



냉장고 속 남은 재료로 만드는 만개의 레시피

재료기반 레시피 추천

목차

a table of contents

01 분석 주제

02 데이터 소개

03 데이터 수집 방법

04 전처리 및 유사도 분석

05 결과 해석





1

”

분석 주제



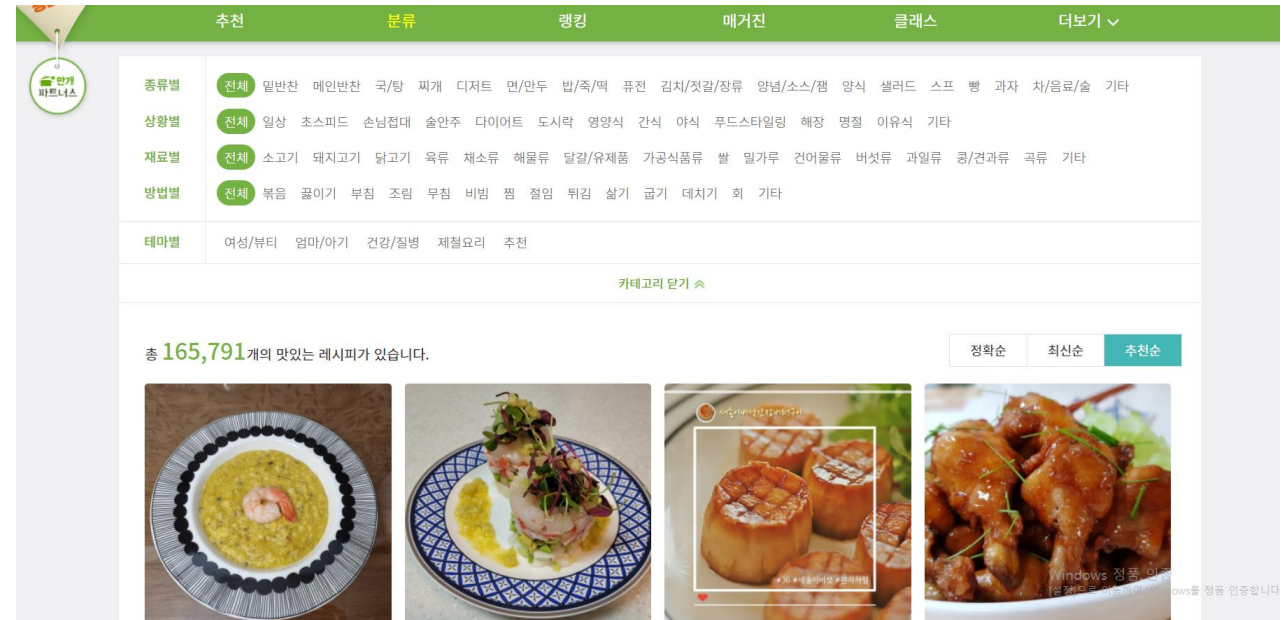
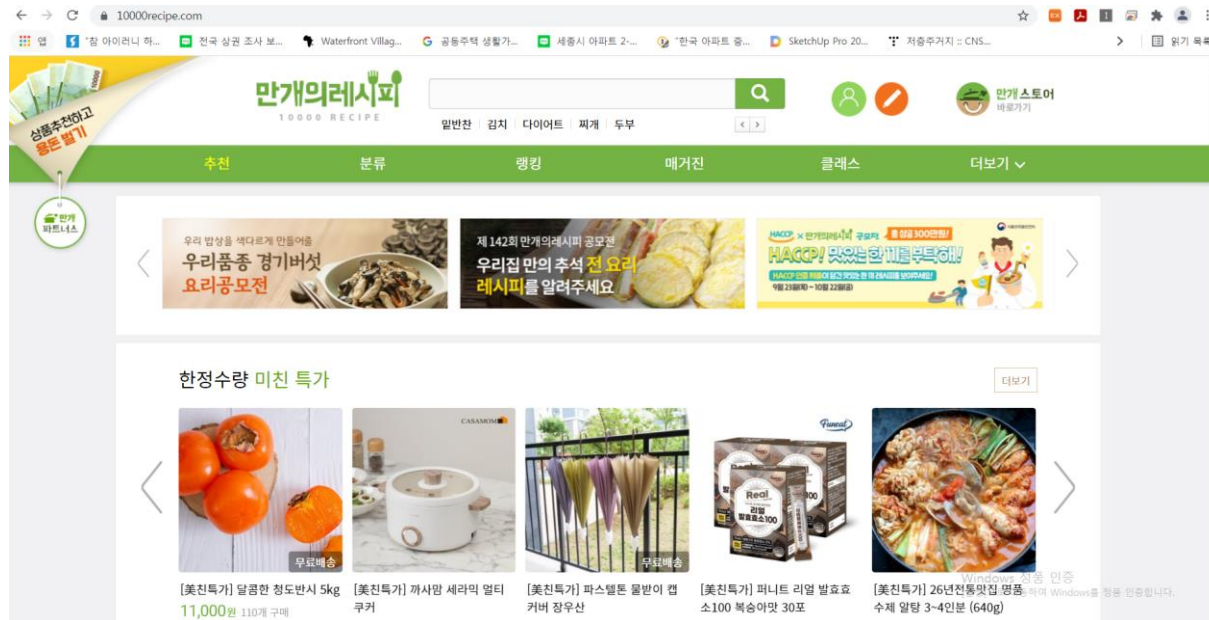
<재료기반레시피추천>

냉장고에 남은 재료로 오늘 무슨 요리를 할 수 있을까?





재료를 입력하고 만개의 레시피 사이트 웹스크래핑으로 레시피를 추천받자!



2

”

데이터 소개





만개의레시피 사이트 웹스크래핑(제목, 재료, URL, PAGE 번호 추출)

총 165,791개의 맛있는 레시피가 있습니다.

정확순

최신순

추천순



HACCP 새우치즈단호박리조또

ekdud1129

★★★★★ (2) 조회수 436



광어삼합타워

슈퍼게미즈

★★★★★ (1) 조회수 95



버섯으로 관자 느낌 내는 방법! 새송이버섯간장버터구이 만들기

루던



닭봉간장조림~ 십년째 만들어 먹는 양념 공유해요!

요리하는러라

재료 Ingredients

[재료]

새송이버섯 ⓘ	2개
버터 ⓘ	30g
어린잎채소 ⓘ	

[소스재료]

간장 ⓘ	1T
맛술 ⓘ	1T
올리고당 ⓘ	1/2T

계량법 안내



3

”

데이터 수집 방법



첫째

DF 혹은 SQL DB에 하나씩 저장

둘째

페이지 별 레시피 URL 추출

셋째

만개의 레시피 사이트내 '분류' 페이지에서 1개의 레시피에 대한 제목, 재료 추출



```
▶ 1 ## https://www.10000recipe.com/recipe/list.html
  2 ## 만개의 레시피
```

```
▶ 1 ## Anaconda - pip install PyMySQL
  2 import pymysql
  3
  4 ## 연속적 print, output clear
  5 ## source : display(target), clear_output(wait=True)
  6 from IPython.display import clear_output
  7
  8 #Console Progress Bar
  9 from tqdm import tqdm, trange, tqdm_notebook
 10
 11 from time import sleep
 12
 13 ## DataFrame O
 14 import numpy as np
 15 # Array
 16 import pandas as pd
 17
 18 ## http Requests
 19 import requests
 20 from bs4 import BeautifulSoup
```




```
1 ## 스크래핑 url
2 target_url = {
3     'domain'      : 'https://www.10000recipe.com', # Target Domain
4     'recipe_list' : '/recipe/list.html',          # Target page url
5     'recipe'      : ''                            # Tager url
6 }
7
8 ## 스크래핑 url 에 보내질 params
9 page_params = {
10     'order'       : 'reco',      # 정렬
11     'cat4'        : '66',        # 분류
12     'cat4_name'   : '빵',        # 분류 이름
13     'page'        : 1,          # Page Number
14     'sequence'    : 0           # Recipe sequence
15 }
```

```
1 ## Csv add Save..
2 # param_dict ex : {'key': '레시피', 'kind' : '일반찬'}
3 def fn_save_csv(param_dict):
4
5     #dict를 리스트로 변환 후 DataFrame 으로 변환.
6     df_save_data = pd.DataFrame([param_dict])
7
8     if(param_dict.get('page_num') == 1 and param_dict.get('sequence') == 1):
9         #to_csv(mode a : 기존 내역에서 추가로 입력한다. , index=False : 인덱스 입력을 x, header=True컬럼명을 입력한다)
10         df_save_data.to_csv('recipe.csv', mode='a', index=False, header=True)
11     else:
12         #to_csv(mode a : 기존 내역에서 추가로 입력한다. , index=False : 인덱스 입력을 x, header=True컬럼명을 입력하지 않는다.)
13         df_save_data.to_csv('recipe.csv', mode='a', index=False, header=False)
```



```

1  ## Function MySql Area.....
2  #-----
3  #      DDL db tool 에서 처리
4  #      Table Schema
5  #-----
6
7  ##### recipe : 요리 정보
8  #drop table if exists recipe;
9  #CREATE TABLE IF NOT EXISTS recipe(
10 #   idx          INT(11)      NOT NULL AUTO_INCREMENT PRIMARY KEY,      # INDEX NO [PK]
11 #   page_num     INT(11)      NOT NULL,                                  # page 번호
12 #   cooking_type VARCHAR(100) NOT NULL,                                  # 요리 종류
13 #   title        VARCHAR(100) NOT NULL,                                  # 제목
14 #   ingredient   VARCHAR(1000),                                           # 재료
15 #   sorce        VARCHAR(1000),                                           # 재료 외 양념장 등
16 #   uri          VARCHAR(500) NOT null,                                   # URL
17 #   insert_dt    datetime      default NOW()                             # 저장일자 시분초
18 #) ENGINE=InnoDB DEFAULT CHARSET=utf8
19 #;
20 #
21 #drop table if exists scraping_error;
22 #create table if not exists scraping_error(
23 #   idx          INT(11)      NOT NULL AUTO_INCREMENT PRIMARY KEY,      # INDEX NO [PK]
24 #   page_num     INT(11),                                               # page 번호
25 #   cooking_type VARCHAR(100),                                           # 요리 종류
26 #   title        VARCHAR(100),                                           # 제목
27 #   error_code   VARCHAR(100),                                           # response error code
28 #   error_str    VARCHAR(2000),                                           # response error str
29 #   uri          VARCHAR(500),                                           # URL
30 #   insert_dt    datetime      default NOW()                             # 저장일자 시분초
31 #) ENGINE=InnoDB DEFAULT CHARSET=utf8
32 #;
33

```




```
37 def fn_insert_sql(params):
38
39     #-----
40     # Database(Mysql) Connect Info
41     #-----
42
43     # mysql 연결을 위한 정보를 저장하고 있는 mysql Connect 객체 생성한다.
44     # host : ip위치(localhost : 현재 pc)
45     # port : mysql port 번호
46     # db   : 데이터베이스 이름
47     # user : 데이터베이스 접근 login id
48     # pw   : 데이터베이스 접근 login id의 password
49     # charset : character Set
50     # autocommit = True : Insert, Updaete, Delete 등이 자동 Commit 된다.
51     # pymysql.cursors.DictCursor : 결과값을 Dict(Key : Value)로 Return한다.
52
53     mysql_conn = pymysql.connect(
54         host='localhost',
55         port=21500,
56         db='mydb', user='root', password='',
57         charset='utf8', autocommit=True,
58         cursorclass=pymysql.cursors.DictCursor
59     )
```



```
61 #return DataFrame
62 rtn = pd.DataFrame:
63 try:
64     #DataBase Sql (Structured Query Language)
65     cursor = mysql_conn.cursor()
66
67     ins_sql = '''
68         INSERT INTO recipe (
69             page_num,
70             seq,
71             cooking_type,
72             title,
73             ingredient,
74             source,
75             uri
76         )
77         VALUES (
78             %,
79             %,
80             %,
81             %,
82             %,
83             %,
84             %
85         )
86     ...
87
88     # 이 때에 cursor에서 database와 연결 및 Query를 보내고, return 값을 받는다.
89     # 실제 이때 연결된다 볼 수 있다.
90     cursor.execute(ins_sql, (
91         params.get('page_num'),
92         params.get('sequence'),
93         params.get('cooking_type'),
94         params.get('title'),
95         params.get('ingredient'),
96         params.get('source'),
97         params.get('uri')
98     ))
99     rtn = cursor.fetchall()
100 # except:
101
102 finally:
103     #Database 연결 해제
104     mysql_conn.close()
105
106 return rtn;
```




```
108 def fn_select_sql():
109
110     #-----
111     # Database(Mysql) Connect Info
112     #-----
113
114     # mysql 연결을 위한 정보를 저장하고 있는 mysql Connect 객체 생성한다.
115     # host : ip위치(localhost : 현재 pc)
116     # port : mysql port 번호
117     # db : 데이터베이스 이름
118     # user : 데이터베이스 접근 login id
119     # pw : 데이터베이스 접근 login id의 password
120     # charset : character Set
121     # autocommit = True : Insert, Update, Delete 등이 자동 Commit 된다.
122     # pymysql.cursors.DictCursor : 결과값을 Dict(Key : Value)로 Return한다.
123     mysql_conn = pymysql.connect(
124         host='localhost',
125         port=21500,
126         db='mydb', user='root', password='',
127         charset='utf8', autocommit=True,
128         cursorclass=pymysql.cursors.DictCursor
129     )
130
131     rtn = pd.DataFrame:
132     try:
133
134         #DataBase Sql (Structured Query Language)
135         cursor = mysql_conn.cursor()
136
137         ins_sql = '''
138             SELECT * FROM recipe
139         '''
140
141         cursor.execute(ins_sql)
142         rtn = cursor.fetchall()
143     # except:
144
145     finally:
146         #Database 연결 해제
147         mysql_conn.close()
148
149     return rtn;
```



```
1 df = pd.DataFrame(fn_select_sql())
2 df.head(5)
```

3]:

	cooking_type	idx	ingredient	insert_dt	page_num	seq	source	title	uri
0	밀반찬	1	새송이버섯,버터, 어린잎채소	2021-10-06 18:23:56	1	1	간장,맛술,올리고당	버섯으로 관자 느낌 내는 방법! 새송이버 섯간장버터구이 만들 기	https://www.10000recipe.com/recipe/6912734
1	밀반찬	2	팽이버섯,청양고 추,식용유,다진마 늘	2021-10-06 18:23:57	1	2	고추가루,설탕,미림, 진간장,굴소스,고추 장,된장,물	500원 이면 밥한끼~ 뚝딱하는 팽이버섯조 림~ 너무 맛있어용 ^^~	https://www.10000recipe.com/recipe/6956266
2	밀반찬	3	건새우,물,마늘종, 소금	2021-10-06 18:23:57	1	3	설탕,간장,식용유,참 기름,물엿,통깨	[마늘종볶음]단짠단짠 자꾸만 손이가요~	https://www.10000recipe.com/recipe/6957535
3	밀반찬	4	베이컨,대파,떡	2021-10-06 18:23:58	1	4	간장,참기름,물엿,기 름	달콤 짭조름♡떡 베 이컨 간장조림♡	https://www.10000recipe.com/recipe/6843136
4	밀반찬	5	풋고추	2021-10-06 18:23:58	1	5	된장,다진 마늘,송송 썰은 대파,참기름,통 깨,매실원액,설탕	아삭아삭한 고추된장 박이	https://www.10000recipe.com/recipe/4030952

```
1 #html Document를 Return 받는 Function
2 def fn_get_html(url, params={}):
3     ## Get html Doc
4
5     ## navigator.userAgent 크롬 개발자 도구(console 입력)에서 확인.
6     ## 현재 요청을 보내는 곳은 내 pc인걸 확인 시켜 준다.
7     headers = {'User-Agent': 'Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/93.0.4577.82'}
8
9     response = requests.get(url, params=params, headers=headers).content
10    # response 된 html doc를 BeautifulSoup로 파싱(변환) 한다.
11    html_soup = BeautifulSoup(response, 'html.parser')
12
13    return html_soup
```



```

1  def fn_get_page(page_num):
2
3      ## Request Url & Params
4      page_url = target_url.get('domain') + target_url.get('recipe_list') # Ex : https://www.10000recipe.com + /recipe/list.html
5      page_params['page'] = page_num # 스크래핑 페이지 번호
6      page_params['sequence'] = 0 # 스크래핑 페이지의 recipe 순번
7
8      print('*****')
9      print('page url : ', page_url, '\npage params : ', page_params)
10     print('*****')
11
12     # BeautifulSoup으로 html doc return
13     html_page_soup = fn_get_html(page_url, page_params)
14
15
16     ## Get Pagination
17     # page 정보 스크래핑 및 다음 페이지 존재여부 확인 가능
18     list_pagination_tag = html_page_soup.find('ul', {'class' : 'pagination'})
19
20     # attrs='aria-label' : None => 속성(Attribute) aria-label이 None(없음) 것들만 찾는다.
21     list_page_tag = list_pagination_tag.find_all('a', attrs={'aria-label': None})
22
23     # attrs='aria-label' : None => 속성(Attribute) aria-label이 Next인 것들만 찾는다.
24     list_page_next = list_pagination_tag.find_all('a', attrs={'aria-label': 'Next'})
25     list_page_num = [x.get_text() for x in list_page_tag] # 페이지 리스트 태그 존재
26
27     ## Get Recipe Url List
28     # 스크래핑 주 목적인 레시피의 url 리스트를 가져온다.
29     list_recipe_tag = html_page_soup.find_all('a', attrs={'class' : 'common_sp_link'})
30     list_recipe_url = [x.attrs['href'] for x in list_recipe_tag]
31
32     ## Progress Bar && Recipe Scraping
33     # 프로그래스 바를 보기 위해 리스트를 tqdm_notebook으로 감싼다.
34     pbar = tqdm_notebook(list_recipe_url)
35

```




```
20 # attrs='aria-label' : None => 속성(Attribute) aria-label이 None(없는) 것들만 찾는다.
21 list_page_tag = list_pagination_tag.find_all('a', attrs={'aria-label': None})
22
23 # attrs='aria-label' : None => 속성(Attribute) aria-label이 Next인 것들만 찾는다.
24 list_page_next = list_pagination_tag.find_all('a', attrs={'aria-label': 'Next'})
25 list_page_num = [x.get_text() for x in list_page_tag] # 페이지 리스트 태그 존재
26
27 ## Get Recipe Url List
28 # 스크래핑 주 목적의 레시피의 url 리스트를 가져온다.
29 list_recipe_tag = html_page_soup.find_all('a', attrs={'class' : 'common_sp_link'})
30 list_recipe_url = [x.attrs['href'] for x in list_recipe_tag]
31
32 ## Progress Bar && Recipe Scriping
33 # 프로그래스 바를 보기 위해 리스트를 tqdm_notebook으로 감싼다.
34 pbar = tqdm_notebook(list_recipe_url)
35
36 # 데이터 가공시 error가 나더라도 계속 진행한다.
37 for f_recipe_url in pbar:
38     try:
39         # 현재 페이지의 레시피의 순서
40         page_params['sequence'] = page_params['sequence'] + 1
41
42         #프로그래스 바의 설명 글을 위한 string Format
43         str_desc = 'type:{0} page : {1}, seq: {2}'.format(
44             page_params.get('cat4_name'),
45             page_params.get('page'),
46             page_params.get('sequence')
47         )
48
49         #프로그래스 바 실행
50         pbar.set_description(str_desc)
51
52         #레시피 List안에 담긴 url들을 하나씩 해당 url로 접근하여 스크래핑 진행
53         fn_get_recipe(f_recipe_url)
54     except:
55         continue
56
57 clear_output(wait=True)
58
59 if (len(list_page_next) > 0):
60     return True
61 else:
62     return False
63
```



```
1 # Recipe 상세 페이지의 스크래핑 및 데이터 저장 Function
2 def fn_get_recipe(recipe_url):
3
4     ## Request Url & Params
5     target_url['recipe'] = recipe_url # Ex : /recipe/6912734
6     recipe_url = target_url.get('domain') + target_url.get('recipe') # Ex : https://www.10000recipe.com + /recipe/6912734
7
8
9     # 데이터 저장을 위한 dict
10    data_params = {
11        'page_num' : page_params.get('page'), # 페이지 번호
12        'sequence' : page_params.get('sequence'), # recipe 순번
13        'cooking_type' : page_params.get('cat4_name'), # 요리 분류 Ex 밀반찬,
14        'title' : '', #
15        'ingredient' : '',
16        'source' : '',
17        'uri' : recipe_url
18    }
19
20    # BeautifulSoup으로 html doc return
21    html_recipe_soup = fn_get_html(recipe_url)
22
23    ## Title
24    data_params['title'] = html_recipe_soup.find('div', {'class' : 'view2_summary st3'}).find('h3').get_text()
25
26    # 데이터 가공시 error가 나더라도 존재하는 데이터만 가지고 저장을 진행하고 다음 프로세스를 진행한다.
27    try:
28
29        ## Recipe
30        recipe_area = html_recipe_soup.find('div', {'id' : 'divConfirmedMaterialArea'})
31
32        ## Recipe - span 제거
33        span_elements = recipe_area.find_all('span')
34        for span in span_elements:
35            span.decompose()
36
37        ## Recipe And Source List
38        list_recipe = [x.get_text(strip=True) for x in recipe_area.find_all('ul')[0].find_all('li')]
39        data_params['ingredient'] = ','.join(list_recipe)
40
41        if(len(recipe_area.find_all('ul')) > 1):
42            list_source = [x.get_text(strip=True) for x in recipe_area.find_all('ul')[1].find_all('li')]
43            data_params['source'] = ','.join(list_source)
44        finally:
45            fn_save_csv(data_params)
46            fn_insert_sql(data_params)
```

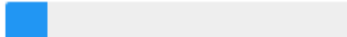


n [*]: **while True:**

```
# 현재 들린 페이지 저장.  
next_page_number = page_params['page']  
  
# 스크래핑 진행하고 나서 다음 페이지 존재여부 확인.  
bool_next_page_existence = fn_get_page(next_page_number)  
  
# 다음 페이지 존재 시  
if(bool_next_page_existence):  
    page_params['page'] = page_params['page'] + 1  
else:  
    # 다음 페이지 존재 하지 않을 시  
    break
```

page url : <https://www.10000recipe.com/recipe/list.html>

page params : {'order': 'reco', 'cat4': '66', 'cat4_name': '빵', 'page': 1, 'sequence': 0}

type:빵 page : 1, seq: 6  12% 5/40 [00:02<00:19, 1.77it/s]

4

”

전처리 및 유사도 분석





첫째

전처리 된 데이터를 받아 결측치 수정

둘째

유저에게 레시피 분류와 재료를 입력 받아 저장

셋째

입력 받은 레시피 분류를 통해 데이터 필터링

넷째

입력 받은 재료를 데이터와 유사도 분석 후 추천



```
In [14]: import pandas as pd
```

```
from sklearn.feature_extraction.text import TfidfVectorizer # TF-IDF 벡터화용
from sklearn.metrics.pairwise import cosine_similarity # 유사도 분석용
```

```
In [15]: # 카테고리 받기 전 결측치 전처리
def recipe_preprocess(recipe):
```

```
    # '소스' 열 결측치 '-' 로 대체
    recipe['source'].fillna('-', inplace=True)
```

```
    # 전처리 데이터 중 '재료' 열에 NaN 이 있는 경우 해당 행 모두 삭제
    recipe.dropna(axis=0, inplace=True, subset=['ingredient'])
```

```
In [16]: # 분류 값 입력
```

```
def get_category():
```

```
    # 카테고리 선택 범위
```

```
    cat_list = ['빵', '메인반찬', '찌개', '국', '탕', '디저트', '밀반찬', '양식', '퓨전']
```

```
    print(f'현재 선택할 수 있는 카테고리는 {cat_list} 입니다.')
```

```
    while(True):
```

```
        cat = input('찾고 싶은 레시피의 분류를 입력해주세요 : ')
```

```
        # 카테고리 선택 범위 밖에 있는 경우 무한 루프를 돌게 됨
```

```
        if cat not in cat_list:
```

```
            print('카테고리에 있는 값이 아닙니다. 다시 입력해주세요')
```

```
        else:
```

```
            break
```

```
    return cat
```




```
In [17]: # 카테고리 선택 후 필터링
def cat_preprocess(cat, recipe):
    # 1. 기존 recipe DF에서 cooking_type열이 cat과 같은 데이터를 DF화
    recipe1 = recipe[(recipe['cooking_type'] == cat)]
    # 기존 인덱스 번호 초기화
    recipe1.reset_index(inplace=True)
    # 2. 1의 DF에서 재료만 리스트 형태로 반환
    origin_recipe = recipe1['ingredient'].tolist()
    # 인덱스를 통해 자료를 뽑기 위한 recipe1 과 vectorizer 추가를 위한 리스트 형태의 origin_recipe 반환
    return recipe1, origin_recipe
```

```
In [33]: # 재료 입력
def get_ingredients():
    cnt = 1
    recipe_list = []
    ingredient = ''
    print('최대 5개까지의 재료를 입력해주시고, 입력이 마무리 된 경우 x 입력해주세요')
    while(cnt < 6):
        ingredient_element = input('재료를 입력해주세요. : ')

        #재료입력에 대소문자 'X' 혹은 공백이 들어올 경우 입력이 마무리 됨
        if ingredient_element == 'x' or ingredient_element=='X' or ingredient_element == ' ':
            break
        else:
            recipe_list.append(ingredient_element)

        cnt += 1

    print(f'선택된 재료들은 {recipe_list} 입니다.')
    input_recipe = ','.join(recipe_list)
    return input_recipe
```



```
In [19]: # 벡터화
def recipe_vectorizer(input_recipe, origin_recipe):

    # input_recipe = 입력받은 레시피 리스트의 str 형태, original_recipe = 레시피 리스트 형태
    origin_recipe.append(input_recipe)

    # TfidfVectorizer() 객체 변수 생성, 광통 모델
    vectorizer = TfidfVectorizer()

    # recipe 재료 데이터 벡터화, .todense 는 반환 값 보고싶을 때 사용(사용 안 할경우 성능 항상 있음)
    vec = vectorizer.fit_transform(origin_recipe)

    return vec
```

```
In [20]: # 벡터화된 레시피 값을 받아 유사도 분석 후 분석된 레시피의 기존 index 값과 유사도 상위 5개를 DataFrame 반환
def sim_indexPreprocess (recipe_vec):

    # 입력된 recipe와 전체 recipe 의 유사도분석
    sim = pd.DataFrame(cosine_similarity(recipe_vec[-1] , recipe_vec))

    # 유사도분석 된 자료를 정렬 후 상위 5개까지 추출 정렬
    sim.sort_values(by=0,axis=1,ascending=False,inplace=True)

    # 유사도 분석 df와 레시피 df의 인덱스 행을 맞추기 위해 행열 전환처리(T=transpose) + 0번을 제외한 유사도 상위 5개 반환
    sim_per = sim.iloc[ : , 1:6].T

    # 컬럼명(= recipe index 번호 추출)
    recipe_index = sim_per.index.tolist()

    return recipe_index, sim_per
```

04 전처리 및 유사도 분석



```
In [21]: # 유사도 결과 출력
def recommendation_result(index_list, ref_recipe):
    print('입력하신 재료와 유사한 레시피 최대 5개를 추출합니다.\n')

    for info in index_list:
        title = ref_recipe.iloc[info]['title'] #제목
        ingredient = ref_recipe.iloc[info]['ingredient'] #재료
        url = ref_recipe.iloc[info]['uri'] # URL
        source = ref_recipe.iloc[info]['source'] # 소스
        similarity = (sim_per.loc[info][0])*100 # 유사도
        print(f'레시피 : {title} (유사도:{similarity:.1f}%)')
        print(f'재 료 : {ingredient}')
        print(f' url : {url}')
        print(f'소 스 : {source}')
        print('\n')
```

```
In [22]: recipe = pd.read_csv('recipe2.csv', encoding='utf-8')
recipe.head()
```

idx	page_num	seq	cooking_type		title	ingredient	source	uri	insert_dt
0	1	1	1	밀반찬	버섯으로 관자 느낌 내는 방법! 새송이버섯간장버 터구이 만들기	새송이버섯,버터, 어린잎채소	간장,맛술,올리고당	https://www.10000recipe.com/recipe/6912734	2021-10-06 18:23:56
1	2	1	2	밀반찬	500원 이면 밥한끼~ 똑딱 하는 팽이버섯조림~ 너무 맛있어용^^~	팽이버섯,청양고 추,식용유,다진마 늘	고추가루,설탕,미림,진 간장,굴소스,고추장,된 장,물	https://www.10000recipe.com/recipe/6956266	2021-10-06 18:23:57
2	3	1	3	밀반찬	[마늘종볶음]단짠단짠 자 꾸만 손이가요~	건새우,물,마늘 종,소금	설탕,간장,식용유,참기 름,물엿,통깨	https://www.10000recipe.com/recipe/6957535	2021-10-06 18:23:57
3	4	1	4	밀반찬	달콤 짭조름♡ 떡 베이컨 간장조림♡	베이컨,대파,떡	간장,참기름,물엿,기름	https://www.10000recipe.com/recipe/6843136	2021-10-06 18:23:58
4	5	1	5	밀반찬	아삭아삭한 고추된장박이	풋고추	된장,다진 마늘,송송썰 은 대파,참기름,통깨,매 실원액,설탕	https://www.10000recipe.com/recipe/4030952	2021-10-06 18:23:58



```
In [23]: #결측치, 행 개수 확인
recipe.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 31961 entries, 0 to 31960
Data columns (total 9 columns):
idx                31961 non-null int64
page_num          31961 non-null int64
seq               31961 non-null int64
cooking_type      31961 non-null object
title             31959 non-null object
ingredient        31412 non-null object
sorce             17739 non-null object
uri              31961 non-null object
insert_dt         31961 non-null object
dtypes: int64(3), object(6)
memory usage: 2.2+ MB
```

```
In [24]: # recipe DataFrame 전처리
recipe_preprocess(recipe)
```

```
In [25]: # 카테고리 입력
cat = get_category()
```

현재 선택할 수 있는 카테고리는 ['빵', '메인반찬', '찌개', '국', '탕', '디저트', '밑반찬', '양식', '퓨전'] 입니다.
찾고 싶은 레시피의 분류를 입력해주세요 : 빵

```
In [26]: # 입력받은 메뉴를 str 형태로 반환 (최대 5개까지 입력)
input_recipe = get_ingredients()
```

최대 5개까지의 재료를 입력해주시고, 입력이 마무리 된 경우 x 입력해주세요
재료를 입력해주세요. : 밀가루
재료를 입력해주세요. : x
선택된 재료들은 ['밀가루'] 입니다.



```
In [27]: ref_recipe, origin_recipe = cat_preprocess(cat, recipe)
```

```
In [28]: # 입력받은 레시피 재료와 기존 레시피 재료 DataFrame을 벡터화
recipe_vec = recipe_vectorizer(input_recipe, origin_recipe)
```

```
In [29]: # 벡터화된 레시피 값을 받아 유사도 분석 후 분석된 레시피의 기존 index 값과 유사도 상위 5개를 DataFrame 반환
recipe_index, sim_per = sim_indexPreprocess(recipe_vec)
```

```
In [30]: # 유사도 결과 출력
recommendation_result(recipe_index, ref_recipe)
```

입력하신 재료와 유사한 레시피 최대 5개를 추출합니다.

레시피 : 손반죽) 쫄깃하고 부드러운 탕종식빵 만들기 (영상/레시피) (유사도:100.0%)

재 료 : 물, 밀가루

url : <https://www.10000recipe.com/recipe/6932249>

소 스 : 강력분, 소금, 설탕, 이스트, 분유, 우유, 버터, 계란

레시피 : 햄프씨드가 쿡쿡박힌 흑미식빵~~정말 고소해요 (유사도:85.7%)

재 료 : 밀가루, 우유

url : <https://www.10000recipe.com/recipe/6895333>

소 스 : 밀가루장, 우유, 달걀, 제빵용흑미가루, 강력분, 햄프씨드, 설탕, 소금, 이스트, 버터

레시피 : 건강 호박파이 만들기 (노버터, 노오일) (유사도:78.0%)

재 료 : 밀가루, 우유, 소금

url : <https://www.10000recipe.com/recipe/6966185>

소 스 : 단호박, 우유, 계란, 계피가루, 생강가루, 후추가루, 정향 가루, 넛맥가루, 설탕, 소금

레시피 : 밥술 우유식빵 (유사도:75.8%)

재 료 : 밀가루, 물, 우유, 설탕, 소금, 프림, 드라이이스트, 밀가루, 버터, 밀가루

url : <https://www.10000recipe.com/recipe/6940268>

소 스 : -



입력받은 재료와 최대한 가까운 레시피에 대한 레시피 URL을 반환

입력하신 재료와 유사한 레시피 최대 5개를 추출합니다.

레시피 : 손반죽) 찰깃하고 부드러운 탈종식빵 만들기 (영상/레시피) (유사도:100.0%)

재 료 : 물, 밀가루

url : <https://www.10000recipe.com/recipe/6932249>

소 스 : 강력분, 소금, 설탕, 이스트, 분유, 우유, 버터, 계란

레시피 : 햄프씨드가 콕콕박힌 흑미식빵~~정말 고슬해요 (유사도:85.7%)

재 료 : 밀가루, 우유

url : <https://www.10000recipe.com/recipe/6895333>

소 스 : 밀가루장, 우유, 달걀, 제빵용흑미가루, 강력분, 햄프씨드, 설탕, 소금, 이스트, 버터

레시피 : 건강 호박파이 만들기 (노버터, 노오일) (유사도:78.0%)

재 료 : 밀가루, 우유, 소금

url : <https://www.10000recipe.com/recipe/6966185>

소 스 : 단호박, 우유, 계란, 계피가루, 생강가루, 후추가루, 정향 가루, 넛맥가루, 설탕, 소금

레시피 : 발쫓 우유식빵 (유사도:75.8%)

재 료 : 밀가루, 물, 우유, 설탕, 소금, 프림, 드라이이스트, 밀가루, 버터, 밀가루

url : <https://www.10000recipe.com/recipe/6940268>

소 스 : -

레시피 : [영상] NO오븐 소보로빵 만들기 : '냉장고를 부탁해' 김풍 '최후의 소보로' (유사도:73.9%)

재 료 : 버터, 설탕, 밀가루, 식빵, 잼

url : <https://www.10000recipe.com/recipe/6843635>

소 스 : -



후속과제

1. 데이터 수집을 더 진행 해 레시피 데이터 유사도의 정확성 향상
2. 현재 구현해 둔 코드를 웹페이지나 어플로 구현

“감사합니다

