

# 치킨 추천시스템

## - 요기요 리뷰 데이터 크롤링

---

20기 김주은

20기 안지완

20기 윤선영

20기 최영우

# 목차

- 주제 소개
- 데이터 소개 & 전처리
- 통계를 이용한 추천
  - 평균평점을 이용한 추천
  - 코사인 유사도 기반
- Matrix factorization
- 리뷰 데이터 분석
  - 리뷰 전처리
  - 리뷰 기반 추천시스템
- 인사이트
- 의의 및 한계
- 느낀 점
- 결론

# 주제 소개

---

- 요기요의 **치킨 리뷰 데이터**를 활용
- 신촌의 다양한 치킨 가게 중 몇 가지의 가게를 추천 시스템을 이용해 추천  
반대로 몇 가지의 치킨 가게들의 보완할 점을 제시
- 많은 리뷰들을 분석해 치킨 가게를 고르는 것을 도움

# 데이터 소개

- 요기요의 치킨 리뷰를 셀레니움을 이용한 크롤링으로 데이터를 수집
- 약 50개 정도의 가게
- 약 13000개의 리뷰



```
#음식점 홈페이지에서 리뷰보기를 클릭하기
browser.find_element(By.XPATH, '/html/body/div[6]/div[2]/div[1]/ul/li[2]/a').click()
time.sleep(3)

AAA = int(browser.find_element(By.XPATH, '/html/body/div[6]/div[2]/div[1]/ul/li[2]/a/span').text)

if AAA > 100:

    #음식점 이름을 저장하기
    name = browser.find_element(By.XPATH, '/html/body/div[6]/div[2]/div[1]/div[1]/div[1]/span').text
    temp.append(name)

    start = 1
    #리뷰 한 음식점에서 100개를 가져옴
    #매를 들어 200개를 가져오고 싶다면 아래 while에서 while <= 200으로 하면됨
    while len(temp) <= 100:
        AAA = browser.find_elements(By.CSS_SELECTOR, '.list-group-item.star-point.ng-scope')
        for i in range(0, 9+1):
            temp.append({start + i : AAA[i].get_attribute('innerText')})
        time.sleep(3)
        start += 10

        browser.find_element(By.CSS_SELECTOR, "body").send_keys(Keys.END) # end키를 통해 스크롤을 맨 아래로 내림
        time.sleep(1)

    # 더보기 클릭
    more_click = browser.find_element(By.CSS_SELECTOR, 'li.list-group-item.btn-more')
    more_click.click()
    time.sleep(3)
    result.append(temp)
#다시 치킨집 리스트로 이동하기
browser.back()
time.sleep(3)
```

# 데이터 전처리













## 1. 가게

- 치킨집이 아닌 다른 식당 제거

## 2. 맛, 양

- 각각 1~5점

- 이상 없음

요기요 등록 음식점 ?		
	<b>굽네치킨&amp;피자-이대역점</b> ★ 4.9   리뷰 1881   사장님댓글 608 요기서결제   12,000원 이상 배달	55~65분
	<b>롯데리아-홍대점</b> ★ 4.8   리뷰 225   사장님댓글 109 요기서결제   11,000원 이상 배달	28~38분
	<b>네네앤봉구스-이대역점</b> ★ 4.6   리뷰 39   사장님댓글 1 요기서결제   8,000원 이상 배달	55~65분
	<b>롯데리아-신촌로터리점</b> ★ 4.6   리뷰 6324   사장님댓글 101 요기서결제   11,000원 이상 배달	36~46분
	<b>네네치킨-이대역점</b> ★ 4.8   리뷰 235 요기서결제   17,500원 이상 배달 4,000원 할인	57~67분
	<b>킹케밥-홍대점</b> ★ 4.4   리뷰 1672   사장님댓글 172 요기서결제   14,900원 이상 배달	53~63분
	<b>파스타어때-마포신촌본점</b> ★ 4.7   리뷰 108 요기서결제   5,000원 이상 배달	50~60분
	<b>킹케밥</b> ★ 4.2   리뷰 186 요기서결제   19,900원 이상 배달	56~66분
	<b>후라이드참잘하는집-서강대점</b> ★ 4.8   리뷰 3202   사장님댓글 3201 요기서결제   11,000원 이상 배달	45~55분
	<b>청년치킨-서교점</b> ★ 4.9   리뷰 2687 요기서결제   15,000원 이상 배달	45~55분
	<b>베이직프라이드치킨-신촌직영점</b> ★ 5.0   리뷰 769   사장님댓글 750 요기서결제   15,000원 이상 배달	45~55분
	<b>푸라닭-신수점</b> ★ 4.8   리뷰 1769   사장님댓글 2 요기서결제   14,900원 이상 배달	45~55분

# 데이터 전처리

## 3. 날짜

- 2일 전, 3일 전을 날짜로 바꿈

가게	맛	양	날짜
후라이드참잘하는집-서강대점	5	5	2일 전
후라이드참잘하는집-서강대점	5	5	3일 전
후라이드참잘하는집-서강대점	5	5	4일 전
후라이드참잘하는집-서강대점	5	5	5일 전
후라이드참잘하는집-서강대점	5	5	5일 전

## 4. 주문

- 옵션 삭제
- 결측치 삭제

가게	맛	양	날짜	주문
교촌치킨-신촌점	5	5	2022년 6월 11일	★ 5 배달 ★ 5
교촌치킨-신촌점	5	5	2022년 6월 7일	★ 5
교촌치킨-신촌점	5	5	2022년 6월 7일	★ 5 배달 ★ 5
교촌치킨-신촌점	5	5	2022년 5월 24일	★ 5 배달 ★ 5
교촌치킨-신촌점	5	5	2022년 5월 20일	★ 5 배달 ★ 5

# 데이터 전처리

---

## 5. 사용자 id

- 사용자 id가 없는 데이터 삭제

## 6. 리뷰

- 뒤에서 자세히 소개

## 7. 총점 추가

- 맛 + 양

# 데이터 전처리

## 8. 크롤링 과정에서 생긴 오류를 이상치 및 결측치 제거를 통해 해결

	가게	맛	양	날짜	주문	사용자id	리뷰	총점
0	후라이드참잘하는집-서강대점	5	5	2023-02-19	핫간장치킨 ( 무 + 소스 + 콜라 )	st	주문해놓고 샤워하고 나왔는데 바로 왔음 배달은 항상 빨라서 좋고 맛은 말할거 없이 ...	10
1	후라이드참잘하는집-서강대점	5	5	2023-02-18	순살 후라이드	yh	맛있어요 너무 맛oㅅ어웁	10
2	후라이드참잘하는집-서강대점	5	5	2023-02-17	반마리 ( 순살 )	id	너무 맛있게 잘 먹었어요. 머스터드 소스 넉넉히 주셔서 모자라지 않게 잘 먹었습니다.	10
3	후라이드참잘하는집-서강대점	5	5	2023-02-16	반마리 ( 순살 )	rl	굳굳굳 잘 먹었습니당	10
4	후라이드참잘하는집-서강대점	5	5	2023-02-16	핫토스치킨 ( 뼈 ) ( 무 + 소스 + 콜라 )	st	시즈닝 팍팍~넘 맛있었요 간만에 치킨 먹은건데 기분 좋게 먹었어요	10

전처리 후의 사용한 데이터 형태



# 통계를 이용한 추천

- 사용자 정보가 없는 경우 기본적으로 추천해주는 시스템
- 맛과 양을 합친 총점 컬럼을 만들고 총점의 평균을 냄
- 총점의 평균으로 k개의 가게를 높은 순 혹은 낮은 순으로 추천

가게	맛	양	총점
컬투치킨-연남사랑점	4.976471	4.980392	9.956863
베이직프라이드치킨-신촌직영점	4.975862	4.979310	9.955172
굽네치킨&피자-남가좌1호점	4.968641	4.979094	9.947735
치킨더홈-서울이대점	4.969466	4.973282	9.942748
페리카나-홍익대점	4.967078	4.975309	9.942387

평점이 높은 순

가게	맛	양	총점
숯미남숯불치킨에미친남자	4.709302	4.697674	9.406977
BBQ-신촌점	4.783133	4.722892	9.506024
호식이두마리치킨-명지대점	4.709898	4.802048	9.511945
BHC-공덕점	4.807560	4.828179	9.635739
전국3대치킨오성통닭-마포신촌점	4.802326	4.856589	9.658915

평점이 낮은 순

# 통계를 이용한 추천

- 가게 별로 '맛' 의 평균

가게

컬투치킨-연남사랑점	4.976471
베이직프라이드치킨-신촌직영점	4.975862
치킨더홈-서울이대점	4.969466
굽네치킨&피자-남가좌1호점	4.968641
페리카나-홍익대점	4.967078

Name: 맛, dtype: float64

평점이 높은 순

가게

전국3대치킨오성통닭-마포신촌점	4.802326
지코바치킨-독립문점	4.795539
BBQ-신촌점	4.783133
호식이두마리치킨-명지대점	4.709898
숯미남숯불치킨에미친남자	4.709302

Name: 맛, dtype: float64

평점이 낮은 순

# 통계를 이용한 추천

- 가게 별로 '양'의 평균

가게	
컬투치킨-연남사랑점	4.980392
베이직프라이드치킨-신촌직영점	4.979310
굽네치킨&피자-남가좌1호점	4.979094
페리카나-홍익대점	4.975309
치킨더홈-서울이대점	4.973282

Name: 양, dtype: float64

평점이 높은 순

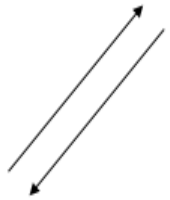
가게	
BHC-공덕점	4.828179
호식이두마리치킨-명지대점	4.802048
마칸마포간풍기-신촌본점	4.796226
BBQ-신촌점	4.722892
숯미남숯불치킨에미친남자	4.697674

Name: 양, dtype: float64

평점이 낮은 순

# 코사인 유사도 기반 추천

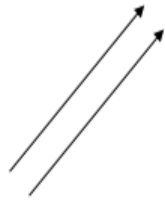
- 두 벡터 간의 코사인 각도를 이용하여 구할 수 있는 두 벡터의 유사도



코사인 유사도 : -1



코사인 유사도 : 0



코사인 유사도 : 1

한 치킨 가게와 다른 치킨 가게들 과의 **유사도**를 구함

사이킷런의 코사인 유사도 라이브러리를 사용

두 벡터 A, B에 대해서 코사인 유사도는 식으로 표현하면 다음과 같습니다.

$$similarity = \cos(\theta) = \frac{A \cdot B}{\|A\| \|B\|} = \frac{\sum_{i=1}^n A_i \times B_i}{\sqrt{\sum_{i=1}^n (A_i)^2} \times \sqrt{\sum_{i=1}^n (B_i)^2}}$$

# 코사인 유사도 기반 추천

가게	BBQ-신 촌점	BHC-공 덕점	BHC-신 촌점	BHC-연 희점	BHC-홍 대초교 점	가마로 강정-광 흥창점	강정이 기가막 하-연희 점	골라먹 는3가지 맛디디 치킨-아 현점	골라먹 는3가지 맛디디 치킨-연 세대점	교촌치 킨-신촌 점	...	티바두 마리치 킨-신수 서강대 점	페리카 나-이대 점	페리카 나-합정 동점	페리카 나-홍익 대점	푸라닭- 신수점	푸라닭- 합정점	피자와 치킨의 러브레 터-서대 문점
가게																		
BBQ- 신촌 점	1.000000	0.307310	0.252003	0.270839	0.228145	0.270469	0.190836	0.241976	0.250211	0.217021	...	0.295799	0.333998	0.175152	0.293273	0.258118	0.209546	0.260812
BHC- 공덕 점	0.307310	1.000000	0.452934	0.401793	0.439990	0.387572	0.286888	0.424000	0.457828	0.375787	...	0.432922	0.458399	0.413469	0.385120	0.427422	0.308774	0.434275
BHC- 신촌 점	0.252003	0.452934	1.000000	0.409942	0.407549	0.408863	0.237211	0.424012	0.402093	0.376753	...	0.387324	0.397057	0.397532	0.428548	0.441348	0.245800	0.397125
BHC- 연희 점	0.270839	0.401793	0.409942	1.000000	0.373160	0.333075	0.290569	0.351458	0.455866	0.329880	...	0.368425	0.349065	0.364345	0.351560	0.451462	0.275057	0.415824
BHC- 홍대 초교 점	0.228145	0.439990	0.407549	0.373160	1.000000	0.324137	0.212351	0.374870	0.418035	0.393758	...	0.433065	0.422654	0.421012	0.414723	0.401410	0.280563	0.442335
가마 로강 정-광 흥창 점	0.270469	0.387572	0.408863	0.333075	0.324137	1.000000	0.270752	0.368104	0.411211	0.322784	...	0.353534	0.411832	0.381549	0.396077	0.382779	0.213072	0.379280

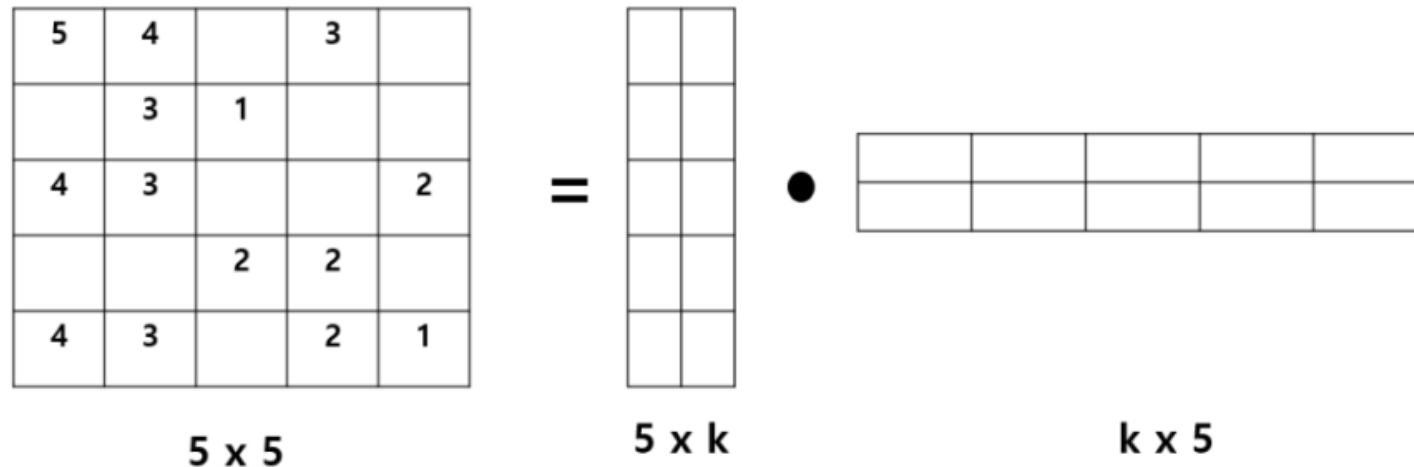
```
# 저번에 시켜먹었던 치킨집과 가장 유사한 치킨집 추천 (양 기준)
def cos_big_recom(title):
    return cosine_matrix[title].sort_values(ascending=False)[:6]

cos_big_recom('BBQ-신촌점') # BBQ 신촌점과 비슷한 치킨집 추천
```

가게  
BBQ-신촌점 1.000000  
굽네치킨&피자-이대역점 0.347956  
자담치킨-신촌점 0.334910  
페리카나-이대점 0.333998  
호식이두마리치킨-명지대점 0.328792  
굽네치킨&피자-남가좌1호점 0.322349  
Name: BBQ-신촌점, dtype: float64

# Matrix Factorization

---



MF 알고리즘은  $R : [ \text{user} \times \text{item} ]$  형태의 full-matrix(평가데이터)를  
 $P : [ \text{user} \times \text{feature} ]$ ,  $Q : [ \text{item} \times \text{feature} ]$  두 개의 행렬로 쪼개서 분석하여  
**비어있는 값(평점)을 예측**하는 방식

# Matrix Factorization

---

1. 잠재요인의 개수  $K$ 를 정한다
2. 임의의 값으로 채워진 두 행렬  $P(m \times k)$ ,  $Q(n \times k)$ 를 생성한다
3. 실제 평점과 예측 평점 오차를 줄여가며  $P$ ,  $Q$ 를 수정한다 (SGD)
4. 기준에 도달 할 때 까지 3과정을 반복한다

# Matrix Factorization

```
#R은 기존 평점 행렬
#R은 P, Q를 초기화하기 위한 상수

def matrix_factorization(R, K, steps=200, learning_rate=0.01, r_lambda = 0.01):
    num_users, num_items = R.shape
    # P와 Q 매트릭스의 크기를 지정하고 정규분포를 가진 랜덤한 값으로 입력합니다.
    np.random.seed(1)
    P = np.random.normal(scale=1./K, size=(num_users, K))
    Q = np.random.normal(scale=1./K, size=(num_items, K))

    break_count = 0

    # R > 0 인 행 위치, 열 위치, 값을 non_zeros 리스트 객체에 저장.
    # NON-ZERO: 사용자가 먹은 처리들만 따로 행렬을 만들음
    non_zeros = [ (i, j, R[i,j]) for i in range(num_users) for j in range(num_items) if R[i,j] > 0 ]

    # SGD기법으로 P와 Q 매트릭스를 계속 업데이트.
    for step in range(steps):
        for i, j, r in non_zeros:
            # 실제 값과 예측 값의 차이인 오류 값 구함
            eij = r - np.dot(P[i, :], Q[j, :].T)
            # Regularization을 반영한 SGD 업데이트 공식 적용
            P[i,:] = P[i,:] + learning_rate*(eij * Q[j, :] - r_lambda*P[i,:])
            Q[j,:] = Q[j,:] + learning_rate*(eij * P[i, :] - r_lambda*Q[j,:])

        rmse = get_rmse(R, P, Q, non_zeros)
        if (step % 10) == 0 :
            print("### iteration step : ", step, " rmse : ", rmse)

    return P, Q, rmse
```

```
# 더 좋은 결과값을 얻기 위한 하이퍼파라미터 튜닝하기

from sklearn.model_selection import ParameterGrid

min = 50000

grid = ParameterGrid({
    'K': [30, 40, 50, 60],
    "learning_rate": [0.01, 0.001, 0.1, 0.02, 0.025],
    'r_lambda': [0.01, 0.02, 0.03, 0.001]
})

err = []

for params in grid:
    try:
        P, Q, rmse = matrix_factorization(ratings_matrix.values, K=params['K'], steps=200, learning_rate=params['learning_rate'], r_lambda=params['r_lambda'])

        if rmse < min:
            min = rmse
            best_parm = params

    except:
        err.append(params)
print(min)
print(best_parm)
```



# Matrix Factorization

```
### iteration step : 0  rmse : 4.8951221380869105
### iteration step : 10  rmse : 1.2430796012716565
### iteration step : 20  rmse : 0.690406339269562
### iteration step : 30  rmse : 0.4650977916214984
### iteration step : 40  rmse : 0.3418230823095762
### iteration step : 50  rmse : 0.2620866359238788
### iteration step : 60  rmse : 0.19978957187206126
### iteration step : 70  rmse : 0.1525514729247504
### iteration step : 80  rmse : 0.12266125979746145
### iteration step : 90  rmse : 0.10412970851938211
### iteration step : 100  rmse : 0.09133001269191322
### iteration step : 110  rmse : 0.08173631565860187
### iteration step : 120  rmse : 0.07423183450591636
### iteration step : 130  rmse : 0.06823198790212999
### iteration step : 140  rmse : 0.063386634942216
### iteration step : 150  rmse : 0.05945658868810413
### iteration step : 160  rmse : 0.056257874551176285
### iteration step : 170  rmse : 0.05364106082678603
### iteration step : 180  rmse : 0.051485376146271945
### iteration step : 190  rmse : 0.04969558738935449
```

하이퍼 파라미터 튜닝 결과

- 0.007 정도로 rmse가 낮게 나옴
- {'K': 60, 'learning\_rate': 0.02, 'r\_lambda': 0.001}

주문

순살치킨 + 순살치킨	5.736646
크리스피치킨	5.272436
치파포 세트 V5	5.240229
후라이드 콤보	5.176459
순한맛1박스	5.170542
닭갈비볶음밥 + 우동세트	5.106739
( KING푸짐 ) 로제닭볶이 2~4인용	5.042074
콘소메이징	5.039892
슈프림	5.037097
크러스터치킨 반 + 달콤 핫	5.034397

## - 라이브러리 비교 (krwordrank vs konlpy)

(‘맛있다’, 6710), (‘먹다’, 5470), (‘맛’, 2710), (‘하다’, 2490), (‘자다’, 2340),  
 1210), (‘시키다’, 1210), (‘같다’, 740), (‘많다’, 710), (‘후라이드’, 700), (‘있  
 ‘않다’, 520), (‘빠르다’, 510), (‘양도’, 510), (‘오다’, 510), (‘진짜’, 490), (‘  
 ‘최고’, 410), (‘바삭’, 400), (‘리뷰’, 390), (‘엄청’, 370), (‘처음’, 360), (‘하  
 말’, 300), (‘튀김’, 290), (‘여기’, 290), (‘맵다’, 270), (‘시간’, 270), (‘역시’,  
 30), (‘되다’, 230), (‘친절하다’, 220), (‘찍다’, 220), (‘코바’, 220), (‘다른’, 2  
 200), (‘주다’, 200), (‘케밥’, 200), (‘자주’, 190), (‘마뽀다’, 190), (‘이벤트’,

konlpy

# 리뷰 기반 추천 시스템

- 전처리 결과 추천에 도움이 될만한 키워드를 뽑는게 어려울 것 같아, 긍/부정 별로 몇 개의 리뷰를 뽑아서 추천에 도움을 주는 방향으로 구현

```
# 긍/부정 비율 확인하기
sentiment = [0 if rank in range(1,4) else 1 for rank in df['맛']]
# 1~3 사이의 값은 0, 4~5는 1로 labeling

pos = len([x for x in sentiment if x==1]) # 긍정 = ?
neg = len([x for x in sentiment if x==0]) # 부정 = ?
print("긍정 비율 : {:.2f}".format(pos/len(sentiment))+", 긍정 개수 : {}개".format(pos))
print("부정 비율 : {:.2f}".format(neg/len(sentiment))+", 부정 개수 : {}개".format(neg))
```

0~3사이의 값은 부정으로,  
4~5 사이의 값은 긍정으로 labeling

	가게	맛	양	날짜	주문	사용자 id	리뷰	label
0	후라이드참잘하는집-서강대점	5	5	2023-02-19	핫간장치킨 ( 무 + 소스 + 콜라 )	st	주문해놓고 사워하고 나왔는데 바로 왔음 배달은 항상 빨라서 좋고 맛은 말할거 없이 ...	1
1	후라이드참잘하는집-서강대점	5	5	2023-02-18	순살 후라이드	yh	맛있어요 너무 맛ㅇ스어울	1
2	후라이드참잘하는집-서강대점	5	5	2023-02-17	반마리 ( 순살 )	id	너무 맛있게 잘 먹었어요. 머스터드 소스 넉넉히 주셔서 모자라지 않게 잘 먹었습니다.	1
3	후라이드참잘하는집-서강대점	5	5	2023-02-16	반마리 ( 순살 )	rl	궁궁궁 잘 먹었습니다	1
4	후라이드참잘하는집-서강대점	5	5	2023-02-16	핫토스치킨 ( 뼈 ) ( 무 + 소스 + 콜라 )	st	시즈닝 짭쪼뽕 맛있었요 간만에 치킨 먹은건데 기분 좋게 먹었어 요	1
...	...	...	...	...	...	...	...	...
13377	치킨시대-명지대점	5	5	2021-12-22	크리스피치킨	sn	맛있어요!튀김도맛있고	1

Label 확인

# 리뷰 기반 추천 시스템

- 긍/부정 비율을 확인해본 결과, 긍정의 비율과 부정의 비율이 큰 차이가 남
- 따라서 **부정적인 의견을 확인하는 것이 중요함**

```
# 긍/부정 비율 확인하기
sentiment = [0 if rank in range(1,4) else 1 for rank in df['맛']]
# 1~3 사이의 값은 0, 4~5는 1로 labeling

pos = len([x for x in sentiment if x==1]) # 긍정 = 1
neg = len([x for x in sentiment if x==0]) # 부정 = 1
print("긍정 비율 : {:.2f}".format(pos/len(sentiment))+", 긍정 개수 : {}개".format(pos))
print("부정 비율 : {:.2f}".format(neg/len(sentiment))+", 부정 개수 : {}개".format(neg))
```

긍정 비율 : 0.98, 긍정 개수 : 13062개  
부정 비율 : 0.02, 부정 개수 : 320개

100개 중 2개가 부정

# 리뷰 기반 추천 시스템

▶ bhc[:5] # 특정 가게의 부정리뷰 상위 5개 출력

]:

	가게	맛	양	날짜	주문	사용자 id	리뷰	label
12936	BHC-공덕점	1	1	2023-01-14	골드킹 콤보	do	다른지점이랑 닭이 다른듯 닭은 너무 말랐고 질기고 짜고 심지어 식어서음	0
12941	BHC-공덕점	1	1	2023-01-03	치킨스넥 ( Chicken's Neck )	yh	치킨스넥 10,000원 주고 안사먹을듯합니다. 맛도 맛이지만(냉동 텐더보다 못함),...	0
12942	BHC-공덕점	2	5	2022-12-30	뿌링클HOT	bc	이번에 너무 퍽퍽하고 시즈닝도 별로 없어서 많이 아쉬웠어요	0
12965	BHC-공덕점	3	3	2022-11-29	골드킹	99	골드킹은 다른치킨보다 더 퍽퍽하군요	0
13076	BHC-공덕점	3	5	2023-01-27	고추장 직화구이	uy	미지근한 정도가 아니라 차갑게 식어서 왔어요. 배달기사님 배정되는거 계산해서 음식 ...	0

label 0 : 부정

주문하려는 가게의 부정/또는 긍정 리뷰를 뽑아서 주문에 도움을 준다  
부정적인 의견을 확인하는 것은 주문을 하는데 도움을 줄 수 있을 것이다

# 인사이트

---

Q1. 그냥 고민하지 말고 프랜차이즈 치킨을 시키는 것이 좋을까?

- 전체 가게의 맛의 평균 : 약 4.88
- 대표적인 프랜차이즈 : bbq, 맘스터치, 굽네치킨, 교촌치킨, bhc
- 해당 프랜차이즈 중 맘스터치와 굽네를 제외하면 **전체 가게의 맛 평균 평점보다 맛 평균 평점이 낮음**

 제일 유명한 프랜차이즈 꼭 시킬 필요가 없고, 소규모의 프랜차이즈나 동네의 치킨집을 시켜도 됨

# 인사이트

---

Q2. 평점이 좋으면 리뷰도 좋은 리뷰만 있을까?

- 그렇지 않음

subQ. 추가로 리뷰의 긍정, 부정 비율을 확인한 결과 **긍정, 부정 비율이 차이가 많이 남**

➡ 웬만하면 긍정의 평가를 하므로 평점 자체만으로는 추천의 정확도가 떨어질 수 있음

# 인사이트

---

Q3. 양과 맛의 평점의 경우 동일할까? 아니면 양은 많은데 맛은 안 좋은 집은 있을까?

- 맛은 5점이지만 양은 4점 이하인 경우, 양은 5점이지만 맛은 4점 이하인 경우 모두 있었음

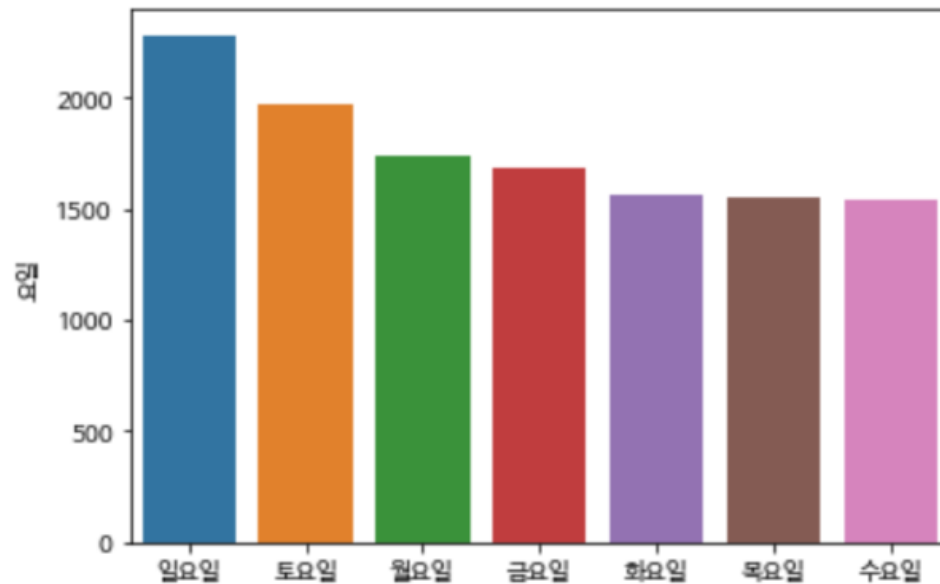
➡ 맛의 평점과 양의 평점이 동일하지 않음을 확인할 수 있음



# 인사이트

Q4. 치킨을 언제 많이 시켜 먹을까? 불금에 치킨일까?

➡ 시각화 결과 : **일요일**>토요일>월요일>금요일>화요일>목요일>수요일



- 1) 불금 < 주말
- 2) 토요일 < 일요일
- 3) 금요일 < 월요일

# 인사이트

---

## Q5. 치킨집은 치즈님이니 보정이 있을까?

- 총점이 10점인 개수, 즉 맛과 양의 평점이 모두 5점인 경우가 약 90%
- 전체 가게의 '맛'의 평균 평점: 약 4.88 / 전체 가게의 '양'의 평균 평점: 약 4.89

 평점이 전체적으로 높다는 것을 알 수 있음

# 의의 및 한계

---

## 의의

- 데이터 크롤링을 이용한 리뷰 데이터 수집
- 통계, matrix factorization, 자연어 처리 등 여러 가지 방식을 이용해 데이터 분석
- 다양한 방법으로 추천에 도움을 줄 수 있는 기능 구현
- 전처리부터 구현까지 전부 구현해볼 수 있었음
- 가설 확인

# 의의 및 한계

---

## 한계

- id가 왼쪽 두 글자만 나와 다른 사람이지만 같은 id로 보임
  - 단순 코사인 유사도 기반으로는 정확한 추천에 어려움이 있음
- 치킨집 외에 다른 가게에 적용을 시켜보지 않아서 치킨 도메인 특성의 영향이 있는지 정확한 확인이 어려움
- 과적합의 위험

# 의의 및 한계

---

## 보완할 점

- id가 왼쪽 두 글자만 나와 더 정확한 추천을 위해 보완
- 특정 가게의 부정 리뷰를 평점 낮은 순으로 정렬해서 확인할 수 있는 기능 추가 가능
- 하이퍼 파라미터 튜닝을 통해서 성능을 올렸지만 과적합의 위험이 있기 때문에 앙상블이나 여러 기법 등 사용
- 데이터의 경우 정보가 적어서 다음에는 좀 더 구체적인 정보의 크롤링의 필요성을 느낌
  - 추가적인 정보가 없어서 추천의 한계가 있음

# 느낀 점

---

- 프로젝트 회의를 여러 번 하면서 엄청난 시간과 소통(...)이 필요하다는 것을 깨달음
- 데이터를 직접 수집하는 것은 시간이 매우 많이 필요
- 정확한 데이터 분석을 위해 자세한 전처리 과정이 필요
  - 전처리 방법에 따라 결과가 크게 달라질 수 있음

# 결론

---

---

**감사합니다**