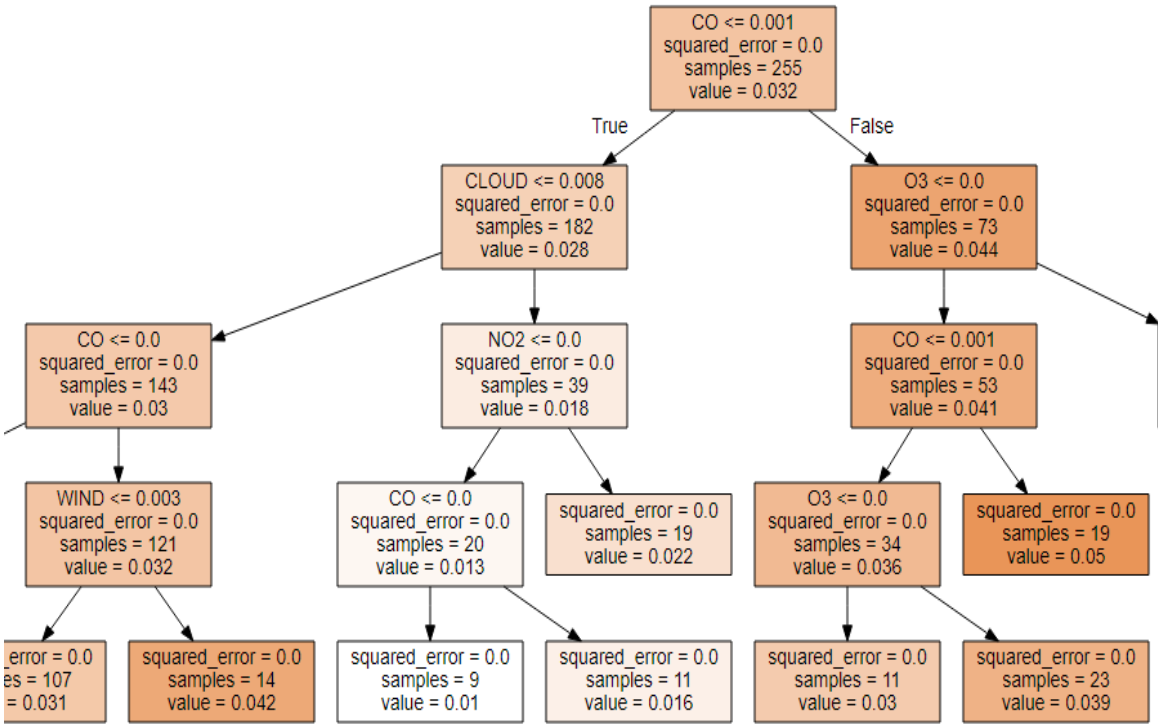
```
# 설명변수 중요도
df_reg_coef = pd.DataFrame({"Coef" : reg_result.params.values[1:]},
                           index = ['O3','CO','SO2','NO2','WIND','HUMIDITY','ATM_PRESS', 'CLOUD'])
df_reg_coef.plot.barh(y="Coef")
```

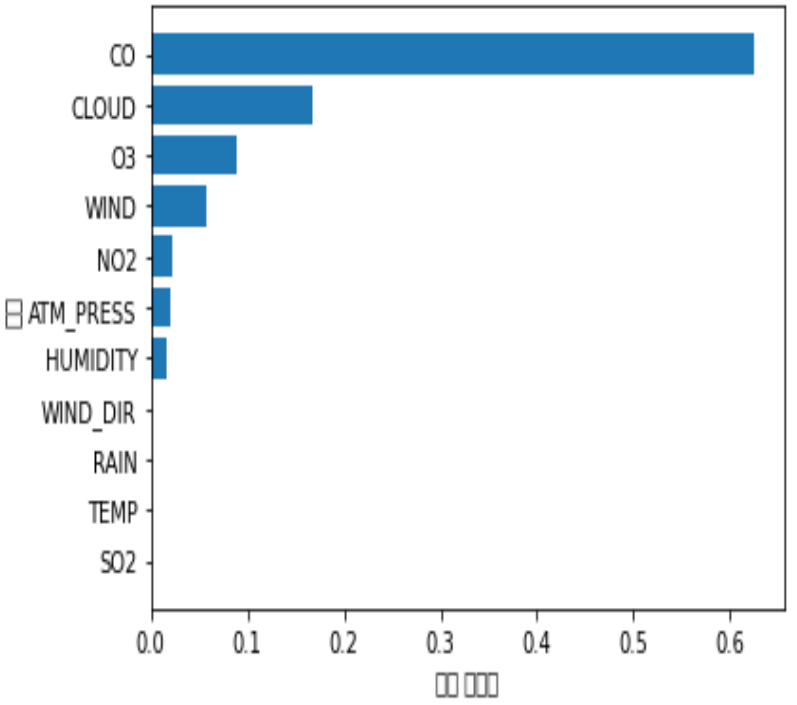
```
<matplotlib.axes._subplots.AxesSubplot at 0x7fd50b1d4790>
```



Coef 설명변수 중요도를 통하여 SO2와 PM10이 강한 연관성이 있음을 파악할 수 있음

Decision Tree

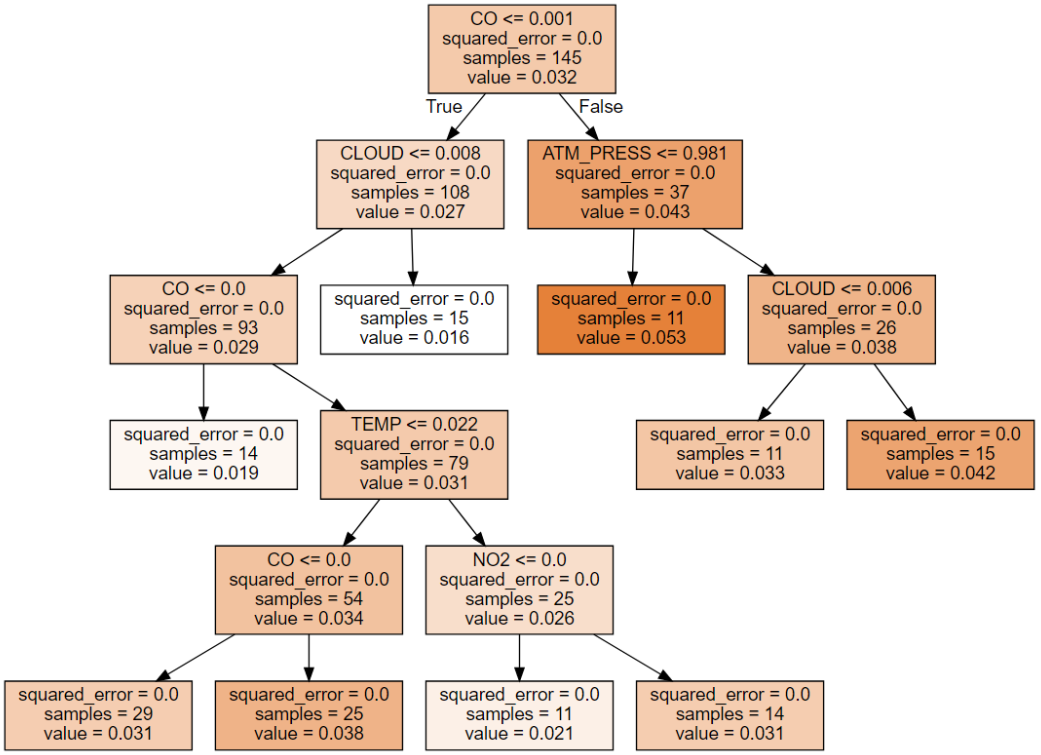


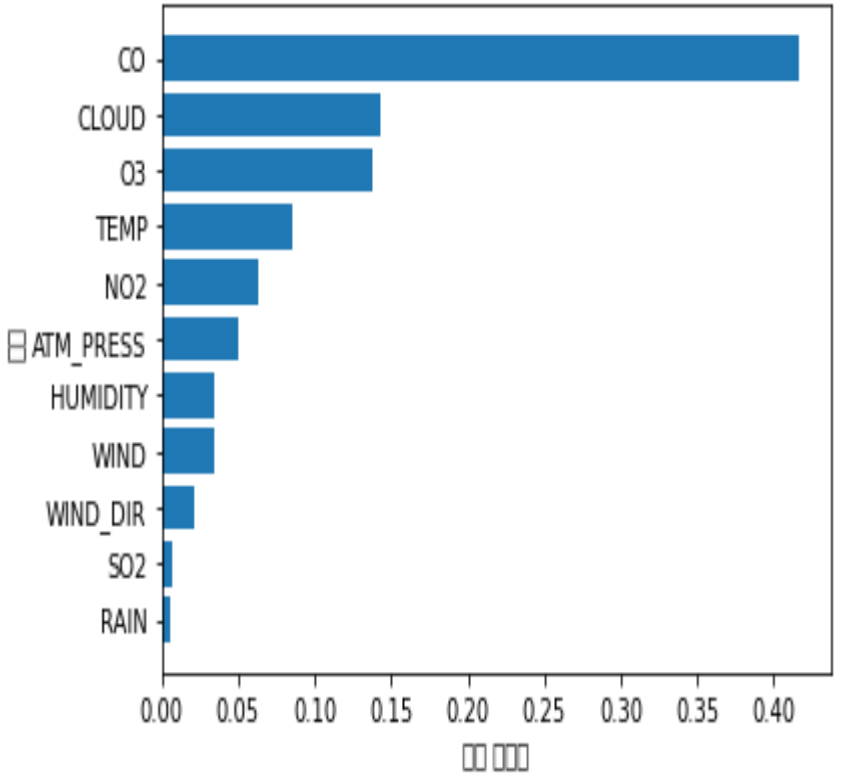



```
# 최종 모델 선정
tree_final = DecisionTreeRegressor(min_samples_leaf = 8, min_samples_split = 20, max_depth = 4, random_state = 1234)
tree_final.fit(df_train_x, df_train_y)
```

중요도에서는 CO, CLOUD, O3, WIND, NO2, ATM_PRESS, HUMIDITY 순으로 높았음

Random Forest






```
rf_final= RandomForestRegressor(random_state=1234,n_estimators=140, min_samples_leaf=10, max_depth = 5)
rf_final.fit(df_train_x, df_train_y)

print('train data의 결정계수:', rf_final.score(df_train_x,df_train_y))
print('test data의 결정계수:', rf_final.score(df_test_x,df_test_y))
```

```
train data의 결정계수: 0.5967738424550431
test data의 결정계수: 0.39682261146948417
```

중요도에서는 CO, CLOUD, O3, TEMP, NO2, ATM_PRESS, HUMIDITY, WIND 등 순으로 높았음

결론 및 대안제시

BEFORE

After

데이터 기반 정책

Normalizer 적용을 통한 Heatmap 출력, 다중선형회귀분석, Decision Tree, Random Forest를 통하여, NO2, CO, SO2 배출 감축을 통해 미세먼지를 줄일 수 있을 것이다