8주차 과제

Text mining(keyword network)

담당교수님: 윤장혁 교수님

과목명: Data Analytics

이름: 박민성

학번: 201611145

전공: 산업공학과

제출일: 2020.05.08

Object

주어진 리뷰데이터를 활용하여 키워드 동시출현 네트워크를 작성, Centrality를 분석한다. 분석을 통해서 연관성이 높은 TOP 5 단어쌍을 선출해 보고, 가장 중요한 node는 무엇인지 알아본다.

보고서 진행순서

진행순서는 크게 다섯 단계로 진행한다.

1. Keyword를 추출

2. 추출한 키워드의 불용처리

3. Centrality 정의

4. 데이터 추출

5. 데이터 시각화

6. 해석 및 코멘트

1. Keyword 추출

분석 툴: Python 활용한다. 주어진 JSON 데이터의 review데이터를 활용해서 분석을 진행했다.

각 데이터는 띄어쓰기를 기준으로 처리하였다. 단어를 어절/어미로 쪼개는 등 더 세세한 방법이 있지만 이 분석에서는 띄어쓰기를 기준으로 한다.

C: > Users > minisong > Desktop > 201611145_박민성_week8.py > ...

```python
import pandas as pd
pd.options.mode.chained_assignment = None

import numpy as np
np.random.seed(0)

from konlpy.tag import Twitter
twitter = Twitter()

import json

from sklearn.feature_extraction.text import TfidfVectorizer, CountVectorizer
from sklearn.metrics.pairwise import linear_kernel, cosine_similarity

import re
import numpy as np
from nltk.corpus import stopwords

import csv

def tokenizer(raw, pos=["Noun","Alpha","Verb","Number"], stopword=[]):
    return [
        word for word, tag in twitter.pos(
            raw,
            norm=True,
            stem=True
            )
            if len(word) > 1 and tag in pos and word not in stopword
        ]
with open("C:\\Users\\minisong\\Desktop\\data_week8.json", 'r', encoding ='utf-8') as f:

    json_data = json.load(f)
stop_words = set(stopwords.words('english'))
aaa = []
listlsit=""
rawdata = []
```

2. 추출한 Keyword의 불용처리

불용처리는 Matrix의 크기를 줄이고, 컴퓨터의 처리 속도 향상 등을 이유로 처리해준다. Sklearn과 konlypy등을 이용해서 내가 분석하려고 하는 단어의 종류를 선택해준다. (명사, 알파벳, 동사, 숫자) 그리고 분석에 필요 없는 문자 ^,> 등을 모두 없애 준다. 또한 단어가 너무 많아서 단어의 출현빈도가 '25'가 되지 않는 단어들을 min_df를 통해 걸러준다. 원활한 분석을 위해 모두 소문자로 변환해서 진행했다.

```
for i in json_data:
    aa=[]
    a=[]
    a= re.sub('[^a-zA-Z]', ' ', i['review_body'])
    a.lower()
    middle=""
    aa=a.split(" ")
    for w in aa:
        if w  not in stop_words:
            middle = middle + w + " "
    rawdata.append(middle)
vectorize = CountVectorizer(
    tokenizer=tokenizer,
    min_df=25
)

X = vectorize.fit_transform(rawdata)

print('fit_transform, (sentence {}, feature {})'.format(X.shape[0], X.shape[1]))


print(X.toarray())

features = vectorize.get_feature_names()
print(features)

a = np.array(X.toarray())
df = pd.DataFrame(a)
df.to_csv("결과데이터 중간.csv",encoding="utf-8-sig",header = False, index = False)
```

3. Centrality 정의, 4. 데이터 추출

결과 테이블은 100개의 문장에서 8개의 feature을 담고 있는 100 x 8 행렬이 나온다. 이를 행렬 A라고 하자. A의 전치행렬과 A를 곱해줘서 8 x 8 행렬, 즉, 단어와 단어의 관계를 보여주는 행렬(B)을 만들고 그 행렬의 원소 값을 분석해서 중요쌍 5개를 선발한다.

또한 Degree centrality를 이용해서 각 노드의 weighted Cd(v)를 계산해서 가장 중요한 노드가 무엇인지 찾아본다.

```
(sentence 100, feature 8)
```

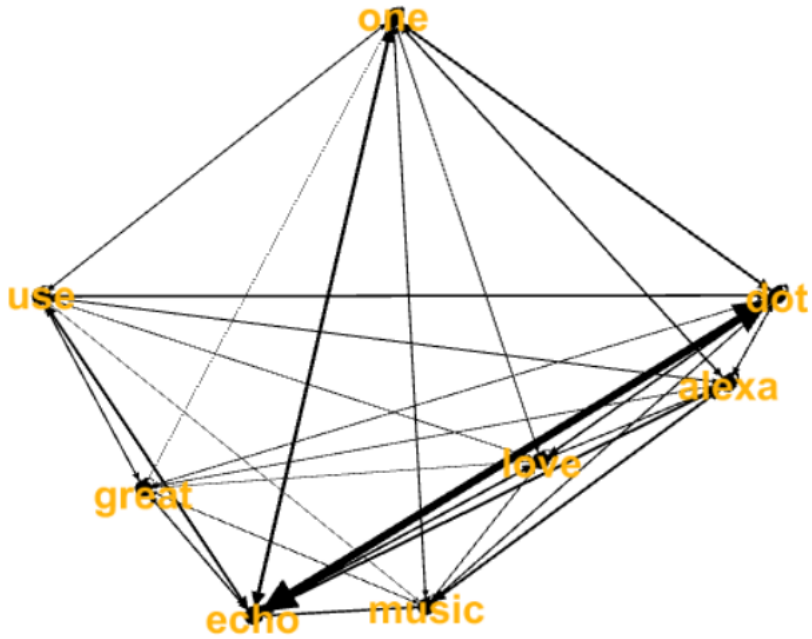
```
['alexa', 'dot', 'echo', 'great', 'love', 'music', 'one', 'use']
```

	alexa	dot	echo	great	love	music	one	use
alexa	90	31	43	23	29	42	34	28
dot	31	100	105	24	36	30	42	38
echo	43	105	168	32	38	36	51	42
great	23	24	32	57	19	23	16	30
love	29	36	38	19	60	25	26	23
music	42	30	36	23	25	63	29	18
one	34	42	51	16	26	29	103	28
use	28	38	42	30	23	18	28	70

예를 들면 'alexa'라는 단어가 나왔을 때 'dot'이라는 단어가 31번 출현했다는 뜻이다. 데이터의 값이 클수록 단어가 같이 등장할 가능성이 높고 연관성이 높은 단어라고 할 수 있다. 이 중에 TOP5의 단어를 알아본다.

5. 데이터 시각화

데이터 시각화툴은 gehpi를 이용했다. Top5를 쉽게 찾아내기 위해서 min_df를 25라는 높은 값을 주었기 때문에 시각화 했을 때 복잡한 모양은 나오지 않았다. 선의 굵기가 굵을수록 두 단어가 같이 출현할 가능성이 높다고 할 수 있겠다.



6. 해석 및 코멘트

위의 결과 테이블에서 보면 가장 동시 출현할 확률이 높은 TOP5 단어쌍은

Top1: (dot, echo) -> 105(동시 출현빈도)

Top2: (one, echo) -> 51

Top3: (echo, alexa) -> 43

Top4: (music, alexa), (dot, one), (use, echo) -> 42 이다.

Degree Centrality 분석

	alexa	dot	echo	great	love	music	one	use
alexa	**0**	**31**	**43**	**23**	**29**	**42**	**34**	**28**
dot	**31**	**0**	**105**	**24**	**36**	**30**	**42**	**38**
echo	**43**	**105**	**0**	**32**	**38**	**36**	**51**	**42**
great	**23**	**24**	**32**	**0**	**19**	**23**	**16**	**30**
love	**29**	**36**	**38**	**19**	**0**	**25**	**26**	**23**
music	**42**	**30**	**36**	**23**	**25**	**0**	**29**	**18**
one	**34**	**42**	**51**	**16**	**26**	**29**	**0**	**28**
use	**28**	**38**	**42**	**30**	**23**	**18**	**28**	**0**
sum	**230**	**306**	**347**	**167**	**196**	**203**	**226**	**207**

앞에서 얻은 행렬은 단어간 동시 출현의 빈도를 알려준다. 이를 가중치로 삼고 nxn에 해당되는 대각원소를 제외하고 더해준다. (self – 제외) 이는 Weighted Degree centrality라고 할 수 있다.

가장 큰 값은 echo이다. 따라서 echo가 가장 중요한 node라고 할 수 있고 실제로 TOP5의 6개의 순서쌍에서도 4쌍이나 있는 것을 확인할 수 있다.