7주차 과제

Vector Space Model

담당교수님: 윤장혁 교수님

과목명: Data Analytics

이름: 박민성

학번: 201611145

전공: 산업공학과

제출일: 2020.05.03

Object : 파이썬 모듈을 이용해서 문장을 분류한후, Vector Space Model적용

Document 3개 Bag of words 기법을 통해서 나눠준다.

```
C: > Users > minisong > Desktop > 7주차.py > ...
1   from konlpy.tag import Okt
2   okt = Okt()
3   import re
4   token = re.sub("(\.)","","경찰청 철창살은 외철창살이냐 쌍철창살이냐 경찰청 철창살이 
5   token =okt.morphs(token)
6   word2index = {}
7   bow = []
8   for voca in token:
9           if voca not in word2index.keys():
10              word2index[voca] = len(word2index)
11              bow.insert(len(word2index)-1,1)
12          else:
13              index = word2index.get(voca)
14              bow[index] = bow[index] +1
15  print(word2index)
16  print(bow)
17
18  print(okt.pos('경찰청 철창살은 외철창살이냐 쌍철창살이냐 경찰청 철창살이 쇠철창살이냐
```

이 document1에 대한 코드의 결과로

```
ions\ms-python.python-2020.4.76186\pythonFiles\lib\python\debugpy\wheels\debugpy\launcher' 'c:\Users\minisong\Desktop\7주차.py'
{'경찰청': 0, '철창': 1, '살': 2, '은': 3, '외': 4, '살이': 5, '냐': 6, '쌍': 7, '쇠': 8, '철': 9, '검찰청': 10, '새': 11, '헌': 12, '창살': 13, ',': 14}
[3, 11, 4, 2, 2, 7, 6, 2, 6, 1, 2, 1, 1, 2, 1]
[('경찰청', 'Noun'), ('철창', 'Noun'), ('살', 'Noun'), ('은', 'Josa'), ('외', 'Noun'), ('철창', 'Noun'), ('살이', 'Noun'), ('냐', 'Josa'), ('쌍', 'Noun'), ('철창', 'N
oun'), ('살이', 'Noun'), ('냐', 'Josa'), ('경찰청', 'Noun'), ('철창', 'Noun'), ('살이', 'Noun'), ('쇠', 'Noun'), ('철창', 'Noun'), ('살이', 'Noun'), ('냐', 'Josa'), (
'철', 'Noun'), ('철창', 'Noun'), ('살이', 'Noun'), ('냐', 'Josa'), ('검찰청', 'Noun'), ('쇠', 'Noun'), ('철창', 'Noun'), ('살', 'Noun'), ('은', 'Josa'), ('새', 'Modif
ier'), ('쇠', 'Noun'), ('철창', 'Noun'), ('살이', 'Noun'), ('냐', 'Josa'), ('헌', 'Modifier'), ('쇠', 'Noun'), ('철창', 'Noun'), ('살이', 'Noun'), ('냐', 'Josa'), ('
경찰청', 'Noun'), ('쇠', 'Noun'), ('창살', 'Noun'), ('외', 'Noun'), ('철창', 'Noun'), ('살', 'Noun'), (',', 'Punctuation'), ('검찰청', 'Noun'), ('쇠', 'Noun'), ('창살
', 'Noun'), ('쌍', 'Noun'), ('철창', 'Noun'), ('살', 'Noun'), ('.', 'Punctuation')]
PS C:\Users\minisong>
```

이러한 결과가 나온다.

결과의 첫째줄을 보면 딕셔너리 형태의 결과가 나오는데 이것은 t1,t2...와 같은 인덱스이고 두번째줄을 보면 하나의 리스트가 나오는데 이것은 각 인덱스에 해당하는 단어의 빈도수를 나타낸다. 불용처리를 위해 단어의 성분에 대해 조사해 보았는데 '은', '냐', 구두점(.)이 발견되어서 이 요소들은 corpus를 구성할 때 제외시켰다.

또한 '살 + 이냐' 를 명사 '살이'로 처리한 부분을 이냐를 제외하고 '살'의 빈도수에 추가해줬다.

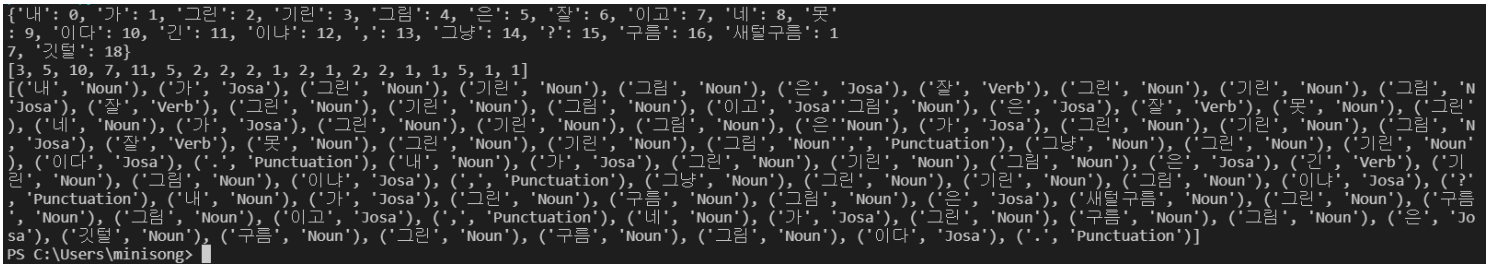
그 결과를 정리해보면

경찰청	철창	살	외	쌍	쇠	철	검찰청	새	헌	창살
3	11	11	2	2	6	1	2	1	1	2

이러한 결과가 나온다.

마찬가지로 document 2,3에 대해서도 진행을 한다.

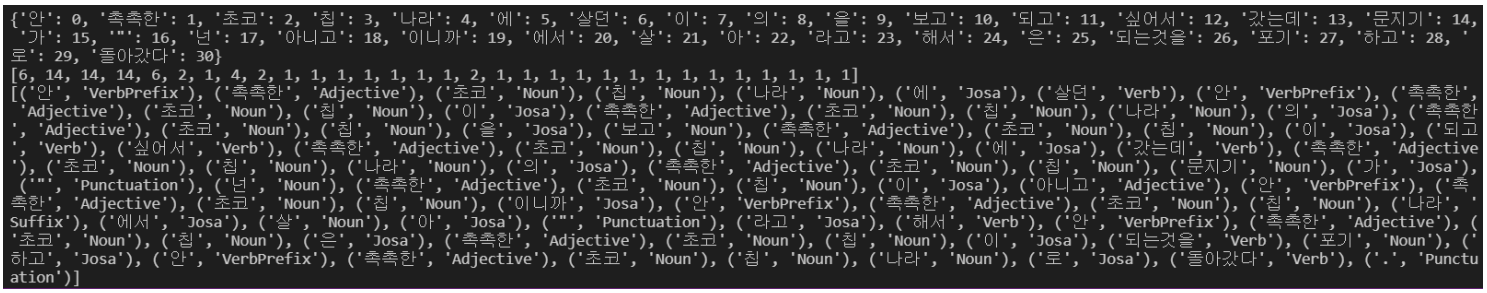
Document 2



마찬가지로 정리하면

내	그린	기린	그림	잘	네	못	긴	그냥	구름	새털구름	깃털
3	10	7	11	2	2	1	1	1	5	1	1

Document 3



안	촉촉한	초코	칩	나라	살던	보고	되고	싶어서	갔는데
6	14	14	14	6	1	1	1	1	1

문지기	넌	아니고	살	해서	되는것을	포기	돌아갔다
1	1	1	1	1	1	1	1

세 document에 요소들을 다합해서 matrix의 형태로 표현한후, 진행한다.

가로가 너무 길어서 두줄에 표현했다.

tf-based VSM representation (boolean)

	경찰청	철창	살	외	쌍	쇠	철	검찰청	새	헌	창살	내	그린	기린	그림	잘	네	못	긴	그냥
d1	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0
d2	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1
d3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

구름	새털구름	깃털	안	촉촉한	초코	칩	나라	살던	보고	되고	싶어서	갔는데	문지기	넌	아니고	살	해서	되는것을	포기	돌아갔다
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Boolean 같은 경우는 단어가 등장하면 1을 할당하고 등장하지 않으면 0을 할당한다.

tf-based VSM representation (simple)

빈도수를 세어서 값을 할당한다.

	경찰청	철창	살	외	쌍	쇠	철	검찰청	새	헌	창살	내	그린	기린	그림	잘	네	못	긴	그냥
d1	3	11	11	2	2	6	1	2	1	1	2	0	0	0	0	0	0	0	0	0
d2	0	0	0	0	0	0	0	0	0	0	0	3	10	7	11	2	2	1	1	1
d3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

구름	새털구름	깃털	안	촉촉한	초코	칩	나라	살던	보고	되고	싶어서	갔는데	문지기	넌	아니고	살	해서	되는것을	포기	돌아갔다
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
5	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	6	14	14	14	6	1	1	1	1	1	1	1	1	1	1	1	1	1

tf-based VSM representation (logarithmatically scaled)

	경찰청	철창	살	외	쌍	쇠	철	검찰청	새	헌	창살	내	그린	기린	그림	잘	네	못	긴	그냥
d1	0.60206	1.079181	1.079181	0.477121	0.477121	0.845098	0.30103	0.477121	0.30103	0.30103	0.477121	0	0	0	0	0	0	0	0	0
d2	0	0	0	0	0	0	0	0	0	0	0	0.60206	1.041393	0.90309	1.079181	0.477121	0.477121	0.30103	0.30103	0.30103
d3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

구름	새털구름	깃털	안	촉촉한	초코	칩	나라	살던	보고	되고	싶어서	갔는데	문지기	넌	아니고	살	해서	되는것을	포기	돌아갔다
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0.778151	0.30103	0.30103	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0.845098	1.176091	1.176091	1.176091	0.845098	0.30103	0.30103	0.30103	0.30103	0.30103	0.30103	0.30103	0.30103	0.30103	0.30103	0.30103	0.30103	0.30103

simple에서 나온 빈도수에 +1을 하여서 로그를 씌워준다.

만약, Similarity를 아래와 같이 구한다면

모든 document 사이에 겹치는 단어가 없으므로 0이 나온다.

$$\boldsymbol{similarilty}(\boldsymbol{A},\boldsymbol{B}) = \cos(\boldsymbol{\theta}) = \frac{\boldsymbol{A}\cdot\boldsymbol{B}}{|\boldsymbol{A}||\boldsymbol{B}|} = \frac{\sum_{i=-1}^{n} \boldsymbol{A_i}\boldsymbol{B_i}}{\sqrt{\sum_{i=-1}^{n} \boldsymbol{A_i}^2}\sqrt{\sum_{i=-1}^{n} \boldsymbol{B_i}^2}}$$