

모듈 импорт

In [1]:

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
plt.rc('font', family='Malgun Gothic')
from tqdm.notebook import tqdm
import warnings
warnings.filterwarnings(action='ignore')
import pickle
```

설명 : make_season(감염병)

2010년부터 2019년까지 모든 감염병에 대한 데이터를 불러들여서
봄, 여름, 가을, 겨울로 구분해 빈도수를 합하여 발생빈도 비율을 통해 계절별 위험 요인을 구함

In [2]:

```
def make_season():

    # 빈도 가져워
    df = pd.read_csv('지역별_2010부터_감염병발생빈도.csv')

    # 월 추가
    df['월'] = df['날짜'].apply(lambda x : x[-2:])
    df['계절'] = df['월']
    for i in tqdm(range(len(df))): # 32640 : 계절별 요인, 2020년은 고려하지 않았고 2010년부터 2019년까지
        if df['월'][i] in ['03', '04', '05']:
            df['계절'][i] = '봄'
        elif df['월'][i] in ['06', '07', '08']:
            df['계절'][i] = '여름'
        elif df['월'][i] in ['09', '10', '11']:
            df['계절'][i] = '가을'
        else:
            df['계절'][i] = '겨울'

    season = df.iloc[range(32640)].groupby('계절').sum().reset_index()
    with open('season.pickle', 'wb') as f:
        pickle.dump(season, f, pickle.HIGHEST_PROTOCOL) # save as season.pickle
    return season
```

필요한 데이터들

In [3]:

```
df = pd.read_csv('지역별_2010부터_감염병발생빈도.csv')
age = pd.read_csv('연령_전체.csv', encoding = 'euc-kr', header = None)
data = pd.read_csv('지역별_2018부터_10만명당감염병발생비율.csv')
prevention = pd.read_excel('감염병 예방.xlsx')
with open('season.pickle', 'rb') as f:
    season = pickle.load(f)
```

설명 : make_season_dict(감염병)

make_season으로 만들어진 데이터를 통해서
특정 감염병에 대해
발생빈도 비율을 통해 계절별 위험 요인을 구함

tfidf 넣어서 계절별 요인 수 구하기

In [4]:

```
# 증가빈도tf-idf
# 계절 요인은 연령보다 세분화되어 있지 않아 수두같은 질병이 등장할 확률이 더 높으므로 최대 스케일이
# 계절 요인이 상대적으로 크게 작용할 경우 질병의 빈도 값을 조절하기 위해 사용(다른 요인이 적용이 안
# 분수에서 분모는 0이 되면 안 되기 때문에(특정 질병이 모든 요인에 없을 수 있다.) +1로 스무딩 처리
# 요인의 갯수가 적어 idf에 로그를 적용하면 음수가 나올 확률이 크므로 로그는 적용하지 않음
# double normalization k = 0.5

def make_season_dict(disease= '수두'):
    season_dict = {}
    cnt = 0
    tmp_for_max = []
    for i in range(len(season['계절'])):
        if season[disease][i] != 0:
            cnt += 1
            tmp_for_max.append(season[disease][i])

    for i in range(len(season['계절'])):
        a = season['계절'][i]
        season_dict[a] = (0.5+ (0.5*(season[disease][i])/max(tmp_for_max))) * (len(season['계절'])/
    return season_dict
```

In [5]:

```
make_season_dict('레지오넬라증')
```

Out [5]:

```
{'가을': 0.7946949602122017,
 '겨울': 0.7225464190981432,
 '봄': 0.6737400530503979,
 '여름': 0.8}
```

설명 : make_age_dict(감염병)

2010년부터 2020년 8월까지 특정 감염병에 대한 데이터를 불러들여서
나이대로 구분해 빈도수를 합하여 발생빈도 비율을 통해 나이별 위험 요인을 구함

In [6]:

```
age = pd.read_csv('연령_전체.csv', encoding = 'euc-kr', header = None)
age
```

Out[6]:

		0	1	2	3	4	5	6	7	8	9	...	55	56	57	58	59	60	61	62	63	64
0	계	0	0	0	0	0	0	0	0	0	0	...	0	3.94	1.27	0.00	0.30	0	0.05	0.08	2.45	0.07
1	0~9세	0	0	0	0	0	0	0	0	0	0	...	0	0.54	0.02	0.00	0.10	0	0.00	0.00	0.02	0.00
2	10~19세	0	0	0	0	0	0	0	0	0	0	...	0	2.16	0.06	0.00	0.10	0	0.00	0.01	0.07	0.00
3	20~29세	0	0	0	0	0	0	0	0	0	0	...	0	7.66	0.42	0.00	0.35	0	0.00	0.14	0.32	0.10
4	30~39세	0	0	0	0	0	0	0	0	0	0	...	0	7.01	1.10	0.00	0.33	0	0.02	0.11	0.26	0.14
5	40~49세	0	0	0	0	0	0	0	0	0	0	...	0	4.12	1.41	0.00	0.33	0	0.07	0.07	0.67	0.07
6	50~59세	0	0	0	0	0	0	0	0	0	0	...	0	3.59	2.46	0.01	0.46	0	0.09	0.15	3.76	0.10
7	60~69세	0	0	0	0	0	0	0	0	0	0	...	0	1.67	3.50	0.00	0.29	0	0.27	0.05	7.87	0.02
8	70세 이상	0	0	0	0	0	0	0	0	0	0	...	0	0.26	2.19	0.00	0.37	0	0.03	0.03	15.58	0.00

9 rows × 65 columns

설명 : make_age_dict(감염병)

2020년 7월(저번달, 특정시점)에 대해 지역별 감염병에 대한 데이터를 불러들여서
1급,2급,3급으로 나뉜 기준을 통해 위험도 계수를 3,2,1 을 곱해주고
10만명당 발생률 * 위험도 계수로, top5를 구함
** 발생하지 않은 감염병은 제외됨 **

tfidf 넣어서 연령별 요인 구하기

In [7]:

```

# 증가빈도tf-idf
# 연령 요인이 상대적으로 크게 작용할 경우 질병의 빈도 값을 조절하기 위해 사용(다른 요인이 적용이 안
# 분수에서 분모는 0이 되면 안 되기 때문에(특정 질병이 모든 요인에 없을 수 있다.) +1로 스무딩 처리
# 요인의 갯수가 적어 idf에 로그를 적용하면 음수가 나올 확률이 크고, 모든 경우의 수를 고려하기 위해
# double normalization k = 0.01

def make_age_dict(disease = '수두'):

    # 불러들이기
    age = pd.read_csv('연령_전체.csv', encoding = 'euc-kr', header = None)

    # 컬럼명 지정
    colum = ['나이', '에볼라바이러스병', '마버그열', '라싸열', '크리미안콩고출혈열', '남아메리카출혈열',
              '중증급성호흡기증후군(SARS)', '중동호흡기증후군(MERS)', '동물인플루엔자 인체감염증', '신종인플루엔자',
              '세균성이질', '장출혈성대장균감염증', 'A형간염', '백일해', '유행성이하선염', '풍진(2018년이전)',
              '페렴구균 감염증', '한센병', '성홍열', '반코마이신내성황색포도알균(VRSA) 감염증', '카바페넴내성효소(KPC)를 생성하는 균주 감염증',
              'C형간염', '말라리아', '레지오넬라증', '비브리오패혈증', '발진티푸스', '발진열', '프프가무시증', '유행성출혈열', '뎅기열', '큐열', '웨스트나일열', '라임병', '진드기매개뇌염', '유비저', '치쿤구니아열',
              '레지오넬라증', 'A형간염', '백일해', '유행성이하선염', '풍진(2018년이전)', '페렴구균 감염증', '한센병', '성홍열', '반코마이신내성황색포도알균(VRSA) 감염증', '카바페넴내성효소(KPC)를 생성하는 균주 감염증', 'C형간염', '말라리아', '레지오넬라증', '비브리오패혈증', '발진티푸스', '발진열', '프프가무시증', '유행성출혈열', '뎅기열', '큐열', '웨스트나일열', '라임병', '진드기매개뇌염', '유비저', '치쿤구니아열']

    # age의 컬럼을 colum으로
    age.columns = colum
    age_dict = {}
    cnt = 0
    tmp_for_max = []
    for i in range(1, len(age['나이'])):
        if (age[disease][i] != 0) or (age[disease][i] != 0.00):
            cnt += 1
            tmp_for_max.append(age[disease][i])

    for i in range(1, len(age['나이'])):
        a = age['나이'][i]
        age_dict[a] = 0.01 + (0.99*(age[disease][i]/max(tmp_for_max))) * ((len(age['나이'])-1)/(1+cnt))
    return age_dict

```

설명 : make_risk_series(location)

2020년 7월(저번달, 특정시점)에 대해 지역별 감염병에 대한 데이터를 불러들여서

1급,2급,3급으로 나뉜 기준을 통해 위험도 계수를 3,2,1 을 곱해주고

10만명당 발생률 * 위험도 계수로, top5를 구함

** 발생하지 않은 감염병은 제외됨 **

In [8]:

```
def make_risk_series(location = '서울강남구'):
    # 위험도 가져오자
    급1 = ['에볼라바이러스병 ', '마버그열', '라싸열', '크리미안콩고출혈열', '남아메리카출혈열', '리프트밸',
           '중증급성호흡기증후군(SARS)', '중증호흡기증후군(MERS)', '동물인플루엔자 인체감염증', '신종인

    급2 = [ '수두', '홍역', '콜레라', '장티푸스', '파라티푸스',
            '세균성이질', '장출혈성대장균감염증', 'A형간염', '백일해', '유행성이하선염', '풍진(2018년이전',
            '페렴구균 감염증', '한센병', '성홍열', '반코마이신내성황색포도알균(VRSA) 감염증', '카바페넴N

    급3 = [ '파상풍', 'B형간염', '일본뇌염',
            'C형간염', '말라리아', '레지오넬라증', '비브리오패혈증', '발진티푸스', '발진열', '쯔쯔가무시증',
            '황열', '댕기열', '큐열', '웨스트나일열', '라임병', '진드기매개뇌염', '유비저', '치쿤구니아열',

    ## input : 계절, 나이,
    col1 = 급1 + 급2 + 급3

    risk = {}
    for disease in col1:
        #감염병 명 : 위험도
        k=0
        if disease in 급1:
            k = 3
        elif disease in 급2:
            k = 2
        else:
            k=1

        # 2020년 7월 감염병 10만명당 발생률 가져오기 #
        data = pd.read_csv('지역별_2018부터_10만명당감염병발생비율.csv')
        data[disease] = data[disease].apply(lambda x : 0 if x == '-' else float(x))
        last = data[data['날짜'] == 202007]
        num = last[last['구분'] == '{} '.format(location)][disease]
        if num.values[0] != 0:
            risk[disease] = (num * k).values[0]
    risk_series = pd.Series(risk)

    return risk_series
```

설명 : final_output(나이, 계절, 위치)

나이와 계절과 위치를 입력하면 특정 지역에 대한 risk_series의 위험도 * 계절요인 * 나이요인을 곱해서 나온 가장 높은 위험도 3개를 예방수칙과 함께 보여줌

결론은 이것만 돌리면 위 함수들을 돌아서 결과가 나옴

In [9]:

```

def final_output(input_age = 41, input_season = '겨울', location = '서울강남구'):

    prevention = pd.read_excel('감염병 예방.xlsx')

    with open('season.pickle', 'rb') as f:
        season = pickle.load(f)

    if input_age < 10:
        input_age = '0~9세'
    elif input_age < 20:
        input_age = '10~19세'
    elif input_age < 30:
        input_age = '20~29세'
    elif input_age < 40:
        input_age = '30~39세'
    elif input_age < 50:
        input_age = '40~49세'
    elif input_age < 60:
        input_age = '50~59세'
    elif input_age < 70:
        input_age = '60~69세'
    else:
        input_age = '70세 이상'

    risk_series = make_risk_series(location)

    danger = {}
    for disease_names in list(risk_series.index):
        # 계절별 요인
        season_dict = make_season_dict(disease_names) #####위에서 사용
        num_season = season_dict[input_season]

        # 나이별 요인
        age_dict = make_age_dict(disease_names) #####위에서 사용
        num_age = age_dict[input_age]

        danger[disease_names] = num_age * num_season * risk_series[disease_names]

    print('-----top3-----')
    print('{} , {} , {} 조심해야할 감염병 3개~'.format(input_age, input_season, location))
    print('')
    print(pd.Series(danger).sort_values()[::-1][:3])
    print('-----top3-----')
    print('')

    top3 = pd.Series(danger).sort_values()[::-1][:3]

    ## top3 예방수칙 보여주기 ##

    for disease_name in top3.index:
        for i in range(len(prevention)):
            if disease_name == prevention['질병'][i]:
                print('※ {}의 예방 수칙 ※'.format(prevention['질병'][i]))
                print('')
                print(prevention['예방수칙'][i])
                print('')
                if prevention['백신'][i] == '0':
                    print('★백신을 접종하세요★')
```

```
print('')

return pd.Series(danger).sort_values()[::-1]
```

In [10]:

```
# 수두
a = final_output(input_age = 15, input_season = '여름', location = '서울중구')
```

```
-----top3-----
10~19세, 여름, 서울중구 조심해야할 감염병 3개~

유행성이하선염      1.436682
장티푸스            0.592940
수두                0.582201
dtype: float64
-----top3-----
```

※ 유행성이하선염의 예방 수칙 ※

사람이 많이 모인곳이나 밀폐된 곳을 피한다.
외출 후 곧바로 손발을 씻고 양치질을 한다.
예방접종을 반드시 맞는다.

★백신을 접종하세요★

※ 장티푸스의 예방 수칙 ※

예방접종을 반드시 맞는다.
철저한 개인위생 관리

★백신을 접종하세요★

※ 수두의 예방 수칙 ※

외출 후엔 반드시 손발을 씻고 양치질을 합니다.
수두환자와 접촉하지 않도록 주의합니다.
수두 생백신으로 예방 접종 가능
수두에 노출되고 3일 이내에 백신을 접종 받으면 발병을 예방하거나 증상을 완화

★백신을 접종하세요★

In [11]:

```
a = final_output(input_age = 75, input_season = '여름', location = '서울중구')
```

```
-----top3-----
70세 이상, 여름, 서울중구 조심해야할 감염병 3개~
```

```
C형간염      1.132080
장티푸스     0.963474
B형간염     0.375790
dtype: float64
```

```
-----top3-----
```

※ C형간염의 예방 수칙 ※

환자의 개인 세면도구(칫솔, 면도기, 손톱 깎기)는 따로 사용
주사바늘이나 피어싱 기구, 침 등을 공동으로 사용금지
감염자와 성교시 가급적이면 콘돔을 사용하도록 한다.

※ 장티푸스의 예방 수칙 ※

예방접종을 반드시 맞는다.
철저한 개인위생 관리

★백신을 접종하세요★

※ B형간염의 예방 수칙 ※

예방접종을 반드시 맞는다.
혈액으로 전염되기 때문에 정맥주사 약물의 남용, 무분별한 성접촉, 면도기, 칫솔, 손톱깎
이 등을 환자와 같이 사용하는 경우, 비위생적인 문신, 피어싱 혹은 침술 등의 시술 등은
주의를 요한다

★백신을 접종하세요★

In [12]:

```
b = final_output(input_age = 75, input_season = '여름', location = '서울노원구')
```

```
-----top3-----  
70세 이상, 여름, 서울노원구 조심해야할 감염병 3개~
```

```
A형간염          0.172605  
장출혈성대장균감염증  0.129466  
유행성이하선염      0.045938  
dtype: float64
```

```
-----top3-----
```

※ A형간염의 예방 수칙 ※

개인위생 관리.

끓인 물을 마시거나 충분한 익힌 음식을 섭취.

외출 후에는 반드시 손을 깨끗이 씻는다.

예방접종을 반드시 맞는다.

★백신을 접종하세요★

※ 장출혈성대장균감염증의 예방 수칙 ※

In [122]:

```
ㄷ = final_output(input_age = 71, input_season = '가을', location = '전남화순군')
```

```
-----top3-----
70세 이상, 가을, 전남화순군 조심해야할 감염병 3개~
```

```
프프가무시증      1.124960
C형간염           1.035014
A형간염           0.193591
dtype: float64
```

```
-----top3-----
```

※ 프프가무시증의 예방 수칙 ※

사람이 많이 모인곳이나 밀폐된 곳을 피한다.
 외출 후 곧바로 손발을 씻고 양치질을 한다.
 2차감염 예방을 위하여 굵지 않도록 하고 손톱을 짧게 깎아준다.
 예방접종을 반드시 맞는다.

★백신을 접종하세요★

※ C형간염의 예방 수칙 ※

환자의 개인 세면도구(칫솔, 면도기, 손톱 깎기)는 따로 사용
 주사바늘이나 피어싱 기구, 침 등을 공동으로 사용금지
 감염자와 성교시 가급적이면 콘돔을 사용하도록 한다.

※ A형간염의 예방 수칙 ※

개인위생 관리.
 끓인 물을 마시거나 충분한 익힌 음식을 섭취.
 외출 후에는 반드시 손을 깨끗이 씻는다.
 예방접종을 반드시 맞는다.

★백신을 접종하세요★

In [125]:

```
d = final_output(input_age = 15, input_season = '가을', location = '충북청주시 청원구')
```

```
-----top3-----
10~19세, 가을, 충북청주시 청원구 조심해야할 감염병 3개~
```

```
유행성이하선염      2.266687
수두                1.231879
A형간염             0.059787
dtype: float64
```

```
-----top3-----
```

※ 유행성이하선염의 예방 수칙 ※

사람이 많이 모인곳이나 밀폐된 곳을 피한다.
외출 후 곧바로 손발을 씻고 양치질을 한다.
예방접종을 반드시 맞는다.

★백신을 접종하세요★

※ 수두의 예방 수칙 ※

외출 후엔 반드시 손발을 씻고 양치질을 합니다.
수두환자와 접촉하지 않도록 주의합니다.
수두 생백신으로 예방 접종 가능
수두에 노출되고 3일 이내에 백신을 접종 받으면 발병을 예방하거나 증상을 완화

★백신을 접종하세요★

※ A형간염의 예방 수칙 ※

개인위생 관리.
끓인 물을 마시거나 충분한 익힌 음식을 섭취.
외출 후에는 반드시 손을 깨끗이 씻는다.
예방접종을 반드시 맞는다.

★백신을 접종하세요★

In []: