기말과제

2019783l4_김상화

인공지능을 위한 데이터 사이언스 입문

기말과제 문제 및 관심사항

기말과제 문제:

14주차 활동과제(12)에서 선택된 주제를 가지고 각자 자신의 관심분야에 대한 부분을 최종적으로 분석 및 시각화한 결과를 코드 및 PPT로 정리해서 제출해 주세요.

관심사항:

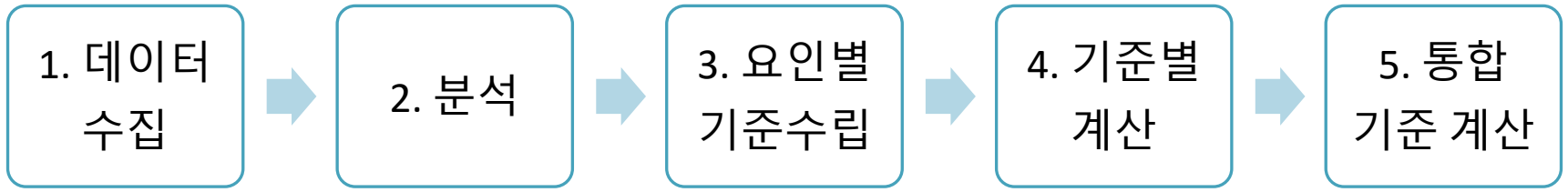
실내 체육 시설 이용이 어려운 요즘,

환경적 요인(미세먼지, 초미세먼지, 기온, 강수량)을 고려했을 때,

1년(2020년 기준) 중 산책하기 좋은 날짜는 몇일이나 되는가?

작업순서

1. 데이터를 수집한다.
2. 각 환경적 요인을 분석한다.
3. 환경적 요인별 산책 가능 기준을 정한다.
4. 각 기준별 산책 가능 일수를 시각화 및 계산한다.
5. 모든 기준을 고려하여 산책 가능 일수를 시각화 및 계산한다.



1. 데이터를 수집한다.

[미세먼지 & 초미세먼지 자료] – <광진구 기준>

최종확정 측정자료 조회(1~6월)

http://www.airkorea.or.kr/web/pastSearch?pMENU_NO=123

우리동네 대기 정보(7~11월)

http://www.airkorea.or.kr/web/realSearch?pMENU_NO=97

[기온 & 강수량 자료] - <서울 기준>

일별 기온

https://www.weather.go.kr/plus/land/current/past_table.jsp?stn=108&yy=2020&obs=07&x=33&y=15

일별 강수량

https://www.weather.go.kr/plus/land/current/past_table.jsp?stn=108&yy=2020&obs=21&x=30&y=12

특이사항:

1) 작업일 기준 전일까지 정보가 확정이므로 2020.01.01 ~ 2020.11.29일까지 자료로 수집
2) 미세먼지 초미세먼지는 한번에 수집이 어려워 나눠 받은 후 통합함
3) 기온, 강수량 정보는 2X2 행렬 형태라 일별 자료로 수정함

2. 각 환경적 요인을 분석한다.

- 시계열 자료이므로 '날짜'를 인덱스이자 date 형식으로 지정
- 오존, 이산화질소, 일산화탄소, 아황산가스 기준도 있지만
  날짜, PM10, PM2.5, 기온, 강수량만으로 분석 진행
- PM10이 미세먼지, PM2.5가 초미세먼지

1. 데이터 로드 ¶

In [2]:
```
data = pd.read_csv('data.csv', index_col=['날짜'], parse_dates=['날짜']
                   , usecols=['날짜','PM10','PM2.5','기온','강수량'])
data.head()
```

Out [2]:

날짜	PM10	PM2.5	기온	강수량
2020-01-01	33	19	-2.2	0.1
2020-01-02	61	34	1.0	NaN
2020-01-03	63	35	-0.1	NaN
2020-01-04	54	30	1.2	NaN
2020-01-05	47	27	1.3	NaN

2. 각 환경적 요인을 분석한다.

- 총 건수는 334건이며, 2020-01-01~2020-11-29까지 존재
- '강수량'에서 비가 오지 않은 날이 0 혹은 null로 표기
- 모두 0값을 가지도록 결측치 처리
- 모두 수치형으로 describe로 기술통계량 확인 가능

2. EDA & 시각화

In [3]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
DatetimeIndex: 334 entries, 2020-01-01 to 2020-11-29
Data columns (total 4 columns):
 #   Column  Non-Null Count  Dtype
---  ------  --------------  -----
 0   PM10    334 non-null    int64
 1   PM2.5   334 non-null    int64
 2   기온      334 non-null    float64
 3   강수량     142 non-null    float64
dtypes: float64(2), int64(2)
memory usage: 13.0 KB
```

In [4]:
```
# 강수량에 NULL 데이터 0 처리
data['강수량'].fillna(0.0, inplace=True)
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
DatetimeIndex: 334 entries, 2020-01-01 to 2020-11-29
Data columns (total 4 columns):
 #   Column  Non-Null Count  Dtype
---  ------  --------------  -----
 0   PM10    334 non-null    int64
 1   PM2.5   334 non-null    int64
 2   기온      334 non-null    float64
 3   강수량     334 non-null    float64
dtypes: float64(2), int64(2)
memory usage: 13.0 KB
```

In [5]:
```
data.describe()
```

Out[5]:

	PM10	PM2.5	기온	강수량
count	334.00000	334.000000	334.000000	334.000000
mean	33.61976	18.940120	14.576647	4.929641
std	17.75126	11.605218	9.000914	15.227135
min	4.00000	1.000000	-8.300000	0.000000
25%	21.00000	10.000000	7.000000	0.000000
50%	32.00000	17.000000	14.500000	0.000000
75%	44.00000	26.000000	22.950000	0.500000
max	124.00000	66.000000	30.200000	103.100000

3. 환경적 요인별 산책 가능 기준을 정한다.

요인	기준	비고
미세먼지	50㎍/㎥ 이하	국내 기준
초미세먼지	15㎍/㎥ 이하	
기온	5~27도 사이	개인 기준
강수량	비가 오지 않아야 함	

국내기준
https://www.airkorea.or.kr/web/contents/contentView/?pMENU_NO=132&cntnts_no=6

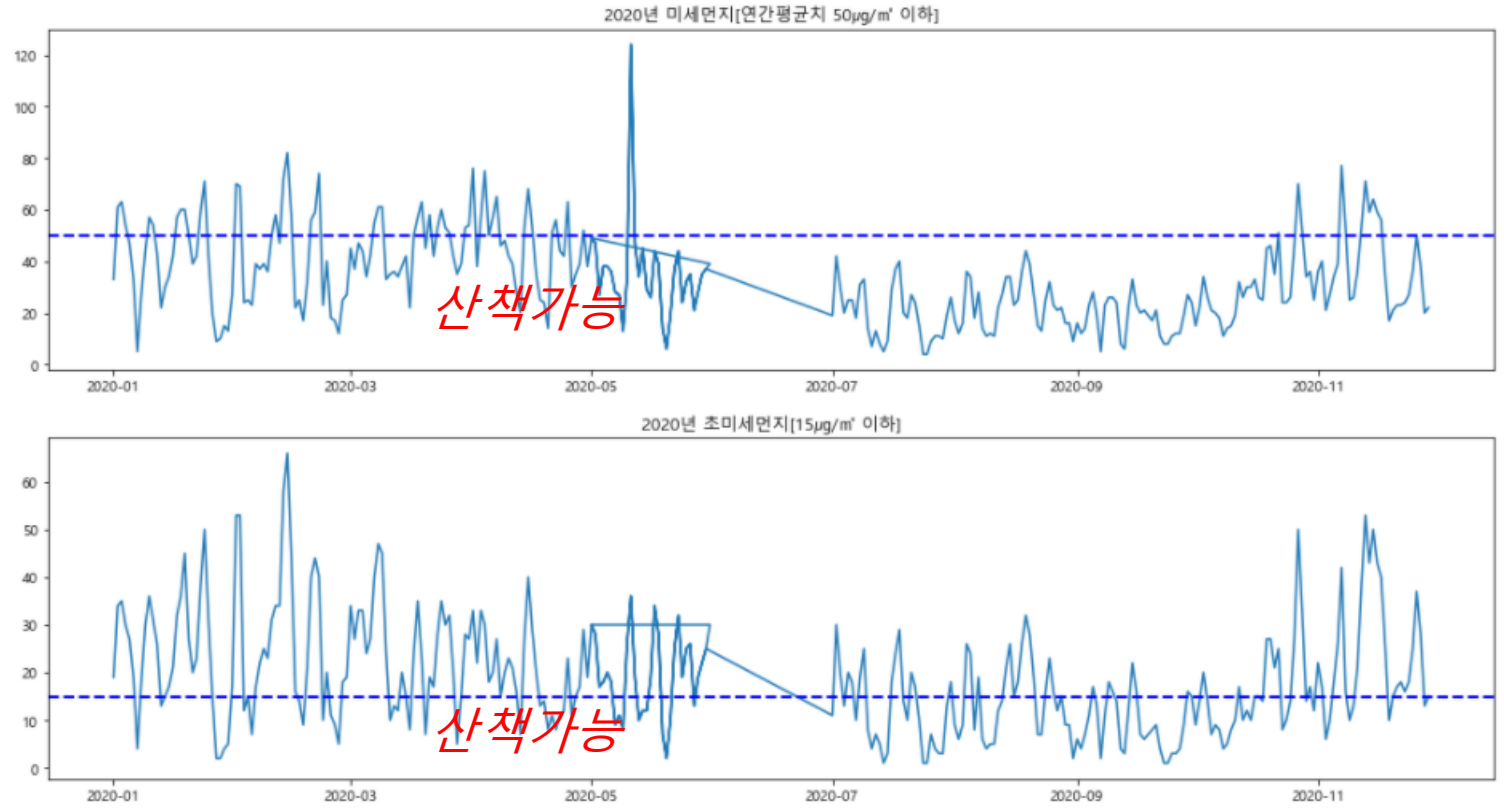
미세먼지 (PM_{10})	연간 평균치	50㎍/㎥ 이하
	24시간 평균치	100㎍/㎥ 이하
초미세먼지 ($PM_{2.5}$)	연간평균치	15㎍/㎥ 이하
	24시간 평균치	35㎍/㎥ 이하

4. 각 기준별 산책 가능 일수를 시각화 및 계산한다.

각 기준별로 산책 가능 날짜 탐색

In [6]:

```
plt.figure(figsize=(18,20))
plt.rc('font', family='Malgun Gothic')
plt.subplot(411)
plt.plot(data['PM10'], label='미세먼지')
plt.axhline(50, color='b', linestyle='dashed', linewidth=2)
plt.title('2020년 미세먼지[연간평균치 50㎍/㎥ 이하]') # 국내 기준

plt.subplot(412)
plt.plot(data['PM2.5'], label='초미세먼지')
plt.axhline(15, color='b', linestyle='dashed', linewidth=2)
plt.title('2020년 초미세먼지[15㎍/㎥ 이하]') # 국내 기준

plt.subplot(413)
plt.plot(data['기온'], label='기온')
plt.axhline(27, color='b', linestyle='dashed', linewidth=2)
plt.axhline(5, color='b', linestyle='dashed', linewidth=2)
plt.title('2020년 기온[개인적 기준: 5~27도]') # 개인적 기준

plt.subplot(414)
plt.plot(data['강수량'], label='강수량')
plt.axhline(0, color='b', linestyle='dashed', linewidth=2)
plt.title('2020년 강수량[개인적 기준: 비 오면 안됨]') # 개인적 기준

plt.legend()
plt.show()
```

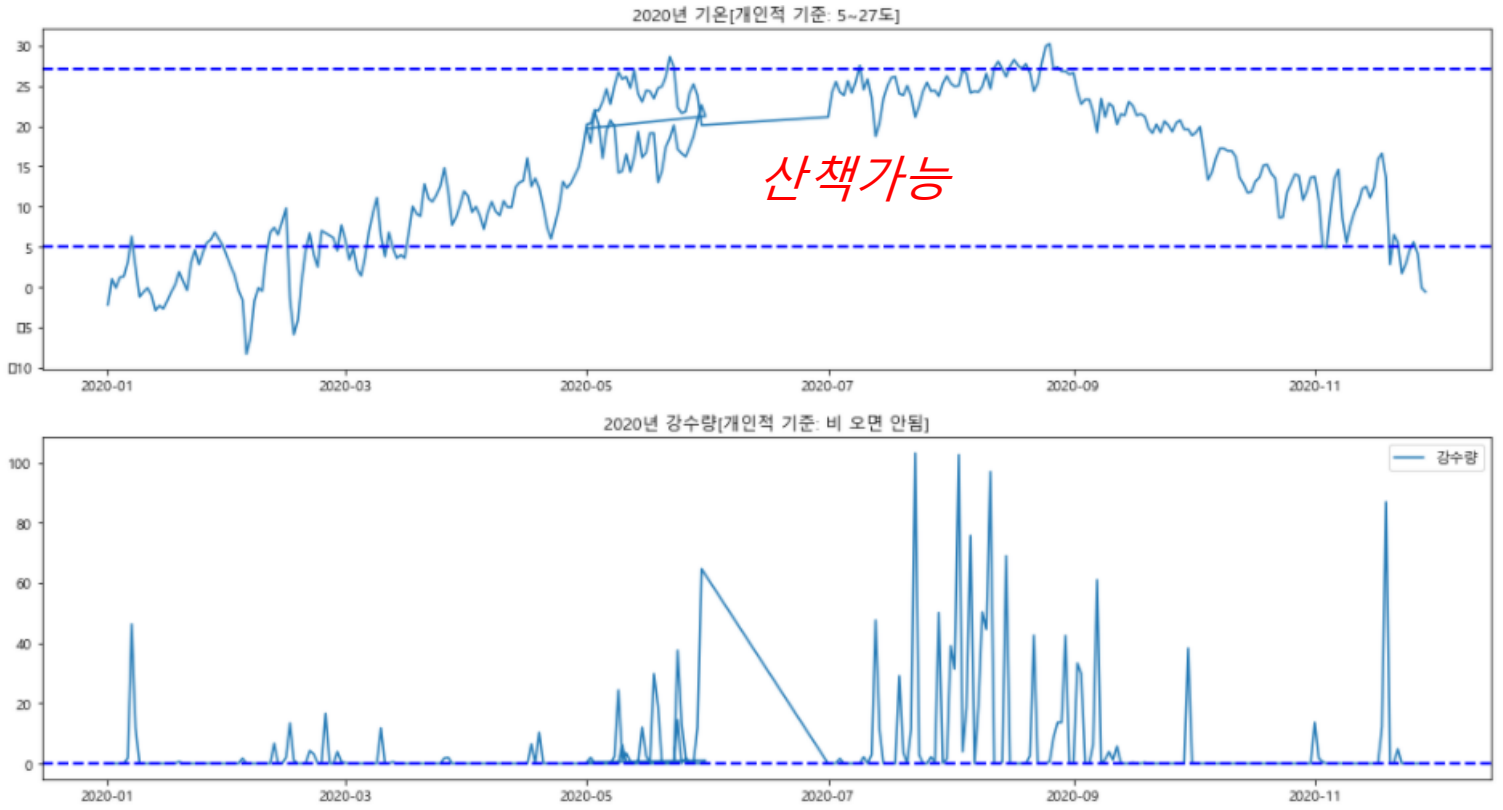


4. 각 기준별 산책 가능 일수를 시각화 및 계산한다.

각 기준별로 산책 가능 날짜 탐색

In [6]:

```
plt.figure(figsize=(18,20))
plt.rc('font', family='Malgun Gothic')
plt.subplot(411)
plt.plot(data['PM10'], label='미세먼지')
plt.axhline(50, color='b', linestyle='dashed', linewidth=2)
plt.title('2020년 미세먼지[연간평균치 50㎍/㎥ 이하]') # 국내 기준

plt.subplot(412)
plt.plot(data['PM2.5'], label='초미세먼지')
plt.axhline(15, color='b', linestyle='dashed', linewidth=2)
plt.title('2020년 초미세먼지[15㎍/㎥ 이하]') # 국내 기준

plt.subplot(413)
plt.plot(data['기온'], label='기온')
plt.axhline(27, color='b', linestyle='dashed', linewidth=2)
plt.axhline(5, color='b', linestyle='dashed', linewidth=2)
plt.title('2020년 기온[개인적 기준: 5~27도]') # 개인적 기준

plt.subplot(414)
plt.plot(data['강수량'], label='강수량')
plt.axhline(0, color='b', linestyle='dashed', linewidth=2)
plt.title('2020년 강수량[개인적 기준: 비 오면 안됨]') # 개인적 기준

plt.legend()
plt.show()
```



산책가능

4. 각 기준별 산책 가능 일수를 시각화 및 계산한다.

In [7]:

```
# 기준별로 산책이 가능했던 날의 비율 계산
산책_가능_미세먼지 = data[data['PM10'] <= 50]['PM10'].count()
산책_가능_초미세먼지 = data[data['PM2.5'] <= 15]['PM2.5'].count()
산책_가능_기온 = data[((data['기온'] >= 5) & (data['기온'] <= 27))]['기온'].count()
산책_가능_강수량 = data[data['강수량'] == 0]['강수량'].count()

print('기준별 산책 가능 일수 및 비율:₩n[미세먼지]₩t {}일₩t {}%₩n[초미세먼지]₩t {}일₩t {}%₩n[기온]₩t₩t {}일₩t {}%₩n[강수량]₩t {}일₩t {}%
                          산책_가능_미세먼지  , np.round(산책_가능_미세먼지/data.shape[0],2)*100
                        , 산책_가능_초미세먼지, np.round(산책_가능_초미세먼지/data.shape[0],2)*100
                        , 산책_가능_기온      , np.round(산책_가능_기온/data.shape[0],2)*100
                        , 산책_가능_강수량    , np.round(산책_가능_강수량/data.shape[0],2)*100))
```

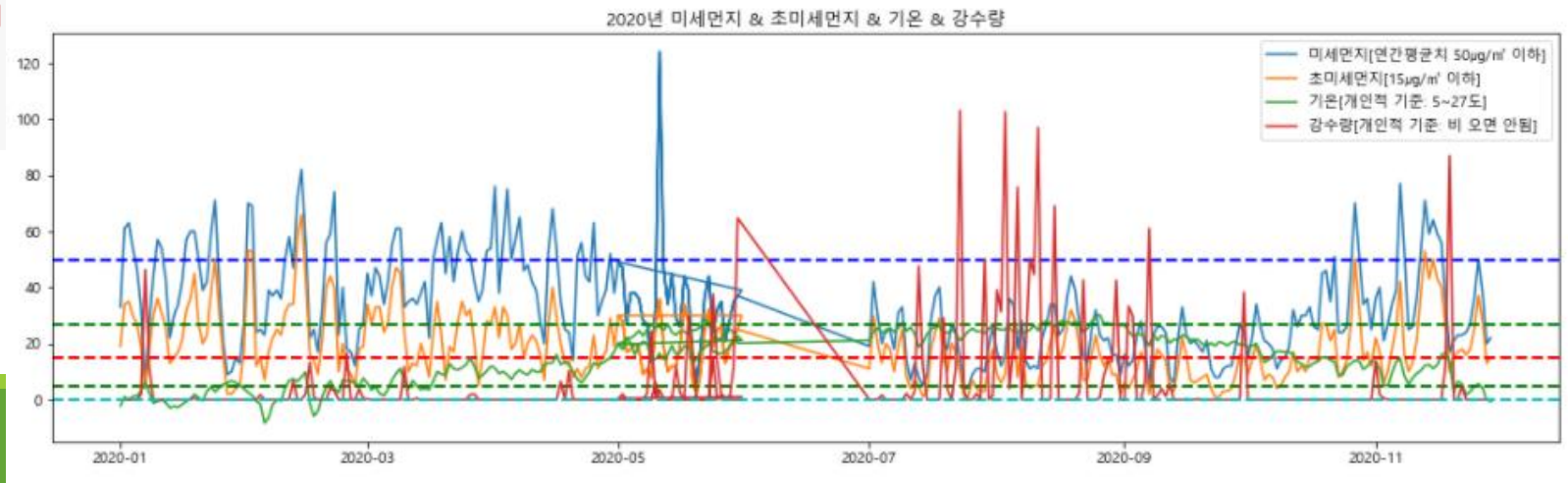
```
기준별 산책 가능 일수 및 비율:
[미세먼지]      278일   83.0%
[초미세먼지]    146일   44.0%
[기온]          257일   77.0%
[강수량]        232일   69.0%
```

5. 모든 기준을 고려하여 산책 가능 일수를 시각화 및 계산한다.

모든 기준을 통합해서 산책 가능 날짜 도출

In [8]:
```
# 기준별 범위가 다르기 때문에 스케일 작업이 필요함을 확인할 수 있음

plt.figure(figsize=(18,5))
plt.rc('font', family='Malgun Gothic')

plt.title('2020년 미세먼지 & 초미세먼지 & 기온 & 강수량')

plt.plot(data['PM10'], label='미세먼지[연간평균치 50㎍/㎥ 이하]')
plt.axhline(50, color='b', linestyle='dashed', linewidth=2)

plt.plot(data['PM2.5'], label='초미세먼지[15㎍/㎥ 이하]')
plt.axhline(15, color='r', linestyle='dashed', linewidth=2)

plt.plot(data['기온'], label='기온[개인적 기준: 5~27도]')
plt.axhline(27, color='g', linestyle='dashed', linewidth=2)
plt.axhline(5, color='g', linestyle='dashed', linewidth=2)

plt.plot(data['강수량'], label='강수량[개인적 기준: 비 오면 안됨]')
plt.axhline(0, color='c', linestyle='dashed', linewidth=2)

plt.legend()
plt.show()
```

추가작업:
기준별 범위가 다르기 때문에 통합 시각화를 위한 스케일링 작업이 필요함



5. 모든 기준을 고려하여 산책 가능 일수를 시각화 및 계산한다.

sklearn 의 MinMaxScaler를 사용하여 환경 요인 데이터가 0~1의 범위를 가짐을 확인

In [9]:
```
# 0~1 범위로 데이터 및 기준 스케일링

from sklearn.preprocessing import MinMaxScaler

scaled = data.copy()
scaler = MinMaxScaler()
scaled[['PM10','PM2.5','기온','강수량']] = scaler.fit_transform(scaled[['PM10','PM2.5','기온','강수량']])
scaled
```

Out [9]:

날짜	PM10	PM2.5	기온	강수량
2020-01-01	0.241667	0.276923	0.158442	0.00097
2020-01-02	0.475000	0.507692	0.241558	0.00000
2020-01-03	0.491667	0.523077	0.212987	0.00000
2020-01-04	0.416667	0.446154	0.246753	0.00000
2020-01-05	0.358333	0.400000	0.249351	0.00000
...	...	...	...	...
2020-11-25	0.266667	0.369231	0.335065	0.00000
2020-11-26	0.383333	0.553846	0.361039	0.00000
2020-11-27	0.283333	0.415385	0.322078	0.00000
2020-11-28	0.133333	0.184615	0.212987	0.00000
2020-11-29	0.150000	0.215385	0.200000	0.00000

334 rows × 4 columns

In [10]:
```
scaled.describe() # 0~1 범위로 조정됨
```

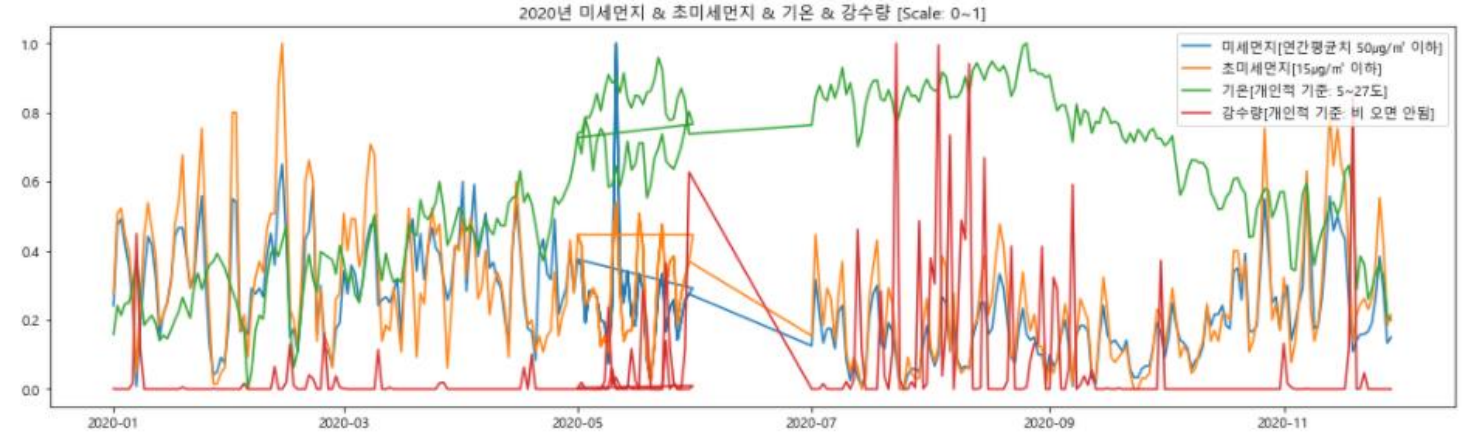
Out [10]:

	PM10	PM2.5	기온	강수량
count	334.000000	334.000000	334.000000	334.000000
mean	0.246831	0.276002	0.594199	0.047814
std	0.147927	0.178542	0.233790	0.147693
min	0.000000	0.000000	0.000000	0.000000
25%	0.141667	0.138462	0.397403	0.000000
50%	0.233333	0.246154	0.592208	0.000000
75%	0.333333	0.384615	0.811688	0.004850
max	1.000000	1.000000	1.000000	1.000000

5. 모든 기준을 고려하여 산책 가능 일수를 시각화 및 계산한다.

In [11]:

```
# 0~1 범위에서 모든 기준 비교

plt.figure(figsize=(18,5))
plt.rc('font', family='Malgun Gothic')

plt.title('2020년 미세먼지 & 초미세먼지 & 기온 & 강수량 [Scale: 0~1]')

plt.plot(scaled['PM10'], label='미세먼지[연간평균치 50㎍/㎥ 이하]')

plt.plot(scaled['PM2.5'], label='초미세먼지[15㎍/㎥ 이하]')

plt.plot(scaled['기온'], label='기온[개인적 기준: 5~27도]')

plt.plot(scaled['강수량'], label='강수량[개인적 기준: 비 오면 안됨]')

plt.legend()
plt.show()
```

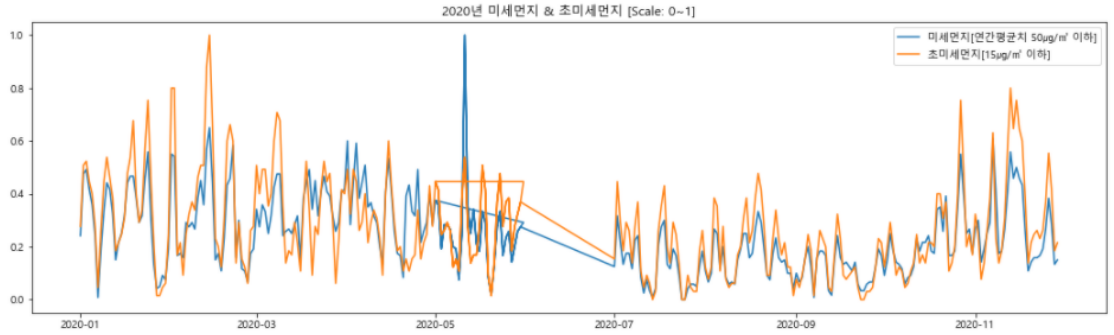


후속작업:
한번에 4가지 요인을 그래프로 비교하기 어려움
요인별로 따로 시각화 및 필요

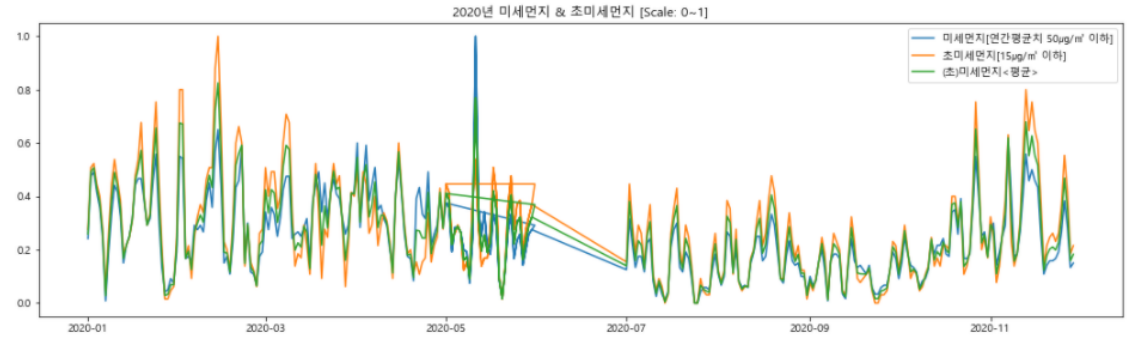
5. 모든 기준을 고려하여 산책 가능 일수를 시각화 및 계산한다.

미세먼지와 초미세먼지가 비슷한 그래프 모양을 가짐으로 평균치를 (초)미세먼지로 생성하여 다른 요인(기온, 강수량)과 비교

In [12]:
```
# 미세먼지 & 초미세먼지

plt.figure(figsize=(18,5))
plt.rc('font', family='Malgun Gothic')

plt.title('2020년 미세먼지 & 초미세먼지 [Scale: 0~1]')

plt.plot(scaled['PM10'], label='미세먼지[연간평균치 50㎍/㎥ 이하]')
plt.plot(scaled['PM2.5'], label='초미세먼지[15㎍/㎥ 이하]')

plt.legend()
plt.show()
```



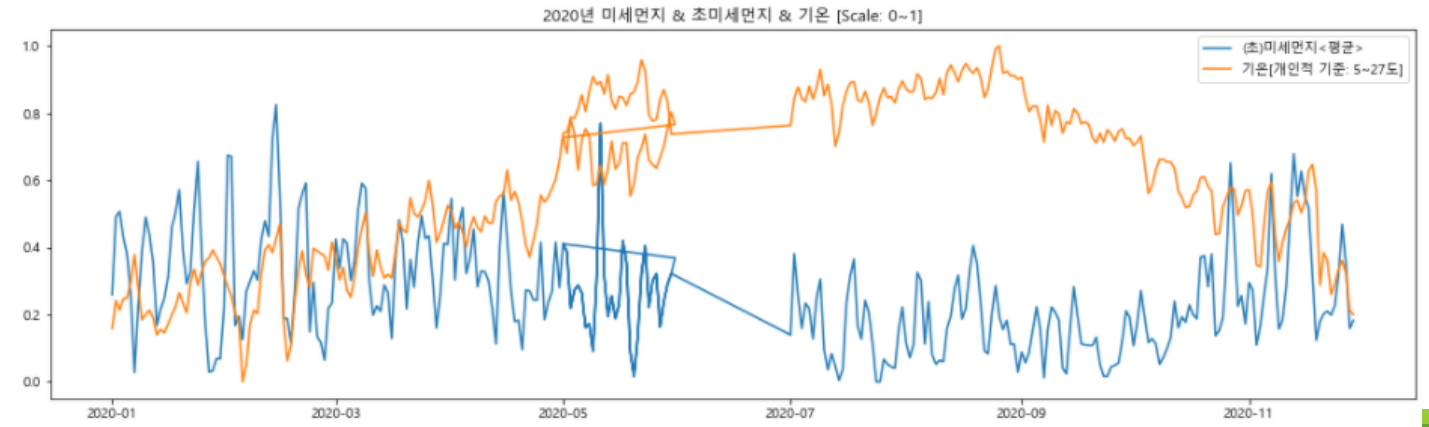
In [13]:
```
# 미세먼지 & 초미세먼지

plt.figure(figsize=(18,5))
plt.rc('font', family='Malgun Gothic')

plt.title('2020년 미세먼지 & 초미세먼지 [Scale: 0~1]')

plt.plot(scaled['PM10'], label='미세먼지[연간평균치 50㎍/㎥ 이하]')
plt.plot(scaled['PM2.5'], label='초미세먼지[15㎍/㎥ 이하]')
plt.plot(((scaled['PM10']+scaled['PM2.5'])/2), label='(초)미세먼지<평균>') # 추세가 비슷함으로 평균값으로 대체 비교

plt.legend()
plt.show()
```



5. 모든 기준을 고려하여 산책 가능 일수를 시각화 및 계산한다.

(초)미세먼지와 기온 사이에 뚜렷한 관계가 있기 보다는 시기적으로 여름에 미세먼지가 덜한 경향이 보이므로 계절적 영향으로 판단됨

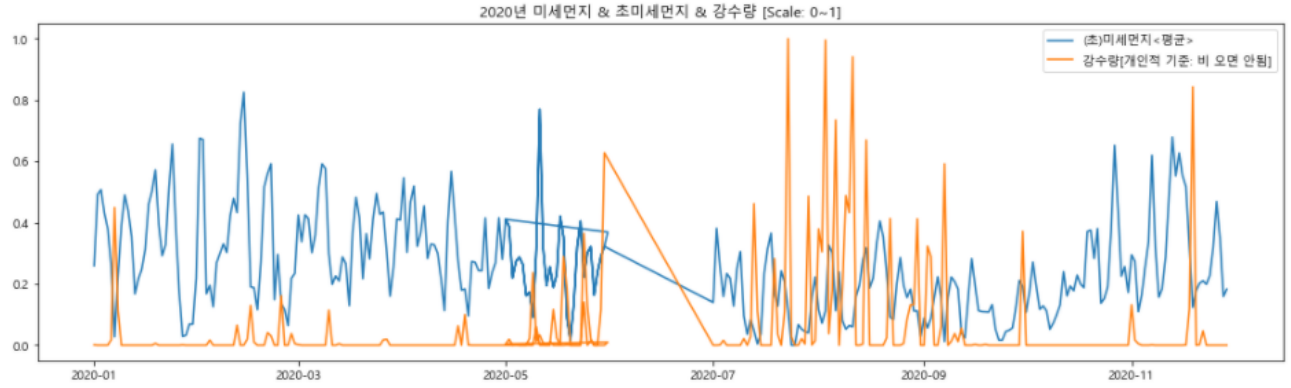
In [14]:
```
# (초)미세먼지 & 기온

plt.figure(figsize=(18,5))
plt.rc('font', family='Malgun Gothic')

plt.title('2020년 미세먼지 & 초미세먼지 & 기온 [Scale: 0~1]')

plt.plot(((scaled['PM10']+scaled['PM2.5'])/2), label='(초)미세먼지<평균>') # 추세가 비슷함으로 평균값으로 대체 비교
plt.plot(scaled['기온'], label='기온[개인적 기준: 5~27도]')

plt.legend()
plt.show()

# (초)미세먼지와 기온 사이에 뚜렷한 관계가 있기 보다는 시기적으로 여름에 미세먼지가 덜한 경향이 보이므로 계절적 영향으로 판단됨
```



5. 모든 기준을 고려하여 산책 가능 일수를 시각화 및 계산한다.

강수량이 있을 때는 확실히 미세먼지가 덜함을 확인 할 수 있으나, 비가오는 날은 산책을 하지 않음으로 최종 결론에선 강수량이 0이면서 미세먼지가 좋은 날만 채택됨

In [15]:
```
# (초)미세먼지 & 강수량

plt.figure(figsize=(18,5))
plt.rc('font', family='Malgun Gothic')

plt.title('2020년 미세먼지 & 초미세먼지 & 강수량 [Scale: 0~1]')

plt.plot(((scaled['PM10']+scaled['PM2.5'])/2), label='(초)미세먼지<평균>') # 추세가 비슷함으로 평균값으로 대체 비교

plt.plot(scaled['강수량'], label='강수량[개인적 기준: 비 오면 안됨]')

plt.legend()
plt.show()

# 강수량이 있을 때는 확실히 미세먼지가 덜함을 확인 할 수 있으나,
# 비가오는 날은 산책을 하지 않음으로 최종 결론에선 강수량이 0이면서 미세먼지가 좋은날만 채택됨
```



5. 모든 기준을 고려하여 산책 가능 일수를 시각화 및 계산한다.

결론: 모든 조건을 만족하는 산책 가능 일수가 23% 밖에 되지 않으므로 산책을 나갈 수 있는 환경 조건이 된다면 최대한 나가야 함

In [16]:
```
# 모든 기준을 만족하는 산책 가능 일수 및 비율

# 기준별 조건(condition)
cond_미세먼지 = data['PM10'] <= 50
cond_초미세먼지 = data['PM2.5'] <= 15
cond_기온 = ((data['기온'] >= 5) & (data['기온'] <= 27))
cond_강수량 = data['강수량'] == 0

day_count = data[cond_미세먼지 & cond_초미세먼지 & cond_기온 & cond_강수량].shape[0]

print('모든 조건을 만족하는 산책 가능한 일수 및 비율: {}일 ₩t{}%'.format(day_count,np.round(day_count/data.shape[0],2)*100))
```

```
모든 조건을 만족하는 산책 가능한 일수 및 비율: 78일	23.0%
```