N421

* 벡터화 (vectorize)

: 텍스트를 컴퓨터가 계산할수 있도록!

① 등장횟수기반 : Bag-of-words (TF, TF-IDF)

② 분포 기반 : word2vec (cbow, skip gram)
Glove
fastText

① 등장횟수기반 단어표현

— 말뭉치에 존재하는 단어의 종류 = feature (차원)

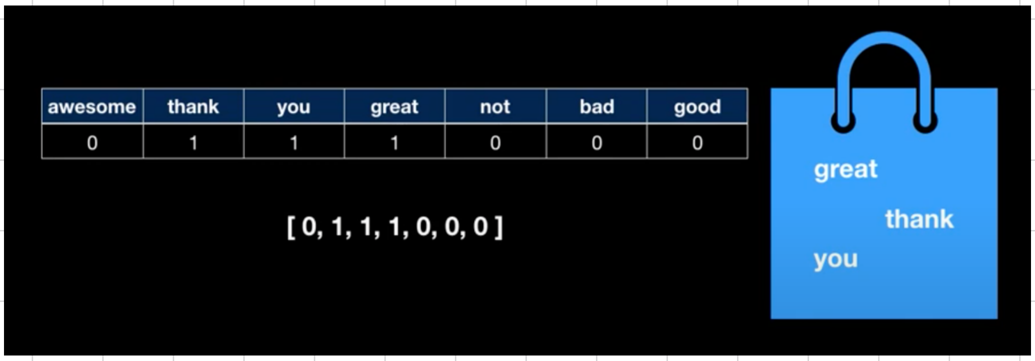
차원(단어)을 줄여야 하는 이유? 차원의 저주

- Bag-of-words (BoW)

1) TF (Term Frequency)

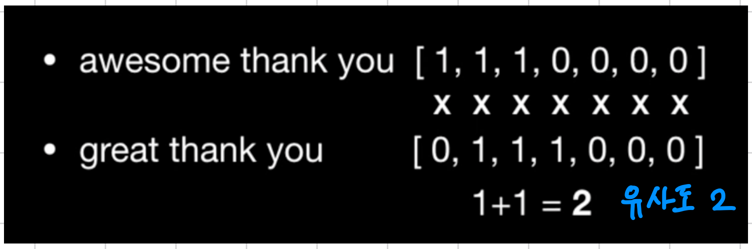
— 문법, 단어순서 무시 only 단어 빈도

— CountVectorizer 사용



awesome	thank	you	great	not	bad	good
0	1	1	1	0	0	0

[0, 1, 1, 1, 0, 0, 0]



```
• awesome thank you  [ 1, 1, 1, 0, 0, 0, 0 ]
                       x  x  x  x  x  x  x
• great thank you    [ 0, 1, 1, 1, 0, 0, 0 ]
                            1+1 = 2  유사도 2
```

2) TF-IDF

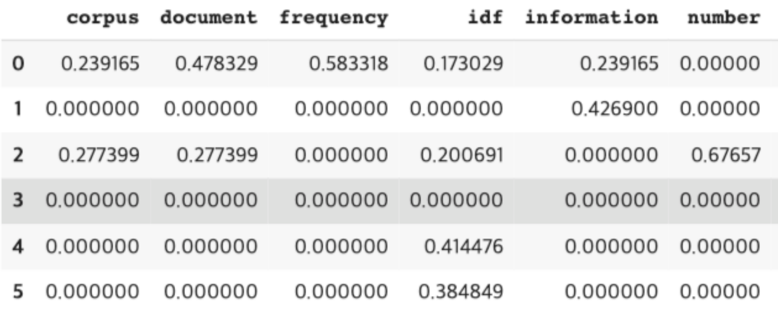
— 특정문서에만 등장하는 단어에 대해 가중치

수식 : $TF\text{-}IDF(w) = \underbrace{TF(w)}_{\text{특정 문서내 단어 "w"수}} \times IDF(w)$

$$IDF(w) = \log\left(\frac{\text{분류예상 문서수}}{\text{w가 들어있는 문서수}}\right)$$

$$\log\left(\frac{n}{1+df(w)}\right)$$

— TfidfVectorizer 사용



	corpus	document	frequency	idf	information	number
0	0.239165	0.478329	0.583318	0.173029	0.239165	0.00000
1	0.000000	0.000000	0.000000	0.000000	0.426900	0.00000
2	0.277399	0.277399	0.000000	0.200691	0.000000	0.67657
3	0.000000	0.000000	0.000000	0.000000	0.000000	0.00000
4	0.000000	0.000000	0.000000	0.414476	0.000000	0.00000
5	0.000000	0.000000	0.000000	0.384849	0.000000	0.00000

→ 실수의 형태로 결과가 나옴. (DTM)

* 토큰화

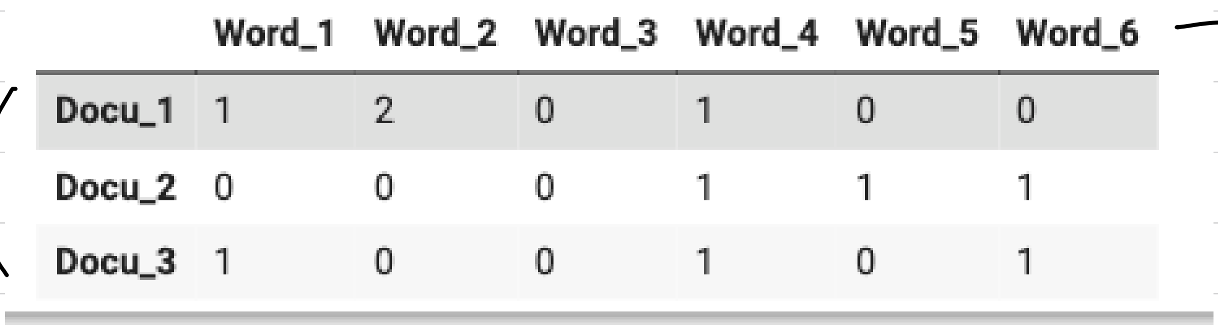
I /am /a /student

토큰 4개 (단어 4개라고 볼수 있다)

- 토큰화를 해야 벡터화를 할수있다
- SpaCy Tokenizer
- 불용어처리, 통계적 트리밍
- 어간 추출 (Stemming), 표제어 추출 (Lemmatization)

* DTM (문서 - 단어 행렬)

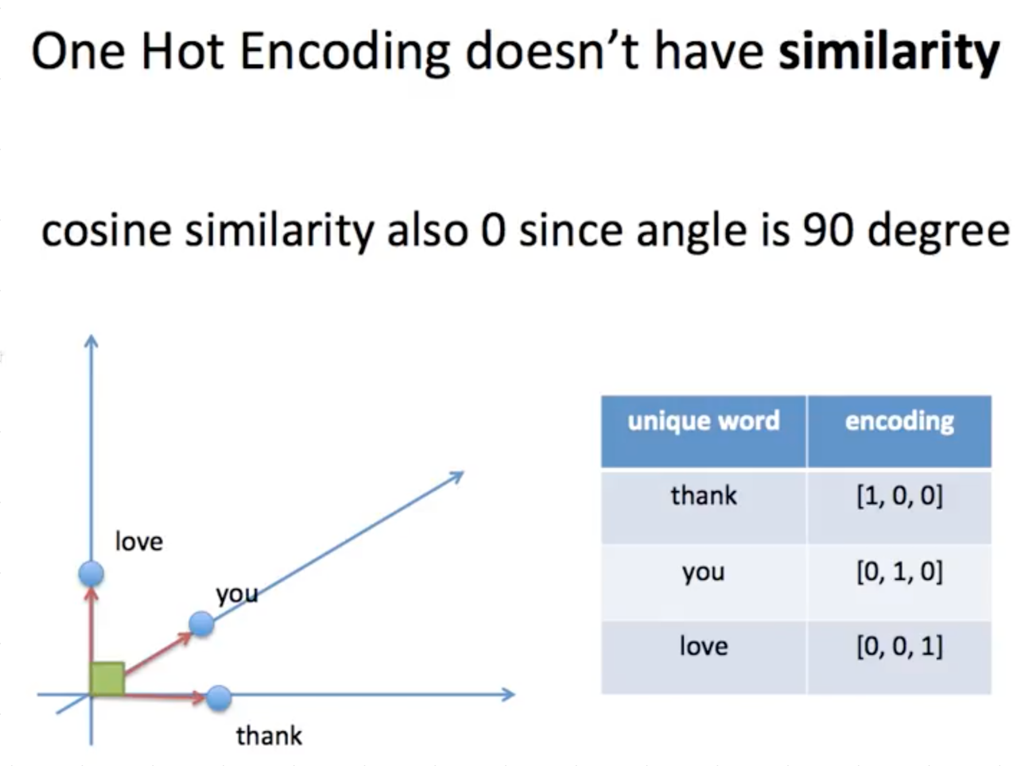
벡터화된 문서는 문서 - 단어 행렬 형태로 나타남.



	Word_1	Word_2	Word_3	Word_4	Word_5	Word_6
Docu_1	1	2	0	1	0	0
Docu_2	0	0	0	1	1	1
Docu_3	1	0	0	1	0	1

→ 단어 (컬럼)

문서 (행)



One Hot Encoding doesn't have **similarity**

cosine similarity also 0 since angle is 90 degree

unique word	encoding
thank	[1, 0, 0]
you	[0, 1, 0]
love	[0, 0, 1]