

텍스트데이터분석 Assignment

Keyword : Climate Change

20192797 임형빈

계획 및 가설

- 1. 기후 변화를 키워드로 naver news, naver blog, cnn news 크롤링 진행
- 2. 크롤링 결과로 워드 클라우드 및 LSA 진행, 기후 변화의 원인과 대책에 대한 결과 수집
- 3. 원인과 대책으로 나온 키워드로 naver blog 크롤링 및 모델링 진행
- 4. 인사이트

계획 및 가설

가설 :

기후변화에 대한 뉴스와 블로그 내용은 기후변화에 대한 단순한 설명이 아닐 것이다.

기후변화가 어떤 형태로 세계에 일어나고 있고, 원인은 무엇이며, 이러한 대책이 필요하다는 점을 알리는 형태의 글들이 많을 것이다.

따라서 뉴스와 블로그에서 기후변화를 키워드로 글들을 크롤링하고, 가설에 따라 나온 내용들을 토대로 원인과 대책에 대한 세부 크롤링을 이어 진행할 예정.

모델링은 워드 클라우드와 LSA를 사용할 예정이다.

1. Climate를 키워드로 cnn news 크롤링

- GPT와 셀레니움, 크롬 드라이버를 활용했다.
- 페이지들을 돌며 우선 기사 링크들을 추출하고, 반복문을 통해 링크에 대해 기사 제목과 원문을 추출함.
- 사용한 파일명: [Cnn_news_crawling.ipynb](#)

1. Climate를 키워드로 cnn news 크롤링

```
from selenium import webdriver
from bs4 import BeautifulSoup as bs
from selenium.webdriver.support.ui import WebDriverWait
from selenium.webdriver.support import expected_conditions as EC
from selenium.webdriver.common.by import By
import time
from selenium.webdriver.chrome.options import Options
import pandas as pd
import os

# Google Chrome WebDriver 경로 설정
webdriver_path = "/Users/hyungson/Desktop/vincent/3-1/textdata/chromedriver_mac_arm64/chromedriver"
# 크롬 옵션 설정
options = Options()
options.add_argument('--headless') # 브라우저 창을 띄우지 않고 실행 (headless 모드)

# page_source를 BeautifulSoup으로 파싱

driver = webdriver.Chrome(webdriver_path, options=options)
driver.implicitly_wait(2)
driver.get('https://www.cnn.com/search?q=climate+change')

links = []

search_keyword = "climate+change"
page_count = 100

for page in range(1, page_count + 1):
    url = f"https://www.cnn.com/search?q={search_keyword}&page={page}&from={(page*10)-10}"
    driver.get(url)

    articles = driver.find_elements(By.CSS_SELECTOR, '.container_link._link')
    for article in articles:
        link = article.get_attribute('href')
        links.append(link)
```

```
result_dict = {'link':links}
df = pd.DataFrame.from_dict(result_dict)
df.to_csv('cnn_links.csv', index = False, encoding='utf-8-sig')

from selenium.common.exceptions import NoSuchElementException

titles = []
contents=[]

for link in links:
    link_driver = webdriver.Chrome(webdriver_path,options=options)
    link_driver.implicitly_wait(2)
    link_driver.get(link)

    try:
        title_elements = link_driver.find_element(By.CSS_SELECTOR, "div.headline_wrapper")
        title = title_elements.text.strip()
        titles.append(title)

        content_elements = link_driver.find_elements(By.CSS_SELECTOR, "div.article_content p")
        content = ' '.join([element.text.strip() for element in content_elements])
        contents.append(content)
    except NoSuchElementException:
        pass

    # Chrome WebDriver 종료
driver.quit()
```

1. Climate를 키워드로 cnn news 크롤링

```
data = {
    'Title': titles,
    'Content': contents
}
df = pd.DataFrame(data)
```

```
df
```

	Title	Content
0	Europe is trying to ditch planes for trains. Here's how...	Editor's Note: Sign up for <i>Unlocking the World...</i>
1	'Murderers' and 'criminals': Meteorologists face heat over climate...	"Murderers." "Criminals." "We are watching you..."
2	Brazil to host COP30 climate summit in the Amazon rainforest	The United Nations has chosen Brazil to host the...
3	Get up to speed on the US debt drama	A version of this story appeared in CNN's <i>What...</i>
4		
...
922	Why coastal communities should fear storm surge	Editor's Note: This story has previously been...
923	Photographer David LaChapelle offers a balm for...	For photographer David LaChapelle, signs are p...
924	Opinion: Here's who John Roberts should blame for...	Editor's Note: Dean Obeidallah, a former attorney...
925	How the Queen's soft power has helped keep the...	When Scots went to the polls to vote for indep...
926	This Palestinian teen went viral for slapping...	Ahed Tamimi rose to prominence in 2017 after a...

927 rows × 2 columns

```
df['Content'][0]
```

'Editor's Note: Sign up for *Unlocking the World*, CNN Travel's weekly newsletter. Get news about destinations opening, inspiration for future adventures, plus the latest in aviation, food and drink, where to stay and other travel developments. Ever since the "flight shame" movement began encouraging travelers to seek greener alternatives to jet planes, many in Europe have been looking to the continent's extensive rail network to replace short-haul air travel. There's definitely been progress. Airlines including Dutch carrier KLM are entering into rail partnerships on certain routes, while countries like Austria and France are seeking to restrict internal routes where trains are available – although the French decree, which was made law in May 2023, has been significantly watered down from its original premise. That's amid a palpable rail revolution on mainland Europe, with new high-speed routes and operators coming online, a reversal in the decline of overnight sleeper services, new tunnel links cutting travel times and new locomotives improving reliability and efficiency. In Spain, Germany and Austria, cheap ticket deals have also played their part. With so much railway investment, it seems as if the trainification of Europe's air transport network is well underway. Surely, it's only a matter of time before the continent is relying almost exclusively on its iron roadways for getting around and the skies are clearer and greener. In reality, that remains a distant dream. But why? As with many efforts to innovate away from environmentally harmful practices, there's good news and bad news. Fixes are being made, but none of them are quick. And there's no sign that Europe's airports are going to get quieter anytime soon. This year got off to a strong start with new legislation promised in France that would ban short-haul flight on a number of domestic routes to help the country cut levels of planet-heating pollution, but though approved by EU officials and then signed into French law in May 2023, the measures are limited in impact. For the ban to apply, the EU insisted the air route in question must have a high speed rail alternative that makes it possible to travel between the two cities in less than two and a half hours. There must also be enough early and late-running trains to enable travelers to spend at least eight hours at the destination. This means that ultimately only three routes were culled: those linking Paris-Orly airport to the cities of Bordeaux, Nantes and Lyon. In a further blow to those hoping for a rail revolution, it turned out that, as it happened, those routes had already been cut in 2020 – the new law just means that they will not be reinstated in the future. So what went wrong? The ruling by the EU's European Commission watered down the original French plans, which would have seen a further five routes ending: From Paris Charles de Gaulle Airport to Bordeaux, Nantes, Lyon and Rennes, as well as a Lyon to Marseilles route. From alternative fuels to rationing trips: A guide to more sustainable flying. The result, say critics, is something that pays lip service to climate concerns without really doing anything about them. "The French flight ban is a symbolic move, but will have very little impact on reducing emissions," Jo Dardenne, aviation director at cleaner transport campaign group Transport & Environment (T&E), told CNN before the law took effect. T&E has estimated that the three routes affected by the ban represent only 0.3% of the emissions produced by flights taking off from mainland France, and 3% of the country's domestic flight emissions (again counting only mainland domestic flights). If the five additional routes that the French authorities wanted to include were added, those figures would be 0.5% and 5% respectively. That doesn't sound like much. But although aviation as a whole currently accounts for around 2.5% of global carbon emissions, its overall contribution to climate change...

1. 기후변화를 키워드로 naver news

- Api를 이용해서, 반복문을 통해 키워드에 해당하는 기사 1000개를 추출할 예정.
- 실습에 사용한 코드와 naver developer의 내용을 참고했다.
- 사용한 파일명: [naver_news.ipynb](#)

1. 기후변화를 키워드로 naver news

```
article_ids = ['newsct_article', 'articeBody']
titles = []
links = []
pubdates = []
contents = []
def getresult():
    url = f'{base_url}?query={encQuery}&display={n_display}&start={start}&sort={sort}'

    my_request = urllib.request.Request(url)
    my_request.add_header("X-Naver-Client-Id",client_id)
    my_request.add_header("X-Naver-Client-Secret",client_secret)

    response = urllib.request.urlopen(my_request)

    rescode = response.getcode()
    if(rescode==200):
        response_body = response.read()
        #   print(response_body.decode('utf-8'))
    else:
        print("Error Code:" + rescode)
    search_result_str = response_body.decode('utf-8')
    search_results = json.loads(search_result_str)

    p = re.compile('https://n.news.naver.com/.+')
    for i, item in enumerate(search_results['items']):
        if p.match(item['link']): ## <link>태그의 문자열이 n.news.naver.com으로 시작하는 결과만 추출
            title = sub_html_special_char(remove_tag(item['title']))
            link = item['link']
            pubdate = item['pubDate']
            titles.append(title)
            links.append(link)
            pubdates.append(pubdate)

            html = urllib.request.urlopen(link)
            bs_obj = BeautifulSoup(html, 'html.parser')
            for article_id in article_ids:
                print(article_id)
                content = bs_obj.find_all('div', {'id':article_id})
                if len(content) > 0:
                    contents.append(content[0].text)
                    break
                else:
                    continue
            time.sleep(3)

    result_dict = {'title': titles, 'link': links, 'pubdate': pubdates, 'content': contents}
    df = pd.DataFrame.from_dict(result_dict)
    return df
```

```
client_id = 'frPEZvEmZbhkQTJR4a9u'
client_secret = '2nOa73V0gW'
query = '기후변화'
display=100
start=1

result_all=pd.DataFrame()
for i in range(0,10):
    start= 1 + 100*i
    result= getresult()

    result_all=pd.concat([result_all,result])
...

```

시간이 매우 오래걸렸고, 중복값이 매우 많이 추출됨. 이후에 naver news 크롤링을 한다면, 코드 수정 후 이용해야겠다.

```
result_all.shape
(2812, 4)

result_all = result_all.drop_duplicates()

result_all.shape
(487, 4)

result_all.to_csv('naver_news_climate_change.csv',encoding='utf-8')
```

코드에 문제가 있는지 1000개 추출이 아닌 2812 개가 추출이 되었고, 대부분이 중복값이었음.
수정 후 네이버 뉴스를 사용하려 했으나
이후 과제 진행 중에는 네이버 뉴스를 사용하지
않아서 수정은 진행하지 않았음.

1. 기후변화를 키워드로 naver blog

- Api를 이용해서, 반복문을 통해 키워드에 해당하는 기사 1000개를 추출할 예정.
- 실습에 사용한 코드와 naver developer의 내용을 참고했다.
- 사용한 파일명: [naver_blog.ipynb](#)

1. 기후변화를 키워드로 naver blog

```
import os
import sys
import urllib.request
import json
import pandas as pd

def getresult(client_id,client_secret,query,display=10,start=1,sort='sim'):
    encText = urllib.parse.quote(query)
    url = "https://openapi.naver.com/v1/search/blog?query=" + encText + \
        "&display=" + str(display) + "&start=" + str(start) + "&sort=" + sort

    request = urllib.request.Request(url)
    request.add_header("X-Naver-Client-Id",client_id)
    request.add_header("X-Naver-Client-Secret",client_secret)
    response = urllib.request.urlopen(request)
    rescode = response.getcode()
    if(rescode==200):
        response_body = response.read()
        response_json = json.loads(response_body)
    else:
        print("Error Code:" + rescode)

    return pd.DataFrame(response_json['items'])['link']
```

```
client_id = 'frPEZvEmZbhkQTJR4a9u'
client_secret = '2nOa73V0gW'
query = '기후변화'
display=100
start=1
sort='sim'
result_all=pd.DataFrame()

for i in range(0,10):
    start= 1 + 100*i
    result= getresult(client_id,client_secret,query,display,start,sort)
    result_all=pd.concat([result_all,result])
```

```
#타이틀과 본문을 iframe에서 모두 추출하도록하기
import selenium
from selenium.webdriver.common.by import By
driver = webdriver.Chrome("/Users/hyungson/Desktop/vincent/3-1/textdata/chromedriver_mac")
from selenium.common.exceptions import NoSuchElementException

titles = []
contents = []
links = []
for i in range(len(result_all)):
    link = result_all['link'][i]
    if 'blog.naver.com' in link:
        driver.get(result_all['link'][i])
        driver.implicitly_wait(2) #time sleep
        iframe = driver.find_element('id', 'mainFrame')
        driver.switch_to.frame(iframe)
        try:
            content = driver.find_element(By.CSS_SELECTOR, 'div.se-main-container').text
            content = content.replace('\n', '') #줄바꿈 제거
            contents.append(content)
            title_tag = driver.find_element(By.TAG_NAME, 'title') #태그로 타이틀을 검색
            title = title_tag.get_attribute('textContent')
            titles.append(title)
            links.append(link)
        except NoSuchElementException:
            pass
driver.quit()
```

	title	content	link
0	부산기후변화체험관 유아 초등 아이와 실내 가볼만 한 곳 : 네이버 블로그	안녕하세요^^여행 인플루언서 텔미예요.부산 북구 실내 아이와 가볼만 한 곳부산기후변...	https://blog.naver.com/tellmememory/223101221334
1	기상청 기자단 활동을 통해 기후변화를 고민하다!: 네이버 블로그	제14기 국민참여기자단 한홀활동증인 모습, 출처=개티이미지뱅크2022년 한 해는 기...	https://blog.naver.com/kma_131/223026067721
2	부산 기후변화체험관 비오는 날 아이랑 갈만한 학명동 기후변화 박물관 : 네이버 블로그	부산 기후변화체험관안녕하세요.비오는 날 일별경에 디녀온 부산 기후변화체험관! 후기입니다.	https://blog.naver.com/yousuyoon/223076516196
3	직접 체험하며 느끼는 수원시기후변화체험교육관두드림 ☆ : 네이버 블로그	우리나라뿐 아니라 전 세계적으로 기온이 점점 상승하고 있다고 하는데요. 작은 실천이...	https://blog.naver.com/suwonloves/223109775737

2. cnn news 크롤링 데이터로 모델링

- 워드 클라우드와 LSA를 진행해서 나온 결과들로 추가적인 크롤링을 진행하겠다.
- 진행한 주피터노트북 파일명: [cnn_wordcloud.ipynb](#)

2. cnn news 크롤링 데이터로 wordcloud

```
import copy
import re

def preprocess_english(text):
    my_text = copy.copy(text)
    my_text = my_text.replace('\n', '')
    sents = nltk.sent_tokenize(my_text)
    tokenizer = TreebankWordTokenizer()
    stopwords = nltk.corpus.stopwords.words('english')

    p = re.compile('[^A-Za-z]')
    result = []
    for sent in sents:
        sent = sent.lower()
        sent = p.sub(' ', sent) # (4) special char removal
        word_tokens = tokenizer.tokenize(sent) # (5) word tokenization
        for token in word_tokens:
            if token not in stopwords:
                result.append(token) # (6) stopwords removal
    result = ' '.join(result)
    return result
```

```
import pandas as pd
cnn = pd.read_csv('cnn_news_927.csv')

pos_data = cnn.copy()

pos_data = pos_data.drop_duplicates()

pos_data.dropna(inplace = True)

pos_data['preprocessed_Content'] = pos_data['Content'].apply(lambda x: preprocess_english(x))
```

길이가 길어서인지 워드 클라우드가 그려지지 않음.

```
cloud = WordCloud(
    background_color = 'white',
    width=800, height=800, )
my_cloud1 = cloud.generate_from_text(all_pos_data)

arr1 = my_cloud1.to_array()

fig = plt.figure(figsize=(10, 10))
plt.imshow(arr1)
plt.axis('off')
plt.show()
# fig.savefig('wordcloud_without_axisoff.png') 생성한 그림 저장하기
```

처음 워드클라우딩을 진행했다가 실패함. 길이가 너무 길어서 인 줄 알아서 데이터 길이가 7000이상은 행을 삭제했는데도 되질 않음. -> data를 반으로 줄여서 각각 진행 -> ppt 만드는 시점에 다시 돌려보니 워드클라우드 정상 작동함.

2. cnn news 크롤링 데이터로 wordcloud

전체 문장을 반으로 나눠서 두번에 걸쳐서 진행하겠음.

```
# 코드 gpt 참고
def split_string(text):
    midpoint = len(text) // 2
    left_half = text[:midpoint]
    right_half = text[midpoint:]

    return left_half, right_half

# 문자열 반으로 분리
left_half, right_half = split_string(all_pos_data)

cloud = WordCloud(font_path = font_path,
                   background_color = 'white',
                   width=800, height=800, )

my_cloud1 = cloud.generate_from_text(left_half)

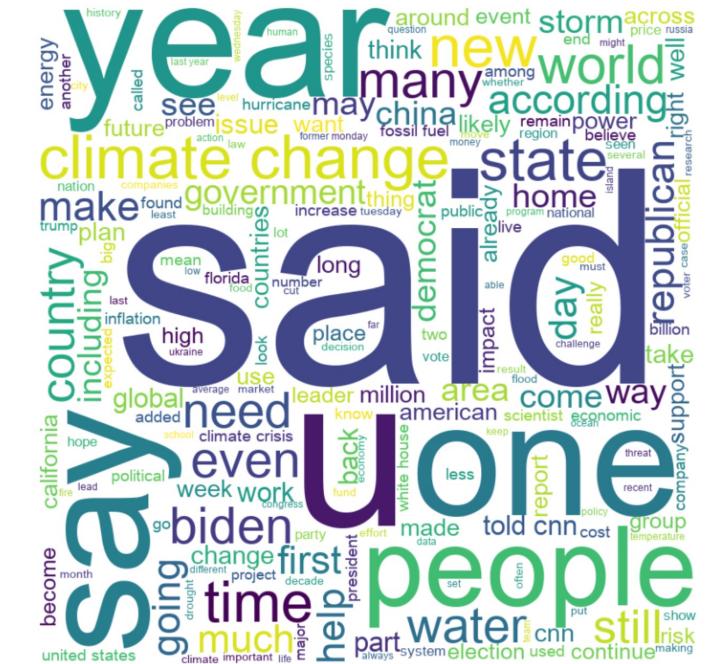
# WordCloud 생성 및 이미지 생성 (첫 번째 절반)
arr1 = my_cloud1.to_array()

# 이미지 출력 (첫 번째 절반)
fig = plt.figure(figsize=(10, 10))
plt.imshow(arr1)
plt.axis('off')
plt.show()
```

```
my_cloud2 = cloud.generate_from_text(right_half)
# WordCloud 생성 및 이미지 생성 (두 번째 절반)
arr2 = my_cloud2.to_array

# 이미지 출력 (두 번째 절반)
fig = plt.figure(figsize=(10, 10))
plt.imshow(arr2)
plt.axis('off')
plt.show()
```

2. cnn news 크롤링 데이터로 wordcloud



왼쪽에서부터 전체, left half, right half

정책이나 정치에 관한 키워드가 많음.

대책이나 원인에 대한 키워드는 가정과 달리 유의미한 결과는 없었다.

2. cnn news 크롤링 데이터로 LSA

```
terms = vectorizer.get_feature_names_out()  
get_keyword_by_topic(model.components_, terms)
```

Topic 0: [('biden', 0.2255), ('water', 0.1375), ('president', 0.1239), ('house', 0.1163), ('energy', 0.0976)]
Topic 1: [('biden', 0.3736), ('house', 0.1819), ('president', 0.1637), ('republicans', 0.1548), ('democrats', 0.1465)]
Topic 2: [('water', 0.4068), ('river', 0.1582), ('california', 0.1394), ('drought', 0.125), ('lake', 0.1222)]
Topic 3: [('china', 0.4216), ('water', 0.2715), ('xi', 0.2231), ('biden', 0.1879), ('chinese', 0.1771)]
Topic 4: [('water', 0.3982), ('energy', 0.1872), ('river', 0.1375), ('oil', 0.1212), ('bank', 0.1184)]
Topic 5: [('hurricane', 0.2102), ('storm', 0.2045), ('bank', 0.1556), ('energy', 0.1532), ('ian', 0.1356)]
Topic 6: [('ice', 0.271), ('biden', 0.2183), ('warming', 0.1443), ('temperatures', 0.1366), ('degrees', 0.1333)]
Topic 7: [('bank', 0.2252), ('inflation', 0.2196), ('fed', 0.2075), ('ice', 0.152), ('debt', 0.1372)]
Topic 8: [('charles', 0.2437), ('king', 0.2061), ('royal', 0.1435), ('ni', 0.1256), ('queen', 0.1255)]
Topic 9: [('biden', 0.3332), ('charles', 0.2607), ('king', 0.2152), ('royal', 0.167), ('queen', 0.1447)]
Topic 10: [('court', 0.2493), ('willow', 0.2255), ('project', 0.2153), ('alaska', 0.1894), ('oil', 0.1809)]
Topic 11: [('charles', 0.1626), ('king', 0.1495), ('ice', 0.148), ('plastic', 0.1332), ('meat', 0.1244)]
Topic 12: [('ice', 0.2381), ('china', 0.173), ('sea', 0.1679), ('water', 0.1346), ('hurricane', 0.1215)]
Topic 13: [('debt', 0.2417), ('court', 0.2007), ('mccarthy', 0.1556), ('pakistan', 0.137), ('ceiling', 0.1311)]
Topic 14: [('ice', 0.307), ('glacier', 0.2081), ('glaciers', 0.1873), ('pakistan', 0.1322), ('sea', 0.1241)]
Topic 15: [('project', 0.2316), ('oil', 0.2144), ('willow', 0.2067), ('energy', 0.1671), ('alaska', 0.1647)]
Topic 16: [('court', 0.2055), ('debt', 0.1675), ('energy', 0.1561), ('ni', 0.1518), ('ukraine', 0.1422)]
Topic 17: [('lula', 0.3086), ('brazil', 0.2491), ('bolsonaro', 0.2364), ('amazon', 0.2247), ('ni', 0.1562)]
Topic 18: [('plastic', 0.2998), ('meat', 0.2295), ('food', 0.1658), ('bags', 0.152), ('oil', 0.1252)]
Topic 19: [('lula', 0.2236), ('brazil', 0.1854), ('bolsonaro', 0.1816), ('court', 0.1535), ('ukraine', 0.1382)]

```
#water + river + california + drought + lake  
#ice + biden + warming + tempeatures + degrees  
#바이든 + 물 + 대통령 + 백악관 + 에너지 -> 짧게 살펴본 결과 환경에 관한 뉴스는 없음  
#2 물 + 강 + 캘리포니아 + 가뭄 + 호수  
#12 빙하 + 중국 + 바다 + 물 + 허리케인 : 8페이지  
#18 플라스틱 + 고기 + 음식 + 가방 + oil -> oil에 대한 번역이 애매해서 검색 결과가 마땅치않은듯.
```

word cloud & LSA 를 사용하여 결과값을 보면:

cnn news는 정치, 정책 등에 관한 키워드가 다수 등장하고, 기후 변화 대책에 대한 키워드는 빈도가 적은 듯함.

하지만 일부 토픽들은 주제에 맞는 단어들에 적합 : 2,12,18

이 주제들을 번역해서 네이버로 다시 검색 시도해보겠음.

2. naver news, blog 크롤링 데이터 모델링

- naver news, blog에서 나온 결과들로 워드 클라우드 진행
- 사용한 파일명: climate_change_wordcloud.ipynb

2. naver news 크롤링 데이터 wordcloud



네이버 뉴스 (키워드 : 기후 변화) 워드 클라우딩을 통해
기후 변화 대책에 관한 키워드 몇개를 선정

탄소 중립, 친환경, 온실가스 감축, 녹색 성장, 기후 기술, 중립 실천(탄소 중립 실천)

전처리 후 워드클라우딩 진행
대책에 관한 키워드 추출

원인에 대한 내용은 별로 없는 듯 했다.

2. naver blog 크롤링 데이터 wordcloud



탄소 중립, 친환경, 온실가스 감축, 녹색 성장, 기후 기술, 지속가능, 환경캠페인

탄소 중립, 친환경, 온실가스감축, 녹색성장, 기후
기술, 지속가능, 환경캠페인을 키워드로
이후의 과정을 진행

3. Keyword로 네이버 블로그 크롤링

- 약 5천여개의 기후변화 대책에 관한 블로그 글을 크롤링 했음.
- 이를 이용해 모델링을 추가로 진행해보았다.
- 사용한 파일명: naver_blog_keyword_crawl.ipynb

3. Cnn news LSA 토픽을 키워드로 네이버 블로그 크롤링

- #2 물 + 강 + 캘리포니아 + 가뭄 + 호수
- #12 빙하 + 중국 + 바다 + 물 + 허리케인

을 키워드로 크롤링진행. 3 토픽 중 1개는 oi에 대한 해석이 모호해서 진행하지 않음.
기름, 유로 해석해서 검색을 해봤지만 관련 글이 적었다.

네이버는 검색어에 a,b가 모두 들어간 결과를 찾고 싶을 땐 a+b 의 식으로 검색을 진행
함. +는 주소 창에 2%의 식으로 대체된 것을 확인함.

#'물%2B강%2B캘리포니아%2B가뭄%2B호수' 이런 식으로 키워드 두개를 가지고 크롤링
진행.

- 사용한 파일명: naver_blog_other_topics.ipynb

3. Keyword로 모델링

Naver blog, news의 키워드로 기후변화 대책에 관한 블로그 글을 크롤링 했음.
Cnn news LSA의 키워드로 블로그 글 크롤링.

전처리 후 데이터는 약 6000여개

이를 이용해 모델링을 추가로 진행해보았다.

사용한 파일명 : keywordcrawling_wordcloud.ipynb

3. Keyword로 모델링

환경 캠페인, 지속가능, 기후 기술 3가지 키워드를 반으로 나눈 데이터 프레임



3. Keyword로 모델링

탄소중립, 친환경, 온실가스감축, 녹색성장 각각에 대한 워드 클라우드



키워드 워드클라우드 인사이트 : ¶

네이버뉴스와 블로그를 통해 나온 대책에 관한 키워드로 워드 클라우드를 추가로 진행해서 살펴보니 세부적인 키워드들이 더 보인다. 블로그 글을 통해 사람들과 기업, 정부가 어떤 분야에서 기후 변화 대책에 대해 노력하고 있는지 대강적으로 살펴볼 수 있음. 네이버 블로그 뿐만 아니라 다양한 플랫폼에서 크롤링을 진행한다면, 우리나라 뿐만 아니라 전세계가 어떤 노력을 어느 분야에서 할 수 있을지 볼 수 있을듯하다.

3. Keyword로 모델링

cnn news LSA 토픽들 워드 클라우드



이번 워드클라우드에서는 유의미한 결과는 딱히 찾아볼 수 없었다.

Cnn news에 대한 데이터가 더 많았을 때
LSA를 했다면 더 유의미하고 연관이 많도록
토픽이 만들어지지 않았을까 생각해본다.

3. Keyword로 모델링

```
In [9]: terms = vectorizer.get_feature_names_out()
get_keyword_by_topic(model.components_, terms)

Topic 0: [('탄소', 0.3649), ('중립', 0.3006), ('감축', 0.182), ('기술', 0.1747), ('온실가스', 0.1653)]
Topic 1: [('감축', 0.223), ('탄소', 0.1916), ('온실가스', 0.1699), ('중립', 0.1695), ('기술', 0.1354)]
Topic 2: [('포인트', 0.3821), ('중립', 0.3765), ('탄소', 0.3513), ('실천', 0.3065), ('가입', 0.1177)]
Topic 3: [('감축', 0.4921), ('온실가스', 0.4071), ('목표', 0.1677), ('사업', 0.1293), ('배출', 0.1256)]
Topic 4: [('세제', 0.3208), ('탄소', 0.1603), ('세탁', 0.1569), ('친환경', 0.1537), ('중립', 0.1487)]
Topic 5: [('포인트', 0.4928), ('기술', 0.3139), ('가입', 0.1609), ('사업', 0.1132), ('지급', 0.1095)]
Topic 6: [('지속', 0.3705), ('가능하다', 0.255), ('가능', 0.1807), ('발전', 0.1239), ('패션', 0.1123)]
Topic 7: [('세제', 0.346), ('사업', 0.2672), ('지원', 0.2244), ('친환경', 0.1655), ('세탁', 0.1632)]
Topic 8: [('녹색', 0.3701), ('성장', 0.2662), ('포인트', 0.2654), ('계획', 0.151), ('목표', 0.1221)]
Topic 9: [('세제', 0.4437), ('세탁', 0.2139), ('지속', 0.2105), ('에너지', 0.1666), ('발전', 0.1384)]
Topic 10: [('교육', 0.3954), ('기후', 0.2114), ('기후변화', 0.1411), ('학교', 0.1316), ('감축', 0.1226)]
Topic 11: [('녹색', 0.1956), ('농산물', 0.1765), ('농업', 0.1667), ('실천', 0.157), ('다이어트', 0.1538)]
Topic 12: [('청소년', 0.4118), ('에너지', 0.3202), ('자동차', 0.2049), ('캠페인', 0.1224), ('유해', 0.1215)]
Topic 13: [('청소년', 0.3782), ('기술', 0.239), ('패션', 0.1663), ('교육', 0.1636), ('기후변화', 0.1359)]
Topic 14: [('에너지', 0.3157), ('교육', 0.2347), ('사용', 0.1312), ('녹색', 0.1269), ('물티슈', 0.1147)]
Topic 15: [('청소년', 0.3216), ('기업', 0.2469), ('테크', 0.1555), ('카본', 0.1483), ('사업', 0.1314)]
Topic 16: [('카본', 0.227), ('기술', 0.2139), ('수소', 0.2), ('다이어트', 0.1702), ('로우', 0.1649)]
Topic 17: [('교육', 0.3001), ('자동차', 0.2314), ('모집', 0.1635), ('플라스틱', 0.1526), ('신청', 0.1423)]
Topic 18: [('물티슈', 0.3354), ('청소년', 0.2871), ('자동차', 0.2136), ('설거지', 0.1682), ('국제', 0.1472)]
Topic 19: [('친환경', 0.2208), ('카본', 0.2002), ('농업', 0.1789), ('수소', 0.1504), ('농산물', 0.1442)]
```

어느정도 기후 변화 및 환경파괴에 악영향을 미치는 키워드도 보이고,

어느 분야에서 친환경적인 움직임이 나타나고 있는지 예상할 수 있다.

하지만, 이건 내가 어느정도의 사전지식이 있어서 해석할 수 있는 것이고,

무지한 분야에서도 정도의 LSA와 워드 클라우딩으로 인과 관계나 상관 관계를 파악할 수 있을지는 모르겠다.

5000여개의 키워드 데이터들을 가지고
LSA를 진행해보았음.

3-1. 추가 크롤링 및 모델링

- 처음에 시작한 cnn news, naver blog, news에 대한 키워드들을 토대로 크롤링과 모델링을 진행했었지만, 네이버 api는 한 키워드에 대해 최대 1000개 추출까지밖에 안되는 단점이 있었다.
- 그래서 cnn news, naver blog, news 분석을 통해 얻지 못한 기후변화의 원인에 대해서 따로 크롤링과 모델링을 진행해보았고,
- 관심있는 분야인 슬로우 패션에 대해서도 크롤링과 모델링을 진행했다.
- 사용한 파일명 : naver_blog_slow_fast_fashion.ipynb, naver_blog_reason.ipynb

3-1. 원인+환경파고로 크롤링 후 모델링



```
terms = vectorizer.get_feature_names_out()
get_keyword_by_topic(model.components_, terms)
```

```
Topic 0: [('지구', 0.203), ('오존층', 0.1436), ('플라스틱', 0.1326), ('오염', 0.1322), ('인간', 0.1237)]
Topic 1: [('여드름', 0.5274), ('피부', 0.3549), ('치료', 0.2723), ('증상', 0.1573), ('경우', 0.1123)]
Topic 2: [('아보카도', 0.7616), ('재배', 0.0954), ('푸드', 0.0748), ('멕시코', 0.0726), ('슈퍼', 0.0694)]
Topic 3: [('오존층', 0.5947), ('아보카도', 0.5161), ('프레온', 0.2045), ('오존', 0.177), ('자외선', 0.1634)]
Topic 4: [('말굽', 0.5118), ('버섯', 0.5046), ('연구', 0.1799), ('증상', 0.1438), ('게르마늄', 0.1244)]
Topic 5: [('플라스틱', 0.5853), ('쓰레기', 0.1947), ('오염', 0.1351), ('담배', 0.1259), ('미세', 0.1164)]
Topic 6: [('건강', 0.1593), ('생명', 0.1541), ('증상', 0.1394), ('자연', 0.12), ('질환', 0.1199)]
Topic 7: [('플라스틱', 0.3759), ('생명', 0.2594), ('자연', 0.2505), ('여드름', 0.2189), ('말리다', 0.1706)]
Topic 8: [('오염', 0.4459), ('환경오염', 0.2051), ('토양', 0.2033), ('담배', 0.1845), ('폐수', 0.1647)]
Topic 9: [('담배', 0.8101), ('금연', 0.3342), ('꽁초', 0.1797), ('흡연', 0.1015), ('연기', 0.0859)]
Topic 10: [('태양광', 0.4575), ('에너지', 0.1924), ('산림', 0.1623), ('발전', 0.1525), ('산지', 0.1461)]
Topic 11: [('플라스틱', 0.2102), ('지구온난화', 0.1863), ('기후변화', 0.1621), ('상승', 0.1539), ('온난화', 0.144)]
Topic 12: [('지구', 0.2122), ('피부', 0.1984), ('에너지', 0.1641), ('채식', 0.1493), ('지구온난화', 0.1239)]
Topic 13: [('피부', 0.4644), ('태양광', 0.3143), ('콜라겐', 0.1526), ('산림', 0.1467), ('생물', 0.1272)]
Topic 14: [('태양광', 0.3127), ('지구', 0.1737), ('여드름', 0.1724), ('동물', 0.1669), ('산림', 0.1239)]
Topic 15: [('꿀벌', 0.8062), ('채식', 0.1417), ('바이러스', 0.1027), ('아마존', 0.0915), ('육식', 0.0787)]
Topic 16: [('꿀벌', 0.4003), ('기후', 0.1219), ('위기', 0.0838), ('읽다', 0.0823), ('변기', 0.0813)]
Topic 17: [('산불', 0.2512), ('쓰레기', 0.1878), ('사랑', 0.1303), ('생물', 0.1296), ('위기', 0.1286)]
Topic 18: [('비트코인', 0.2864), ('채굴', 0.2491), ('변기', 0.1933), ('봉수', 0.1733), ('배관', 0.1683)]
Topic 19: [('생물', 0.2809), ('다양성', 0.2032), ('생태계', 0.1577), ('채식', 0.1469), ('태양광', 0.1352)]
```

워드 클라우드로는 플라스틱 하나 건질 수 있었음

LSA로 토픽을 추출해보니 어느정도 원인 키워드가 보인다.

플라스틱, 아보카도, 담배꽁초, 육식, 비트코인 채굴, 폐수 등

3-2. 슬로우패션, 패스트 패션 각각 크롤링 후 모델링

```
from nltk.tokenize import TreebankWordTokenizer
from sklearn.feature_extraction.text import TfidfVectorizer

tfidf = TfidfVectorizer()
tfidf_mat = tfidf.fit_transform(tmp_data['preprocessed_content'])

tfidf_mat
<1330x15311 sparse matrix of type '<class 'numpy.float64'>'  
with 250032 stored elements in Compressed Sparse Row format>

from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression

X = tfidf_mat.toarray()
y = tmp_data['class'].to_numpy()

X_trn, X_tst, y_trn, y_tst = train_test_split(X, y, test_size=0.2, stratify=y)

model = LogisticRegression()
model.fit(X_trn, y_trn)

LogisticRegression()
In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.  
On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.

y_pred = model.predict(X_tst)

from sklearn.metrics import accuracy_score

accuracy_score(y_tst, y_pred)
0.10526315789473684
```

Text classification으로, 두 데이터를 구별할 수 있는지 진행해봤음.

슬로우 패션은 친환경적인 키워드가 많을 것이고, 패스트 패션은 환경파괴적인 키워드가 많을 것이라고 가정했었고, 모델링을 진행해봤지만,

정확도가 0.1로 매우 낮았다.

3-2. 슬로우패션, 패스트 패션 각각 크롤링 후 모델링

정확도가 매우 낮음. 각 파일이 slow인지 fast인지에 대한 명확한 특징이 존재하지 않는 것일 수도 있는 것으로 보인다.

같은 파일에서 워드클라우드까지 진행해보겠음.



왼쪽부터 fast, slow fashion (파일에는 같이 진행한 워드클라우드도 있음)

비교 결과, 패스트패션이라는 말도 슬로우 패션이라는 말이 생기고 나서 생긴 느낌이다.

특정한 패션 브랜드나 종류를 패스트 패션이라 하는 것이 아닌, 슬로우 패션이 추구하는 것과 반대되는 패션 산업을 패스트 패션이라 칭하는 것 같음.

양 워드 클라우드에서 모두 환경과 슬로우 패션이라는 키워드 들이 등장하며, 슬로우 패션 브랜드 명들이 추출되는 것을 볼 수 있음. ([Text classification 정확도 낮은 이유인듯](#))

원래 이번 키워드를 선택한 목적은 슬로우 패션을 지향하는 브랜드 명을 찾고 싶었던 것이었으나, 전처리 후 브랜드명에 손상이 가서 별로 추출이 안되는 것 같다. (몇몇 보이긴 함.)

고유명사에 대한 전처리를 따로 처리해줄 수 있으면 좋겠다는 생각을 얻었다.

3-2. 슬로우패션, 패스트 패션 각각 크롤링 후 모델링

```
terms = vectorizer.get_feature_names_out()
get_keyword_by_topic(model.components_, terms)

Topic 0: [('패스트', 0.2063), ('환경', 0.1881), ('의류', 0.1589), ('입다', 0.1339), ('친환경', 0.13)]
Topic 1: [('카페', 0.2057), ('팬츠', 0.197), ('애시드', 0.1652), ('퍼티', 0.1255), ('오어', 0.1228)]
Topic 2: [('카페', 0.4344), ('먹다', 0.175), ('영도', 0.1354), ('맛있다', 0.1284), ('이드', 0.1193)]
Topic 3: [('퍼티', 0.4348), ('오어', 0.4303), ('팬츠', 0.2963), ('복각', 0.1044), ('일본', 0.0923)]
Topic 4: [('요가', 0.3078), ('나이스', 0.2211), ('브라', 0.1642), ('운동', 0.1418), ('요가복', 0.1346)]
Topic 5: [('애시드', 0.397), ('볼캡', 0.3468), ('패스트', 0.1579), ('모자', 0.1562), ('의류', 0.0928)]
Topic 6: [('요가', 0.3745), ('나이스', 0.2722), ('브라', 0.1943), ('포크', 0.1767), ('운동', 0.1758)]
Topic 7: [('요가', 0.2095), ('카페', 0.2079), ('애시드', 0.173), ('친환경', 0.1646), ('비건', 0.1611)]
Topic 8: [('수선', 0.3959), ('가게', 0.2834), ('비건', 0.2281), ('책방', 0.22), ('재봉틀', 0.2061)]
Topic 9: [('클럽', 0.339), ('tedi', 0.2364), ('영도', 0.2166), ('마을', 0.2032), ('슬로', 0.1307)]
Topic 10: [('클럽', 0.2118), ('레시피', 0.1593), ('수선', 0.1532), ('tedi', 0.1436), ('터틀', 0.1418)]
Topic 11: [('행사', 0.2611), ('에콰도르', 0.2385), ('사진', 0.2294), ('레시피', 0.2213), ('동영상', 0.1849)]
Topic 12: [('투맨', 0.2982), ('빈티', 0.2037), ('오어', 0.1955), ('포크', 0.1694), ('볼캡', 0.1331)]
Topic 13: [('에콰도르', 0.2288), ('동영상', 0.2056), ('케이크', 0.1832), ('사진', 0.1785), ('터틀', 0.17)]
Topic 14: [('제로', 0.2275), ('이스트', 0.2009), ('에콰도르', 0.1965), ('지구', 0.1963), ('사진', 0.1749)]
Topic 15: [('투맨', 0.3998), ('산업', 0.1183), ('텐션', 0.1079), ('비건', 0.0948), ('가죽', 0.0939)]
Topic 16: [('포레스트', 0.4205), ('삼청동', 0.2567), ('카페', 0.1949), ('루프', 0.1111), ('라떼', 0.1072)]
Topic 17: [('투맨', 0.3262), ('제로', 0.1959), ('이스트', 0.1843), ('패스트', 0.1288), ('케이크', 0.1138)]
Topic 18: [('지갑', 0.2184), ('가죽', 0.1895), ('먹다', 0.1844), ('볼캡', 0.1411), ('비건', 0.1344)]
Topic 19: [('팬츠', 0.2328), ('선글라스', 0.2106), ('젠틀', 0.1577), ('몬스터', 0.1576), ('볼캡', 0.1373)]
```

LSA 결과 유의미한 결과는 없었다.

카페는 왜 이렇게 많은지.....?

추가로 궁금증이 생겨서 슬로우패션 + 카페라는 키워드로 따로 검색을 해봤다.

슬로우패션이라는 패션 산업은 요새 인기를 끌고 있는 산업이고, 젊은 층에 주로 겨냥한다.

카페 산업과 더불어서 브랜딩을 하는 슬로우패션 브랜드가 많은 것으로 보인다.

전처리, vectorization, 크롤링 관련

```
import copy

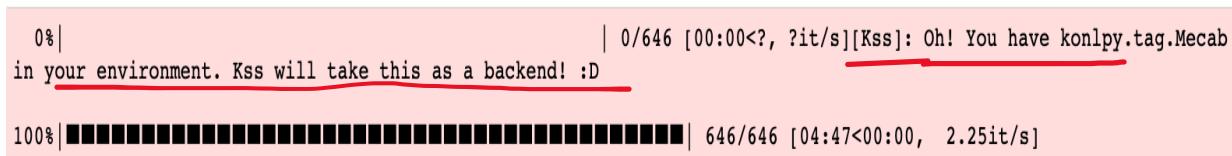
with open('stopwords.txt', 'r', encoding='utf-8') as f:
    stopwords = f.readlines()
stopwords = [x.replace('\n', '') for x in stopwords] # stopword 파일의 줄바꿈 문자 제거
okt = Okt()

def preprocess_korean(text):
    my_text = copy.copy(text)
    my_text = my_text.replace('\n', '')
    spacer = pykospacing.Spacing()
    my_text = spacer(my_text) # (3) 띄어쓰기 교정
    sents = kss.split_sentences(my_text) # (4) sentence tokenization

    p = re.compile('[^ㄱ-ㅎㅏ-ㅣ가-힣]')
    results = []
    for sent in sents:
        result = []
        tokens = okt.morphs(sent, stem=True) # (6) word tokenization
        for token in tokens:
            token = p.sub('', token) # (5) 특수문자 제거
            if token not in stopwords:
                result.append(token) # (7) stopword removal
        results.extend(result)
    result = ' '.join(results)

return result
```

- 전처리 코드는 실습에 사용한 코드를 사용했고, 데이터 길이가 길었을 때, 걸리는 시간이 굉장히 길었다. 여러 시간에 걸쳐서 꾸준히 진행했다. 특히, 전처리 코드 중에 kss를 이용하는 코드가 있었는데, mecab을 backend로 사용할 수 있다는 메시지가 코드 중에 떠서, 설치 후에 진행한 결과 속도는 향상됐다.
- Tqdm 모듈로, 코드에 적용해서 진행상태를 알아보도록 했다.
- Progress_apply 함수로 데이터프레임 행에 전처리 함수를 적용시켜 사용.



CPU times: user 4min 46s, sys: 24.5 s, total: 5min 10s
Wall time: 4min 47s

전처리, vectorization, 크롤링 관련

```
vectorizer = TfidfVectorizer( max_features=2000, max_df=0.5, smooth_idf=True) # 상위 2000개의 단어에 대해서만  
X = vectorizer.fit_transform(df['preprocessed_content'])
```

```
# slow fashion은 label = 1, fast fashion label=0으로 해서 classification 진행  
tmp_data_1['class'] = 0  
tmp_data_2['class'] = 1
```

```
tmp_data = pd.concat([tmp_data_1, tmp_data_2])
```

```
from nltk.tokenize import TreebankWordTokenizer  
from sklearn.feature_extraction.text import TfidfVectorizer
```

```
tfidf = TfidfVectorizer()  
tfidf_mat = tfidf.fit_transform(tmp_data['preprocessed_content'])
```

```
tfidf_mat
```

```
<1330x15311 sparse matrix of type '<class 'numpy.float64'>'  
with 250032 stored elements in Compressed Sparse Row format>
```

- LSA와 classification 을 사용할 때는 TfidfVectorizer를 사용함.

4. 인사이트

- 워드 클라우드를 주로 사용하여 결과를 얻었다. 결과에서 유의미한 인사이트를 얻기 힘들었다. 하다, 이다, 은, 는 등의 빈도 높은 단어들이 많이 추출되어서 방해받은 영향이 큰 것 같다. 다음에 워드 클라우드를 이용한다면, 위와 같은 빈도 높은 단어들을 추가로 제거하는 작업이 필요할 것 같다. 이후 명사들로만 워드 클라우드를 진행한다면 전보다는 많은 인사이트를 얻을 것으로 보인다.
- LSA 결과로 나온 토픽들은 꽤나 흥미로웠지만, 더 많은 데이터와 전처리 후에 진행하고, 다른 모델링 방법들과 합쳐서 사용한다면 유의미한 결과를 얻기 좋을 것 같다.
- Text classification을 더 이용해보고 싶었지만, 주제 상 해당 모델을 사용할 일이 적었다.
- 기후 변화에 대한 키워드로 여러 방법들을 시도해본 결과, 기후 변화에 대한 원인과 대책들에 대해서 추가적으로 얻은 정보들도 꽤 있다. 특히 아보카도라는 식품이 환경 파괴를 일으킨다는 것을 이번에 처음 알았다.
- 이번에 이용한 플랫폼은 네이버 api와 cnn news 인데, 다음엔 더 다양한 플랫폼에서 크롤링을 진행 해보고 싶다.