NAVER 데이터랩 크롤링

주제 설정

- 데이터 분석 용이
- 현재 트렌드 분석
- 키워드 별 대중 관심도 분석

라이브러리 사용

- import pandas as pd
- import urllib.request
- import json
- import matplotlib.pyplot as plt
- from konlpy.tag import Komoran
- import nltk

API 추출

```
url = "https://openapi.naver.com/v1/datalab/search"

body = "{\
        \"startDate\":\"2023-01-01\",\
        \"endDate\":\"2023-04-06\",\
        \"timeUnit\":\"date\",\
        \"keywordGroups\":[{\"groupName\":\"파이썬\",\"keywords\":[\"파이썬\",\"python\",\"python홈\",\"python공부\",\"pycharm\",\"jupyterNotebook\",\"주피터노트북\",\"python라이
                           {\"groupName\":\"자바\",\"keywords\":[\"자바\",\"java\"]},\
                           {\"groupName\":\"자바스크립트\",\"keywords\":[\"javascript\",\"Node.js\"]},\
                           {\"groupName\":\"데이터베이스\",\"keywords\":[\"데이터베이스\",\"sql\",\"oracle\"]}\
                          ],\
        \"device\":\"mo\",\
        \"ages\":[\"3\",\"5\"],\
        \"gender\":\"m\"\
        }";

request = urllib.request.Request(url)
request.add_header("X-Naver-Client-Id",client_id)
request.add_header("X-Naver-Client-Secret",client_secret)
request.add_header("Content-Type","application/json")

response = urllib.request.urlopen(request, data=body.encode("utf-8"))

rescode = response.getcode()

if(rescode==200):
    response_body = response.read()
    response_data = response_body.decode('utf-8')
else:
    print("Error Code:" + rescode)

result = json.loads(response_data)

print(result)

date = [a['period'] for a in result['results'][0]['data']]
ratio_data1 = [a['ratio'] for a in result['results'][0]['data']]
ratio_data2 = [a['ratio'] for a in result['results'][1]['data']]
ratio_data3 = [a['ratio'] for a in result['results'][2]['data']]
ratio_data4 = [a['ratio'] for a in result['results'][3]['data']]

datalab = pd.DataFrame({'date':date,
                        'searching_result':ratio_data1,
                        'searching_result2':ratio_data2,
                        'searching_result3':ratio_data3,
                        'searching_result4':ratio_data4})

datalab
```

	date	searching_result	searching_result2	searching_result3	searching_result4
0	2023-01-01	22.73972	7.94520	0.82191	17.53424
1	2023-01-02	37.26027	11.23287	4.38356	19.17808
2	2023-01-03	30.41095	12.32876	4.10958	19.72602
3	2023-01-04	31.78082	11.23287	4.93150	23.83561
4	2023-01-05	30.95890	10.13698	3.28767	20.27397
...	...	...	...	...	...
91	2023-04-02	25.20547	5.20547	3.01369	16.43835
92	2023-04-03	44.93150	10.95890	6.30136	23.01369
93	2023-04-04	43.56164	12.05479	5.20547	26.84931
94	2023-04-05	36.43835	12.32876	3.01369	27.67123
95	2023-04-06	40.00000	11.23287	4.65753	21.64383

96 rows × 5 columns

columns 변환

```
datalab.columns
```

```
Index(['date', 'searching_result', 'searching_result2', 'searching_result3',
       'searching_result4'],
      dtype='object')
```

```
datalab = datalab.rename(columns={'date': '검색날짜',
                                  'searching_result': '검색결과1',
                                  'searching_result2': '검색결과2',
                                  'searching_result3': '검색결과3',
                                  'searching_result4': '검색결과4'})
datalab
```

	검색날짜	검색결과1	검색결과2	검색결과3	검색결과4
0	2023-01-01	22.73972	7.94520	0.82191	17.53424
1	2023-01-02	37.26027	11.23287	4.38356	19.17808
2	2023-01-03	30.41095	12.32876	4.10958	19.72602
3	2023-01-04	31.78082	11.23287	4.93150	23.83561
4	2023-01-05	30.95890	10.13698	3.28767	20.27397
...	...	...	...	...	...
91	2023-04-02	25.20547	5.20547	3.01369	16.43835
92	2023-04-03	44.93150	10.95890	6.30136	23.01369
93	2023-04-04	43.56164	12.05479	5.20547	26.84931
94	2023-04-05	36.43835	12.32876	3.01369	27.67123
95	2023-04-06	40.00000	11.23287	4.65753	21.64383

96 rows × 5 columns

```
datalab.columns
```

```
Index(['검색날짜', '검색결과1', '검색결과2', '검색결과3', '검색결과4'], dtype='object')
```

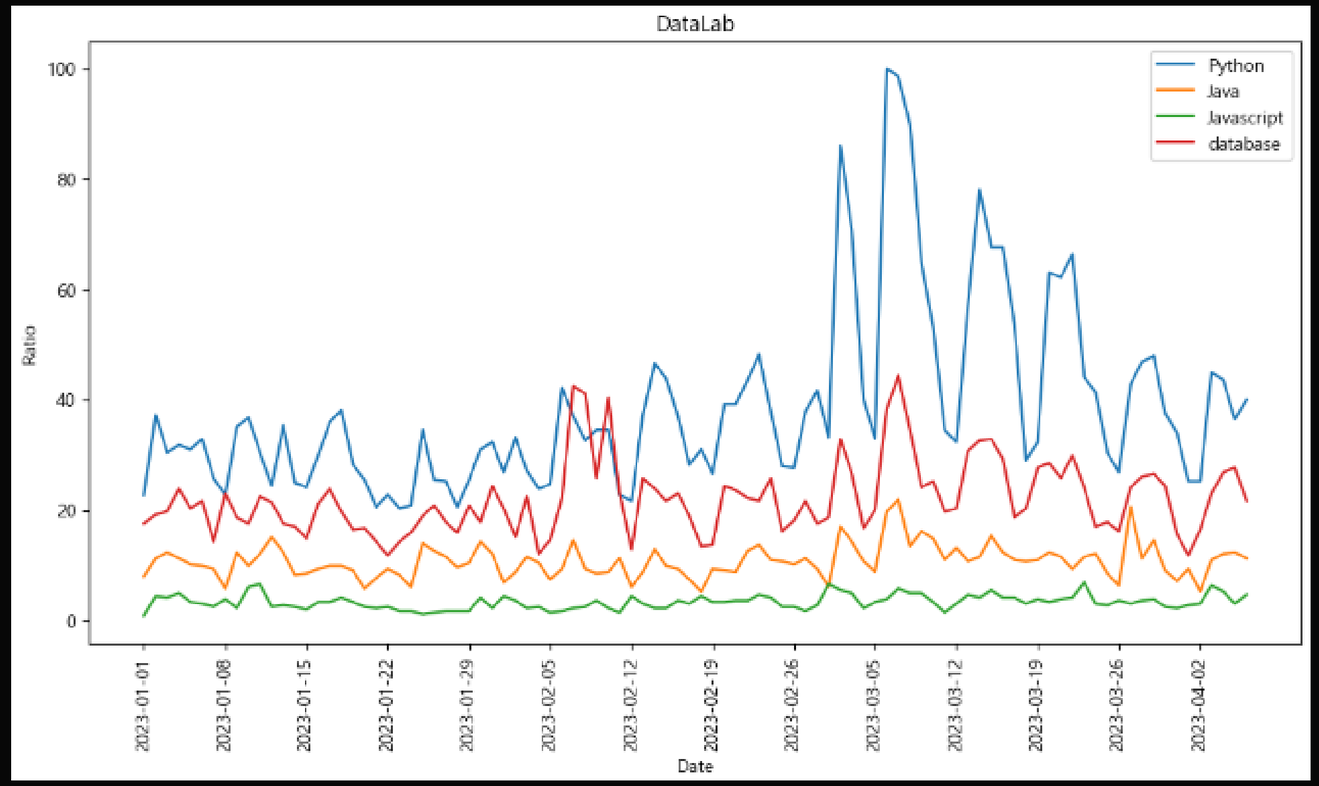
.csv로 가공 및 저장

```
datalab.to_csv('./../data/naver.csv')

print('파일 저장 완료')
```

```
파일 저장 완료
```

2023년 검색량

```python
import matplotlib.pyplot as plt

# 그래프 설정
plt.figure(figsize=(12,6))
plt.title("DataLab")
plt.xlabel("Date")
plt.ylabel("Ratio")

# 그래프 그리기
plt.plot(date, ratio_data1, label='Python')
plt.plot(date, ratio_data2, label='Java')
plt.plot(date, ratio_data3, label='Javascript')
plt.plot(date, ratio_data4, label='database')

# x축 설정 가독성을 위해 7일 간격으로 설정
plt.xticks(date[::7], rotation='vertical')

# 범례 추가
plt.legend(loc='upper right')

# 그래프 출력
plt.show()
```



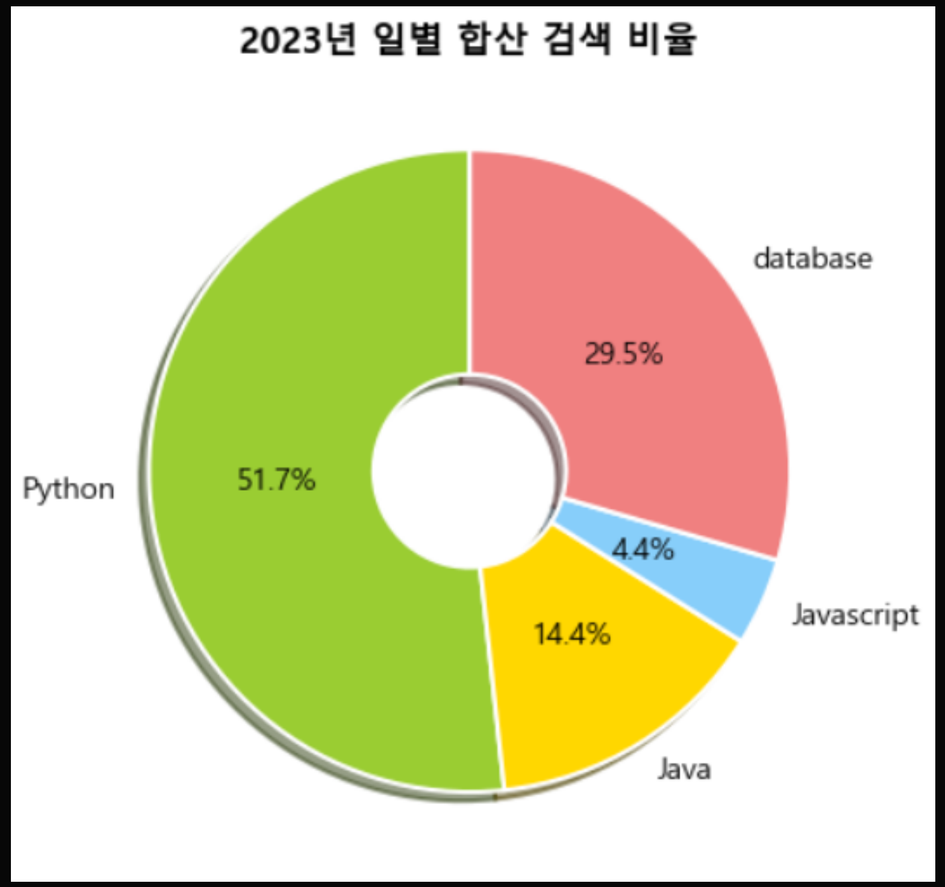
일별 합산 검색량

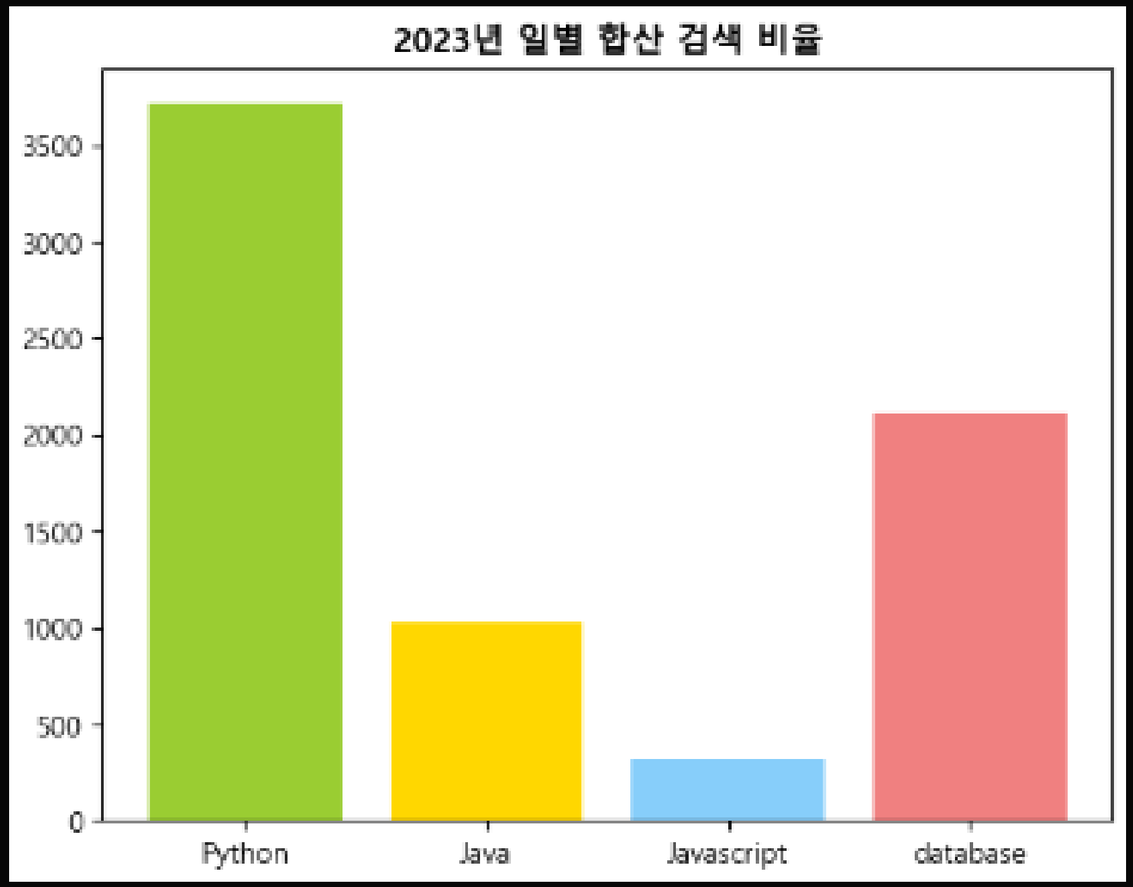
```python
import pandas as pd
import matplotlib.pyplot as plt
plt.rc('font', family='Malgun Gothic')

# 데이터 불러오기
df = pd.read_csv('./../data/naver.csv', sep=',')

df = df.set_index('검색날짜')

total1 = df['검색결과1'].sum()
total2 = df['검색결과2'].sum()
total3 = df['검색결과3'].sum()
total4 = df['검색결과4'].sum()

totals = [total1, total2, total3, total4]

# 각 항목별 색상 설정
colors = ['yellowgreen', 'gold', 'lightskyblue', 'lightcoral']
```

```python
# 파이차트 그리기
labels = ['Python', 'Java', 'Javascript', 'database']
wedgeprops = {'width': 0.7, 'edgecolor': 'w', 'linewidth': 1.5}
plt.pie(totals, labels=labels, colors=colors, wedgeprops=wedgeprops, autopct='%1.1f%%', shadow=True, startangle=90)
plt.title('2023년 일별 합산 검색 비율', fontweight='bold')
plt.show()
```

```python
# 바 차트 그리기
labels = ['Python', 'Java', 'Javascript', 'database']
values = [total1, total2, total3, total4]
colors = ['yellowgreen', 'gold', 'lightskyblue', 'lightcoral']

plt.bar(labels, values, color=colors)
plt.title('2023년 일별 합산 검색 비율', fontweight='bold')

plt.show()
```

시각화


2023년 일별 합산 검색 비율
database
29.5%
Python
51.7%
4.4%
Javascript
14.4%
Java


2023년 일별 합산 검색 비율
3500
3000
2500
2000
1500
1000
500
0
Python
Java
Javascript
database

def 활용

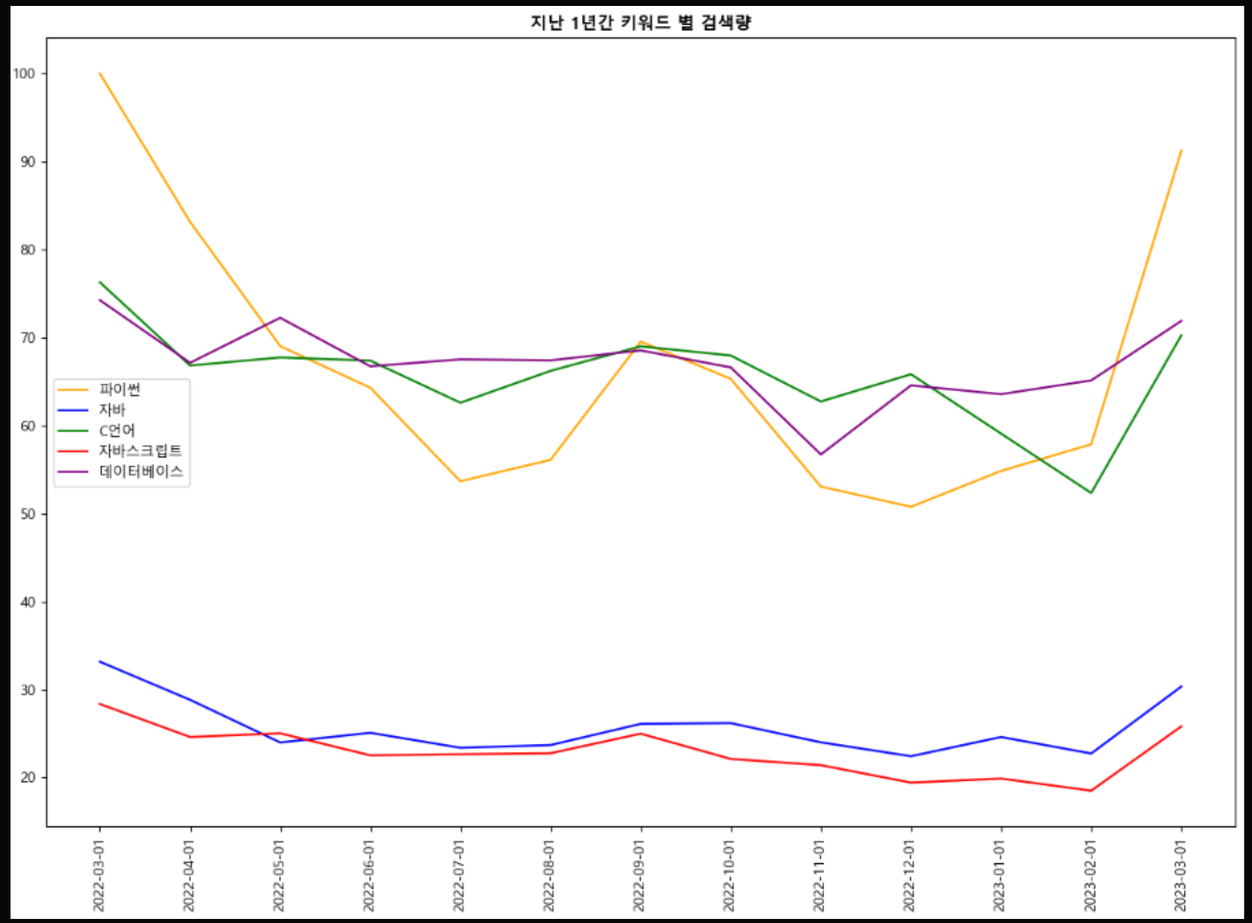
```
url = "https://openapi.naver.com/v1/datalab/search"

body = "{\
        \"startDate\":\"2023-01-01\",\
        \"endDate\":\"2023-04-06\",\
        \"timeUnit\":\"date\",\
        \"keywordGroups\":[{\"groupName\":\"파이썬\",\"keywords\":[\"파이썬\",\"python\",\"python특징\",\"python공부\",\"pycharm\",\"jupyterNotebook\",\"주피터노트북\",\"python라이
                           {\"groupName\":\"자바\",\"keywords\":[\"자바\",\"java\"]},\
                           {\"groupName\":\"자바스크립트\",\"keywords\":[\"javascript\",\"Node.js\"]},\
                           {\"groupName\":\"데이터베이스\",\"keywords\":[\"데이터베이스\",\"sql\",\"oracle\"]}\
                          ],\
        \"device\":\"mo\",\
        \"ages\":[\"3\",\"5\"],\
        \"gender\":\"m\"\
        }";

request = urllib.request.Request(url)
request.add_header("X-Naver-Client-Id",client_id)
request.add_header("X-Naver-Client-Secret",client_secret)
request.add_header("Content-Type","application/json")

response = urllib.request.urlopen(request, data=body.encode("utf-8"))

rescode = response.getcode()

if(rescode==200):
    response_body = response.read()
    response_data = response_body.decode('utf-8')
else:
    print("Error Code:" + rescode)

result = json.loads(response_data)

print(result)

date = [a['period'] for a in result['results'][0]['data']]
ratio_data1 = [a['ratio'] for a in result['results'][0]['data']]
ratio_data2 = [a['ratio'] for a in result['results'][1]['data']]
ratio_data3 = [a['ratio'] for a in result['results'][2]['data']]
ratio_data4 = [a['ratio'] for a in result['results'][3]['data']]

datalab = pd.DataFrame({'date':date,
                        'searching_result':ratio_data1,
                        'searching_result2':ratio_data2,
                        'searching_result3':ratio_data3,
                        'searching_result4':ratio_data4})

datalab
```

```
def getresult(startDate, endDate, timeUnit, keywordGroups, device, gender, ages):
    url = "https://openapi.naver.com/v1/datalab/search";

    body_dict = {}  # 검색 정보를 저장할 변수
    body_dict['startDate'] = startDate
    body_dict['endDate'] = endDate
    body_dict['timeUnit'] = timeUnit
    body_dict['keywordGroups'] = keywordGroups
    body_dict['device'] = device
    body_dict['gender'] = gender
    body_dict['ages'] = ages

    body = str(body_dict).replace("'", '"')  # ' 문자로는 에러가 발생해서 " 로 변환
```

```
client_id = "W3fPWgvUru_vubkZQOUV"
client_secret = "a55vs5QmYf"

startDate='2022-03-01'
endDate='2023-03-31'
timeUnit='month'  #'date','week','month'
keywordGroups=[ # 메인 키워드 1개 / 세부 키워드 20개까지 가능
    {'groupName':'파이썬', 'keywords':['파이썬','파이썬크롤링','파이썬프로젝트']},
    {'groupName':'자바', 'keywords':['자바','자바 연산자','자바 기초','java']},
    {'groupName':'C언어', 'keywords':['C#','C','C++']},
    {'groupName':'자바스크립트', 'keywords':['javascript','Node.js','자바스크립트']},
    {'groupName':'데이터베이스', 'keywords':['database','oracle','sql','오라클']}
]
device='mo'  #'pc','mo'
gender='m'   #'m','f'
ages=['3','4','5']  #1: 0~12세, 2: 13~18세, 3: 19~24세, 4: 25~29세,
#5: 30~34세, 6: 35~39세, 7: 40~44세, 8: 45~49세, 9: 50~54세, 10: 55~59세, 11: 60세 이상

getresult(startDate,endDate,timeUnit,keywordGroups,device,gender,ages)
```

꺽은선 그래프

```python
# 결과데이터중 'data' 와 'title'만 따로 DataFrame으로 저장
response_results = pd.DataFrame()
for data in response_json['results']:
    result = pd.DataFrame(data['data'])
    result['title'] = data['title']

    response_results = pd.concat([response_results, result])
    response_results.to_csv('./../data/naverdata.csv')

print('파일이 저장되었습니다.')

# title별로 그래프를 그리기 위한부분
titles = response_results['title'].unique()

 # 그래프 컬러 설정
colors = ['orange', 'blue', 'green', 'red', 'purple']

plt.figure(figsize=(15, 10))

for i, title in enumerate(titles):
    data = response_results.loc[response_results['title'] == title, :]

    plt.plot(data['period'], data['ratio'], label=title, color=colors[i % len(colors)])

    # plt.bar(data['period'], data['ratio'], label=title, color=colors[i % len(colors)])

    plt.plot(linewidth=5)

    plt.xticks(rotation=90)
    plt.legend()
    plt.title('지난 1년간 키워드 별 검색량', fontweight='bold')
plt.show()
```



word cloud

```
dataInFolder = './../data/'
filename = dataInFolder + 'Programming Language.txt'

language = open(filename, encoding='UTF-8').read()

print(language)

# url : https://konlpy.org/ko/latest/
from konlpy.tag import Komoran

# 사용자 정의 단어들을 정하시고, 사용자 정의 사전에 추가하도록 합니다.
lan_dict = dataInFolder + 'lan_dic.txt' # 사용자 정의 사전
komo = Komoran(userdic=lan_dict) # 객체 생성
token_list = komo.nouns(language) # nouns : 명사 추출

print('토큰 목록')
print(token_list)

# 불용어(stopword) : 빈도는 많지만, 분석시 중요하지 않다고 판단하는 단어들
stop_dict = dataInFolder + 'stop_dict.txt' # 불용어 파일
stop_file = open(stop_dict, encoding='UTF-8').readlines()
stop_words = [word.strip() for word in stop_file]

print('불용어 리스트')
print(stop_words)

new_token_list = [word for word in token_list if word not in stop_words]
print('불용어 제외된 토큰 목록')
print(new_token_list)

# 집합을 이용하여 불용어로 처리된 내용을 확인합니다.
set_token_list = set(token_list)
set_new_token_list = set(new_token_list)
diff = set_token_list.difference(set_new_token_list)
print('불용어 처리된 단어 확인')
print(diff)
```

```
import nltk

nltk_token = nltk.Text(tokens=new_
bindo_size = 500 # 출현 빈도 수
token_data = nltk_token.vocab().mo
print(
print(
wordli
# 단어
for wo
    if
print(
print(
```

```
# token_data 파일로 저장하기
import pandas as pd
savedWordFile = dataInFolder + 'word_list.csv'
                                                                단어','빈도수'])
                                                                49', index=False)

import numpy as np
from PIL import Image # PIL : Python Image Library
from wordcloud import WordCloud

alice_color_file = dataInFolder + 'python02.png' # 워드 클라우드가 그려질 이미지
alice_color_array = np.array(Image.open(alice_color_file)) # 이미지 배열

word_dict = dict(wordlist) # 단어 사전
                                                                :barcount]

font_name = 'malgun.ttf' # 글꼴

mycloud = WordCloud(font_path=font_name, mask=alice_color_array, background_color='white')
mycloud = mycloud.generate_from_frequencies(word_dict)
                                                                , use_index=True, legend=False)
                                                                 단어', size=20)

# ImageColorGenerator : 컬러 이미지의 색상 톤을 유지하고자 할 때 사용되는 라이브러리
from wordcloud import ImageColorGenerator
color_generator = ImageColorGenerator(alice_color_array)

mycloud = mycloud.recolor(color_func=color_generator)

plt.figure(figsize=(25,25)) # 새 도화지 준비
plt.axis('off') # 그래프 테두리 없애기
plt.imshow(mycloud)
                                                                loud/index.html

cloudFileName = dataInFolder + 'word_cloud.png'
plt.savefig(cloudFileName)
print(cloudFileName + '그래프가 생성되었습니다.')

print('finished')
```


출처 필요
자료형
연결
영역
세대
고급 프로그래밍 언어
절대
컴퓨터
발전
동작
정의
선언
어셈블리어
내부
표준
이후
언어
지시
운영 체제
고급
구현
기계어
정보
레벨
이용
정수형
프로그래머
지원
함수
의미
사용
메모리
저장
픽셀
구조
배열
본문
델파이
유닉스
제공
초기
실행
프로그래밍 언어
특정
프로그램
인간
최초
일반
사각형
단위
데이터
개발
높이
프로그래밍 언어의 역사
너비
형식
사용자
작성
현재
알고리즘
부분
필요
C++
모듈
관련
상위
바이트

Q & A

감사합니다