In [1]:

```
import pandas as pd
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.metrics.pairwise import cosine_similarity
import pymysql
from itertools import chain
from collections import defaultdict
import numpy as np
```

In [2]:

```
conn = pymysql.connect(host = 'localhost',
                       port=3306,
                       user = 'root',
                       password = 'rootpass',
                       db = 'jeju')
```

In [3]:

```
sql_input = "SELECT * FROM jeju_data_web where contentscd = '음식점'"
sql_df = pd.read_sql_query(sql_input, conn)
```

```
C:₩Users₩TJW.conda₩envs₩py38₩lib₩site-packages₩pandas₩io₩sql.py:761: UserWarning: pa
ndas only support SQLAlchemy connectable(engine/connection) ordatabase string URI or
sqlite3 DBAPI2 connectionother DBAPI2 objects are not tested, please consider using
SQLAlchemy
  warnings.warn(
```

In [4]:

```
sql_input = "SELECT * FROM jeju_data_web where contentscd = '음식점'"
sql_df = pd.read_sql_query(sql_input, conn)
```

```
C:₩Users₩TJW.conda₩envs₩py38₩lib₩site-packages₩pandas₩io₩sql.py:761: UserWarning: pa
ndas only support SQLAlchemy connectable(engine/connection) ordatabase string URI or
sqlite3 DBAPI2 connectionother DBAPI2 objects are not tested, please consider using
SQLAlchemy
  warnings.warn(
```

In [5]:

```
sql_df
```

5	13418405	우진해장국	126.519969	33.511539	제주특별자치도 제주시 서사로 11	064-757-3393	리를 갈아 넣고 푹 끓여 갈색 빛깔이 나는 제주식 해장국을 만날 수 있는...	고사리육개장, 사골해장국, 녹두빈대떡, 육개장	phinf.pstatic.ne
6	1011125170	오는정김밥	126.567598	33.249664	제주특별자치도 서귀포시 동문동로 2	064-762-8927	안녕하세요 제주도 오는 정김밥입니다. 홀에서는 드실수 없고 포장판매만 가능한	김밥	phinf.pstatic.ne

In [9]:

```
review_body = pd.read_csv('./review_body_preprocessed.csv')
review_stats = pd.read_csv('./review_stats_preprocessed.csv')
```

In [10]:

```
review_voted_keywords = pd.read_csv('./review_voted_keywords_preprocessed.csv')
```

In [11]:

```
review_voted_keywords
```

Out[11]:

	id	votedKeywords_음식이 맛있어요	votedKeywords_재료가 신선해요	votedKeywords_친절해요	votedKeywords_뷰가 좋아요	voted가성
0	12883219	114.0	56.0	49.0	49.0	
1	1207652081	17.0	13.0	7.0	NaN	
2	35269176	41.0	23.0	18.0	5.0	
3	11710933	23.0	12.0	17.0	15.0	
4	1480037450	30.0	9.0	16.0	13.0	
...	...	...	...	...	...	
12016	31507955	15.0	3.0	6.0	1.0	
12017	1231669980	43.0	19.0	18.0	15.0	
12018	1910356482	NaN	NaN	3.0	2.0	
12019	1277648050	NaN	NaN	124.0	720.0	
12020	16964053	11.0	2.0	8.0	NaN	

12021 rows × 83 columns

In [12]:

```
review_voted_keywords.fillna(0,inplace=True)
```

In [13]:

```
top3_review_stats=pd.read_csv('./top3_food_review_voted_keywords_preprocessed.csv')
```

In [14]:

```
top3_review_stats.fillna(0,inplace=True)
```

In [15]:

```
total_review_stats = pd.concat([review_voted_keywords,top3_review_stats],axis=0,ignore_index=True)
```

In [16]:

```
total_review_stats.drop_duplicates(subset='id',ignore_index=True,inplace=True)
```

In [17]:

```
total_review_stats[total_review_stats['id']==1000671392]
```

Out[17]:

	id	votedKeywords_음식이 맛있어요	votedKeywords_재료가 신선해요	votedKeywords_친절해요	votedKeywords_뷰가 좋아요	voted가성
12021	1000671392	0.0	0.0	225.0	417.0	

1 rows × 83 columns

In [18]:

```
food_db=pd.read_csv('../../data/naver_crawling/음식점db_final_concat_Cafe.csv')
```

```
C:₩Users₩TJW₩AppData₩Local₩Temp₩ipykernel_13836₩3588702470.py:1: DtypeWarning: Column
s (3,10,21,25,27,34,38,39,43) have mixed types. Specify dtype option on import or se
t low_memory=False.
  food_db=pd.read_csv('../../data/naver_crawling/음식점db_final_concat_Cafe.csv')
```

In [19]:

```
food_db['categories']=food_db.loc[:,'categories'].apply(lambda x: eval(x))
```

In [20]:

```
food_db['category1'] = food_db.loc[:,'categories'].apply(lambda x: x[0])
food_db['category2'] = food_db.loc[:,'categories'].apply(lambda x: x[-1])
```

In [21]:

```
food_db.shape
```

Out[21]:

```
(16477, 46)
```

In [22]:

```
right_join_df=food_db[['id','category1','category2','categories']]
```

In [23]:

```
search_db=pd.merge(left=total_review_stats,right=right_join_df,how='left',right_on='id',left_on='id
```

In [24]:

```
search_db
```

Out[24]:

	id	votedKeywords_음식이 맛있어요	votedKeywords_재료가 신선해요	votedKeywords_친절해요	votedKeywords_뷰가 좋아요	voted가성
0	12883219	114.0	56.0	49.0	49.0	
1	1207652081	17.0	13.0	7.0	0.0	
2	35269176	41.0	23.0	18.0	5.0	
3	11710933	23.0	12.0	17.0	15.0	
4	1480037450	30.0	9.0	16.0	13.0	
...	...	...	...	...	...	
12028	1828477580	0.0	0.0	238.0	892.0	
12029	1273416923	0.0	0.0	262.0	733.0	
12030	1516216333	0.0	0.0	257.0	699.0	
12031	37191637	0.0	0.0	89.0	392.0	
12032	1431450188	0.0	0.0	902.0	1435.0	

12033 rows × 86 columns

In [25]:

```
id_index_df=search_db.set_index('id')
```

In [26]:

```
def get_same_category(restauarant_id):
    category_nm=id_index_df.loc[restauarant_id,'categories']
    conditions = (search_db['category1'].isin(category_nm)) |(search_db['category2'].isin(category_r
    result_df=search_db[conditions]
    result_df.reset_index(drop=True,inplace=True)
    return result_df
```

In [27]:

```
def get_sim_id_by_cossim(restauarant_id):
    df1=get_same_category(restauarant_id)
    print(f'{restauarant_id}같은 카테고리 식당들 조회 완료')
    df1.fillna(0,inplace=True)
    df2= df1.drop(['id','category1','category2','categories'],axis=1)
    arr=df2.to_numpy()
    #id를 통해 몇번째 index에 존재하는지 확인할 수 있게 dictionary 생성
    i2d = dict(zip(df1['id'],df1.index))
    idx = i2d.get(restauarant_id)
    #코사인유사도 행렬 생성
    cosine_sim = cosine_similarity(arr, arr)
    print('코사인유사도 행렬 생성완료')
    #idx를 통해 유사도 값 조회
    sim_scores = list(enumerate(cosine_sim[idx]))
    #코사인유사도를 기준으로 내림차순 정렬 0 번째는 본인
    sorted_scores=sorted(sim_scores,key=lambda x: x[1],reverse=True)
    #유사도 상위 3개 그리고 0.85이상인 값들의 index값만 담는다.
    index_lsts = [idx[0] for idx in sorted_scores[1:4] if idx[1] > 0.85]
    #인덱스를 통해 id 값 조회
    if index_lsts is None:
        sim_idlsts = None
    else:
        sim_idlsts=df1.iloc[index_lsts,0].values

    return (restauarant_id,sim_idlsts)
```

In [28]:

```
food_db[food_db['id'] == 19873758]['categories']
```

Out[28]:

```
11564    [한식, 해물,생선요리]
Name: categories, dtype: object
```

In [29]:

```
td = get_same_category(16886040)
```

In [30]:

```
td.fillna(0,inplace=True)
```

```
C:₩Users₩TJ₩AppData₩Local₩Temp₩ipykernel_13836₩1811282120.py:1: SettingWithCopyWarni
ng:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/u
ser_guide/indexing.html#returning-a-view-versus-a-copy (https://pandas.pydata.org/pa
ndas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy)
  td.fillna(0,inplace=True)
```

In [31]:

```
td2 = td.drop(['id','category1','category2','categories'],axis=1)
arr=td2.to_numpy()
```

In [32]:

```
i2d = dict(zip(td['id'],td.index))
idx = i2d.get(16886040)
#코사인유사도 행렬 생성
cosine_sim = cosine_similarity(arr, arr)
sim_scores = list(enumerate(cosine_sim[idx]))
```

In [33]:

```
td.iloc[980,0]
```

Out[33]:

```
1516216333
```

In [34]:

```
restaurants_lsts = []
sims_id = []
for i,v in enumerate(sql_df['id']):
    original_id,sim_id_results = get_sim_id_by_cossim(v)
    restaurants_lsts.append(original_id)
    sims_id.append(sim_id_results)
    print(f'{i}번째 종료')
```

```
코사인유사도 행렬 생성완료
1번째 종료
1927504039같은 카테고리 식당들 조회 완료

C:WUsersWTJWAppDataWLocalWTempWipykernel_13836W2893858214.py:4: SettingWithCopyWar
ning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stabl
e/user_guide/indexing.html#returning-a-view-versus-a-copy (https://pandas.pydata.o
rg/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy)
  df1.fillna(0,inplace=True)

코사인유사도 행렬 생성완료
2번째 종료
37060300같은 카테고리 식당들 조회 완료

C:WUsersWTJWAppDataWLocalWTempWipykernel_13836W2893858214.py:4: SettingWithCopyWar
ning:
A value is trying to be set on a copy of a slice from a DataFrame
```

In [35]:

```
result_df=pd.DataFrame({'id':restaurants_lsts,
             'sims_id':sims_id})
```

In [36]:

```
sim_rank = []


for index,id_lsts in enumerate(result_df['sims_id']):
    tmp_dict = {'id':result_df['id'][index]}
    for num,id_v in enumerate(id_lsts):
        condition = (food_db['id'] == id_v)
        tmp_df=food_db[condition]
        name = tmp_df['name'].values[0]
        imageURL = tmp_df['imageURL'].values[0]
        id_values = tmp_df['id'].values[0]
        if imageURL is np.NaN:
            result = name+',,'+str(id_values)
        else:
            result = name + ',' + imageURL + ','+str(id_values)

        if num == 0:
            tmp_dict['sim_rank1']= result
        elif num == 1:
            tmp_dict['sim_rank2']= result
        elif num == 2:
            tmp_dict['sim_rank3'] = result

    sim_rank.append(tmp_dict)
```

In [37]:

```
final_df =pd.DataFrame(sim_rank)
```

In [38]:

```
final_df.shape
```

Out[38]:

(56, 4)

In [39]:

```
final_df['sim_rank1'][0]
```

Out[39]:

'표선어촌식당,https://ldb-phinf.pstatic.net/20191208_274/15757683091844Rxra_JPEG/aAkTWbxQ6Avlsq6viCTeedsh.jpg,32166291'

In [40]:

```
sql_df.shape
```

Out[40]:

(56, 15)

In [59]:

```
f_df=pd.merge(left=sql_df,right=final_df,how='inner',left_on='id',right_on='id')
```

In [61]:

```
f_df.to_csv('./리뷰키워드기반음식점추천추가.csv',index=False)
```

In [60]:

```
f_df['sim_rank1'][0]
```

Out[60]:

```
'표선어촌식당,https://ldb-phinf.pstatic.net/20191208_274/15757683091844Rxra_JPEG/aAkTWbxQ6AvIsq6viCTeedsh.jpg,32166291'
```