



텍스트마이닝

텍스트마이닝(text mining)

- 문자로 된 데이터에서 가치 있는 정보를 얻어 내는 분석기법
 - 형태소 분석: 가장 먼저하는 작업, 문장을 구성하는 어절들이 어떤 품사인지 파악
 - 어절의 품사를 파악한 후 명사, 동사, 형용사 등 의미를 지닌 품사를 추출, 단어 사용확인



KoNLPy 라이브러리 설치하기

■ 설치 기본 사항 확인하기

LG PC	
장치 이름	DESKTOP-B1R4JPJ
프로세서	Intel(R) Pentium(R) CPU 4415U @ 2.30GHz 2.30 GHz
설치된 RAM	4.00GB(3.86GB 사용 가능)
장치 ID	2A0AD030-B964-47BD-9C05-3E11A7984101
제품 ID	00325-80109-77037-AAOEM
시스템 종류	64비트 운영 체제, x64 기반 프로세서
펜 및 터치	이 디스플레이에 사용할 수 있는 펜 또는 터치 식 입력이 없습니다.

■ 아나콘다 파이썬의 버전 확인 : 아나콘다 프롬프트 창에서 확인

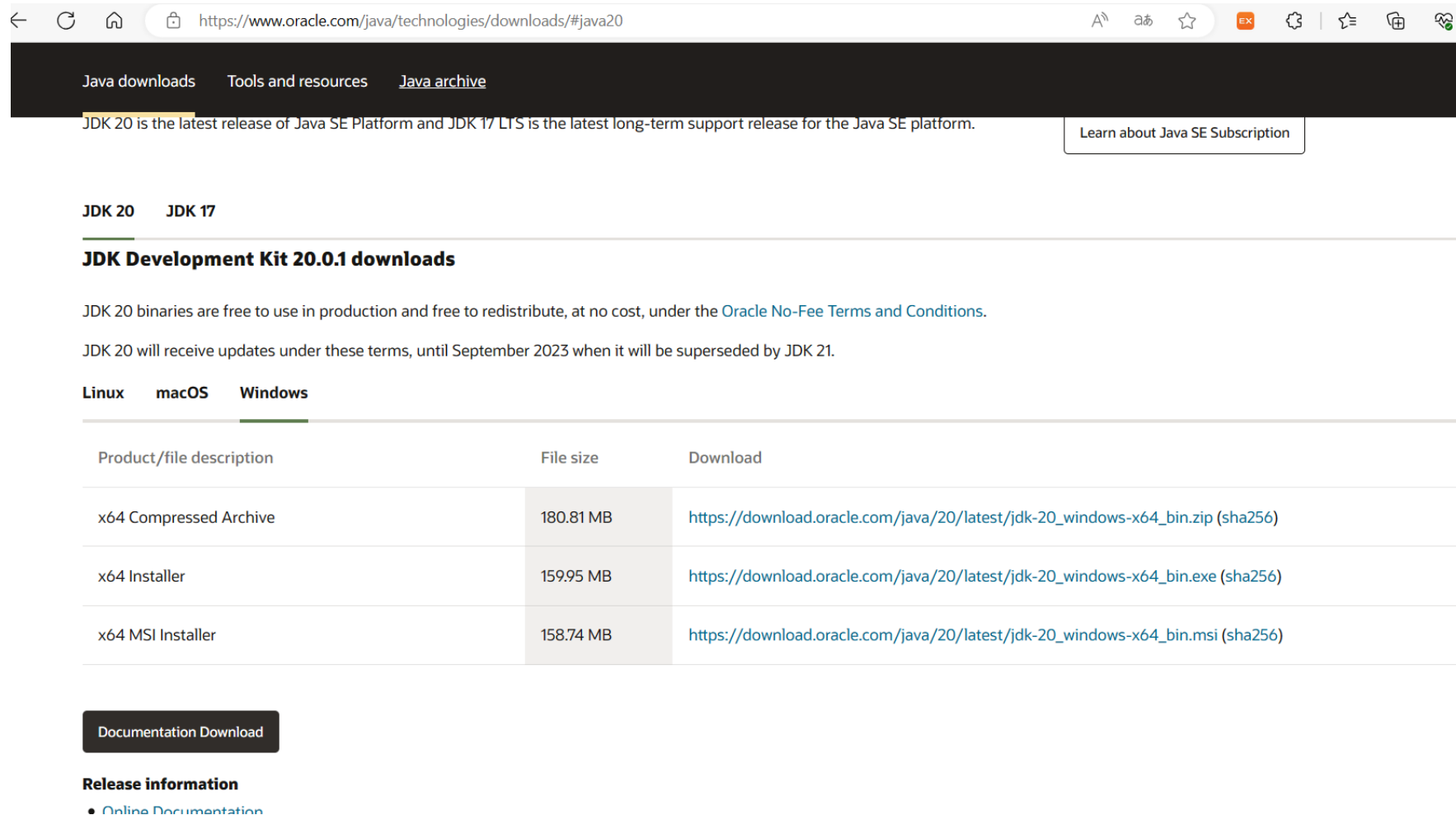
```
Anaconda Prompt (anaconda3)

(base) C:\Users\LG>python --version
Python 3.8.8

(base) C:\Users\LG>
```

KoNLPy 라이브러리 설치하기

■ JDK설치하기



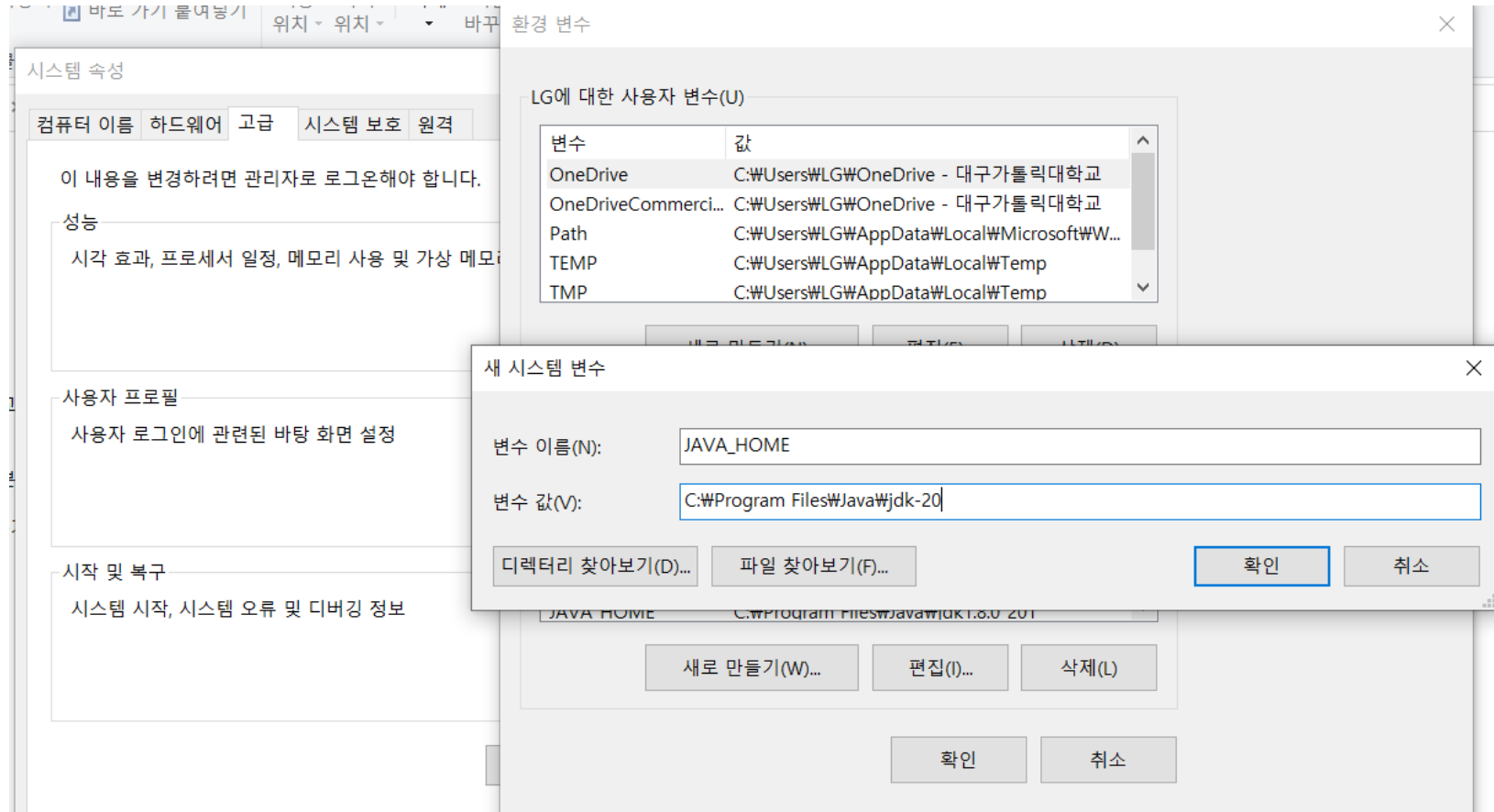
The screenshot shows the Oracle Java Downloads page for JDK 20. The browser address bar displays the URL <https://www.oracle.com/java/technologies/downloads/#java20>. The page features a navigation bar with links for "Java downloads", "Tools and resources", and "Java archive". A banner at the top states: "JDK 20 is the latest release of Java SE Platform and JDK 17 LTS is the latest long-term support release for the Java SE platform." Below this, there are tabs for "JDK 20" and "JDK 17", with "JDK 20" selected. The main heading is "JDK Development Kit 20.0.1 downloads". A paragraph explains that JDK 20 binaries are free to use in production and free to redistribute, at no cost, under the [Oracle No-Fee Terms and Conditions](#). It also notes that JDK 20 will receive updates under these terms, until September 2023 when it will be superseded by JDK 21. Below this, there are tabs for "Linux", "macOS", and "Windows", with "Windows" selected. A table lists the available download options for Windows:

Product/file description	File size	Download
x64 Compressed Archive	180.81 MB	https://download.oracle.com/java/20/latest/jdk-20_windows-x64_bin.zip (sha256)
x64 Installer	159.95 MB	https://download.oracle.com/java/20/latest/jdk-20_windows-x64_bin.exe (sha256)
x64 MSI Installer	158.74 MB	https://download.oracle.com/java/20/latest/jdk-20_windows-x64_bin.msi (sha256)

Below the table, there is a button labeled "Documentation Download". At the bottom, under the heading "Release information", there is a link for "Online Documentation".

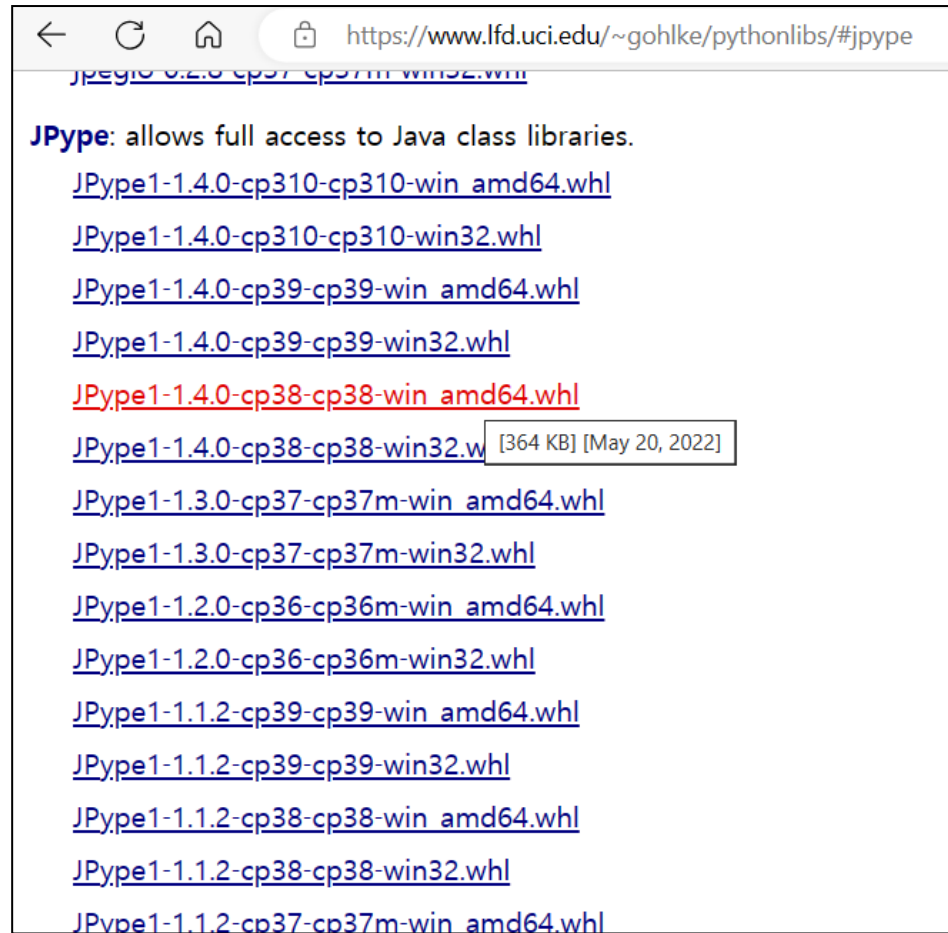
KoNLPy 라이브러리 설치하기

■ JAVA_HOME 설정하기



KoNLPy 라이브러리 설치하기

- JType1 설치하기: 파이썬에서 JDK 사용하기 위해 JType.whl 파일 설치
 - 아나콘다 버전과 윈도우 버전 일치하는 파일을 선택



KoNLPy 라이브러리 설치하기

■ 다운로드한 파일을 소스파일이 있는 폴더로 이동 후 설치 실행

```
(base) C:\Users\LG\PycharmProjects\source\My_Python>pip install JPype1-1.4.0-cp38-cp38-win_amd64.whl
Processing c:\Users\LG\PycharmProjects\source\my_python\jpype1-1.4.0-cp38-cp38-win_amd64.whl
Installing collected packages: JPype1
  Attempting uninstall: JPype1
    Found existing installation: JPype1 1.3.0
    Uninstalling JPype1-1.3.0:
      Successfully uninstalled JPype1-1.3.0
Successfully installed JPype1-1.4.0
```

■ koNLPy설치하기

```
(base) C:\Users\LG\PycharmProjects\source\My_Python>pip install koNLPy
Collecting koNLPy
  Using cached konlpy-0.6.0-py2.py3-none-any.whl (19.4 MB)
Requirement already satisfied: JPype1>=0.7.0 in c:\Users\LG\anaconda3\lib\site-packages (from koNLPy) (1.4.0)
Requirement already satisfied: lxml>=4.1.0 in c:\Users\LG\anaconda3\lib\site-packages (from koNLPy) (4.6.3)
Requirement already satisfied: numpy>=1.6 in c:\Users\LG\anaconda3\lib\site-packages (from koNLPy) (1.20.1)
Installing collected packages: koNLPy
Successfully installed koNLPy-0.6.0
```

KoNLPy 라이브러리 설치하기

■ 설치가 안될 경우

파이썬에서 한글 자연어 처리를 위해서 KoNLPy 패키지를 설치 및 사용하려고 합니다.

해당 패키지를 사용하기 위해서는 JDK(JAVA), JPyype1, KoNLPy의 순차적인 설치가 필요한데,
위 라이브러리 및 패키지를 모두 설치하고 코드를 실행시켰을 때
아나콘다 환경에서는 종종 다음과 같은 오류가 발생하기도 합니다.

ValueError: No JVM shared library file (jvm.dll) found. Try setting up the JAVA_HOME environment variable properly.

솔루션은 JDK 설치 시 설정했던 **JAVA_HOME** 환경변수를 제대로 인식해주도록 하는 것인데,
아무리 환경변수를 제대로 설정하더라도 끝까지 인식을 못하는 경우가 많더군요.

그래서 아예 **jpyype**의 **jvm**을 찾는 코드에서 직접 **JDK**의 경로를 넣어주도록 하였습니다.

1. 디렉토리 찾아가기

C:\ProgramData\Anaconda3\Lib\site-packages\jpyype_jvmfinder.py

jpyype 패키지의 jvm을 찾는 소스코드는 보통 다음과 같은 디렉토리 경로에 위치해있습니다.
종종 숨김 폴더 옵션 때문에 Program Data가 보이지 않을 수도 있으니, 그럴땐 숨김 폴더 옵션을
해제해주시면 됩니다.

KoNLPy 라이브러리 설치하기

2. 소스코드 수정

```
def _get_from_java_home(self):  
    """  
    Retrieves the Java library path according to the JAVA_HOME environment  
    variable  
  
    Returns:  
    | The path to the JVM library, or None  
    """  
    # Get the environment variable  
    #java_home = os.getenv("JAVA_HOME")  
    java_home = "C:\\Program Files\\Java\\jdk-14.0.1"  
    if java_home and os.path.exists(java_home):  
        # Get the real installation path  
        java_home = os.path.realpath(java_home)  
  
        if not os.path.exists(java_home):  
            java_home = os.getenv("JAVA_HOME")  
  
    # Look for the library file  
    return self.find_libjvm(java_home)
```

- 1) 해당 소스코드를 오픈한 후 `_get_from_java_home()` 메소드를 찾습니다.
- 2) 기존에 있던 `JAVA_HOME` 환경 변수로 `JDK` 경로를 찾던 코드를 주석처리 해줍니다.
- 3) 직접 `JDK` 설치 경로를 입력해줍니다. 이때 `os.getenv()`를 사용하는 것이 아닌, 문자열 형태 그대로 입력하는 것이 중요합니다. `os.getenv()` 메소드를 이용하면 여전히 계속 오류가 발생하기도 하기 때문입니다.

텍스트 마이닝

- 예제 불러오기(오바마대통령 연설문)
 - 한글의 문서파일을 위해 UTF-8로 지정하기

```
In [24]: 1 obama=open('Obama.txt',encoding='UTF-8').read() #Obama연설문 불러오기
          2 obama
```

Out[24]: 'WuffeffBarack Hussein Obama 대통령 당선 연설문\n\n아직도 미국이 무한한 가능성의 나라라는 것을 의심하는 사람이 있다면, 아직도 이 나라의 선조들이 꾸었던 꿈들이 살아있는가에 대한 의문을\n\n품은 사람이 있다면, 그리고 민주주의의 힘을 믿지 못하는 사람들이\n\n있다면, 바로 오늘 밤 여러분이 그 답을 보여줬습니다.\n\n투표소였던 학교와 교회들을 휘감았던 긴 줄들, 역사상 유례 없던 최다 투표율, 세 시간이고 네 시간이고 투표하기 위해 기다렸던 사람들; 바로 지금이 변화의 시기이며 자신의 목소리가 바로 그 변화라는 굳은 믿음 하에 인생 처음으로 투표했던 사람들, 이 모두가 사람들이 품었던 의문들에 대한 답입니다.\n\n젊은이, 늙은이, 빈자, 부자, 민주당, 공화당, 흑인, 백인, 라틴계 미국인, 동양인, 아메리카 인디언, 동성애자, 이성애자, 장애를 가진 자들, 장애가 없는 자들 - 우리 모두가 사람들이 품었던 의문들에 답했습니다.\n\n오늘은 세계에 미국은 단순한 붉은 주(공화당)와 푸른 주(민주당)의 집합이 아닌 통일된 미국이라는 것을 알리는 전보와도 같았습니다.\n\n오늘은 우리가 이를 수 있는 일들에 대해 조금 더 냉소적이 되어야 한다고, 걱정해야 한다고, 그리고 우리가 가진 것에 대해 의심을 품어야 한다고 계속하여 세뇌 당했던 평범한 자들 마저 역사의 기다란 호에 손을\n\n얹어 미래에 대한 희망을 향해 그 길을 꺾은 날입니다. 이것이 우리의\n\n답입니다. \n\n이 길에 오기까지는 오랜 시간이 걸렸지만, 우리가 이 중요한 시기에\n\n오늘 밤 선거에서 내린 결정 때문에 미국은 변화 할 것입니다.\n\n저는 방금 맥케인 의원님께 굉장히 기쁜 소식을 받았습니다.\n\n그는 이 캠페인에서 오랫동안 열심히 싸워 주셨으며, 그가 사랑하는\n\n이 국가를 위해서는 더욱이나 오랫동안 열심히 싸워 주셨습니다. 그는 국가를 위해 우리가 상상도 할 수 없는 희생을 하셨으며, 우리는 맥케인 의원님 같은 분들의 용기와 사심 없는 지도력 때문에 훨씬 살기 좋은\n\n국가 되었습니다. 저는 그와 페일런 부지사가 이루었던 모든 업적을 축하하고 싶습니다.

텍스트 마이닝

- 불필요한 문자 제거하기
 - 특수문자, 공백 등 분석대상이 아닌 문자를 공백으로 바꾸기

```
In [25]: 1 import re                #불필요한 문자 제거를 위한 문자처리 패키지 불러오기
          2 obama=re.sub('[^가-힣]', ' ', obama) #한글이 아닌 모든 문자를 의하는 규칙적용하여 제거하기(정규표현식)
          3 obama
```

```
Out[25]: ' 대통령 당선 연설문 아직도 미국이 무한한 가능성의 나라라는 것을 의심하는 사람이 있다면 아직도 이 나라의 선조들이
꾸었던 꿈들이 살아있는가에 대한 의문을 품은 사람이 있다면 그리고 민주주의의 힘을 믿지 못하는 사람들이 있다면 바로 오늘밤 여러분이 그
답을 보여줬습니다 투표소였던 학교와 교회들을 휘감았던 긴 줄들 역사상 유례 없던 최다 투표율 세 시간이고 네 시간이고 투표하기 위해 기
다렸던 사람들 바로 지금이 변화의 시기이며 자신의 목소리가 바로 그 변화라는 굳은 믿음 하에 인생 처음으로 투표했던 사람들 이 모두가 사람
들이 품었던 의문들에 대한 답입니다 젊은이 늙은이 빈자 부자 민주당 공화당 흑인 백인 라틴계 미국인 동양인 아메리카 인디언 동
성애자 이성애자 장애를 가진 자들 장애가 없는 자들 우리 모두가 사람들이 품었던 의문들에 답했습니다 오늘은 세계에 미국은 단순한 붉
은 주 공화당 와 푸른 주 민주당 의 집합이 아닌 통일된 미국이라는 것을 알리는 전보와도 같았습니다 오늘은 우리가 이룰 수 있는 일들에 대해
조금 더 냉소적이 되어야 한다고 걱정해야 한다고 그리고 우리가 가진 것에 대해 의심을 품어야 한다고 계속하여 세뇌 당했던 평범한 자들 마저
역사의 기다란 호에 손을 얹어 미래에 대한 희망을 향해 그 길을 걷은 날입니다 이것이 우리의 답입니다 이 길에 오기까지는 오랜 시간이
걸렸지만 우리가 이 중요한 시기에 오늘 밤 선거에서 내린 결정 때문에 미국은 변화 할 것입니다 저는 방금 맥케인 의원님께 굉장히 기쁨 있
는 전화를 받았습니다 그는 이 캠페인에서 오랫동안 열심히 싸워 주셨으며 그가 사랑하는 이 국가를 위해서는 더욱이나 오랫동안 열심히 싸워
주셨습니다 그는 국가를 위해 우리가 상상도 할 수 없는 희생을 하셨으며 우리는 맥케인 의원님 같은 분들의 용기와 사심 없는 지도력 때문에
```

텍스트 마이닝

- 명사 추출하기

- 형태소 중 명사를 추출하여 내용파악, 분석 단위로 사용

```
In [26]: 1 #명사를 추출하기
          2 import konlpy
          3 hannanum=konlpy.tag.Hannanum()
          4
          5 nouns=hannanum.nouns(obama)
          6 nouns
```

```
Out[26]: ['대통령',
          '당선',
          '연설문',
          '미국',
          '무한',
          '가능성',
          '것',
          '의심',
          '사람',
          '나라',
          '선조들',
          '꿈',
          '살아있는가',
          '의문',
          '사람',
          '민주주의',
          '힘',
          '사람',
          '오늘밤',
          '정리하다']
```

텍스트 마이닝

- 처리 단위 변환(데이터프레임으로 변환)

```
In [27]: 1 import pandas as pd
          2 df_word=pd.DataFrame({'word': nouns}) #데이터프레임으로 변환하기
          3 df_word
```

Out[27]:

	word
0	대통령
1	당선
2	연설문
3	미국
4	무한
...	...
1049	가호
1050	당신
1051	가정
1052	미합중국
1053	바

1054 rows × 1 columns

텍스트 마이닝

- 단어 빈도표 만들기
 - 빈도표 중 단어 글자수를 제한하여 의미없는 경우 제거

```
In [28]: 1 df_word['count']=df_word['word'].str.len() #글자수 카운트하여 열 추가하기
          2 df_word
```

Out[28]:

	word	count
0	대통령	3
1	당선	2
2	연설문	3
3	미국	2
4	무한	2
...
1049	가호	2
1050	당신	2
1051	가정	2
1052	미합중국	4
1053	바	1

1054 rows × 2 columns

```
In [29]: 1 df_word=df_word.query('count >=2') #두글자 이상만 표현하기
          2 df_word.sort_values('count')
```

Out[29]:

	word	count
417	용감	2
564	안주	2
563	예전	2
562	우리	2
560	기회	2
...
132	오랫동안	4
12	살아있는가	5
973	미국인들이	5
253	엑슬로드에	5
412	아프가니스탄	6

744 rows × 2 columns

텍스트 마이닝

- 단어의 사용빈도를 구하여 빈도순으로 정렬

```
In [33]: 1 # 단어별 분리한 것을 빈도수에 따라 내림차순 정렬하기
          2 df_word=df_word.groupby('word',as_index=False)##
          3 .agg(n=('word', 'count'))##
          4 .sort_values('n',ascending=False)
          5 df_word
```

Out[33]:

	word	n
238	우리	65
133	미국	18
49	그녀	16
230	오늘	15
218	여러분	14
...
150	백인	1
147	반대	1
146	바이든에	1
145	믿음	1
197	시련	1

394 rows × 2 columns

텍스트 마이닝

- 단어 빈도 막대그래프 만들기(seaborn 차트)
 - 사용빈도 중 상위 20개를 추출

```
In [34]: 1 #단어빈도 막대 그래프만들기위한 상위20개만 추출  
2 top20=df_word.head(20)  
3 top20
```

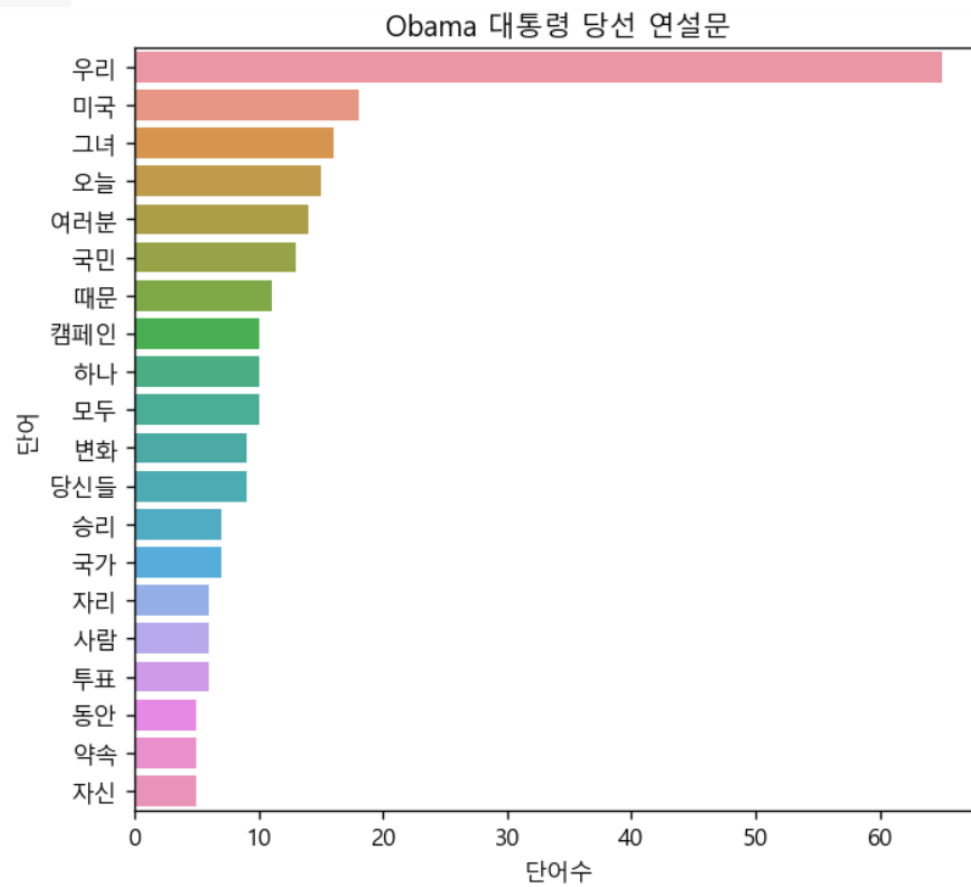
Out[34]:

	word	n
238	우리	65
133	미국	18
49	그녀	16
230	오늘	15
218	여러분	14
44	국민	13
106	때문	11
347	캠페인	10
368	하나	10
118	모두	10
157	변화	9
83	당신들	9
194	승리	7

텍스트 마이닝

In [43]:

```
1 # 상위20개의 막대그래프
2 import seaborn as sns
3 import matplotlib.pyplot as plt
4
5 plt.rcParams.update({'font.family' : 'Malgun Gothic',
6                      'figure.dpi' : '120',
7                      'figure.figsize' : [6.5,6]})
8 ax=sns.barplot(data=top20, y='word',x='n')
9 ax.set_ylabel('단어')
10 ax.set_xlabel('단어수')
11 ax.set_title('Obama 대통령 당선 연설문')
```



텍스트 마이닝

- 워드 클라우드 : 단어의 빈도를 구름 모양으로 표현한 그래프
- 워드 클라우드 패키지 설치

```
In [44]: 1 pip install wordcloud
```

- 데이터프레임의 형태를 딕셔너리 데이터형으로 변환

```
In [45]: 1 dic_word=df_word.set_index('word').to_dict()['n'] #위의 데이터프레임을 딕셔너리로 변환  
2 dic_word
```

```
Out[45]: {'우리': 65,  
          '미국': 18,  
          '그녀': 16,  
          '오늘': 15,  
          '여러분': 14,  
          '국민': 13,  
          '때문': 11,  
          '캠페인': 10,  
          '하나': 10,  
          '모두': 10,  
          '변화': 9,  
          '당신들': 9,  
          '승리': 7,  
          '국가': 7,  
          '자리': 6,  
          '사람': 6,  
          '투표': 6,  
          '동안': 5,  
          '약속': 5,  
          '지식': 5}
```

텍스트 마이닝

- 워드 클라우드 만들기

```
In [68]: 1 #워드클라우드 모양만들기
2 from wordcloud import WordCloud
3
4 wc=WordCloud(random_state=1234,
5               font_path = "c:/Windows/fonts/malgun.ttf",
6               width=400,
7               height=400,
8               background_color='white')
9
10 #워드 클라우드 만들어 출력하기(테두리선 없애기)
11 img_wordcloud=wc.generate_from_frequencies(dic_word)
12 plt.figure(figsize=(5,5))
13 plt.axis('off')
14 plt.imshow(img_wordcloud)
```

Out[68]: <matplotlib.image.AxesImage at 0x243acca29d0>



텍스트 마이닝

- 워드 클라우드 모양 변환
 - 구름 모양의 이미지 파일을 이용하여 워드 클라우드 만들기
 - 마스크로 사용될 이미지는 테두리가 뚜렷하고 어두운색으로 지정

- 마스크 만들기

```
In [65]: 1 import PIL          #이미지 삽입을 위한 불러오기 패키지  
2 icon=PIL.Image.open('cloud.png')  
3 icon
```

Out[65]:



텍스트 마이닝

- 이미지파일을 mask하기

In [66]:

```
1 import numpy as np      #그림으로 마스크형태를 만들
2 img=PIL.Image.new('RGB', icon.size, (255,255,255))
3 img.paste(icon, icon)
4 img=np.array(img)
```

- 구름 모양으로 형태 변환한 결과

In [67]:

```
1 wc1=WordCloud(random_state=1234,
2               font_path = "c:/Windows/fonts/malgun.ttf",
3               width=200,
4               height=200,
5               background_color='white',
6               mask=img)
7
8 #워드 클라우드 만들어 출력하기(테두리선 없애기)
9 img_wordcloud1=wc1.generate_from_frequencies(dic_word)
10 plt.figure(figsize=(5,5))
11 plt.axis('off')
12 plt.imshow(img_wordcloud1)
```

Out[67]: <matplotlib.image.AxesImage at 0x243ada9ce80>

