

3주차
CountVectorizer
Tf-idf
WordCloud

+

02



목차

01

Counters

02

CountVectorizer

03

Tf-idf

04

wordCloud

01

Counters



Collections_Counters



컨테이너에 동일한 값의 자료가 몇개인지를 파악하는데 사용하는 객체

```
import collections
from collections import Counter
```

01 Counter 객체 선언

```
counter = collections.Counter()
```

02 기본 메소드

`most_common()` :

입력된 값의 요소들 중 빈도수(frequency)가 높은 순으로
상위 n개를 리스트(list) 안의 튜플(tuple) 형태로 반환한다

```
vocab = sorted(counts, key=counts.get, reverse=True):
```

Count Vectorizer

빈도수 기반

```
from sklearn.feature_extraction.text import CountVectorizer
```

```
vectorizer = CountVectorizer(stop_words="english",  
analyzer= {'word', 'char', 'char_wb'},tokenizer="""nltk.word_tokenize""")
```

1. 단어들의 카운트(출현 빈도)로 여러 문서들을 벡터화
2. 모두 소문자로 반환하기 때문에 대/소 신경 안써도 됨

```
vectorizer.fit_transform(text)
```

3. fit_transform 이용해서 학습

CountVectorizer는 이러한 작업을 하기 위한 다음과 같은 인수를 가질 수 있다.

stop_words : 문자열 {'english'}, 리스트 또는 None (디폴트)
stop words 목록. 'english'이면 영어용 스탑 워드 사용.

analyzer : 문자열 {'word', 'char', 'char_wb'} 또는 함수
단어 n-그램, 문자 n-그램, 단어 내의 문자 n-그램

token_pattern : string 토큰 정의용 정규 표현식

tokenizer : 함수 또는 None (디폴트) 토큰 생성 함수 .

ngram_range : (min_n, max_n) 튜플 n-그램 범위

max_df : 정수 또는 [0.0, 1.0] 사이의 실수. 디폴트 1 단어장에 포함되기 위한 최대 빈도

min_df : 정수 또는 [0.0, 1.0] 사이의 실수. 디폴트 1 단어장에 포함되기 위한 최소 빈도

Tf-idf

어떤 단어가 특정 문서에서 얼마나 중요한지를 수치화하는 모델

-> CountVectorizer 는 조사, 관사 등의 의미없는 단어에 높은 수치 부여 확률 높아짐.

개별 문서에서 자주 등장하는 단어에 높은 가중치를 주되,
모든 문서에서 전반적으로 자주 등장하는 단어에는 패널티 값을 부여.

$$\text{tf-idf}(d, t) = \text{tf}(d, t) \cdot \text{idf}(t)$$

- $\text{tf}(d, t)$: term frequency. 특정한 단어의 빈도수
- $\text{idf}(t)$: inverse document frequency. 특정한 단어가 들어 있는 문서의 수에 반비례하는 수

$$\text{idf}(d, t) = \log \frac{n}{1 + \text{df}(t)}$$

- n : 전체 문서의 수
- $\text{df}(t)$: 단어 t 를 가진 문서의 수

```
from sklearn.feature_extraction.text import TfidfVectorizer
```

```
tfidf = TfidfVectorizer().fit_transform(corpus).toarray()
```

`vectorizer.vocabulary_` : vectorizer가 학습한 단어사전 출력

```
sorted(vectorizer.vocabulary_.items())
```

```
vectorizer.get_feature_names()
```

```
array = vectorizer.transform(text).toarray()
```

#array type으로 변환하여 출력

Word Cloud

중요한 단어나 키워드를 직관적으로 보여주는 시각화 도구

```
!pip install wordcloud
```

```
from wordcloud import WordCloud
import matplotlib.pyplot as plt
%matplotlib inline
%config InlineBackend.figure_format = 'retina'
```

```
wordcloud = WordCloud(background_color='white')
wordcloud = wordcloud.generate_from_text(text)
```



실습파일로 가봅시다!