**OCR을 활용한 카페메뉴 추천 웹**

**<1. Selenium을 이용한 YOGIYO Web Crawling>**

**#파이썬 라이브러리(사전 설치필요)**

import time

import pickle

import pandas as pd

from tqdm import tqdm #상태바 나타내는 라이브러리

from tqdm import trange

import warnings

warnings.filterwarnings('ignore')

**# 크롤링 패키지**

import requests

from selenium import webdriver

**#요기요에서 제공하는 파스쿠찌매장(서울) 주소 리스트**

test\_list = ['서울 종로구 대학로 142', '서울시 종로구 관수동 3-1', '서울시 종로구 세종대로 23길 54', … , '서울특별시 강남구 도곡동 953 다림빌딩']

**#크롤링한 총 결과 담을 데이터프레임 생성(가져올 컬럼 : 매장이름, 사용자 아이디, 시킨 메뉴, 리뷰글, 총 평점, 맛 평점, 양 평점, 배달 평점, 날짜)**

df\_result = pd.DataFrame(columns=['Restaurant','UserID','Menu','Review',

'Total','Taste','Quantity','Delivery','Date'])

**#웹 드라이버(Chrome, Firefox,... 설치필요)**

**#크롬 :** [**http://chromedriver.chromium.org/downloads**](http://chromedriver.chromium.org/downloads)

**#(‘크롬 웹 드라이버저장된주소\chromedriver.exe’)**

wd = webdriver.Chrome('C:\chromedriver.exe')

**#Chrome webdriver - 요기요 페이지 크롤링 실행**

for i in test\_list:

wd.get('https://www.yogiyo.co.kr/')

print('요기요 페이지 시작’')

time.sleep(1.5)

**#요기요 페이지 주소창에 test\_list입력**

wd.find\_element\_by\_xpath('//\*[@id="search"]/div/form/input').click()

wd.find\_element\_by\_xpath('//\*[@class="btn-search-location-cancel btn-search-location btn btn-default"]').click()

wd.find\_element\_by\_name('address\_input').send\_keys(i)

wd.find\_element\_by\_xpath('//\*[@id="button\_search\_address"]').click()

time.sleep(2)

**#파스쿠찌 이미지로 파스쿠찌 리뷰 찾기**

wd.find\_element\_by\_xpath('//\*[@style="background-image: url(\'/media/franchise\_logos/파스쿠치\_20180419\_Franchise이미지약정서\_crop\_200x200.png\'), url(\'image/default\_restaurant\_logo.png\');"]').click()

time.sleep(1.5)

**#해당 파스쿠찌 매장의 클린리뷰창**

wd.find\_element\_by\_xpath('//\*[@id="content"]/div[2]/div[1]/ul/li[2]/a').click()

wd.find\_elements\_by\_css\_selector('#review > li.list-group-item.star-point.ng-scope')

**#리뷰 불러오기**

review\_count = int(wd.find\_element\_by\_xpath('//\*[@id="content"]/div[2]/div[1]/ul/li[2]/a/span').text)

click\_count = int((review\_count/10))

print('모든 리뷰 불러오기 시작')

**#더보기 클릭**

for \_ in trange(click\_count):

try:

wd.find\_element\_by\_class\_name('btn-more').click()

time.sleep(2)

except Exception as e:

pass

print('모든 리뷰 보기 완료’')

time.sleep(2)

reviews = wd.find\_elements\_by\_css\_selector('#review > li.list-group-item.star-point.ng-scope')

time.sleep(2)

**#크롤링한 한 지점의 결과 담을 데이터프레임**

df = pd.DataFrame(columns=['Restaurant','UserID','Menu','Review',

'Total','Taste','Quantity','Delivery','Date'])

**#해당 지점의 리뷰 수 만큼 데이터를 가져옴(가져올 컬럼 : 매장이름, 사용자 아이디, 시킨 메뉴, 리뷰글, 총 평점, 맛 평점, 양 평점, 배달 평점, 날짜)**

for review in tqdm(reviews):

try:

df.loc[len(df)] = {

'Restaurant':wd.find\_element\_by\_class\_name('restaurant-name').text,

'UserID':review.find\_element\_by\_css\_selector('span.review-id.ng-binding').text,

'Menu':review.find\_element\_by\_css\_selector('div.order-items.default.ng-binding').text,

'Review':review.find\_element\_by\_css\_selector('p').text,

'Total':str(len(review.find\_elements\_by\_css\_selector('div > span.total > span.full.ng-scope'))),

'Taste':review.find\_element\_by\_css\_selector('div:nth-child(2) > div > span.category > span:nth-child(3)').text,

'Quantity':review.find\_element\_by\_css\_selector('div:nth-child(2) > div > span.category > span:nth-child(6)').text,

'Delivery':review.find\_element\_by\_css\_selector('div:nth-child(2) > div > span.category > span:nth-child(9)').text,

'Date':review.find\_element\_by\_css\_selector('div:nth-child(1) > span.review-time.ng-binding').text,

}

except Exception as e:

print('리뷰 페이지 에러')

print(e)

pass

time.sleep(2)

print('크롤링 완료')

time.sleep(2)

**#크롤링 결과 피클에 입력**

pickle.dump(df, open('./data/first.pkl','wb'))

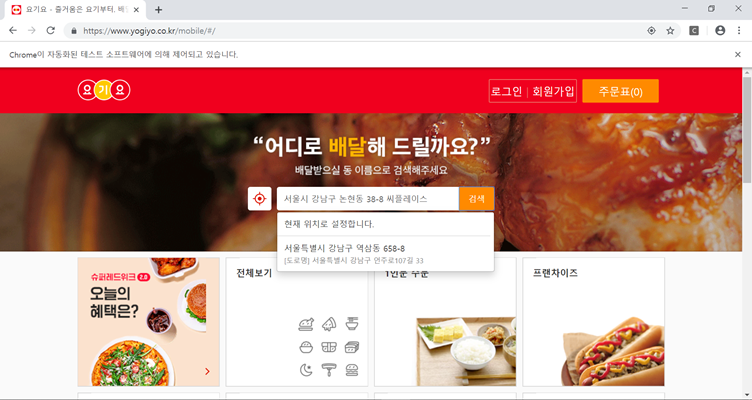
**#피클에서 가져오기**

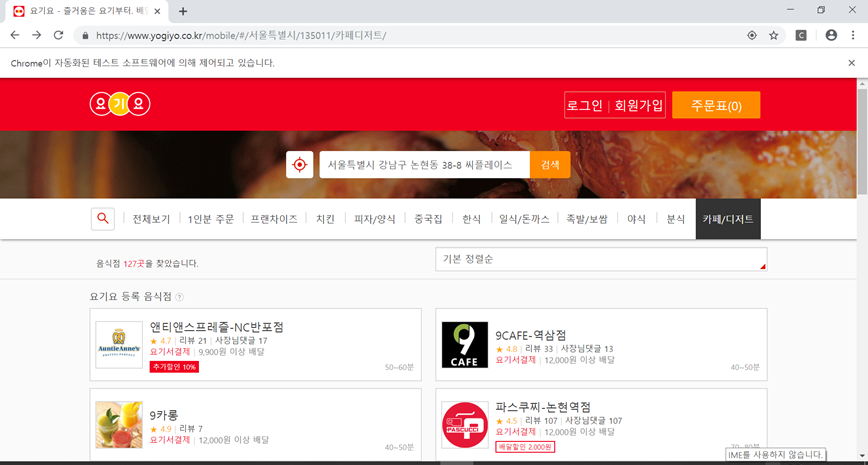
df = pickle.load(open('./data/first.pkl','rb'))

**#데이터 프레임 결합**

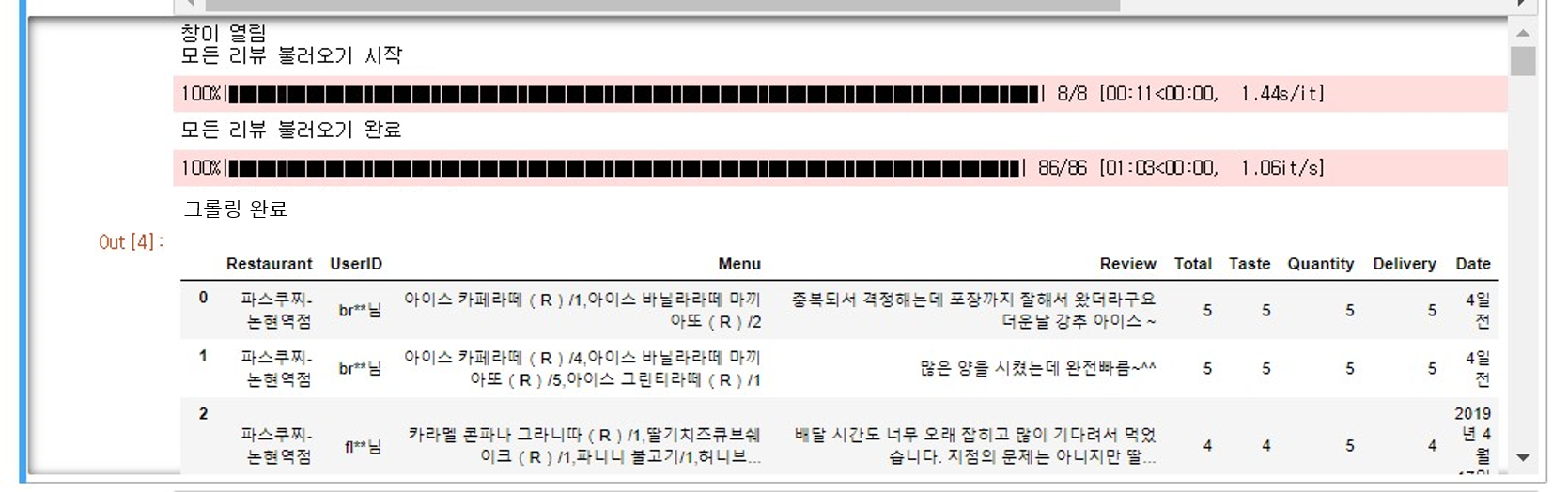
df\_result = pd.concat([df\_result,df])

display(df\_result)









**#크롤링 총 결과 데이터 프레임 CSV파일로 저장**

df\_result.to\_csv("C:\example\clawringexample\_20190516\_True.csv",header=True,index=True)

df\_result.to\_csv("C:\example\clawringexample\_20190516\_False.csv",header=False,index=False)

**<2. Crawling결과 데이터 전처리>**

import numpy as np

import pandas as pd

import datetime

**##1. 크롤링한 결과 csv 파일로 불러오기**

df = pd.read\_csv("./data/cafe/clawringexample\_06\_True.csv")

**##2. 메뉴 결측값 제거**

df=df.dropna(axis=0) # 값이 없는 행 모두 제거

**##3. 메뉴 별 리뷰 분리(메뉴 3개 시켰다면 메뉴 하나씩으로 행 구분)**

df = \

(df.set\_index(df.columns.drop('Menu',1).tolist())

.Menu.str.split(',', expand=True)

.stack()

.reset\_index()

.rename(columns={0:'Menu'})

.loc[:, df.columns]

)

**##4. 메뉴 이름 외 값 제거**

df[['Menu','Option']] = df.Menu.str.split("/",expand=True)

del df['Option']

df[['Menu','Option']] = df.Menu.str.split("（",expand=True)

del df['Option']

**##5. 정리된 데이터프레임(df)에서 우리가 필요한 컬럼만 불러오기**

df=df[["Restaurant","Menu","Review","Total","Taste","Quantity","Date"]]

**# Date가 문자열로 되어있는데 날짜 데이터로 변환하기**

df["Date"] = pd.to\_datetime(df["Date"])

**##6. 기상청에서 다운받은 날짜별 날씨 데이터로 비여부 판단**

**# AvgTemp : 평균기온 / RainDuration : 강수지속시간(일)**

**#RainMaxHour : 1시간 최대 강수량 / RainAday : 일강수량**

w = pd.read\_csv("./data/cafe/weather.csv")

**#문자로 되어있는 Date의 데이터를 날짜 형식으로 바꾸기**

w["Date"] = pd.to\_datetime(w["Date"])

w = pd.DataFrame(w, columns=["Date","AvgTemp","RainDuration","RainMaxHour","RainAday","Rain"])

**#비오는 기준(임의로 정함)**

**#강수지속시간(일)>3(시간) or 1시간 최대 강수량>=5(mm) or 일강수량>=5(mm) -> 비온날 True**

w["Rain"] = (w["RainDuration"]) > 3 | (w["RainDuration"] >=5 ) | (w["RainAday"]>=5)

**##7. 크롤링 데이터와 날씨 데이터를 Date 컬럼을 기준으로 합침**

data=pd.merge(df, w , on="Date", how="inner")

data2 = data[["Date","Restaurant","Menu","Review","Taste","Quantity","Rain"]]

data3 = data2.sort\_values(by="Date", axis =0 , ascending=False, inplace=False)

**##8. 크롤링 + 비여부 판단 데이터 최종 (띄어쓰기 제거 모든 메뉴 한 단어로 만들기)**

menu\_list=[]

for i in (data3["Menu"]):

menu\_list.append(i.replace(' ',''))

data3["Menu"] = menu\_list

**## 9. 메뉴별 빈도수 구해보기**

data4=data3.groupby(["Menu"]).count()['Total']

Menu\_in\_Rv = pd.DataFrame(data4)

Menu\_in\_Rv.reset\_index(level=['Menu'], inplace = True)

# Menu\_in\_Rv.to\_csv("./data/cafe/Menu\_in\_Rv.csv", header = True, index = False)

**<3. OCR api 이용해서 이미지에서 텍스트 추출>**

**#ocr계정생성 후 실행**

import os

import io

import re

import numpy as np

import pandas as pd

from google.cloud import vision

from google.cloud.vision import types

**# cmd 창에 setx GOOGLE\_APPLICATION\_CREDENTIALS "파일경로/파일명.json"**

**# 혹은 cmd 창에 set GOOGLE\_APPLICATION\_CREDENTIALS = "파일경로/파일명.json"**

**# 해서 성공 뜨면 연결된 것.**

**#google과 사용자를 연결하는 코드**

client = vision.ImageAnnotatorClient()

**#분석하려는 이미지 파일 경로를 잡는 코드**

**#(os.path.dirname('파일경로'), '이미지파일명')**

file\_name = os.path.join(

os.path.dirname('C:'),

'/example/example2.jpg')

**#이미지 로드**

with io.open(file\_name, 'rb') as image\_file:

content = image\_file.read()

image = types.Image(content=content)

**#이미지 파일에서 text를 감지**

response = client.text\_detection(image=image)

texts = response.text\_annotations

my\_list = list()

**# 이미지에서 추출한 texts들 중 description만 뽑아내기**

for text in texts:

result = text.description

my\_list.append(result)

data = my\_list[0]

data1 = data.replace('\n',' ')

data2 = data1.replace('(',' ')

data3 = data2.replace(')',' ')

data4 = data3.replace('/',' ')

data5 = data4.split(' ')

df = pd.DataFrame(data5, columns=["총리스트"])

df1 = pd.DataFrame(columns=["0","1","2","3"])

df2 = pd.DataFrame(columns=["0","1","2","3"])

df1.loc[0, '3'] ="아메리카노"

df1.loc[1, '2'] ="아이스"

df1.loc[1, '3'] ="아메리카노"

...

df1.loc[2, '2'] ="아이스"

df1.loc[2, '3'] ="카페라떼"

df1.loc[3, '3'] ="카페라떼"

**#df1에 있는 파스쿠찌 모든 메뉴 한 단어로 만드는 과정 => Allmenu에 담음**

menu=df1[['0','1','2','3']].astype(str).sum(axis=1)

menu = menu.str.replace('nan', '')

m2=menu.unique()

Allmenu = pd.DataFrame(m2, columns = ["AllMenu"])

display(Allmenu)

for i in range(0,len(df.index)):

for i2 in range(0,len(df1.index)):

if df1.loc[i2,'0'] == df.loc[i,"총리스트"]:

df2.loc[i2,'0'] = df.loc[i,"총리스트"]

elif df1.loc[i2,'1'] == df.loc[i,"총리스트"]:

df2.loc[i2,'1'] = df.loc[i,"총리스트"]

elif df1.loc[i2,'2'] == df.loc[i,"총리스트"]:

df2.loc[i2,'2'] = df.loc[i,"총리스트"]

elif df1.loc[i2,'3'] == df.loc[i,"총리스트"]:

df2.loc[i2,'3'] = df.loc[i,"총리스트"]

**# 사진에서 추출한 텍스트 메뉴랑 비교해서 한 단어로 만들기**

df3=df2.sort\_index()

match=df3[['0','1','2','3']].astype(str).sum(axis=1) #match -> array 형식 astype -> 문자로 변환

match = match.str.replace('nan', '') # 난값을 공백

match = match.unique() # 중복메뉴 (아이스 4개행 중복 걸러줌)

imgtxt = pd.DataFrame(match, columns = ["imgtxt"])

**<4. 2에서 만든 데이터에서 Review의 텍스트를 NLU API 돌려서 수치화 하기>**

import json

from watson\_developer\_cloud import NaturalLanguageUnderstandingV1

from watson\_developer\_cloud.natural\_language\_understanding\_v1 import Features, SentimentOptions

import pandas as pd

**# 1에서 저장한 Data.csv 파일 불러오기**

Data = pd.read\_csv("./data/cafe/Data.csv")

R=Data["Review"]

natural\_language\_understanding = NaturalLanguageUnderstandingV1(

version='2018-11-16',

iam\_apikey='개인 apikey',

url='https://gateway.watsonplatform.net/natural-language-understanding/api'

)

\_list=list()

for txt in R:

response = natural\_language\_understanding.analyze(text= txt,\

features=Features(sentiment=SentimentOptions())).get\_result()

\_keys=response["sentiment"]["document"]["score"]

\_list.append(\_keys)

Data["Review"] = \_list

**# NLU 수치 정규화**

Data["Review"] = MinMaxScaler().fit\_transform(Data[["Review"]].values)

**Data.to\_csv("./data/cafe/Data\_list.csv", header = True, index = True)**

**##생성된 csv파일 아래서 데이터베이스에 넣기(정규화 전)**

**# 메뉴별로 groupby해서 평균 구하기**

Data01 = Data.groupby(['Menu']).mean()

Data01["NoRain"] = 1 - Data01["Rain"]

#Data01.to\_csv("./data/cafe/Data02.csv", header = True, index = True)

**<5. 최종 데이터 전처리 과정 (모든 컬럼 0 ~ 1로 정규화 하고 가중치 부여해서 Result 값 뽑아내기)>**

from sklearn.preprocessing import MinMaxScaler

**## 1. 데이터 불러오기**

Data = pd.read\_csv('./data/cafe/Data02.csv')

**## 1-2. 빈도수 데이터**

Fr = pd.read\_csv('./data/cafe/Menu\_in\_Rv.csv')

**## 2. 빈도수 정규화 후 데이터 추가**

Data["Frequency"]=Fr["Total"]/Fr["Total"].sum()

Frequency = MinMaxScaler().fit\_transform(Data[["Frequency"]].values)

Data["Frequency"]=Frequency

Data.to\_csv('./data/cafe/Result\_data.csv', header=True, index=True)

**## 3. 각 컬럼에 가중치 부여해서 최종 Result 값 구하기 ( 비여부에 따라 두 가지 결과 )**

**# 3-1. 비 올 때 결과 값**

Data["Rain\_Result"]=Data["Frequency"]\*0.5 + Data["Review"]\*0.25 + \

Data["Taste"]\*0.15 + Data["Rain"]\*0.07 + Data["Quantity"]\*0.03

Data.sort\_values("Menu",ascending=True)

Result\_Rain=Data.drop("NoRain", axis = 1 , inplace = False)

Result\_Rain.to\_csv('./data/cafe/Result\_data\_Rain.csv', header=True, index=True)

**# 3-2. 비 오지 않을 때 결과 값**

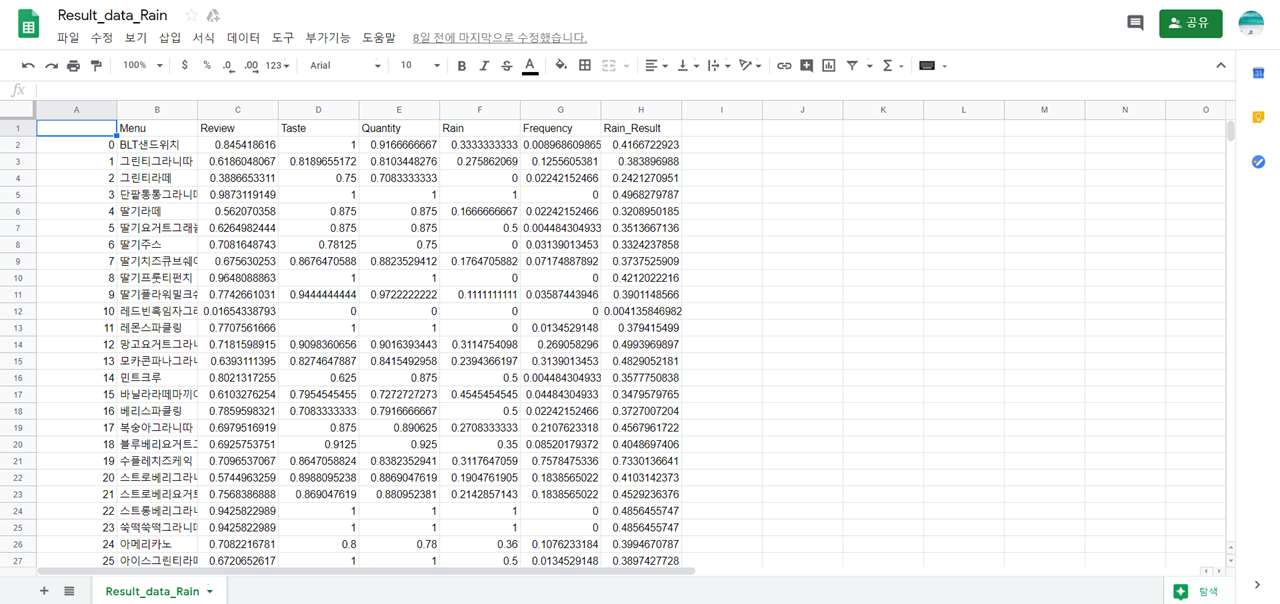
Data["NoRain\_Result"]=Data["Frequency"]\*0.5 + Data["Review"]\*0.25 + \

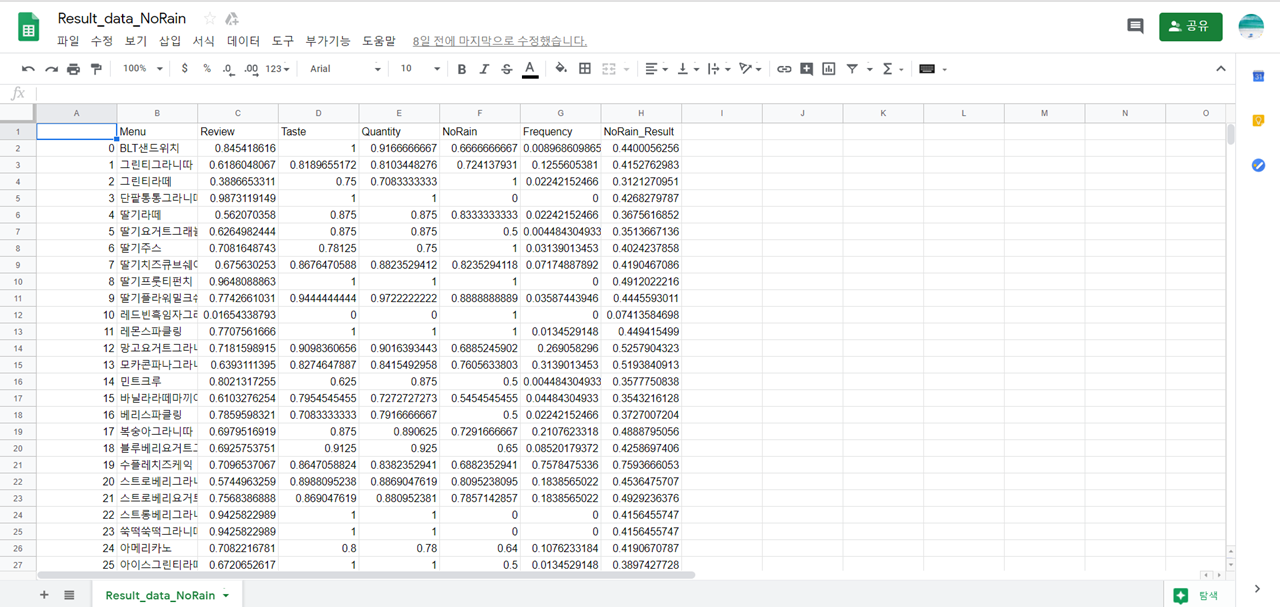
Data["Taste"]\*0.15 + Data["NoRain"]\*0.07 + Data["Quantity"]\*0.03

Data.sort\_values("Menu",ascending=True)

Result\_NoRain=Data.drop("Rain", axis = 1 , inplace = False)

Result\_NoRain.to\_csv('./data/cafe/Result\_data\_NoRain.csv', header=True, index=True)





**<6. Oracle Database에 저장>**

**#전처리 CSV파일 DB에 넣기( Result\_data\_Rain.csv, Result\_data\_NoRain.csv)**

**##Oracle DB에 테이블 만들기(오라클 DB에서 쿼리문 실행)**

create table data\_rain (

Menu varchar2(50),

Review number(7, 6),

Taste number(7, 6),

Quantity number(7, 6),

Rain number(7, 6),

Frequency number(7, 6),

RainResult number(7, 6)

);

create table data\_no\_rain (

Menu varchar2(50),

Review number(7, 6),

Taste number(7, 6),

Quantity number(7, 6),

NoRain number(7, 6),

Frequency number(7, 6),

NoRainResult number(7, 6)

);

**##파이썬코드로 CSV파일 DB에 넣기**

**#Result\_data\_Rain.csv**

import cx\_Oracle

import csv

con = cx\_Oracle.connect("dator/me@localhost:1521/xe") #oracle 접속 세부사항

cur = con.cursor()

with open('./data//Result\_data\_Rain.csv', "r", encoding='UTF8') as csv\_file: #csv파일로 저장된 크롤링 결과 불러오기

csv\_reader = csv.reader(csv\_file, delimiter=',')

next(csv\_reader) #헤더생략

for lines in csv\_reader:

sql\_insert= "insert into data\_rain values(:id1, :id2, :id3, :id4, :id5, :id6, :id7)"

cur.execute(sql\_insert, id1 = lines[1], id2 = lines[2], id3 = lines[3], id4 = lines[4], id5 = lines[5],id6 = lines[6],

id7 = lines[7]) #첫번째 열(인덱스)생략

cur.close()

con.commit()

con.close()

**#Result\_data\_NoRain.csv**

import cx\_Oracle

import csv

con = cx\_Oracle.connect("dator/me@localhost:1521/xe") #oracle 접속 세부사항

cur = con.cursor()

with open('./data//Result\_data\_NoRain.csv', "r", encoding='UTF8') as csv\_file: #csv파일로 저장된 크롤링 결과 불러오기

csv\_reader = csv.reader(csv\_file, delimiter=',')

next(csv\_reader) #헤더생략

for lines in csv\_reader:

sql\_insert= "insert into data\_no\_rain values(:id1, :id2, :id3, :id4, :id5, :id6, :id7)"

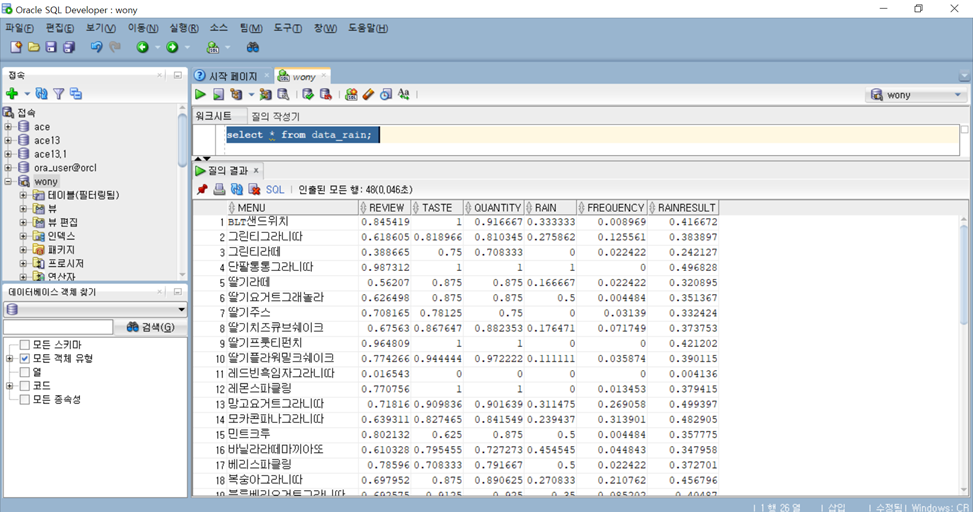
cur.execute(sql\_insert, id1 = lines[1], id2 = lines[2], id3 = lines[3], id4 = lines[4], id5 = lines[5],id6 = lines[6],

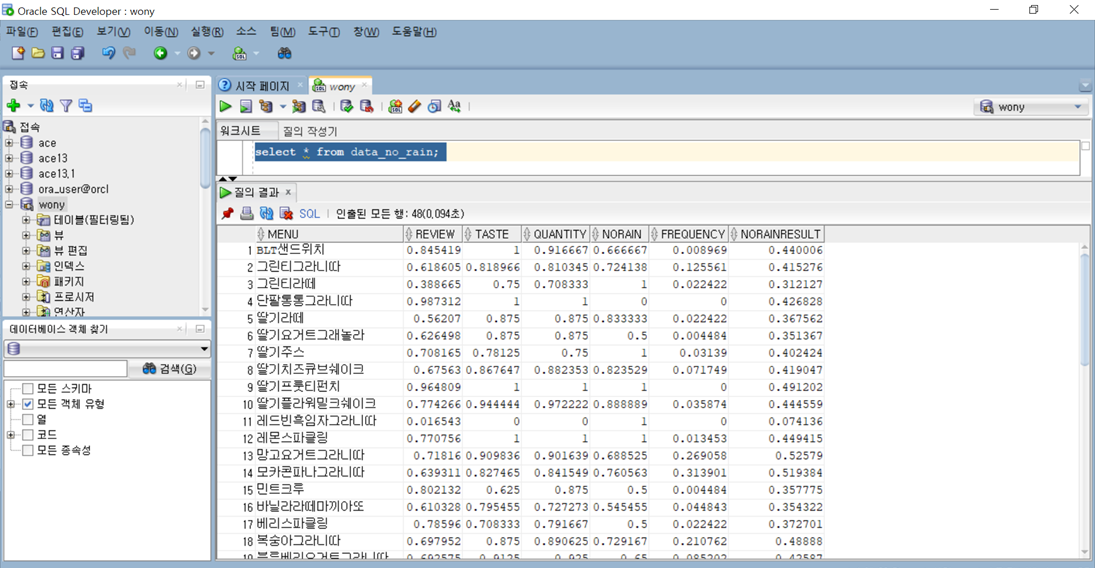
id7 = lines[7]) #첫번째 열(인덱스)생략

cur.close()

con.commit()

con.close()





###################################################################

**#정규화전 csv파일 DB에 넣기**

**#프로젝트의 최종 결과값(ex 아메리카노)이 나온 근거(각 컬럼들 예를들면 빈도수, 리뷰...)들의 수치를 나타내줄때 사용**

import cx\_Oracle #라이브러리 설치

import csv

con = cx\_Oracle.connect("dator/me@localhost:1521/xe") #oracle developer 접속 세부정보

cur = con.cursor()

with open(‘**./data/cafe/Data\_list.csv**', "r", encoding='UTF8') as csv\_file:

csv\_reader = csv.reader(csv\_file, delimiter=',')

next(csv\_reader)

for lines in csv\_reader: #lines는 한 컬럼을 의미함

sql\_insert= "insert into info5=10 values(:id1, :id2, :id3, :id4, :id5)"

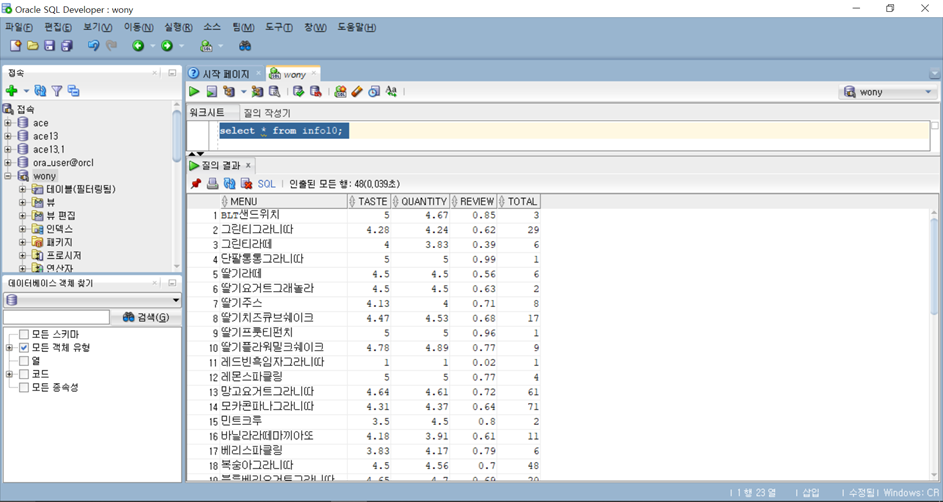
cur.execute(sql\_insert, id1 = lines[1], id2 = lines[2], id3 = lines[3],\

id4 = lines[4], id5 = lines[5])

cur.close()

con.commit()

con.close()



**<7. Django 웹 페이지 구현 원리 정리 >**

1. Views.py : 웹 페이지에 출력할 것들 함수로 만드는 곳
2. Urls.py : views.py에서 만든 함수들이 연결될 페이지 주소 만드는 곳
3. Html : 각 페이지 출력 및 디자인 설정
4. Models.py : sqlite3와 연결할 DB class 생성하는 곳
5. Admin.py : admin에 4에서 만든 것 등록
6. Setting.py : 각종 setting

**1.시작 페이지**

**1)views.py**

def index(request):

return render(request, 'posts/index.html')

**2)urls.py**

path('start/', views.index)

**3)index.html**

**2.사진 업로드 페이지**

**1)views.py**

class CreatePostView(CreateView):

model = Post

form\_class = PostForm

template\_name = 'posts/post.html'

success\_url = reverse\_lazy('result')

**2)urls.py**

path('start/post/', CreatePostView.as\_view(), name='add\_post')

**3)post.html**

**3.OCR 결과 텍스트와 이미지 띄우는 페이지**

**1)views.py**

def index01(request):

image = Post.objects.all()

key = Post.objects.all()

image\_db01=key[len(key)-1].cover

image\_db=str(image\_db01)

client = vision.ImageAnnotatorClient()

file\_name = os.path.join(

os.path.dirname('C:'), '/project/uploading/media/' + image\_db)

with io.open(file\_name, 'rb') as image\_file:

content = image\_file.read()

image = types.Image(content=content)

response = client.text\_detection(image=image)

texts = response.text\_annotations

my\_list = list()

for text in texts:

result = text.description

my\_list.append(result)

data = my\_list[0]

data1 = data.replace('\n',' ')

data2 = data1.replace('(',' ')

data3 = data2.replace(')',' ')

data4 = data3.replace('/',' ')

data4 = data3.replace('그라니때','그라니따')

data4 = data3.replace('Sparkling','스파클링')

data5 = data4.split(' ')

df = pd.DataFrame(data5, columns=["총리스트"])

df1 = pd.DataFrame(columns=["0","1","2","3"])

df2 = pd.DataFrame(columns=["0","1","2","3"])

df1.loc[0, '3'] ="아메리카노"

df1.loc[1, '2'] ="아이스"

…….

df1.loc[116,'2']="청포도"

df1.loc[116,'3']="Sparkling"

menu=df1[['0','1','2','3']].astype(str).sum(axis=1)

menu = menu.str.replace('nan', '')

m2=menu.unique()

Allmenu = pd.DataFrame(m2, columns = ["AllMenu"])

for i in range(0,len(df.index)):

for i2 in range(0,len(df1.index)):

if df1.loc[i2,'0'] == df.loc[i,"총리스트"]:

df2.loc[i2,'0'] = df.loc[i,"총리스트"]

elif df1.loc[i2,'1'] == df.loc[i,"총리스트"]:

df2.loc[i2,'1'] = df.loc[i,"총리스트"]

elif df1.loc[i2,'2'] == df.loc[i,"총리스트"]:

df2.loc[i2,'2'] = df.loc[i,"총리스트"]

elif df1.loc[i2,'3'] == df.loc[i,"총리스트"]:

df2.loc[i2,'3'] = df.loc[i,"총리스트"]

df3=df2.sort\_index()

match=df3[['0','1','2','3']].astype(str).sum(axis=1)

match = match.str.replace('nan', '')

match = match.unique()

imgtxt = pd.DataFrame(match, columns = ["imgtxt"])

results = list()

for s1 in m2:

for s2 in match:

if s1 == s2:

results.append(s1)

context = {'texts' : results}

# index2.html 창에서 {{texts}}로 결과 불러오기

return render(request, 'posts/index2.html', context)

**2)urls.py**

path('start/post/result/', views.index01, name = 'result'),

**3)index2.html**

**4)models.py ( sqllite3와 연동되는 DB 생성하기 ) ( admin 등록은 admin.py에서)**

from django.db import models

class Post(models.Model):

cover = models.ImageField(upload\_to='images/')

🡺 python manage.py createsuperuser # admin 계정 생성

🡺 python manage.py makemigrations # DB 생성

🡺 python manage.py migrate # DB 연결

**4.최종 결과 ( 추천메뉴 및 5순위 메뉴 ), 1순위 메뉴 점수 / 날씨 ( 맑은 날, 비 오는 날 ) / 오늘 날짜**

**1)views.py**

def index02(request):

image = Post.objects.all()

key = Post.objects.all()

image\_db01=key[len(key)-1].cover

image\_db=str(image\_db01)

client = vision.ImageAnnotatorClient()

file\_name = os.path.join( os.path.dirname('C:'), '/project/uploading/media/' + image\_db)

with io.open(file\_name, 'rb') as image\_file:

content = image\_file.read()

image = types.Image(content=content)

response = client.text\_detection(image=image)

texts = response.text\_annotations

my\_list = list()

for text in texts:

result = text.description

my\_list.append(result)

data = my\_list[0]

data1 = data.replace('\n',' ')

data2 = data1.replace('(',' ')

data3 = data2.replace(')',' ')

data4 = data3.replace('/',' ')

data4 = data3.replace('그라니때','그라니따')

data4 = data3.replace('Sparkling','스파클링')

data5 = data4.split(' ')

df = pd.DataFrame(data5, columns=["총리스트"])

df1 = pd.DataFrame(columns=["0","1","2","3"])

df2 = pd.DataFrame(columns=["0","1","2","3"])

df1.loc[0, '3'] ="아메리카노"

df1.loc[1, '2'] ="아이스"

…….

df1.loc[116,'2']="청포도"

df1.loc[116,'3']="Sparkling"

menu=df1[['0','1','2','3']].astype(str).sum(axis=1)

menu = menu.str.replace('nan', '')

m2=menu.unique()

Allmenu = pd.DataFrame(m2, columns = ["AllMenu"])

for i in range(0,len(df.index)):

for i2 in range(0,len(df1.index)):

if df1.loc[i2,'0'] == df.loc[i,"총리스트"]:

df2.loc[i2,'0'] = df.loc[i,"총리스트"]

elif df1.loc[i2,'1'] == df.loc[i,"총리스트"]:

df2.loc[i2,'1'] = df.loc[i,"총리스트"]

elif df1.loc[i2,'2'] == df.loc[i,"총리스트"]:

df2.loc[i2,'2'] = df.loc[i,"총리스트"]

elif df1.loc[i2,'3'] == df.loc[i,"총리스트"]:

df2.loc[i2,'3'] = df.loc[i,"총리스트"]

df3=df2.sort\_index()

match=df3[['0','1','2','3']].astype(str).sum(axis=1)

match = match.str.replace('nan', '')

match = match.unique()

imgtxt = pd.DataFrame(match, columns = ["imgtxt"])

**# 기상청 API로 현재 날씨( 한 시간 전 ) 비 여부 불러오기**

year = datetime.now().year

month = datetime.now().month

if month < 10:

month = '0' + str(month)

day = datetime.now().day

if day < 10:

day ='0' + str(day)

hour = datetime.now().hour-1 # 1시간 전 날씨를 불러옴.

if hour < 10:

hour = '0' + str(hour)

minute = datetime.now().minute

if minute < 10:

minute = '0' + str(minute)

time\_number = str(hour)+str(minute)

**#api 호출 #60-127 서울시**

url = 'http://newsky2.kma.go.kr/service/SecndSrtpdFrcstInfoService2/ForecastGrib'

queryParams = '?' + urlencode({quote\_plus('ServiceKey'): 'vNTq1lksHC5gAxJtE2JHfHXFsESjQxUWA7TJyvVUvX3wHAmgc9UDnLIPz94tPZRfidRUEd9Mm8fCOl4rHoakUw==',

quote\_plus('base\_date') : str(year)+str(month)+str(day),

quote\_plus('base\_time') : str(time\_number),

quote\_plus('nx') : '60',

quote\_plus('ny') : '127', #60-127 서울시

quote\_plus('numOfRows') : '10',

quote\_plus('pageNo') : '1',

quote\_plus('\_type') : 'json' })

request1 = Request(url + queryParams)

request1.get\_method = lambda: 'GET'

response\_body = urlopen(request1).read()

**#json type으로 변형**

my\_json = response\_body.decode('utf8').replace("'", '"')

Weather = json.loads(my\_json)

Weather

**#카테고리 확인용 PYT = 강수 여부**

C = Weather["response"]["body"]["items"]["item"][0]["category"]

**#날씨 값 #pyt = 강수 형태 0=맑음, 1 비, 2 눈비, 3 눈**

R = Weather["response"]["body"]["items"]["item"][0]["obsrValue"]

if R == 3:

R = 0 # 눈-> 비가 안온 날 0

if R == 2:

R = 1 # 진눈깨비 -> 비온날 1

**#강수 여부의 값이 아닐 경우 Error가 뜨도록 설정.**

if C != 'PTY':

C = 'Error'

if C == 'Error':

R = 'Error'

con = cx\_Oracle.connect("dator/me@localhost:1521/XE")

cur=con.cursor()

if R == 0:

cur.execute("SELECT menu FROM data\_no\_rain ORDER BY NoRainResult DESC")

else:

cur.execute("SELECT menu FROM data\_rain ORDER BY RainResult DESC")

stt=()

sttdf = pd.DataFrame(columns=["순위메뉴"])

i6=0

for row in cur:

stt+=row

sttdf.loc[i6,"순위메뉴"]=stt[i6]

i6+=1

result\_finish = pd.DataFrame(columns=["최종결과","순위"])

for i7 in range(0,len(imgtxt.index)):

for i8 in range(0,len(sttdf.index)):

if sttdf.loc[i8,'순위메뉴'] == imgtxt.loc[i7,"imgtxt"]:

result\_finish.loc[i8,'최종결과'] = imgtxt.loc[i7,"imgtxt"]

result\_finish.loc[i8,'순위']=i8

result\_finish01=result\_finish.sort\_index()

final=result\_finish01.iloc[0,0]

flist = list()

for i9 in range(0,2):

final2 = result\_finish01.iloc[i9,0]

flist.append(final2)

if R == 0:

a = "맑은 날"

else:

a = "비 오는 날"

today = date.today().strftime('%Y-%m-%d')

info = cur.execute("SELECT \* FROM info5")

rows = cur.fetchall()

df1 = pd.DataFrame(rows, columns =["메뉴","맛평가","양평가","리뷰","빈도수"] )

df1["빈도수"]=df1["빈도수"].astype(int)

qq=str(final)

for z in range(0, len(df1)):

if df1.loc[z,"메뉴"] == qq:

aa = df1.loc[z]

dict1 = "메뉴: {} 맛평가: {}, 양평가: {}, 리뷰: {}, 빈도수:{}".format(aa[0], aa[1] ,aa[2], aa[3], aa[4])

context = {'recommend' : final, 'final2' : flist, 'weather': a, 'date':today,'info':dict1 }

return render(request, 'posts/index3.html', context)

**2)urls.py**

path('start/post/result/result1/', views.index02)

**3)index3.html**

