<빅데이터 미니프로젝트 보고서>

유튜브 크롤링

휴먼지능정보공학과 201710799 이 지민

1. 서 론

-다양하고, 형식을 갖추지 않은 자유로운 데이터들이 많은 유튜브를 크롤링하는 것을 주제로 정했다.
웹상에서 raw데이터를 수집하고, 정제하고, 분석하고, 시각화하는 전반적인 과정을 진행하였다.

2. 프로젝트의 내용, 방법

-프로젝트 기술서

what

먹방 유튜버들의 영상정보를 웹 크롤링하여, 유튜버들이 많이 먹는 음식종류를 알아본다.
그리고 데이터를 순위별로 알아보기 쉽게 시각화한다.

Who
신메뉴, 유행음식을 파악해 먹방 유튜버와 먹방시청자에게 마케팅을 하는 것에 도움을 준다.

Why
먹방은 신메뉴나 유행에 예민하기 때문에 사람들에게 관심있는 음식의 종류를 알 수 있다.

How
1) 50명의 먹방 유튜버의 영상정보를 selenium, bratifulSoup 모듈을 이용하여 유튜브의 영상목록 url을 이용하여 크롤링을 한다. 영상정보를 크롤링할 때, 웹상에서 스크롤이 자동으로 200번이 내려가며 모든 영상의 정보가 저장된다. 영상의 정보 중에서 영상제목, 유튜버이름, 조회수 등을 txt파일로 저장한다.

youtube crawling Ae Jung.ipynb	하루 전	1.61 MB
youtube crawling amatta.ipynb	하루 전	1.38 MB
youtube crawling army.ipynb	하루 전	65 kB
youtube crawling Boki.ipynb	하루 전	108 kB
youtube crawling Bonggil.ipynb	하루 전	77.2 kB
youtube crawling changbae.ipynb	하루 전	4.7 MB
youtube crawling chiyoon.ipynb	하루 전	1.94 MB
youtube crawling ddeonggae.ipynb	하루 전	8.41 MB
youtube crawling Donam.ipynb	하루 전	3.93 MB
youtube crawling dorothy.ipynb	하루 전	6.09 MB
youtube crawling eating.ipynb	하루 전	3.11 MB
youtube crawling flowerpig.ipynb	하루 전	5.2 MB
youtube crawling fran.ipynb	하루 전	8.89 MB
youtube crawling Fume.ipynb	하루 전	1.6 MB
youtube crawling G-IN.ipynb	하루 전	1.6 MB
youtube crawling Gamsa.ipynb	하루 전	4.12 MB
youtube crawling GONGSAM.ipynb	하루 전	1.54 MB
youtube crawling Gyun.ipynb	하루 전	2.41 MB
youtube crawling hamzy.ipynb	한 시간 전	141 kB
youtube crawling hongsound.ipynb	하루 전	6 MB
youtube crawling Hongyu.ipynb	하루 전	1.45 MB
youtube crawling HwaRong.ipynb	하루 전	946 kB
youtube crawling Hyeji.ipynb	하루 전	1.13 MB

crawling

In [1]:

```python
from selenium import webdriver
from selenium.webdriver.common.keys import Keys
from bs4 import BeautifulSoup
import time
...

driver = webdriver.Chrome('C:/Users/jmlee/Code/chromedriver_win32/chromedriver.exe')
driver.get('https://www.youtube.com/channel/UC-Bsa2ivAGMq7bsSPrPGFVA/videos')

#여기서부터
num_of_pagedowns=200
body = driver.find_element_by_xpath('/html/body')

#자동으로 스크롤을 내려서 더이상 영상이 로드 안될때까지 내려줌.
while num_of_pagedowns:
    body.send_keys(Keys.PAGE_DOWN)
    time.sleep(0.3)
    num_of_pagedowns -= 1
    try:
        driver.find_element_by_xpath("""//*[@id="feed-main-what_to_watch"]/button""").click()
    except:
        None

page = driver.page_source
page
```


```
CccykSIdFavLiCrMZ//cHQKnRZW09dIdVoBHYuJmGj2wZ7nDg+xE4RQMQ+KulzKM3PvYAIAAABeeyJvcmlnaW4iOiJodHRwczovL2dvb2dsZS5jb206NDQzIiwiZmVhdHVyZSI6Il
IYkNvbXBvbmVudHNWMCIsImV4cGlyeSI6MTYwMzgzNjc3MiwiaXNTdWJkb21haW4iOnRydWV9"><script>var ytcfg = {d: function() {return (window.yt && yt.con
fig_) || ytcfg.data_ || (ytcfg.data_ = {});},get: function(k, o) {return (k in ytcfg.d()) ? ytcfg.d()[k] : o;},set: function() {var a = ar
guments;if (a.length > 1) {ytcfg.d()[a[0]] = a[1];} else {for (var k in a[0]) {ytcfg.d()[k] = a[0][k];}}}};window.ytcfg.set(₩'EMERGENCY_BA
SE_URL₩', "₩₩/error_204?client.version=2.20191127.01.00₩₩u0026client.name=1₩₩u0026level=ERROR₩₩u0026t=jserror");</script><script>window.Sh
adyCSS = {disableRuntime: true};window.ShadyDOM = {force: true};window.ShadyDOM.preferPerformance = true;window.ShadyDOM.noPatch = true;wi
ndow.Polymer = (window.Polymer || {});window.Polymer.legacyOptimizations = true;</script><link rel="shortcut icon" href="https://s.ytimg.c
om/yts/img/favicon-vfl8qSV2F.ico" type="image/x-icon"><link rel="icon" href="https://s.ytimg.com/yts/img/favicon_32-vflOogEID.png" sizes
="32x32"><link rel="icon" href="https://s.ytimg.com/yts/img/favicon_48-vflVjB_Qk.png" sizes="48x48"><link rel="icon" href="https://s.ytim
g.com/yts/img/favicon_96-vflW9Ec0w.png" sizes="96x96"><link rel="icon" href="https://s.ytimg.com/yts/img/favicon_144-vflLAfaB.png" sizes
="144x144"><title>입짧은햇님 - YouTube</title><script>var ytcsi = {gt: function(n) {n = (n || ₩'₩') + ₩'data_₩';return ytcsi[n] || (ytcsi
[n] = {tick: {},info: {}});},now: (window.performance && window.performance.timing &&window.performance.now && window.performance.timing.r
```


In [4]:

```python
from bs4 import BeautifulSoup
#태그, 클래스
soup = BeautifulSoup(page,'lxml')
all_title = soup.find_all('a','yt-simple-endpoint style-scope ytd-grid-video-renderer')
title = [soup.find_all('a','yt-simple-endpoint style-scope ytd-grid-video-renderer')[n].string for n in range(0,len(all_title))]
title
```

Out[4]:

```
['리얼먹방:) 굽네는 치트킹 햄지는 치트키.....라임인정?? (ft.허니멜로피자,소주)|Chicken with Spicy powder|REAL SOUND|ASMR MUKBANG|',
 '리얼먹방:) 보쌈먹방!! 이 영상보고 참으면 다이어트 의지 인정😜(ft.생굴,소주)|Bossam & Oysters|REAL SOUND|ASMR MUKBANG|EATING SHOW
|',
 '리얼먹방:) 밥보다많은 꼬막!! 꼬막비빔밥 먹방(ft.청국장)|Spicy Blood cockle bibimbap|REAL SOUND|ASMR MUKBANG|EATING SHOW|',
 '리얼먹방:) 오늘은 조촐하게 햄버거로 한끼 때울게요🍔🍟|Hamburger+French fries|REAL SOUND|ASMR MUKBANG|EATING SHOW|',
 '리얼먹방:) 뻑뻑살 싫어해서 윙봉 시켰어요😭 (ft.60개 고추윙봉,라이트불닭볶음면)|Spicy Chicken|REAL SOUND|ASMR MUKBANG|EATING SHOW
|',
 '리얼먹방:) 햄지25시★편의점 아니고.....편의집(?) 먹방!!|Korea Convenience Store Food|REAL SOUND|ASMR MUKBANG|',
 '리얼먹방:) 매운 쭈꾸미삼겹살 먹방!!★마무리는 날치알 볶음밥 |Spicy Stir-fried Octopus|REAL SOUND|ASMR MUKBANG|EATING SHOW|',
 '리얼먹방:) 조개찜에 혼술★ 마무리는 당연히 칼국수!!|Steamed clams|REAL SOUND|ASMR MUKBANG|EATING SHOW|',
 '리얼먹방:) 도가니탕&수육 날씨가 쌀쌀할땐 국밥이 최고!!!(ft.깍두기)|REAL SOUND|ASMR MUKBANG|EATING SHOW|',
 '리얼먹방:) 매콤~한 쫄면에 치즈돈까스 !! 궁극의 조합| jjolmyeon(spicy noodles)|REAL SOUND|ASMR MUKBANG|EATING SHOW|',
 '리얼먹방:) 콩나물 팍팍!! 얼큰~한 알탕 먹방!! |Spicy Altang(Fish₩xa0Roe₩xa0Soup)|REAL SOUND|ASMR MUKBANG|EATING SHOW|',
 '리얼먹방:) 피맥(?) NoNo~난 피소|해무르니까...(ft.전주불백피자,진로이즈백)|Pizza&Soju|REAL SOUND|ASMR MUKBANG|EATING SHOW|',
 '리얼먹방:) 송이버섯🍄 짜장라면.....FLEX ★ ft.군만두,파김치|Spicy jjajang Ramen|ASMR MUKBANG|EATING SHOW|',
 '리얼먹방:) 대창이 이정도는 들어가줘야 대창덮밥 아님?? (ft.불닭소스,막창)★디저트는 쿠키😋|Grilled Cow intestine|REAL SOUND|ASMR MUK
BANG',
 '리얼먹방:) 입맛없을땐 대패삼겹살&비빔면★디저트는 RED키위|Spring Rolls|REAL SOUND|ASMR MUKBANG|EATING SHOW|',
 '리얼먹방:) 연어장으로 연어덮밥 만들어먹기★청양고추와 함께라면 문제없음😎|Marinated Salmon|REAL SOUND|ASMR MUKBANG|EATING SHOW
|',
```

```
youtube_video_list = []

x = 0 ----> #조회수의 index
y = 1 ----> #업로드 시점의 index

for i in range(0,len(all_title)):
    roww = []
    roww.append(title[i])
#     roww.append(video_time[i].strip())
    roww.append(channel)
    roww.append(sub_num)
    roww.append(view_num[x])
    x += 2 ----> #조회수만 append
    roww.append(view_num[y])
    y += 2 ----> #업로드 시점만 append
    roww.append(extract_date)
    youtube_video_list.append(roww) ----> #2차원 list로 만들어줌

youtube_video_list
```

```
 '구독자 104만명',
 '조회수 266만회',
 '3주 전',
 '2019/11/17 20:22:40'],
['리얼먹방:) 대창이 이정도는 들어가줘야 대창덮밥 아님?? (ft.불닭소스.막창)★디저트는 쿠키 | Grilled Cow intestine | REAL SOUND | ASMR MUKBANG',
 '[햄지]Hamzy',
 '구독자 104만명',
 '조회수 129만회',
 '3주 전',
 '2019/11/17 20:22:40'],
['리얼먹방:) 입맛없을땐 대패삼겹살&비빔면★디저트는 RED커튼 | Spring Rolls | REAL SOUND | ASMR MUKBANG | EATING SHOW | ',
 '[햄지]Hamzy',
 '구독자 104만명',
 '조회수 112만회',
 '4주 전',
 '2019/11/17 20:22:40'],
['리얼먹방:) 연어장으로 연어덮밥 만들어먹기★청양고추와 함께라면 문제없음 | Marinated Salmon | REAL SOUND | ASMR MUKBANG | EATING SHOW | ',
 '[햄지]Hamzy',
```

2) spark를 이용해 각 유튜버 별 영상정보를 txt로 저장하고, 그 txt를 불러와 정규표현식을 이용해 이모티콘, 특수문자를 제거한다. 그래도 제거되지 않은 문자 또는 음식명 추출에 방해되는 단어는 stopword로 제거한다. 제거한 후 rdd를 만들고 음식명을 key로 하여 key별 빈도수를 추출한다. 추출하고 난 결과는 50명 모두 하나의 txt(Frequency.txt)에 한번에 저장한다.

```
#정규화한다음 빈도수 추출
hanzy=hanzystopRdd\
    .flatMap(lambda x:x.split(','))\
    .map(lambda x:(x,1))\
    .reduceByKey(lambda x,y:x+y)\
    .map(lambda x:(x[1],x[0]))\
    .sortByKey(False)\
    .take(20)
```

```
print(hanzy)
```

```
[(5, '볶음밥'), (4, '매운'), (4, '실비김치'), (3, '소주'), (3, '불닭소스'), (3, '쭈꾸미'), (3, '스파게티'), (3, '짜파게티'), (3, '홍각김치'), (2, '생굴'), (2, '꼬막비빔밥'), (2, '날치알'), (2, '수육'), (2, '짜두기'), (2, '막창'), (2, '함께라면'), (2, '왕만두'), (2, '김치먹방'), (2, '짜스타'), (2, '된장찌개')]
```

```
#파일저장
f=open("C:₩₩Users₩₩jmlee₩₩Code₩₩Frequency.txt",'a',-1,"utf-8")
A=len(hanzy)
print (A)
for i in range (0,A):
    f.writelines(str(hanzy[i]))
    f.writelines("₩n")
f.close()
```

```
20
```

3) 50명의 분석결과가 있는 Frequency.txt를 이용해 분석을 한다. txt를 불러와 list를 만들고, 음식이름을 key로 빈도수 추출을 하려고 했는데, txt의 list속이 str형태여서 빈도수가 이상하게 구해졌다.(괄호와, 숫자도 key로 계산됨.) 따라서 list의 for문을 돌리면서 list속을 tuple형태로 바꿔주고 음식을 key로 설정하여 빈도수를 계산하였다.

```
FrequencyRdd=spark.sparkContext.textFile(os.path.join("crawling txt","Frequency.txt"))
```

```
FrequencyList=FrequencyRdd.collect()
# print(FrequencyList)
print(FrequencyList)
len(FrequencyList)
```


["(11, '치킨')", "(6, '피자')", "(6, '신메뉴')", "(5, '간장게장')", "(4, '치즈')", "(4, '케이크')", "(3, '막창')", "(3, '떡볶이')", "(3, '핫도그')", "(3, '불고기')", "(3, '새우튀김')", "(3, '목은지')", "(3, '새우장')", "(3, '도미노')", "(3, '감성카초')", "(3, '탕수육')", "(3, '교촌')", "(2, '소곱창')", "(2, '대창')", "(2, '소시지')", "(2, '라면')", "(2, '짜장면')", "(2, '비빔밥')", "(2, '닭발')", "(2, '통삼겹살')", "(2, '마시멜로우')", "(2, '연어회')", "(2, '수육')", "(10, '골조합')", "(5, '집밥')", "(4, '신메뉴')", "(4, '간장새우')", "(3, '치즈')", "(3, '매콤한')", "(3, '연어')", "(3, '군만두')", "(3, '허니콤보')", "(2, '대창')", "(2, '돈까스')", "(2, '분식')", "(2, '치킨')", "(2, '찹쌀떡')", "(2, '시원한')", "(2, '맥주')", "(2, '전복')", "(2, '불닭볶음면')", "(2, '흰쌀밥')", "(2, '광어')", "(2, '닭껍질튀김')", "(2, '새우튀김')", "(2, '날치알')", "(2, '디저트')", "(2, '통삼겹살')", "(2, '짜파게티')", "(2, '파김치')", "(2, '꾸덕꾸덕')", "(2, '중국당면')", "(2, '주먹밥')", "(6, '치즈볼')", "(5, '라면')", "(4, '스팸')", "(4, '계란후라이')", "(4, '김치')", "(4, '연어')", "(4, '떡볶이')", "(3, '수육')", "(3, '삼겹살')", "(3, '김')", "(3, '신제품')", "(3, '까르보불닭')", "(2, '볼고기')", "(2, '반올림피자샵')", "(2, '한우')", "(2, '뼝')", "(2, '명랑핫도그')", "(2, '칠리새우')", "(16, '매운')", "(6, '중국당면')", "(6, '청양고추')", "(5, '전주진김치')", "(4, '매콤한')", "(4, '김치')", "(4, '금치김치')", "(3, '떡만둣국')", "(3, '당면사리')", "(3, '엽떡')", "(3, '떡당면')", "(3, '휙귀면')", "(3, '파김치')", "(3, '얼큰한')", "(3, '짜두기')", "(3, '중국당면도')", "(3, '실비김치')", "(3, '금치')", "(3, '송주불냉면')", "(2, '한우')", "(2, '흰쌀밥')", "(2, '빕밥')", "(2, '마라탕볘')", "(2, '날치알')", "(2, '치즈핵불닭')", "(5, '투썸플레이스')", "(5, '생크림')", "(4, '짜장면')", "(4, '누텔라')", "(4, '치즈볼')", "(4, '치즈롯탄')", "(4, '투썸')", "(3, '뿌링치즈볼')", "(3, '통스팸')", "(3, '명랑핫도그')", "(3, '통삼겹')", "(3, '김치')", "(3, '케이크')", "(3, '스팸')", "(3, '불닭볶음면')", "(3, '초콜릿')", "(3, '아이스크림')", "(3, '삼겹살')", "(3, '떡볶이')", "(2, '간장새우장')", "(2, '샌드위치')", "(2, '크림새우')", "(2, '티라미슈')", "(2, '뿌링핫도그')", "(2, '김치찌개')", "(2, '계란말이')", "(2, '도넛')", "(2, '바나나')", "(2, '바삭바삭한')", "(11, '떡갈비')", "(11, '삼겹살')", "(11, '김밥')", "(9, '오이소박이')", "(9, '라면')", "(8, '국밥')", "(8, '고추장찌개')", "(8, '짬뽕')", "(8, '중국당면')", "(8, '김치')", "(8, '매운')", "(8, '순대')", "(7, '대패삼겹살')", "(7, '비빔밥')", "(7, '새우튀김')", "(6, '부대찌개')", "(6, '파김치')", "(6, '매콤')", "(28, '치킨')", "(15, '신메뉴')", "(12, '떡볶이')", "(10, '뿌링클')", "(10, '치즈볼')", "(9, '매운')", "(8, '짜파게티')", "(8, '중국당면')", "(7, '마카롱')", "(6, '치킨매니아')", "(5, '탕후루')", "(5, '라면')", "(5, '치즈')", "(5, '핵불닭볶음면')", "(5, '실비김치')", "(5, '매운맛')", "(5, '엽떡')", "(5, '아이스크림')", "(5, '신메뉴')", "(12, '떡볶이')", "(10, '뿌링클')", "(10, '치즈볼')", "(9, '매운')", "(8, '짜파게티')", "(8, '중국당면')", "(7, '마카롱')", "(6, '치킨매니아')", "(5, '탕후루')", "(5, '라면')", "(5, '치즈')", "(5, '핵불닭볶음면')", "(5, '실비김치')", "(5, '매운맛')", "(5, '엽떡')", "(5, '아이스크림')", "(5,


```
from ast import literal_eval

#str을 tuple형태로 만듦.
list=[]
for i in FrequencyList:
    t = literal_eval(i)
    list.append(t)

print(list)
print(type(list[0]))
```


[(11, '치킨'), (6, '피자'), (6, '신메뉴'), (5, '간장게장'), (4, '치즈'), (4, '케이크'), (3, '막창'), (3, '떡볶이'), (3, '핫도그'), (3, '불고기'), (3, '새우튀김'), (3, '목은지'), (3, '새우장'), (3, '도미노'), (3, '감성카초'), (3, '탕수육'), (3, '교촌'), (2, '소곱창'), (2, '대창'), (2, '소시지'), (2, '라면'), (2, '짜장면'), (2, '비빔밥'), (2, '닭발'), (2, '통삼겹살'), (2, '마시멜로우'), (2, '연어회'), (2, '수육'), (10, '골조합'), (5, '집밥'), (4, '신메뉴'), (4, '간장새우'), (3, '치즈'), (3, '매콤한'), (3, '연어'), (3, '군만두'), (3, '허니콤보'), (2, '대창'), (2, '돈까스'), (2, '분식'), (2, '치킨'), (2, '찹쌀떡'), (2, '시원한'), (2, '맥주'), (2, '전복'), (2, '불닭볶음면'), (2, '흰쌀밥'), (2, '광어'), (2, '닭껍질튀김'), (2, '새우튀김'), (2, '날치알'), (2, '디저트'), (2, '통삼겹살'), (2, '짜파게티'), (2, '파김치'), (2, '꾸덕꾸덕'), (2, '중국당면'), (2, '주먹밥'), (6, '치즈볼'), (5, '라면'), (4, '스팸'), (4, '계란후라이'), (4, '김치'), (4, '연어'), (4, '떡볶이'), (3, '수육'), (3, '삼겹살'), (3, '김'), (3, '신제품'), (3, '까르보불닭'), (2, '볼고기'), (2, '반올림피자샵'), (2, '한우'), (2, '뼝'), (2, '명랑핫도그'), (2, '칠리새우'), (16, '매운'), (6, '중국당면'), (6, '청양고추'), (5, '전주진김치'), (4, '매콤한'), (4, '김치'), (4, '금치김치'), (3, '떡만둣국'), (3, '당면사리'), (3, '엽떡'), (3, '떡당면'), (3, '휙귀면'), (3, '파김치'), (3, '얼큰한'), (3, '짜두기'), (3, '중국당면도'), (3, '실비김치'), (3, '금치'), (3, '송주불냉면'), (2, '한우'), (2, '흰쌀밥'), (2, '빕밥'), (2, '마라탕볘'), (2, '날치알'), (2, '치즈핵불닭'), (5, '투썸플레이스'), (5, '생크림'), (4, '짜장면'), (4, '누텔라'), (4, '치즈볼'), (4, '치즈롯탄'), (4, '투썸'), (3, '뿌링치즈볼'), (3, '통스팸'), (3, '명랑핫도그'), (3, '통삼겹'), (3, '김치'), (3, '케이크'), (3, '스팸'), (3, '불닭볶음면'), (3, '초콜릿'), (3, '아이스크림'), (3, '삼겹살'), (3, '떡볶이'), (2, '간장새우장'), (2, '샌드위치'), (2, '크림새우'), (2, '티라미슈'), (2, '뿌링핫도그'), (2, '김치찌개'), (2, '계란말이'), (2, '도넛'), (2, '바나나'), (2, '바삭바삭한'), (11, '떡갈비'), (11, '삼겹살'), (11, '김밥'), (9, '오이소박이'), (9, '라면'), (8, '국밥'), (8, '고추장찌개'), (8, '짬뽕'), (8, '중국당면'), (8, '김치'), (8, '매운'), (8, '순대'), (7, '대패삼겹살'), (7, '비빔밥'), (7, '새우튀김'), (6, '부대찌개'), (6, '파김치'), (6, '매콤'), (28, '치킨'), (15, '신메뉴'), (12, '떡볶이'), (10, '뿌링클'), (10, '치즈볼'), (9, '매운'), (8, '짜파게티'), (8, '중국당면'), (7, '마카롱'), (6, '치킨매니아'), (5, '탕후루'), (5, '라면'), (5, '치즈'), (5, '핵불닭볶음면'), (5, '실비김치'), (5, '매운맛'), (5, '엽떡'), (5, '아이스크림'), (5,

```
result=myRdd\
    .map(lambda x:(x[1],x[0]))\
    .reduceByKey(lambda x,y:x+y)\
    .map(lambda x:(x[1],x[0]))\
    .sortByKey(False)\
    .collect()
```

```
print(result)
```

```
[(441, '신메뉴'), (424, '치킨'), (285, '매운'), (279, '라면'), (257, '떡볶이'), (229, '중국당면'), (223, '짜장면'), (210, '볼닭볶음면'), (1
92, '추억밥'), (179, '삼겹살'), (176, '탕수육'), (173, '파김치'), (171, '짜파게티'), (171, '치즈볼'), (153, '케이크'), (152, '청양고추'),
(149, '김치'), (145, '치즈'), (133, '밥'), (109, '실비김치'), (101, '새우튀김'), (101, '김밥'), (99, '피자'), (86, '참통'), (96, '항'), (9
6, '마카롱'), (96, '양념치킨'), (96, '챕사이신'), (89, '군만두'), (84, '튀김'), (83, '계란후라이'), (83, '배추김치'), (80, '깨롭한'), (78,
'볶음밥'), (73, '보밥'), (72, '매운맛'), (69, '청량핫도그'), (68, '시원한'), (67, '아이스크림'), (67, '계란찜'), (66, '집밥'), (65, '돈까
스'), (65, '뿌링클'), (64, '총각김치'), (63, '스윔'), (61, '짤버거'), (59, '비빔밥'), (59, '옐기떡볶이'), (56, '순대'), (56, '죽발'), (54,
'치즈스틱'), (54, '신전떡볶이'), (52, '새우'), (50, '염떡'), (50, '작두기'), (46, '얼큰한'), (45, '숭주불냉면'), (44, '롯데리아'), (44, '초
밥'), (43, '김치찌개'), (43, '중식'), (42, '당면사리'), (42, '스테이크'), (41, '허니콤보'), (41, '분식'), (41, '닭다리'), (41, '바삭한'),
(40, '간장계장'), (40, '스파게티'), (39, '생크림'), (38, '떡도날드'), (38, '치킨집'), (35, '핫도그'), (34, '된장찌개'), (33, '파스타'), (3
3, '소세지'), (33, '한우'), (33, '쟁반국수'), (32, '마라바게트'), (32, '간장치킨'), (31, '네네치킨'), (31, '짤라'), (31, '꿀'), (31, '가리
비'), (31, '마라탕'), (30, '알타리김치'), (29, '떡갈비'), (29, '물냉면'), (28, '뿌링치즈볼'), (28, '막국수'), (27, '계란말이'), (27, '얼라
면'), (27, '치킨먹방'), (26, '한식'), (26, '오돌뼈'), (25, '김치'), (25, '부대찌개'), (25, '탕후루'), (25, '제육볶음'), (25, '육회'), (24,
'닭발'), (24, '도시락'), (24, '모듬튀김'), (24, '페리카나'), (23, '신제품'), (23, '통삼겹'), (23, '치밥'), (23, '멕시카나'), (23, '한식먹
방'), (23, '함소터치'), (23, '뚜레쥬르'), (23, '과자'), (22, '초콜릿'), (22, '김치볶'), (22, '힐하사소세지'), (21, '꿀조합'), (21, '전복'),
(21, '피자헛'), (21, '고기'), (21, '과일'), (21, '너구리'), (20, '수육'), (20, '까르보나라'), (20, '행이버섯'), (19, '연어'), (19, '핵불닭
볶음면'), (19, '버거킹'), (19, '배추볼김치'), (19, '우동사리'), (19, '컵라면'), (19, '당면'), (19, '소주'), (19, '동치치한'), (18, '햇일'),
(18, '자메이카'), (18, '탕죽발'), (18, '김피탕'), (17, '까르보불닭'), (17, '감장김치'), (17, '칩챕먹방'), (17, '왕한우'), (17, '몰리져스'),
(17, '꿀면'), (17, '고스트피피'), (17, '감자튀김'), (17, '무뼈닭발'), (17, '하겐다즈'), (17, '후라이드치킨'), (16, '대창'), (16, '처돌박
이'), (16, '황금올리브'), (16, '딸기'), (16, '바삭바삭'), (16, '우동'), (16, '주꾸미'), (15, '한우'), (15, '모짜렐라'), (15, '설빙'), (15,
'양념계장'), (15, '라면사리'), (15, '자메이카치킨'), (15, '문어'), (15, '옐기닭볶음탕'), (14, '마시멜로우'), (14, '샌드위치'), (14, '휘핑크
림'), (14, '부추김치'), (14, '낙지'), (14, '곱창'), (14, '김치피자창수육'), (13, '막창'), (13, '디저트'), (13, '김'), (13, '천주진김치'),
(13, '달달한'), (13, '떡'), (13, '비빔면'), (13, '매운닭발'), (13, '아이스크림케이크'), (13, '오잉탕'), (13, '신라면'), (13, '후라이드'),
(12, '뿌링클닭다리'), (12, '초코케이크'), (12, '비엔나'), (12, '오징어'), (12, '핵불닭'), (12, '통통'), (12, '불족발'), (12, '고촌레드콤보
```

```
tDf=resultDf.withColumn("Food",resultDf['_2'].cast("string")).drop('_2')
ResultDf=tDf.withColumn("Frequency",tDf['_1'].cast("integer")).drop('_1')
```

```
ResultDf.show()
```

```
+----------+---------+
|      Food|Frequency|
+----------+---------+
|    신메뉴|      441|
|      치킨|      424|
|      매운|      285|
|      라면|      279|
|    떡볶이|      257|
|  중국당면|      229|
|    짜장면|      223|
|볼닭볶음면|      210|
|    추억밥|      192|
|    삼겹살|      179|
|    탕수육|      176|
|    파김치|      173|
|  짜파게티|      171|
|    치즈볼|      171|
|    케이크|      153|
|  청양고추|      152|
|      김치|      149|
|      치즈|      145|
|      밥  |      123|
|  실비김치|      109|
+----------+---------+
only showing top 20 rows
```

4) 빈도수별로 데이터결과를 시각화 하기위해 Frequency.txt에서 compile,sub정규표현식으로 음식이름data만 추출해 only_Food.txt로 저장하였다. 또한 동의어를 처리하고 싶어 replace정규표현식을 사용했다. wordcloud모듈을 사용하여 동의어 처리된 음식이름data를 replace_Food.txt로 저장하여, txt에서 data의 빈도수를 자동 계산해 빈도수순으로 data를 시각화하였다.(Konply, pygame등의 모듈을 이용)

wordcloud

```
#wordcloud를 위해 한글만 남김.
import re
b=[]
for j in FrequencyList:
    text=re.compile('[^ ㄱ-ㅣ가-힣]+')
    result = text.sub('', j)
    b.append(result)
print(b)
print(type(b))
```

```
['  치킨', '  피자', '  신메뉴', '  간장계장', '  치즈', '  케이크', '  막창', '  떡볶이', '  핫도그', '  불고기', '  새우튀김', '  묵은지', '  새우
장', '  도미노', '  감성커츠', '  탕수육', '  고촌', '  소금창', '  대창', '  소시지', '  라면', '  짜장면', '  비빔밥', '  닭발', '  통삼겹살', '
마시멜로우', '  연어회', '  수육', '  꿀조합', '  집밥', '  신메뉴', '  간장새우', '  치즈', '  매콤한', '  연어', '  군만두', '  허니콤보', '  대
창', '  돈까스', '  분식', '  치킨', '  참쌀떡', '  시원한', '  맥주', '  전복', '  볼닭볶음면', '  흰쌀밥', '  광어', '  닭껍질튀김', '  새우튀김',
'  날치알', '  디저트', '  통삼겹살', '  짜파게티', '  파김치', '  꾸덕꾸덕', '  중국당면', '  주먹밥', '  치즈볼', '  라면', '  스윔', '  계란후라
이', '  김치', '  연어', '  떡볶이', '  수육', '  삼겹살', '  김', '  신제품', '  까르보불닭', '  불고기', '  반올림피자샵', '  한우', '  뽕', '  명
랑핫도그', '  칠리새우', '  매운', '  중국당면', '  청양고추', '  천주진김치', '  매콤한', '  김치', '  금치김치', '  떡만둣국', '  당면사리', '
염떡', '  떡당면', '  훠궈면', '  파김치', '  얼큰한', '  작두기', '  중국당면도', '  실비김치', '  금치', '  숭주불냉면', '  한우', '  흰쌀밥', '
쌉밥', '  마라탕에', '  날치알', '  치즈핵불닭', '  투썸몰레이스', '  생크림', '  짜장면', '  누텔라', '  치즈볼', '  치즈폭탄', '  루빙', '  뿌링
치즈볼', '  통스팸', '  명랑핫도그', '  통삼겹', '  김치', '  케이크', '  스팸', '  볼닭볶음면', '  초콜릿', '  아이스크림', '  삼겹살', '  떡볶
이', '  간장새우장', '  샌드위치', '  크림새우', '  타라미슈', '  뿌링핫도그', '  김치찌개', '  계란말이', '  도넛', '  바나나', '  바삭바삭한', '
떡갈비', '  삼겹살', '  김밥', '  오이소박이', '  라면', '  국밥', '  고추장찌개', '  깡통', '  중국당면', '  김치', '  매운', '  순대', '  대패삼겹
살', '  비빔밥', '  새우튀김', '  부대찌개', '  파김치', '  매콤', '  치킨', '  신메뉴', '  떡볶이', '  뿌링클', '  치즈볼', '  매운', '  짜파게티',
'  중국당면', '  마카롱', '  치킨매니아', '  탕후루', '  라면', '  치즈', '  핵불닭볶음면', '  실비김치', '  매운맛', '  염떡', '  아이스크림', '
떡도날드', '  숭주불냉면', '  모짜렐라', '  육개장', '  네네치킨', '  불맥타이거새우', '  처갓집', '  파스타', '  도시락', '  파김치', '  까르보나
라', '  통삼겹', '  아이스크림', '  케이크', '  치킨', '  멸리', '  마카롱', '  치즈', '  치즈볼', '  설빙', '  시원한', '  닭다리', '  탕수육', '
짜장면', '  스윔', '  초코', '  주먹밥', '  얼큰한', '  계란후라이', '  매운', '  치즈스틱', '  초콜릿', '  부드러운', '  핫도그', '  달달한', '  폰
즈폰즈한', '  버거킹', '  옐기떡볶이', '  뿌링클', '  김밥', '  신전떡볶이', '  생크림', '  소세지', '  츄파스', '  마쉬멜로우', '  바삭한', '
딱', '  신메뉴', '  닭다리', '  옐기떡볶이', '  처갓집', '  멸라면', '  비비큐', '  고봉밥', '  중국당면', '  양념치킨', '  치즈볼', '  떡볶이', '
```

```
#음식이름만 파일에 저장
f=open("C:\Users\minle\OneDrive\Desktop\only_Food.txt",'w',-1,"utf-8")
```

```
7fdb8bbd05cd8f337ee05f4d1509c226b0
```

```
#replace 이용해서 동의어 처리
FoodRdd=FoodRdd.map(lambda x:x.replace(u"묵은지",u"김치").replace(u"파김치",u"김치").replace(u"천주진김치",u"김치").replace(u"금치김치",u"김
ListFood=FoodRdd.collect()
print(ListFood)
```

```
[' 치킨', ' 피자', ' 신메뉴', ' 계장', ' 치즈', ' 케이크', ' 막창', ' 떡볶이', ' 핫도그', ' 불고기', ' 새우튀김', ' 김치', ' 새우장', ' 도
미노', ' 감성커츠', ' 탕수육', ' 고촌', ' 소금창', ' 대창', ' 소시지', ' 라면', ' 짜장면', ' 비빔밥', ' 닭발', ' 통삼겹살', ' 마시멜로우',
' 연어회', ' 수육', ' 꿀조합', ' 집밥', ' 신메뉴', ' 간장새우', ' 치즈', ' 매콤한', ' 연어', ' 군만두', ' 허니콤보', ' 대창', ' 돈까스', '
분식', ' 치킨', ' 참쌀떡', ' 시원한', ' 맥주', ' 전복', ' 볼닭볶음면', ' 흰쌀밥', ' 광어', ' 닭껍질튀김', ' 새우튀김', ' 날치알', ' 디저
트', ' 통삼겹살', ' 짜파게티', ' 김치', ' 꾸덕꾸덕', ' 중국당면', ' 주먹밥', ' 치즈볼', ' 라면', ' 스윔', ' 계란후라이', ' 김치', ' 연어',
' 떡볶이', ' 수육', ' 삼겹살', ' 김', ' 신제품', ' 까르보불닭', ' 불고기', ' 피자', ' 한우', ' 뽕', ' 명랑핫도그', ' 칠리새우', ' 매운', '
중국당면', ' 청양고추', ' 김치', ' 매콤한', ' 김치', ' 김치', ' 떡만둣국', ' 당면사리', ' 염떡', ' 떡당면', ' 훠궈면', ' 김치', ' 얼큰한',
' 작두기', ' 중국당면도', ' 김치', ' 김치', ' 숭주불냉면', ' 한우', ' 흰쌀밥', ' 쌉밥', ' 마라탕에', ' 날치알', ' 치즈핵불닭', ' 투썸몰레이
스', ' 생크림', ' 짜장면', ' 누텔라', ' 치즈볼', ' 치즈폭탄', ' 루빙', ' 뿌링치즈볼', ' 통스팸', ' 명랑핫도그', ' 통삼겹', ' 김치', ' 케이
크', ' 스팸', ' 볼닭볶음면', ' 초콜릿', ' 아이스크림', ' 삼겹살', ' 떡볶이', ' 간장새우장', ' 샌드위치', ' 크림새우', ' 타라미슈', ' 뿌링핫
도그', ' 김치찌개', ' 계란말이', ' 도넛', ' 바나나', ' 바삭바삭한', ' 떡갈비', ' 삼겹살', ' 김밥', ' 오이소박이', ' 라면', ' 국밥', ' 고추
장찌개', ' 깡통', ' 중국당면', ' 김치', ' 매운', ' 순대', ' 대패삼겹살', ' 비빔밥', ' 새우튀김', ' 부대찌개', ' 김치', ' 매콤', ' 치킨', '
신메뉴', ' 떡볶이', ' 뿌링클', ' 치즈볼', ' 매운', ' 짜파게티', ' 중국당면', ' 마카롱', ' 치킨', ' 탕후루', ' 라면', ' 치즈', ' 핵불닭볶음
면', ' 김치', ' 매운맛', ' 염떡', ' 아이스크림', ' 떡도날드', ' 숭주불냉면', ' 모짜렐라', ' 육개장', ' 치킨', ' 불맥타이거새우', ' 처갓집',
' 파스타', ' 도시락', ' 김치', ' 까르보나라', ' 통삼겹', ' 아이스크림', ' 케이크', ' 치킨', ' 멸리', ' 마카롱', ' 치즈', ' 치즈볼', ' 설
빙', ' 시원한', ' 치킨', ' 탕수육', ' 짜장면', ' 스윔', ' 초코', ' 주먹밥', ' 얼큰한', ' 계란후라이', ' 매운', ' 치즈스틱', ' 초콜릿', ' 부
드러운', ' 핫도그', ' 달달한', ' 폰즈폰즈한', ' 버거킹', ' 옐기떡볶이', ' 뿌링클', ' 김밥', ' 신전떡볶이', ' 생크림', ' 소세지', ' 츄파스',
' 마쉬멜로우', ' 바삭한', ' 딱', ' 신메뉴', ' 치킨', ' 옐기떡볶이', ' 처갓집', ' 멸라면', ' 비비큐', ' 고봉밥', ' 중국당면', ' 치킨', ' 치
즈볼', ' 떡볶이', ' 명랑핫도그', ' 회', ' 짜장면', ' 처돌박이', ' 계장', ' 치밥', ' 볼닭볶음면', ' 마카롱', ' 피자', ' 꿍카통', ' 부대찌
개', ' 황금올리브', ' 비빔면', ' 멕시카나', ' 볶음밥', ' 계장', ' 새우장', ' 청양고추', ' 중국당면', ' 검사이신', ' 매운', ' 숭주불냉면', '
김치', ' 매운맛', ' 가리비', ' 반호이', ' 맹초', ' 볼닭볶음면', ' 밥', ' 한국당면', ' 짜장면', ' 김치', ' 오징어짓갈', ' 비빔밥', ' 새우',
' 라면', ' 베트남땡초가루', ' 삼겹살', ' 총각김치', ' 전복', ' 용수당면', ' 김장김치', ' 주먹밥', ' 챕사이신비빔밥', ' 소면', ' 김치', ' 김
치찜', ' 김치', ' 매운', ' 집밥', ' 밥', ' 김치', ' 한식먹방', ' 김치', ' 김치에', ' 매운맛', ' 집밥먹방', ' 배추볼김치', ' 시원한', ' 라
면', ' 회', ' 얼큰한', ' 비빔밥', ' 떡볶이', ' 김치', ' 육배기', ' 총각김치', ' 김치에', ' 치킨', ' 짜파게티', ' 계란후라이', ' 김치', ' 검
사이신', ' 치즈', ' 뜨끈한', ' 치킨', ' 뽕', ' 당면사리', ' 라면', ' 신메뉴', ' 주먹밥', ' 치즈볼', ' 새우튀김', ' 킬바사소세지', ' 생크
림', ' 중국당면', ' 소세지', ' 우동사리', ' 케이크', ' 총각김치', ' 라면사리', ' 명랑핫도그', ' 왕만두', ' 매운', ' 휘핑크림', ' 마카롱', '
밥', ' 볼닭볶음면', ' 후랑콜치킨', ' 케이크', ' 얼라면', ' 허니콤보', ' 짜파게티', ' 뿌링치즈볼', ' 멸티져스', ' 배추김치', ' 김치', ' 짜파
게티', ' 작두기', ' 알타리김치', ' 삼겹살', ' 쪗일', ' 한식', ' 제육볶음', ' 부추김치', ' 롯데리아', ' 비엔나', ' 스파게티', ' 얼무김치', '
비빔밥', ' 햇반', ' 짜장밥', ' 소고기', ' 총각김치', ' 김밥', ' 신메뉴', ' 생선까스', ' 오징어초무침', ' 모듬튀김', ' 떡갈비', ' 컵라면', '
```

```
#replace후 저장
f=open("C:₩Users₩jmlee₩Code₩replace_Food.txt",'w',-1,"utf-8")
A=len(List)
print (A)
for i in range (0,A):
    f.writelines(str(List[i]))
    f.writelines("₩n")
f.close()
```

```
1320
```

```
FoodRdd_=spark.sparkContext.textFile(os.path.join("C:₩Users₩jmlee₩Code₩replace_Food.txt"))
```

```
FoodList_=FoodRdd_.collect()
# print(FrequencyList)
FoodList_[0]
len(FoodList_)
```

```
1320
```

```
# -*- encoding:utf8 -*-
import pytagcloud
import random
import webbrowser
from konlpy.tag import Twitter
from collections import Counter
import matplotlib.font_manager as fm

######################################################
# get_tags 새함수 만듦:
# 기능 1. 댓글별로 명사만 추출
# 기능 2. 명사 빈도수 집계
# 기능 3. 단어구름에 표시할 명사에 3가지 시각화 속성
#        (색상'color', 단어'tag', 크기'size')부여

# 입력변수- text : 댓글, ntags : 표시할 단어수, multiplier : 크기가중치
def get_tags(text, ntags=20, multiplier=2):
    t = Twitter()
    nouns = []

# 모든 댓글에서 명사만 추출하고 nouns변수에 누적해 저장함
    for sentence in text:
        for noun in t.nouns(sentence):
            nouns.append(noun)
            # 각 명사별로 빈도계산
            count = Counter(nouns)
    # n : 명사, c : 빈도
    return [{'color': color(),'tag':n,'size':2*c*multiplier} for n,c in count.most_common(ntags)]

# draw_cloud 새함수 만듦:
# 기능 1. pytagclud 모듈을 사용해 단어구름 이미지를 만듦
# 기능 2. 단어구름 이미지를 파일로 저장함
# 기능 3. 화면에 단어구름을 표시함

# 입력변수 tags : get_tags()에서 리턴되는 color, tag, size(이미지크기) 값이 전달됨.
# fontname : Noto Sans CJK - 한글폰트
def draw_cloud(tags, filename, fontname='BMHANNA',size1 = (1300,800)):
    pytagcloud.create_tag_image(tags,filename,fontname=fontname,size=size1)
```

3. 프로젝트 결과

```
tDf=resultDf.withColumn("Food",resultDf['_2'].cast("string")).drop('_2')
ResultDf=tDf.withColumn("Frequency",tDf['_1'].cast("integer")).drop('_1')
```

```
ResultDf.show()
```

```
+---------+---------+
|     Food|Frequency|
+---------+---------+
|   신메뉴|      441|
|     치킨|      424|
|     매운|      285|
|     라면|      279|
|   떡볶이|      257|
| 중국당면|      229|
|   짜장면|      223|
|불닭볶음면|      210|
|   주먹밥|      192|
|   삼겹살|      179|
|   탕수육|      176|
|   파김치|      173|
| 짜파게티|      171|
|   치즈볼|      171|
|   케이크|      153|
| 청양고추|      152|
|     김치|      149|
|     치즈|      145|
|      밥|      123|
| 실비김치|      109|
+---------+---------+
only showing top 20 rows
```

왼쪽은 raw데이터로 구한 음식이름별 빈도수이고, 오른쪽은 Frequency.txt의 data에서 정규표현식을 거쳐 음식이름별 빈도수를 토대로 데이터 시각화한 모습이다. 치킨, 매운, 라면, 떡볶이 등 순으로 음식빈도가 많이 나왔다. 또한 프로젝트를 하면서 느낀점은 웹상에 데이터는 정말 깨끗하지 않은 형태이고, 다양한 방법으로 여러번 정제해야 가치있는 데이터를 얻을 수 있겠다고 생각했다. 내가 생각한 방법으로 정제하고 분석하려면 data형태가 맞지 않거나, 원하지 않는 data까지 섞여있는 등 다양한 문제가 발생했다. 이런 문제를 해결하고 고민하는 과정이 유익했던 것 같다. 또한 수집과 정제와 분석을 잘한다면 무한한 가치가 있고, 다양한 의미가 숨겨져 있을 것이라는 기대를 하게 되었다.