데이터 분석 - 자연어 처리와 Word Cloud 시각화

shlee2227 · 2023년 5월 31일

경기미래기술학교 AI과정 _ TIL

목록 보기
20/54

교육 정보

- 교육 명: 경기미래기술학교 AI 교육
- 교육 기간: 2023.05.08 ~ 2023.10.31
- 오늘의 커리큘럼: 빅데이터 기초 활용 역량 강화 (5/10~6/9) - 데이터 분석
- 강사: 조미정 강사님 (빅데이터, 머신러닝, 인공지능)
- 강의 계획:
  1. 파이썬 언어 기초 프로그래밍
  2. 크롤링 - 데이터 분석을 위한 데이터 수집(파이썬으로 진행)
  3. 탐색적 데이터 분석, 분석 실습
  - 분석은 파이썬만으로는 할 수 없으므로 분석 라이브러리/시각화 라이브러리를 통해 분석
  4. 통계기반 데이터 분석
  5. 미니프로젝트

1. 자연어 처리

KoNLPy 자연어 처리 패키지

> - KoNLPy (Korean Morph Parser Library, 코엔엘파이)KoNLPy Documentation
>   - 한국어 정보 처리를 위한 파이썬 패키지
>   - 패키지 안에 5가지 형태소 분석기
>   - Hannanum: 한나눔. KAIST Semantic Web Research Center 개발. 링크
>   - Kkma: 꼬꼬마. 서울대학교 IDS(Intelligent Data Systems) 연구실 개발. 링크
>   - Komoran: 코모란. Shineware에서 개발. 링크
>   - Mecab: 메카브. 일본어용 형태소 분석기를 한국어를 사용할 수 있도록 수정. 링크
>   - Open Korean Text: 오픈 소스 한국어 분석기. 과거 트위터 형태소 분석기. (Twitter) 링크
> - mecab이 빠르지만 설치가 여러움 (일본어용이라)
> - 한나눔이랑 Okt가 가장 사용하기 편하고 빠름
> - 공식문서 참고하여 OS 맞춰 설치 (colab은 우분투)

모듈 설치

- KoNLPy : pip install konlpy
- JDK 설치 : Java JDK로 검색해서 OS에 맞춰 설치
- JAVA_HOME 설정 : 환경변수(path) 설정 필요

```
!pip install konlpy
!sudo apt-get install g++ openjdk-8-jdk python3-dev python3-pip curl
! python3 -m pip install --upgrade pip
! python3 -m pip install konlpy
import konlpy
```

한글 자연어 처리 기초

- 다음과 같은 메서드를 공통적으로 제공
  - nouns : 명사 추출
  - morphs : 형태소 추출
  - pos : 품사 부착

KkM

- 문장 추출

```
from konlpy.tag import Kkma
```

```
kkma = Kkma()
kkma.sentences('한국어 분석을 공부하고 있습니다. 안녕하세요.')
#
#결과
['한국어 분석을 공부하고 있습니다.', '안녕하세요.']
```

- 명사 추출

```
kkma.nouns('한국어 분석을 공부하고 있습니다. 안녕하세요.')
#
#결과
['한국어', '분석', '공부']
```

- 형태소 추출

```
kkma.morphs('한국어 분석을 공부하고 있습니다. 안녕하세요.')
#
#결과
['한국어', '분석', '을', '공부', '하', '고', '있', '습니다', '.', '안녕하', '세요', '.']
```

- 품사 추출

```
kkma.pos('한국어 분석을 공부하고 있습니다. 안녕하세요.')
#
#결과
[('한국어', 'NNG'),
 ('분석', 'NNG'),
 ('을', 'JKO'),
 ('공부', 'NNG'),
 ('하', 'XSV'),
 ('고', 'ECE'),
 ('있', 'VXV'),
 ('습니다', 'EFN'),
 ('.', 'SF'),
 ('안녕하', 'VA'),
 ('세요', 'EFN'),
 ('.', 'SF')]
```

- 품사 태그의 기호와 의미

```
kkma.tagset
#
#결과
{'EC': '연결 어미',
 'ECD': '의존적 연결 어미',
 'ECE': '대등 연결 어미',
 'ECS': '보조적 연결 어미',
 'EF': '종결 어미',
 'EFA': '청유형 종결 어미',
 'EFI': '감탄형 종결 어미',
```

```
'EFN': '평서형 종결 어미',
'EFO': '명령형 종결 어미',
'EFQ': '의문형 종결 어미',
'EFR': '존칭형 종결 어미',
'EP': '선어말 어미',
'EPH': '존칭 선어말 어미',
'EPP': '공손 선어말 어미',
'EPT': '시제 선어말 어미',
'ET': '전성 어미',
'ETD': '관형형 전성 어미',
'ETN': '명사형 전성 어미',
'IC': '감탄사',
'JC': '접속 조사',
'JK': '조사',
'JKC': '보격 조사',
'JKG': '관형격 조사',
'JKI': '호격 조사',
'JKM': '부사격 조사',
'JKO': '목적격 조사',
'JKQ': '인용격 조사',
'JKS': '주격 조사',
'JX': '보조사',
'MA': '부사',
'MAC': '접속 부사',
'MAG': '일반 부사',
'MD': '관형사',
'MDN': '수 관형사',
'MDT': '일반 관형사',
'NN': '명사',
'NNB': '일반 의존 명사',
'NNG': '보통명사',
'NNM': '단위 의존 명사',
'NNP': '고유명사',
'NP': '대명사',
'NR': '수사',
'OH': '한자',
'OL': '외국어',
'ON': '숫자',
'SE': '줄임표',
'SF': '마침표, 물음표, 느낌표',
'SO': '붙임표(물결,숨김,빠짐)',
'SP': '쉼표,가운뎃점,콜론,빗금',
'SS': '따옴표,괄호표,줄표',
'SW': '기타기호 (논리수학기호,화폐기호)',
'UN': '명사추정범주',
'VA': '형용사',
'VC': '지정사',
'VCN': "부정 지정사, 형용사 '아니다'",
'VCP': "긍정 지정사, 서술격 조사 '이다'",
'VV': '동사',
'VX': '보조 용언',
'VXA': '보조 형용사',
'VXV': '보조 동사',
'XP': '접두사',
'XPN': '체언 접두사',
'XPV': '용언 접두사',
```

```
 'XR': '어근',
 'XSA': '형용사 파생 접미사',
 'XSN': '명사파생 접미사',
 'XSV': '동사 파생 접미사'}
```

Okt

- 명사 추출

```
from konlpy.tag import Okt
okt = Okt()
okt.nouns('한국어 분석을 공부하고 있습니다. 안녕하세요.')
#
#결과
['한국어', '분석', '공부']
```

- 형태소 추출

```
okt.morphs('한국어 분석을 공부하고 있습니다. 안녕하세요.')
#
#결과
['한국어', '분석', '을', '공부', '하고', '있습니다', '.', '안녕하세요', '.']
```

- 품사 부착

```
okt.pos('한국어 분석을 공부하고 있습니다. 안녕하세요.')
#
#결과
[('한국어', 'Noun'),
 ('분석', 'Noun'),
 ('을', 'Josa'),
 ('공부', 'Noun'),
 ('하고', 'Josa'),
 ('있습니다', 'Adjective'),
 ('.', 'Punctuation'),
 ('안녕하세요', 'Adjective'),
 ('.', 'Punctuation')]
```

- 품사 태그의 기호와 의미

```
okt.tagset
#
#결과
{'Adjective': '형용사',
 'Adverb': '부사',
 'Alpha': '알파벳',
 'Conjunction': '접속사',
 'Determiner': '관형사',
 'Eomi': '어미',
```

```
 'Exclamation': '감탄사',
 'Foreign': '외국어, 한자 및 기타기호',
 'Hashtag': '트위터 해쉬태그',
 'Josa': '조사',
 'KoreanParticle': '(ex: ㅋㅋ)',
 'Noun': '명사',
 'Number': '숫자',
 'PreEomi': '선어말어미',
 'Punctuation': '구두점',
 'ScreenName': '트위터 아이디',
 'Suffix': '접미사',
 'Unknown': '미등록어',
 'Verb': '동사'}
```

2. Word Cloud

이상한나라 엘리스 텍스트 분석(ENG)

- 이상한나라엘리스 원문 : https://raw.githubusercontent.com/mjcho7/dataset/main/alice_novel.txt
- 이상한나라엘리스 이미지 : https://raw.githubusercontent.com/mjcho7/dataset/main/alice_mask.png

1. 객체 생성

```
!wget https://raw.githubusercontent.com/mjcho7/dataset/main/alice_novel.txt
text = open('/content/alice_novel.txt').read()
```

2. 키워드 분석

```
#워드클라우드 객체 생성
wordcloud1 = WordCloud().generate(text1)
# 확인
rel_freq = wordcloud1.words_
print(list(rel_freq.items())[:10])
# 출력
plt.imshow(wordcloud1)
plt.axis('off')
plt.show()
```

3. 불용어 처리

```
# 위에 출력된거 보고 불용어 추가
from wordcloud import STOPWORDS
stopwords1 = set(STOPWORDS)
stopwords1.add("said")
stopwords1.add("Alice")
```

```
#제목이 너무 많이 나오거나 하는 경우는 필요에 따라 제거
# 추가된 불용어 적용하여 다시 출력
wordcloud1 = WordCloud(stopwords = stopwords1).generate(text1)
plt.imshow(wordcloud1)
plt.axis('off')
plt.show()
```

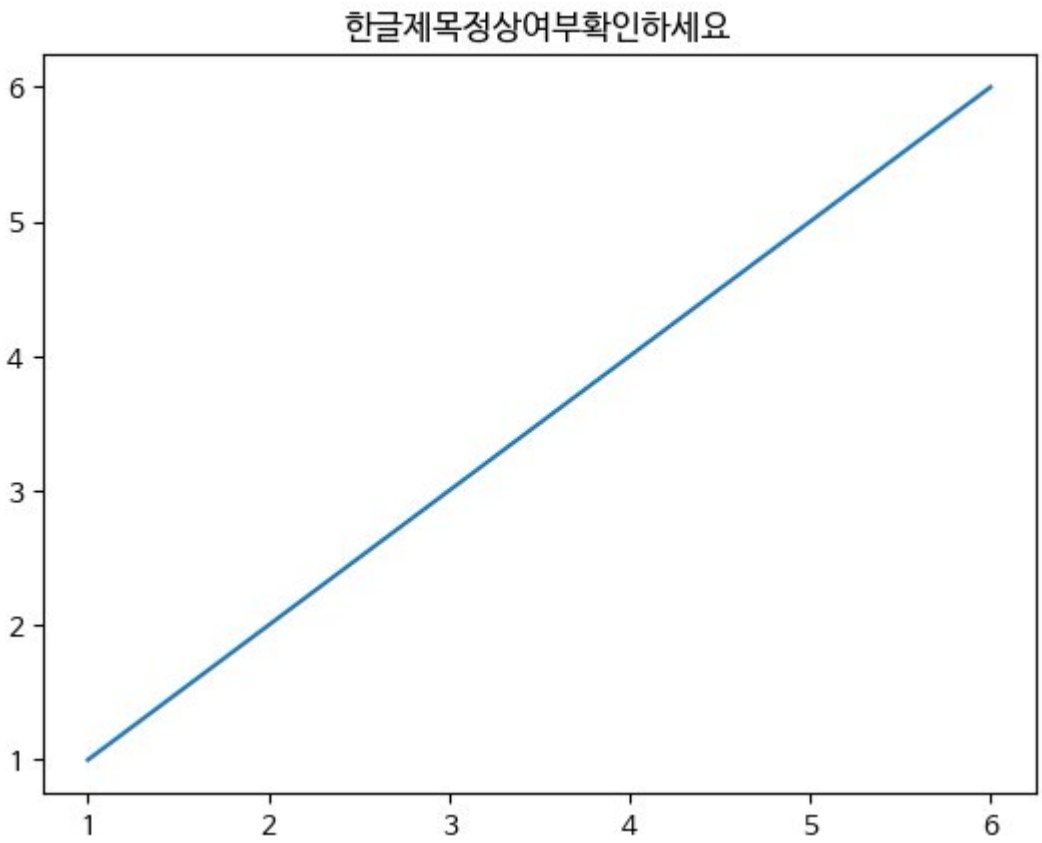
4. 워드클라우드 형태 변경 (이미지 기반)

```
# 사용할 이미지 다운로드
!wget https://raw.githubusercontent.com/mjcho7/dataset/main/alice_mask.png
#
from PIL import Image
import numpy as np
image = Image.open('alice_mask.png')
al_mask = np.array(image)
print(al_mask.shape)
# 이미지를 900*900 픽셀마다 값 지정해서 숫자로 변환해준것
wordcloud1 = WordCloud(stopwords = stopwords1, mask = al_mask, background_color = 'white').gen
#이미지 적용하여 다시 출력(배경색도 변경)
plt.imshow(wordcloud1)
plt.axis('off')
plt.show()
```

한국 법률 말뭉치 분석(KOR)

1. import 및 한글 폰트 설치

```
#글꼴 설치
!sudo apt-get install -y fonts-nanum # 나눔폰트 설치
!sudo fc-cache -fv # 폰트가 캐시에 저장되므로 위에서 설치한 폰트가 적용되도록 폰트 캐시 재 조성
!rm ~/.cache/matplotlib -rf # matplotlib의 폰트 캐시를 삭제
#이거 한 다음에 런타임을 재시작 해야함 (런타임-다시시작)- 다시시작하면 다 날아가니까 맨 처음에
import matplotlib as mpl
import matplotlib.pyplot as plt
# 폰트 적용
plt.rc('font', family='NanumBarunGothic')
x = [1, 2, 3, 4, 5, 6]
y = [1, 2, 3, 4, 5, 6]
plt.plot(x,y)
plt.title("한글제목정상여부확인하세요")
plt.show()
```



2. KoNLPy 및 WordCloud 설치

- jdk 설치

```
!sudo apt-get install g++ openjdk-8-jdk python3-dev python3-pip curl
```

- Konlpy 설치

```
! python3 -m pip install --upgrade pip
! python3 -m pip install konlpy
```

- wordcloud 설치

```
!pip install wordcloud
```

3. 키워드 분석

한국 법률 말뭉치 데이터를 사용

- 객체 생성

```
from konlpy.corpus import kolaw
kor_law = kolaw.open('constitution.txt').read()
print(type(kor_law), len(kor_law))
#
#결과
<class 'str'> 18884
```

- 형태소 분석
  - 형태소 분석을 통해 조사, 접속사 등의 제거 가능
  - 하지만 한국어는 명사에서도 상당히 많은 불필요한 단어들이 포함
  - 사용자가 직접 불용어 사전을 유지하면서 불필요한 단어 제거 필요

```
from konlpy.tag import Okt
okt = Okt()
nouns = okt.nouns(kor_law)
```

- 불용어 처리
  - 원하는 불용어를 str-> list하여 적용하는 방법

```
stopwords = '제 월 그 이 바 및 안 때 정 조 거나 경우'
stopwords = stopwords.split(' ') # 띄어쓰기를 기준으로 split해서 리스트화하여 원래 변수에 저장
#
#결과
['제', '월', '그', '이', '바', '및', '안', '때', '정', '조', '거나', '경우']
```

- 위에서 설정한 붕용어 + 한글자 짜리 단어를 불용어로 간주하여 모두 제거

```
nouns = [word for word in nouns if word not in stopwords]
# 내가 설정한 불용어 제거
nouns = [word for word in nouns if len(word) > 1]
# 단어가 1자인것을 제거
```

- 단어 빈도수 측정
  - **한국어는 빈도수를 바로 넣어줘야 워드클라우드 생성 가능**
  - 단어 빈도수 측정에는 collections 라이브러리의 Counter 함수를 이용
  - collections 라이브러리는 내장 라이브러리로 별도 설치가 필요없음
  - Counter를 이용하면 각 단어와 각 단어의 빈도 수를 딕셔너리로 편리하게 생성 가능
  - **전체 텍스트에 액세스할 필요가 없이** 빈도수로 워드클라우드 생성

```
from collections import Counter
nouns_counter = Counter(nouns)
top50 = dict(nouns_counter.most_common(50)) # dict이 편하니까 dict으로 지정
top50 # 높은값부터 정렬된 값 50개 나옴
#
#결과
{'법률': 127,
 '대통령': 83,
 '국가': 73,
  ...
 '동의': 11}
```

4. 워드클라우드

- WordCloud를 이용해 객체를 생성해주고, generate_from_frequencies() 함수로 빈도 수에 따라 워드클라우드 생성
- 워드클라우드를 시각화할 때는 이미지 시각화 함수인 imshow() 함수를 사용해야 함

```
from wordcloud import WordCloud
wordcloud = WordCloud(font_path='./font/NanumBarunGothic.ttf', background_color='white').gener
plt.imshow(wordcloud)
plt.axis('off')
plt.show()
```

팔로우

LSH

:D

다음 포스트
탐색적 데이터 분석 - 기상정보를 활용한 공공자전거 수요 분석

이전 포스트
데이터 분석 - 시각화 기초

0개의 댓글

댓글을 작성하세요

댓글 작성