

개인  
프로젝트

# 더 나은 치킨을 먹자

| Chicken2Vec |

— • —

멀티캠퍼스 C반 서상원

목표.

# 우리동네에는 무슨치킨을 많이먹을까?

요기요(Yogiyo)의 리뷰/주문 데이터를 크롤링

```
from selenium import webdriver
from wordcloud import WordCloud
import matplotlib.pyplot as plt
import pandas as pd
import time
```

```
## 데이터 가져오기 ##
home_name_list = []
review_list = []
ordered_list = []

one_wait = 1
second_wait = 2
ten_wait = 10

middle_point = 0

for arr in range(9,50):
    address = "서울특별시 서대문구 북가좌동 327-3 경성하이빌"
    driver = webdriver.Chrome(executable_path="chromedriver.exe")
    driver.implicitly_wait(second_wait)
    Yogiyo_URL = "https://www.yogiyo.co.kr/mobile/#/"
    driver.get(Yogiyo_URL)

    time.sleep(1)
    #내 주소 입력
    input_field = driver.find_element_by_xpath('//*[@id="search"]/div/form/input')
    driver.execute_script("arguments[0].value = ''", input_field)
    driver.implicitly_wait(second_wait)
    driver.find_element_by_xpath('//*[@id="search"]/div/form/input').send_keys(address)
    driver.implicitly_wait(second_wait)
    driver.find_element_by_xpath('//*[@id="button_search_address"]/button[2]').click()
    driver.implicitly_wait(second_wait)
    # 치킨선택
    path = '//*[@id="category"]/ul/li[5]'
    element = driver.find_element_by_xpath(path)
    driver.execute_script("arguments[0].click();", element)

    #음식점 개수
    restaurant_count = int(driver.find_element_by_css_selector("#restaurant_count").text)
    scroll_count = int((restaurant_count/20))

    # Get scroll height
    last_height = driver.execute_script("return document.body.scrollHeight")

    while True:
        # Scroll down to bottom
        driver.execute_script("window.scrollTo(0, document.body.scrollHeight);")
```

● 사용자에게 주소를 입력받는다

25%

\*요기요에서 제공하는 GPS시스템으로 주소가 자동입력되면 사용자 주소를 무시하고 진행한다.

● 음식 카테고리중 '치킨' 메뉴로  
이동하여 스크롤을 하단까지 내린후  
모든 음식점에 대한 리뷰페이지를 로드한다.

50%

\* 서대문구 북가좌동 기준으로 150개의 치킨집이 있다.

● 각 치킨집의 리뷰페이지로  
이동 후, 스크롤을 하단까지 내려  
모든 리뷰를 크롤링한다.

100%

\* 각 치킨집당 상이한 개수로 총 25000여건의 리뷰를 크롤링하였다.

**질문.**

## 각 치킨집 별로 무슨 치킨이 많이 팔렸을까?

## BHC치킨집 리뷰를 대상으로 워드클라우드 생성

```
from wordcloud import WordCloud
from PIL import Image
import matplotlib.pyplot as plt
import numpy as np
import time

### 모든 Name List를 Join 할 것 ## 우리동네에는 무슨차린이 제일 많이 팔렸을까? ##
for result_DF in result_DF_list:
    result_text = " ".join(result_DF['object'])
    dak_mask = np.array(Image.open("dak.jpg"))

    wordclo = WordCloud(background_color="white", mask=dak_mask, font_path="C:\\Windows\\Fonts\\HMKMAHD.ttf", max_words=len(result_DF))
    wordclo = wordclo.generate(result_text)

    plt.figure(figsize=(20,20))
    plt.imshow(wordclo, interpolation="bilinear")
    plt.axis("off")
    break
```

서대문구 북가좌동의 BHC치킨집에서는  
부링클치킨이 많이 팔리는것을 알 수 있었다.



**질문.**

서대문구 북가좌동 전체 치킨집에서는 무슨메뉴가 가장많이 팔렸을까?

## 크롤링된 전체 치킨집 주문상품에 대하여 워드 클라우드 생성



질문.

# 치킨을 추천받을 수 있을까?

전체 리뷰데이터를 원형으로 형태소분석한 후, 명사와 형용사에 대해서 word2vec사전을 생성한다.

```
## 원형 추출과 매핑되는 메뉴 Dataframe ###
Compare_Chicken_dataframe = pd.DataFrame({"review":sentence_list , "item": Total_Chicken_menu_list})
Compare_Chicken_dataframe = Compare_Chicken_dataframe.dropna()
Compare_Chicken_dataframe
```

	review	item
0	[부, 립을, 감추, 이다]	부킹클BHC
1	[마, 라칸, 양념, 물다, 마르다, 할, 도, 너무, 막하다, 아쉽다, 집...	마라칸BHC
2	[너무, 맛있다, 잘, 먹다]	순살부킹클BHC
3	[]	부킹클BHC
4	[부킹, 감자, 맛있다]	해바라기팟프라이드치킨BHC
5	[진짜하다, 맛, 도, 좋다]	부킹클다리,BHC
6	[튀김, 도, 좋다, 너무, 맛있다]	부킹클날개,BHC
7	[맛있다]	부킹클BHC
8	[부킹클, 은, 먹다, 맛있다, 식지, 않다, 파릇하다, 잘, 먹다]	부킹클 + 팔간스먹BHC
9	[사이드, 메뉴, 만, 맛있다, 깔끔하다, 포장, 하다, 배달, 서다, 조아영]	팔간스먹BHC
10	[순, 살프, 순식간, 다, 먹다, 맛있다]	순살부킹클BHC
11	[맛있다, 비, 안, 오다, 주문, 하다, 기사, 남, 복다, 비, 많이, 맞다, ...]	부킹클HOTBHC
12	[아, 좋아하다, 썩, 립을]	부킹클BHC
13	[완다리, 많이, 다이어트, 하다]	부킹클 + 치즈볼BHC
14	[배달, 굿, 이구, 요, 부, 립을, 치즈볼, 잘, 바삭, 하다, 맛있다, 먹다]	치하오 + 부킹치즈볼BHC
15	[부, 립을, 잘, 하다, 얇이, 맛있다, 치킨, 너무, 튀기다, 너무, 막막하다,....]	부킹클BHC
16	[배달, 엄청, 팔르다, 분도, 안되다, 오다, 부, 립을, 핫도그, 도, 존득존,....]	부킹클BHC
17	[맛있다]	부킹클BHC
18	[바람, 불다, 날씨, 이다, 자다, 배달, 해주다, 감사하다, 치킨, 도, 넘다,....]	부킹클BHC
19	[최고]	부킹클BHC
20	[맛있다, 마르다, 할, 좋다]	마라칸BHC
21	[요기, 요, 배달, 첫, 주문, 이다, 배달, 도, 빠르다, 치킨, 도, 마, 정...]	순살부킹클BHC
22	[좋다, 좋다, 좋다, 맛있다]	해바라기프라이드치킨BHC
23	[맛있다, 후루룩, 폭박, 먹다, 사이드, 만, 끝맛]	부킹치즈볼BHC
24	[언제나, 깔끔하다, 배달, 맛있다, 먹다]	순살부킹클BHC
25	[은, 일리다, 맛있다, 먹다]	부킹클 + 치즈볼BHC
26	[먹다, 닭, 냄새, 나다, 위, 부리다, 가루, 먹다, 그렇다, 뭘, 예는, 이럴...]	부킹클BHC
27	[맛, 나다, 먹다]	해바라기프라이드치킨BHC
28	[집, 은, 곱질, 엄청, 두껍다, 살프음, 인데, 여기다, 진짜, 최고, 살도얇루...]	해바라기닭다리프라이드치킨BHC
29	[오글, 도, 역시, 빠르다, 빠르다, 맛있다, 아들, 좋아하다]	부킹클다리,BHC
...	...	...
3508	[느끼하다, 맛있다, 알, 많다]	글드치즈치킨mymy
3609	[알도, 많다, 맛, 도, 괜찮다]	어니언치킨mymy
3610	[배달, 은, 약속, 한, 오다, 치즈, 너무, 끝다, 치킨, 살, 도, 너무, 각...	글드치즈치킨mymy

질문.

## 치킨별 특징을 방해하는 단어들을 어떻게 처리할까?

치킨의 메뉴와 상관없는 불용어 (EX: 주문, 맛있다, 먹다)와 한국어 불용어에 대해서 필터링 수행

```
In [16]: len(stop_word_list)
```

```
Out[16]: 675
```

```
chicken_stopwords = ["배달", "맛", "여기", "주문", "치킨", "맛있다", "먹다"]
```

한국어 불용어 675개와 치킨리뷰만을 위한 불용어를 생성하여 형태소 분석 결과에서 지워준다.

질문.

## 치킨을 추천받을 수 있을까?

만들어진 사전에 대해서 Word2Vec모델을 생성한다.

이후 사용자에게 키워드를 입력받아서, 그와 유사한 키워드를 포함하고 있는 리뷰의 주문상품을 리스트로 리턴한다.

```
: from gensim.models import Word2Vec
model = Word2Vec(sentences=Chicken2Vec, size=1000, window=10, min_count=3, workers=4, sg=0)

: keyword = input()
치즈볼

: search_base=model.wv.most_similar(keyword)
print(search_base)

[('치킨', 0.9999840259552002), ('주문', 0.9999839067459106), ('분', 0.9999833703041077), ('처음', 0.9999830722808838),
('살', 0.9999830722808838), ('아니다', 0.9999830722808838), ('여기', 0.9999829530715942), ('닭', 0.9999828338623047), ('무',
0.9999827146530151), ('때', 0.9999827146530151)]

: ### a에서 명사 활용사 원본을 가져오는 부분 ###
search_base_list = []
for search_base_item in search_base:
    search_base_list.append(search_base_item[0])

surgest_item = []

for compare_one in Compare_Chicken_dataframe["review"]:
    for search_base_item in search_base_list:
        if(search_base_item in compare_one):
            surgest_item.append(search_base_item)

surgest_item = set(surgest_item)
surgest_item = list(surgest_item)
surgest_item

: ['아니다', '분', '처음', '닭', '살', '주문', '치킨', '무']
```

결과.

## '치즈볼' 키워드 입력에 대한 치킨추천

결과는 좋지않았다!

```
In [80]: item = []
for number in range (len(Compare_Chicken_dataframe)):
    Compare = Compare_Chicken_dataframe["review"][number]
    for surgest_item_one in surgest_item:
        if(surgest_item_one in Compare):
            item.append(Compare_Chicken_dataframe["item"][number])
item = set(item)
item
```

```
{ '잇 만만치킨 (잇후라이드 + 잇간성) jeadam',
  '잇 후라이드치킨 jeadam',
  '잇소이바베큐BHC',
  '잇순살 치킨 jeadam',
  '잇순살치킨 jeadam',
  '잇슈프림 양념치킨 cheagotzip',
  '잇후라이드 치킨 jeadam',
  '해바라기 잇후라이드치킨BHC',
  '해바라기 후라이드치킨BHC',
  '허니순살[R] kochyn',
  '허니오리지날 kochyn',
  '허니콤보 kochyn',
  '황금올리브치킨BBQ',
  '후라이드 반 / 양념 반BHC',
  '후라이드 치킨 jeadam',
  '후라이드치킨 jeadam',
  '후라이드치킨 mymy',
  '후라이드 + 뿌링 치즈볼BHC',
  '(할인) 뿌링클BHC',
  '(할인) 후라이드BHC' }
```





## 왜 그럴까?

### Hypothesis

치즈볼에 대한 치킨  
추천은 부링클BHC가 되어야한다.

### Reason 1

치킨 불용어가 제대로 처리되지  
않았다.



치킨 리스트가  
올바르게 추천되지 않은  
이유?

### Reason 2

사용자의 리뷰에서 명사  
형용사를 추출하여도 치킨의 특색을 반영할 만한 단어를 뽑을 수 없었다.  
Raw data의 문제..

### Reason 3

데이터 크롤링의 페이지 리뷰를 참조하는 시간을 5초로 주었는데  
25000건의 리뷰를 크롤링하기위해선 35시간이 걸린다.  
모델 구현에는 전체 크롤링데이터를 넣지 않아서 정확하지 않은 것 같다.

A red speech bubble with a white outline, containing the Korean text '감사합니다!'.

감사합니다!