#가상 크롬드라이버를 불러옴

driver = webdriver.Chrome("C:\\basic\\chromedriver")

import konlpy

from konlpy.tag import Okt

from wordcloud import WordCloud

import matplotlib.pyplot as plt

from konlpy.tag import Twitter

from collections import Counter

import pandas as pd

import numpy as np

from selenium import webdriver

from tqdm import tqdm\_notebook

import requests

import pickle

import re

import ast

from bs4 import BeautifulSoup

from urllib.request import urlopen

import urllib

import time

import os

import sys

import urllib.request

import requests

################################ 기사 제목으로 워드클라우드 만들기

news\_data = []

page\_count = 20

client\_id = "6hpj2dSbrPbbzyW\_zffS"

client\_secret = "1hoQ1IuJ2N"

encText = urllib.parse.quote("keyword")

for idx in range(page\_count):

url = "https://openapi.naver.com/v1/search/news?query=" + encText+ "&start=" + str(idx \* 50 +1)

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):

result = requests.get(response.geturl(),

headers = {"X-Naver-Client-Id": client\_id,

"X-Naver-Client-Secret" : client\_secret}

)

news\_data.append(result.json())

else:

print("error code:" + rescode)

news\_title= dict.fromkeys(range(20))

for i in range(len(news\_data)):

news\_title[i] = []

for a in news\_data[i]['items']:

news\_title[i].append(a['title'])

title\_list= list(news\_title.values())

# 명사에 해당하는 단어만 모아서 리스트에 저장하기 위해 빈 리스트를 생성합니다.

noun\_list = []

for sentence in title\_list:

# 정규표현식 함수를 통해 문장에 포함된 영문, 숫자는 제외시킵니다.

sentence = re.sub('[a-zA-z0-9]','', str(sentence) )

# 명사(Noun)에 해당하는 단어만 noun\_list에 append합니다.

for word, tag in okt.pos(sentence):

if tag in ['Noun']: # Noun, Adjective, Verb 등등 조건을 바꿀 수 있음

# 한글자짜리 명사는 제외합니다.

if len(word) == 1:

continue

else:

noun\_list.append(word)

# Counter를 사용해서 글자 빈도수 뽑아내기

count = Counter(noun\_list)

words = dict(count.most\_common(n=50)) # 빈도수가 높은 상위 50개의 단어만 포함

# 워드클라우드 만들기

from wordcloud import WordCloud

import matplotlib.pyplot as plt

wordcloud = WordCloud(font\_path = "C:/Windows/Fonts/gulim.ttc",background\_color='white',colormap = 'rainbow\_r',width=800,height=800).generate\_from\_frequencies(words)

fig = plt.figure(figsize=(10, 10)) # 출력한 그래프 사이즈 설정

plt.imshow(wordcloud, interpolation="bilinear")

plt.axis('off')

plt.show()

#png로 결과 저장하기

#fig.savefig('삼성 wordcloud', bbox\_inches='tight', pad\_inches=0)

#########################연관검색어로 워드클라우드 만들기

def naver\_related\_search(keyword):

url = 'https://search.naver.com/search.naver?sm=top\_hty&fbm=1&ie=utf8&query=' + urlparser.quote(keyword)

response = requests.get(url)

if response.status\_code != 200:

return '실패'

else:

dom = BeautifulSoup(response.text, 'html.parser')

related\_search1 = dom.select('#nx\_footer\_related\_keywords > div > div.related\_srch > ul')

related\_search2 = dom.select('#nx\_right\_related\_keywords > div > div.related\_srch > ul')

result = [i.text.strip() for i in related\_search1] + [i.text.strip() for i in related\_search2]

return result

word = ['주식','재무제표','투자','주가','주식 투자','기업분석','코스피']

word\_result = []

for i in word:

a = naver\_related\_search(i)

word\_result.append(a)

word\_result = np.ravel(word\_result, order='C')

# 명사에 해당하는 단어만 모아서 리스트에 저장하기 위해 빈 리스트를 생성합니다.

noun\_list = []

for sentence in word\_result:

# 정규표현식 함수를 통해 문장에 포함된 영문, 숫자는 제외시킵니다.

sentence = re.sub('[a-zA-z0-9]','', str(sentence) )

# 명사(Noun)에 해당하는 단어만 noun\_list에 append합니다.

for word, tag in okt.pos(sentence):

if tag in ['Noun']: # Noun, Adjective, Verb 등등 조건을 바꿀 수 있음

# 한글자짜리 명사는 제외합니다.

if len(word) == 1:

continue

else:

noun\_list.append(word)

# Counter를 사용해서 글자 빈도수 뽑아내기

count = Counter(noun\_list)

words = dict(count.most\_common(n=50)) # 빈도수가 높은 상위 50개의 단어만 포함

# 워드클라우드 만들기

from wordcloud import WordCloud

import matplotlib.pyplot as plt

wordcloud = WordCloud(font\_path = "C:/Windows/Fonts/gulim.ttc",background\_color='white',colormap = 'rainbow\_r',width=800,height=800).generate\_from\_frequencies(words)

fig = plt.figure(figsize=(10, 10)) # 출력한 그래프 사이즈 설정

plt.imshow(wordcloud, interpolation="bilinear")

plt.axis('off')

plt.show()

#png로 결과 저장하기

#fig.savefig('wordcloud\_2', bbox\_inches='tight', pad\_inches=0)