Colab 환경에서의 자연어 처리 시작하기

1-1 한글 폰트 설정

1-2 한글 적용 확인

1-3 konlpy 설치

1-4 한글 엔진을 이용한 간단한 예제

In [1]:

```
%matplotlib inline
import matplotlib as mpl            # 기본 설정 만지는 용도
import matplotlib.pyplot as plt     # 그래프 그리는 용도
import matplotlib.font_manager as fm # 폰트 관련 용도
```

colab 환경에서 한글 적용을 위한 나눔 고딕 설치

In [2]:

```
### 나눔 고딕 설치
!apt-get update -qq   # 설치를 업데이트   -qq  : 로그를 최소한으로
!apt-get install fonts-nanum* -qq # 설치한다. fonts-nanum*   =>  ttf-nanum, ttf-nanum-coding, ttf-na
```

```
Selecting previously unselected package fonts-nanum.
(Reading database ... 155501 files and directories currently installed.)
Preparing to unpack .../fonts-nanum_20170925-1_all.deb ...
Unpacking fonts-nanum (20170925-1) ...
Selecting previously unselected package fonts-nanum-eco.
Preparing to unpack .../fonts-nanum-eco_1.000-6_all.deb ...
Unpacking fonts-nanum-eco (1.000-6) ...
Selecting previously unselected package fonts-nanum-extra.
Preparing to unpack .../fonts-nanum-extra_20170925-1_all.deb ...
Unpacking fonts-nanum-extra (20170925-1) ...
Selecting previously unselected package fonts-nanum-coding.
Preparing to unpack .../fonts-nanum-coding_2.5-1_all.deb ...
Unpacking fonts-nanum-coding (2.5-1) ...
Setting up fonts-nanum-extra (20170925-1) ...
Setting up fonts-nanum (20170925-1) ...
Setting up fonts-nanum-coding (2.5-1) ...
Setting up fonts-nanum-eco (1.000-6) ...
Processing triggers for fontconfig (2.12.6-0ubuntu2) ...
```

In [3]:

```
path = '/usr/share/fonts/truetype/nanum/NanumGothicEco.ttf'
font_name = fm.FontProperties(fname=path, size=10).get_name()
print(font_name)
plt.rc('font', family=font_name)

# 우선 fm._rebuild() 를 해주고 # 폰트 매니저 재빌드가 필요하다.
fm._rebuild()
```

NanumGothic Eco

런타임 재기동 후,

- (방법 1) CTRL + M . 을 실행
- (방법 2) 메뉴의 런타임 선택 후, 런타임 다시 시작 선택
- 데이터 준비
- 라이브러리 import
- 폰트 설정 후, 확인

In [1]:

```
%matplotlib inline
import matplotlib as mpl # 기본 설정 만지는 용도
import matplotlib.pyplot as plt # 그래프 그리는 용도
import matplotlib.font_manager as fm # 폰트 관련 용도
import numpy as np

path = '/usr/share/fonts/truetype/nanum/NanumGothicEco.ttf' # 설치된 나눔글꼴중 원하는 녀석의 전체
font_name = fm.FontProperties(fname=path, size=10).get_name()
print(font_name)
plt.rc('font', family=font_name)

## 음수 표시되도록 설정
mpl.rcParams['axes.unicode_minus'] = False
```

NanumGothic Eco

In [2]:

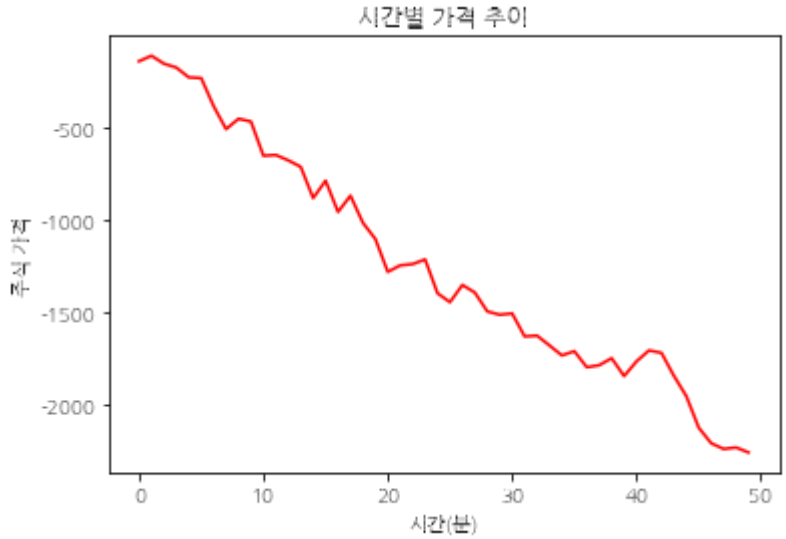
```
# 데이터 준비
data = np.random.randint(-200, 100, 50).cumsum()
data
```

Out[2]:

```
array([ -139,  -109,  -152,  -173,  -226,  -230,  -382,  -506,  -450,
        -465,  -650,  -646,  -675,  -710,  -879,  -785,  -954,  -866,
       -1013, -1100, -1278, -1243, -1236, -1211, -1394, -1442, -1349,
       -1390, -1492, -1510, -1503, -1627, -1623, -1676, -1730, -1707,
       -1794, -1783, -1745, -1842, -1762, -1703, -1716, -1840, -1950,
       -2121, -2205, -2236, -2228, -2255])
```

In [3]:

```
# 그래프를 그려보자. 이번에는 정상
plt.plot(range(50), data, 'r')
plt.title('시간별 가격 추이')
plt.ylabel('주식 가격')
plt.xlabel('시간(분)')
plt.style.use('seaborn-pastel')
plt.show()
```



1-3 konlpy 소개 및 설치

- 설치 : pip install konlpy
- 웹 사이트 : https://konlpy.org/ko/latest/ (https://konlpy.org/ko/latest/)
- KoNLPy는 파이썬 프로그래밍 언어로 사용이 가능.

In [9]:

```
!pip install konlpy
```

```
Requirement already satisfied: konlpy in /usr/local/lib/python3.7/dist-packages (0.
6.0)
Requirement already satisfied: numpy>=1.6 in /usr/local/lib/python3.7/dist-packages
(from konlpy) (1.21.6)
Requirement already satisfied: JPype1>=0.7.0 in /usr/local/lib/python3.7/dist-packag
es (from konlpy) (1.3.0)
Requirement already satisfied: lxml>=4.1.0 in /usr/local/lib/python3.7/dist-packages
(from konlpy) (4.2.6)
Requirement already satisfied: typing-extensions in /usr/local/lib/python3.7/dist-pa
ckages (from JPype1>=0.7.0->konlpy) (4.2.0)
```

In [10]:

```
import konlpy
```

In [11]:

```
import nltk
import matplotlib.pyplot as plt
import numpy as np
```

꼬꼬마를 이용한 분석

- 문장 분석
- 명사 분석
- 형태소 분석

문장 분석

- [].sentences() : 주어진 텍스트를 문장 단위로 쪼개어 리스트로 반환.

In [12]:

```
from konlpy.tag import Kkma
k = Kkma()
k.sentences("안녕하세요! 오늘은 한글 분석을 시작합니다.")
```

Out[12]:

```
['안녕하세요!', '오늘은 한글 분석을 시작합니다.']
```

명사 분석

- [].nouns(텍스트) : 주어진 텍스트에서 명사를 추출해서 리스트로 반환

In [13]:

```
k.nouns("안녕하세요! 오늘은 한글 분석을 시작합니다.")
```

Out[13]:

```
['안녕', '오늘', '한글', '분석']
```

형태소 분석

- http://kkma.snu.ac.kr/documents/index.jsp?doc=postag (http://kkma.snu.ac.kr/documents/index.jsp?doc=postag) : 한글 형태소 분석기 품사 태그표
- (예) NNG : 일반 명사, XSV : 동사 파생 접미사, EFN : 평서형 종결 어미

In [14]:

```
k.pos("안녕하세요! 오늘은 한글 분석을 시작합니다.")
```

Out[14]:

```
[('안녕', 'NNG'),
 ('하', 'XSV'),
 ('세요', 'EFN'),
 ('!', 'SF'),
 ('오늘', 'NNG'),
 ('은', 'JX'),
 ('한글', 'NNG'),
 ('분석', 'NNG'),
 ('을', 'JKO'),
 ('시작하', 'VV'),
 ('ㅂ니다', 'EFN'),
 ('.', 'SF')]
```

여러가지 엔진 사용해 보기

- Hannanum
- Okt(Open Korean Text) (예전 - Twitter)
- Kkma
- Mecab
- Komoran

In [16]:

```
from konlpy.tag import Hannanum
hannanum = Hannanum()
hannanum.pos("아버지가방에 들어가신다")
```

Out[16]:

```
[('아버지가방', 'N'), ('에', 'J'), ('들', 'P'), ('어', 'E'), ('가', 'P'), ('시ㄴ다',
'E')]
```

In [17]:

```
from konlpy.tag import Kkma
k = Kkma()
k.pos("아버지가방에 들어가신다")
```

Out[17]:

```
[('아버지', 'NNG'),
 ('가방', 'NNG'),
 ('에', 'JKM'),
 ('들어가', 'VV'),
 ('시', 'EPH'),
 ('ㄴ다', 'EFN')]
```

In [18]:

```
from konlpy.tag import Komoran
k = Komoran()
k.pos("아버지가방에 들어가신다")
```

Out[18]:

```
[('아버지', 'NNG'),
 ('가방', 'NNP'),
 ('에', 'JKB'),
 ('들어가', 'VV'),
 ('시', 'EP'),
 ('ㄴ다', 'EC')]
```

In [19]:

```
from konlpy.tag import Okt
okt = Okt()
okt.pos("아버지가방에 들어가신다")
```

Out[19]:

```
[('아버지', 'Noun'), ('가방', 'Noun'), ('에', 'Josa'), ('들어가신다', 'Verb')]
```

- Mecab는 설치후 진행해야 함.

(실습) 나는 밥을 먹는다 를 품사 태깅을 해보자

말뭉치 사용, 명사 추출

In [1]:

```
from konlpy.corpus import kolaw
c = kolaw.open('constitution.txt').read()
print( c[:15])
```

```
대한민국헌법

유구한 역사와
```

In [6]:

```
from konlpy.tag import Kkma
from konlpy.tag import Komoran
from konlpy.tag import Hannanum
from konlpy.tag import Okt
import time
```

In [7]:

```
start = time.time()
okt = Okt()
print( len( okt.nouns(c) ) )
print("time :", time.time() - start)  # 현재 - 시작 = 실행 시간
```

```
3882
time : 10.826172590255737
```

In [8]:

```
start = time.time()
komoran = Komoran()
print( len( komoran.nouns(c) ) )
print("time :", time.time() - start)  # 현재 - 시작 = 실행 시간
```

```
3358
time : 6.645719766616821
```

In [9]:

```
start = time.time()
kkma = Kkma()
print( len( kkma.nouns(c) ) )
print("time :", time.time() - start)  # 현재 - 시작 = 실행 시간
```

```
1571
time : 41.09953784942627
```

In [10]:

```
han = Hannanum()
start = time.time()
print( len( han.nouns(c) ) )
print("time :", time.time() - start)  # 현재 - 시작 = 실행 시간
```

```
3178
time : 7.618913412094116
```

여러 엔진 비교 분석 (대한 민국 헌법 텍스트)

- 런타임 재기동 후
- 순서도 바꿔보는 등의 몇차례 실행.
  - 명사 추출
    - 정확도를 떠나 주어진 텍스트에서 okt가 가장 많은 단어 추출
    - 시간은 Komoran()-코모란이 가장 빨랐다.
    - 시간은 Kkma()가 가장 오래 걸리고, 단어 추출도 가장 작다.
    - 속도 Komoran > Hannanum > Okt > Kkma

실습1 - 아래 텍스트에 대해서도 비교 분석을 해보기.

```
from konlpy.corpus import kobill
d = kobill.open('1809890.txt').read()
```

생각해 보기

- 명사 추출은 어떠할까?

말뭉치 사용해 보기

말뭉치(corpus) 사용

- kolaw : 한국 법률 말뭉치 : constitution.txt
- kobill : 대한민국 국회 의안 말뭉치. 파일 ID는 의안 번호를 의미
  - 1809890.txt ~ 1809899.txt

In [ ]:

```
from konlpy.corpus import kolaw
c = kolaw.open('constitution.txt').read()
print( c[:15])
```

대한민국헌법

유구한 역사와

In [ ]:

```
from konlpy.corpus import kobill
d = kobill.open('1809890.txt').read()
print( d[:20])
```

지방공무원법 일부개정법률안

(정의화

Ref

- KoNLPy : 파이썬 한국어 NLP
  - https://www.konlpy.org/en/latest/ (https://www.konlpy.org/en/latest/)

- konlpy를 사용한 사용 예시
  - https://konlpy.org/ko/latest/examples/ (https://konlpy.org/ko/latest/examples/)
- 사전 : https://konlpy.org/ko/latest/data/ (https://konlpy.org/ko/latest/data/)
- mecab 설치
  - https://konlpy-ko.readthedocs.io/ko/v0.4.3/install/ (https://konlpy-ko.readthedocs.io/ko/v0.4.3/install/)

이력

No	날짜	내용	ver

No	날짜	내용	ver
01	21/09/30	내용 업데이트	1.0
01	22/04/27	내용 업데이트	1.1