HW5

Jae Young Kim

ID : 2918465924

Section 1 - Goals of the project.

My goal of the project is to make a gourmet map of La hamburger restaurants. As there were no map which shows the general concentration of the famous restaurants and the detail information of the restaurants, I want to make a map which shows the details including open hour, current weather, temperature, score and so on.

In this process, I used sentiment analysis to the scraped reviews (the description in the blog I scraped) and comments (comments of the visitors). I wanted to compare the score in the blog and the sentiment scores of scraped reviews and comments. Therefore, I used scatter plot matrix to visualize and analyze the rating and scores.

Finally, I wanted to see whether the number of comments is an influential variable to predict the rating of the restaurant or not. Therefore, I used regression plot to check it and visualize it.

Section 2 - Description of what your code does. Try to include high-level flow diagrams in order to understand the code flow easily.

In 'project1_scrape_web_page.py', I scraped data from https://www.timeout.com/los-angeles/restaurants/the-best-burgers-in-los-angeles. From the article, I scraped the subtitles of the article. It looks like "Single Burger at Everson Royce Bar". And extracted famous menu and restaurant name. In the blog, there were introducing pages of each restaurants such as 'https://www.timeout.com/los-angeles/bars/everson-royce-bar'. I scraped address, review of the restaurant, website of the restaurant, rated score from the blog from all the introducing pages

In 'project2_yelp_api.py', I scraped business_id, the number of comments, rating in Yelp, open hour, comments from visitors from yelp-api.

In 'project3_nlp_google_api.py', I did sentiment analysis of the reviews and comments of each restaurants by using google natural language api. I got the sentiment scores and magnitude of the reviews and comments of the restaurants.

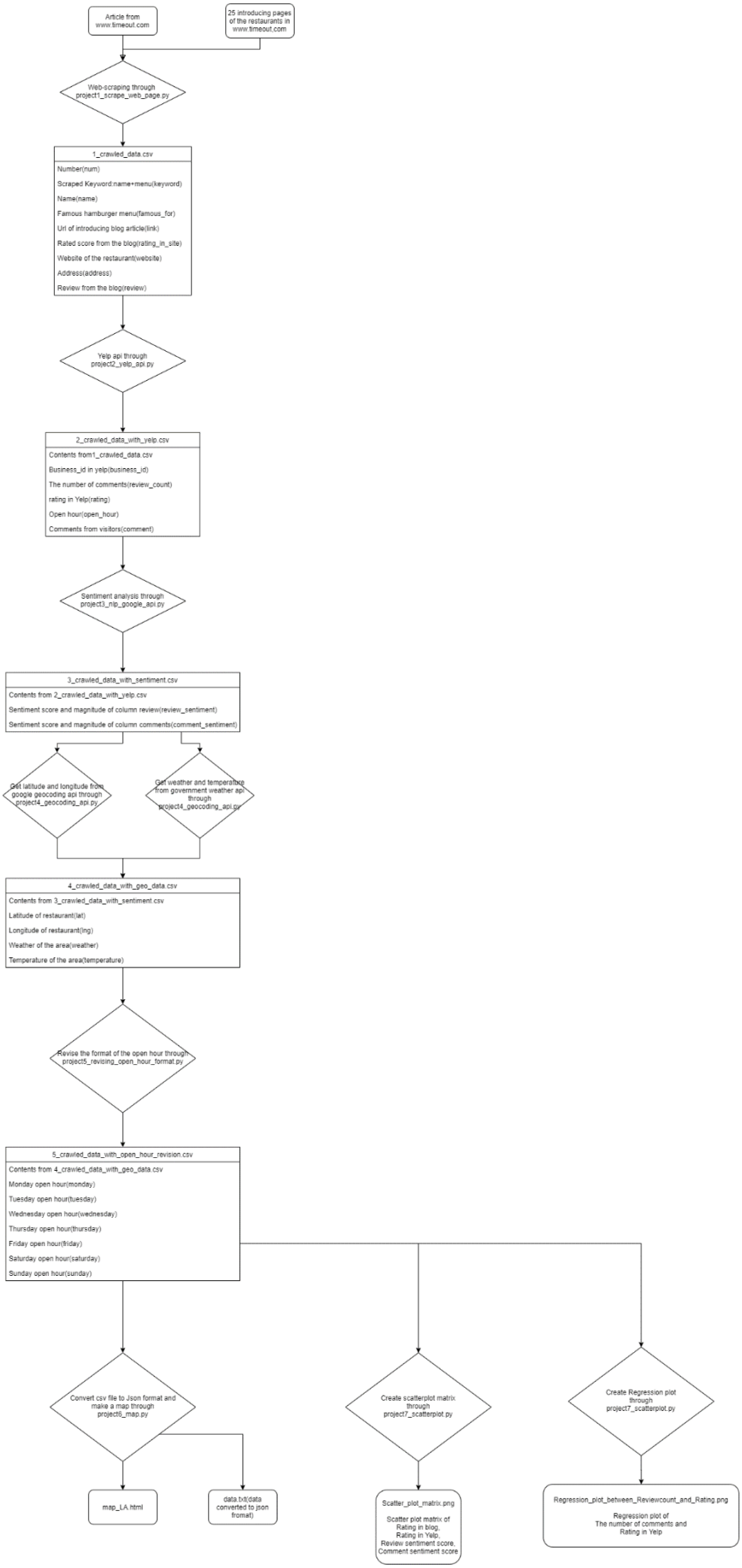
In 'project4_geocoding_api.py', I used google geocoding api to get the latitude and longitude of the restaurants by using the scraped address from 'project1_scrape_web_page.py'. Then I used government weather api to get the weather and temperature of the given latitude and longitude.

In 'project5_revising_open_hour_format.py', as I wanted to change the format of the open hour list stored in the csv file, I scraped open hour again by using the code I used in 'proejct2_yelp_api.py'. In this code, the open hour list, expressed as a string, was divided and

stored for each day of the week.
In ‘project6_map.py’, I converted data frame to json format to show the data on the map. The map was stored as an html file format.

In ‘project7_scatterplot.py’, I made scatter plot matrix of 4 scores of the restaurants : rated score I scraped from www.timeout.com, rated score I scraped from yelp, sentiment score of review from www.timeout.com, sentiment score of comments scraped from yelp.
In addition, I created a regression plot between the number of comments and rated score in yelp.

All the processes described so far are summarized in the diagram below. As it is quite complex, I included ‘Diagram of project.png’ in the zip file. You can zoom it.


Article from www.timeout.com
25 introducing pages of the restaurants in www.timeout.com
Web-scraping through project1_scrape_web_page.py
1_crawled_data.csv
Number(num)
Scraped Keyword:name+menu(keyword)
Name(name)
Famous hamburger menu(famous_for)
Url of introducing blog article(link)
Rated score from the blog(rating_in_site)
Website of the restaurant(website)
Address(address)
Review from the blog(review)
Yelp api through project2_yelp_api.py
2_crawled_data_with_yelp.csv
Contents from1_crawled_data.csv
Business_id in yelp(business_id)
The number of comments(review_count)
rating in Yelp(rating)
Open hour(open_hour)
Comments from visitors(comment)
Sentiment analysis through project3_nlp_google_api.py
3_crawled_data_with_sentiment.csv
Contents from 2_crawled_data_with_yelp.csv
Sentiment score and magnitude of column review(review_sentiment)
Sentiment score and magnitude of column comments(comment_sentiment)
Get latitude and longitude from google geocoding api through project4_geocoding_api.py
Get weather and temperature from government weather api through project4_geocoding_api.py
4_crawled_data_with_geo_data.csv
Contents from 3_crawled_data_with_sentiment.csv
Latitude of restaurant(lat)
Longitude of restaurant(lng)
Weather of the area(weather)
Temperature of the area(temperature)
Revise the format of the open hour through project5_revising_open_hour_format.py
5_crawled_data_with_open_hour_revision.csv
Contents from 4_crawled_data_with_geo_data.csv
Monday open hour(monday)
Tuesday open hour(tuesday)
Wednesday open hour(wednesday)
Thursday open hour(thursday)
Friday open hour(friday)
Saturday open hour(saturday)
Sunday open hour(sunday)
Convert csv file to Json format and make a map through project6_map.py
Create scatterplot matrix through project7_scatterplot.py
Create Regression plot through project7_scatterplot.py
map_LA.html
data.txt(data converted to json fromat)
Scatter_plot_matrix.png
Scatter plot matrix of
Rating in blog,
Rating in Yelp,
Review sentiment score,
Comment sentiment score
Regression_plot_between_Reviewcount_and_Rating.png
Regression plot of
The number of comments and
Rating in Yelp

Section 3 - Description of your code itself.

In 'project1_scrape_web_page.py', I scraped www.timeout.com/los-angeles/restaurants/the-best-burgers-in-los-angeles'. I started with scraping names and famous burgers from this webpage. In the scraped data, since there were some non-english characters, I removed or replaced with corresponding English character. Then as there were links to the introducing page in www.timeout.com such as 'www.timeout.com/los-angeles/bars/everson-royce-bar', I scraped review, address, web address of the restaurants. Then saved the scraped data as a csv file.

In 'project2_yelp_api.py', I got data from yelp api. At first, I sent query to yelp api by using the name of the restaurant. Then the api searched the restaurant name in yelp and sent me back the result as json file. Then I extracted business_id(restaurants' id in yelp), the number of comments of the visitors and rated score in yelp. In addition, I got open hour of the restaurants. Then since the format of the open hour I got was not good, I changed the format. The changed format looks like this.

['On Monday open : 1130 close 0200 open overnight : True', 'On Tuesday open : 1130 close 0200 open overnight : True', 'On Wednesday open : 1130 close 0200 open overnight : True', 'On Thursday open : 1130 close 0200 open overnight : True', 'On Friday open : 1130 close 0200 open overnight : True', 'On Saturday open : 1700 close 0200 open overnight : True', 'On Sunday open : 1400 close 0000 open overnight : False']

Then used another query to scrapes the comments of the visitors. The yelp api provided at most 3 comments. Then I saved the data into a csv file.

In 'project3_nlp_google_api.py', I used google natural language api to get the sentiment score of the review and comments of the visitors of the restaurants. So I defined sentiment_analysis function which use query to natural language api and get the sentiment score and magnitude of the given text. Sentiment score is from -1 to 1 and range of magnitude is 0 to infinity.

Then I got sentiment score of the review and visitors' comments. Then I saved the data into a csv file.

In 'project4_geocoding_api.py', I got data from google geocoding api and government weather api. From geocoding api, I got the latitude and longitude of the restaurants by using address of the restaurants. Then with the latitude and longitude, I used government weather api and got weather and temperature of the area. Then as the unit of the temperature is Fahrenheit, I attached F at the end of the temperature. Then stored the data into a csv file.

In 'project5_revising_open_hour_format.py', as the format of the open_hour_list was not adequate to display on the map, I scraped the open_hour_list again by using the code I used in 'project2_yelp_api.py'. Then I divided the open_hour_list and stored by days.(format : ['1130-

0200'])

Then in ‘project6_map.py’, I created a map by using the data I scraped. At first, I calculated average latitude and longitude of the restaurants to set the middle point of the map. Then to use folium package and show the data on the map by using Geojson function, I changed the data format from data frame to json. To do this process I defined a function named dataframe_to_json which converts dataframe I collected to a json file named data. Then I stored the json file in .txt format. I defined a function named save_json to do this. After that, I defined a function named “build_map”. In this function, I used folium package and created a map with 3 layers. Each of them is shown in section6 and the map is stored as an html file named ‘map_LA.html’

In ‘project7_scatterplot.py’, I used seaborn and matplotlib library to make scatterplot matrix and regression plot. At first, as there were many restaurants which are not rated in ‘www.timeout.com’, I changed the ‘no_rating’ value to 0. I defined a function named “imputation” to do this imputation. As all the values were stored as string, I tried to convert it to float and if it is not possible, I transferred it to 0.

As sentiment scores are stored as a format (sentiment score, sentiment magnitude), to select sentiment score, I defined a function named select_sentiment_score and selected sentiement score except sentiment magnitude.

Then I selected rated score in ‘www.timeout.com’, rated score in yelp, sentiment score of the reviews and the comments. Then plotted as a scatter plot matrix. I defined a function “dataframe_for_scatter_plot_matrix” to run this.

Then made a regression plot of the number of comments and rating in yelp by using seaborn.

Section 4(Optional) - Description of any additional packages that you may have used apart from the ones that are discussed in the class.

I used package named ‘yelpapi’ which allows me to get data from yelp api.

I used package named ‘google.cloud’, ‘google.cloud.language’ and ‘googlemaps’ which allows me to have connection with google cloud server and get the result of my query from google api especially, from natural language api and geocoding api.

I used package named ‘folium’ which allows me to create a map and display what I want to show on the map.

I used package named ‘matplotlib’ and ’seaborn’ which allows me to create plot and do visualization.

Section 5(Optional) - Additional procedures to be followed by the

grader in order to run the code.

To run 'project2_yelp_api.py', you need valid account of yelp and api_key. In 'project2_yelp_api.py', I included my own api_key. Therefore, if you just run the code, it is possible to run the code with my yelp account. However, if you want to run the code with your own account, you have to authenticate your own account. In 'https://www.yelp.com/developers/documentation/v3/authentication', you can create your own api_key by creating app. When you create app and get your own api key,

```
from yelpapi import YelpAPI
import requests
import pandas as pd

df=pd.read_csv("1_crawled_data.csv")
business_id_list=[]
review_count_list=[]
rating_list=[]
open_hour_list=[]
comment_list=[]



###Account_Info
api_key='NRW4IhT6TdVSpTXi5YbwSqMdgt_YIbPIJthLFM6iL1jc-Y5OHyHtv_IPVg4oDw_WWIDNN4h7JzzZwfcFQykneec_AzgwrV1dHaTOY6bJrouZIpoPbCLuPbBDk6CaXnYx'
url = "https://api.yelp.com/v3//businesses/{id}/reviews"
yelp_id='X1LOAXqPtWgGUUmFYSw-YQ'
yelp_api = YelpAPI(api_key)

###Search Restaurants
for i in range(len(df['name'])):

    search_results = yelp_api.search_query(term=df['name'][i], location='Los Angeles, CA')

```

You can add your api_key on the 'Account_info' section.

To run my code, especially to run 'project3_nlp_google_api.py' and 'project4_geocoding_api.py', you need valid account of google cloud platform. For 'project4_geocoding_api.py', when you have api_key, it is possible to run the code and I included my api_key. Therefore, you can run without your account.(If you want to use your own account, you have to create project on geocoding api) However, if you want to run 'project3_nlp_google_api.py', you have to authenticate your account and create json file including the information of your account. You have to follow the following steps to do this.

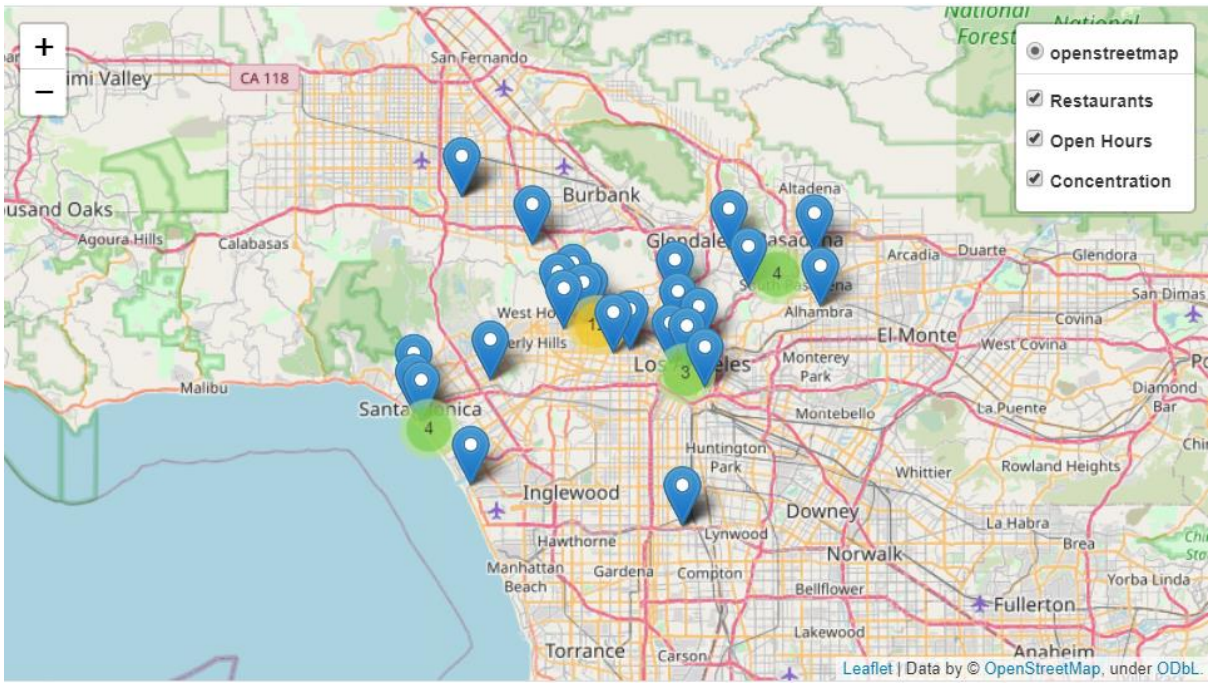
Google Cloud Console→Create a project →Search "Cloud Natural Language API"→Create Credential→Create a Service account→Create Key→Key type JSON→JSON file will be downloaded→register the path of the JSON file as an environment path of [GOOGLE_APPLICATION_CREDENTIALS](It can be different in MAC pc. I used Windows)
File named 'vibrant-period-272906-d44462db6203.json' is the json file for my project. I included mine in the zip folder.
After this, you have to install 'google.cloud.language', 'googlemaps' package. I included requirement.txt which contains all the information about the packages and versions. You can use it to install all the packages at once.

If you installed all the packages in requirement.txt, other codes would run.
When you install packages with requirement.txt sometimes there can be error 'permission denied'. If so, you can use 'pip install -r requirements.txt --user'. Then your environment will install all the packages.

Section 6 - Summary/Presentation of results

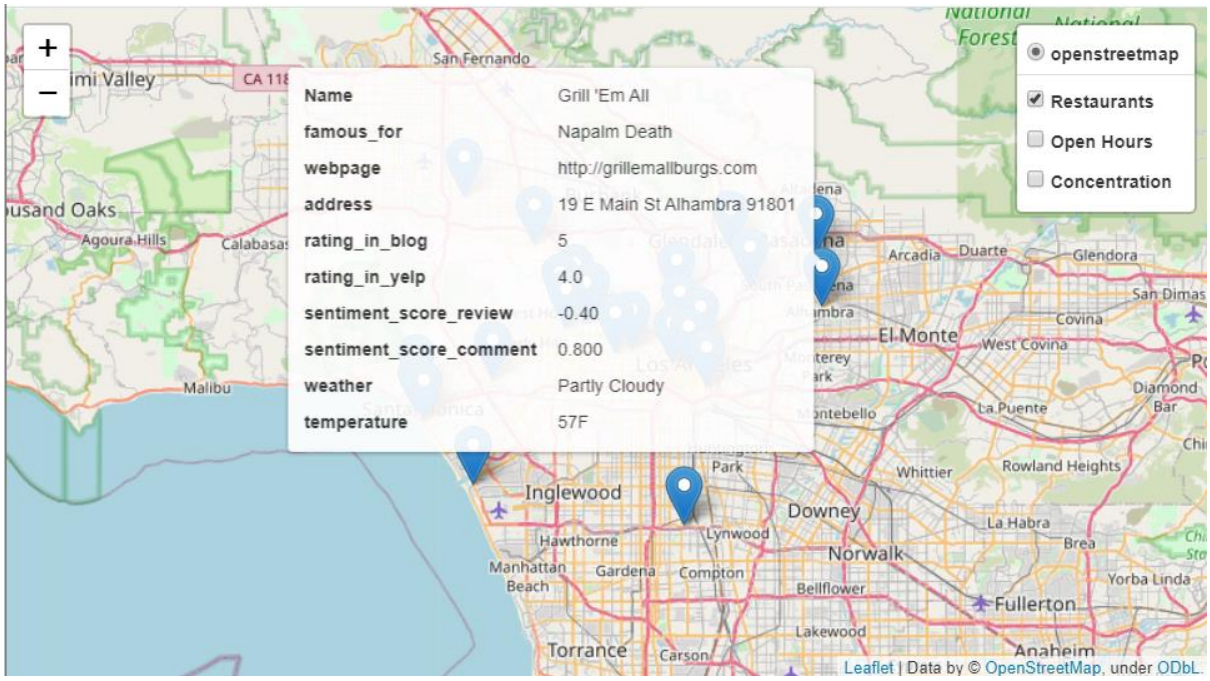
By showing the data on the map, I created a map file named 'map_LA.html'. It has 3 layers : 'Restaurants', 'Open hours', 'Concentration'.

If you open the html file, you will see the picture below.



This is a map showing all the 3 layers.

On the upper right side, there is an index. You can click the box to show/remove the layer from the map. On the 'Restaurants' layer, it shows the Marker of the restaurants and each of them shows the detailed information of the restaurant.



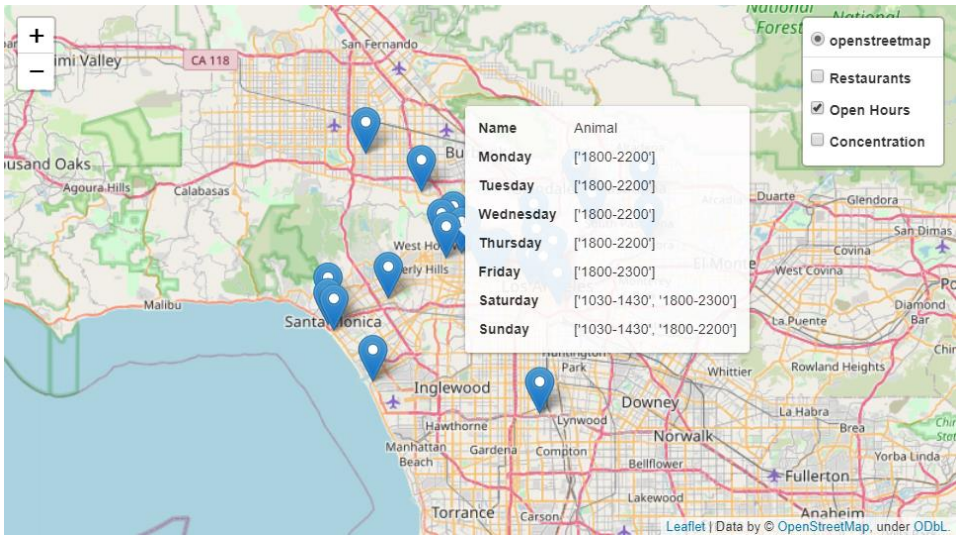
If you hover mouse pointer on the Marker, it shows like the above picture.

It shows the name, famous hamburger, webpage, address, rating_in_blog(rated score in www.timeout.com blog), rating_in_yelp(rated score in yelp), sentiment_score_review(sentiment score of the review in www.timeout.com blog), sentiment_score_comment(sentiment score of the 3 comments of the visitors of the restaurants scraped from yelp), weather, temperature of the restaurant.

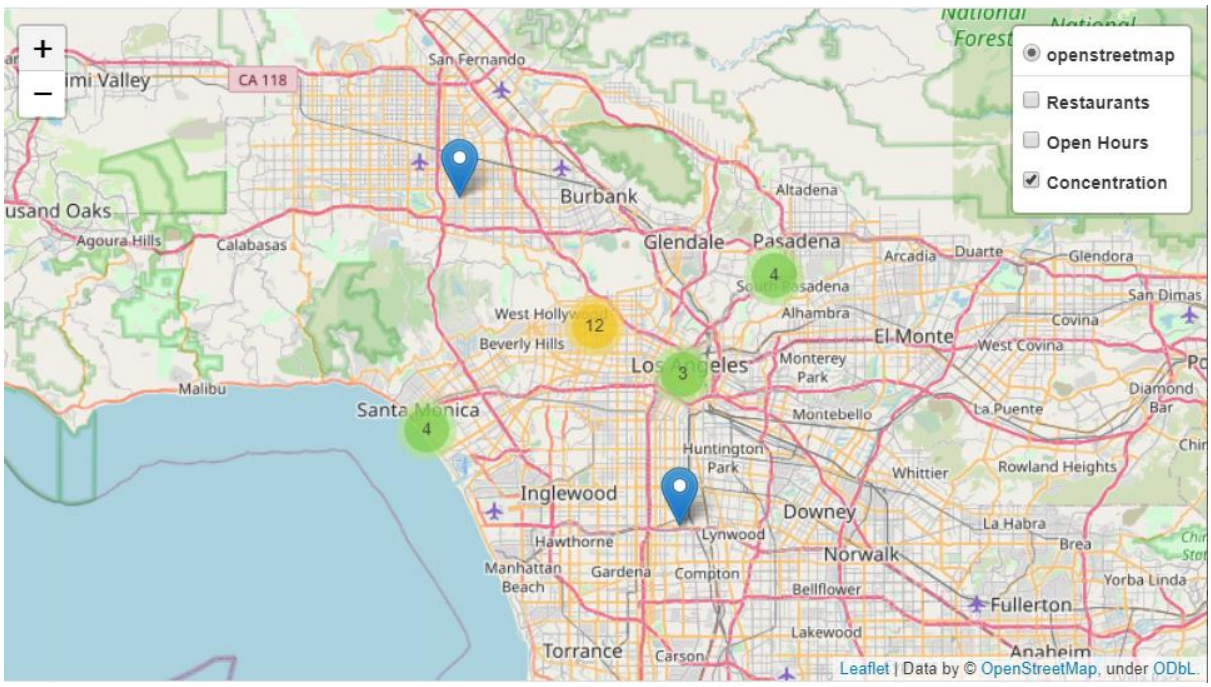
The user can get information and compare 4 different rated scores. Score of 'rating_in_blog' and 'rating_in_yelp' are from 1 to 5. Score of 'sentiment_score_review ' and 'sentiment_score_comment' are form -1 to 1.

Furthermore, the weather and temperature of the area around the restaurant is included. Therefore, the user can this information to decide whether the user will go to the restaurant immediately or not.

In the second layer, 'Open Hours', it shows daily open and close time of the restaurants.

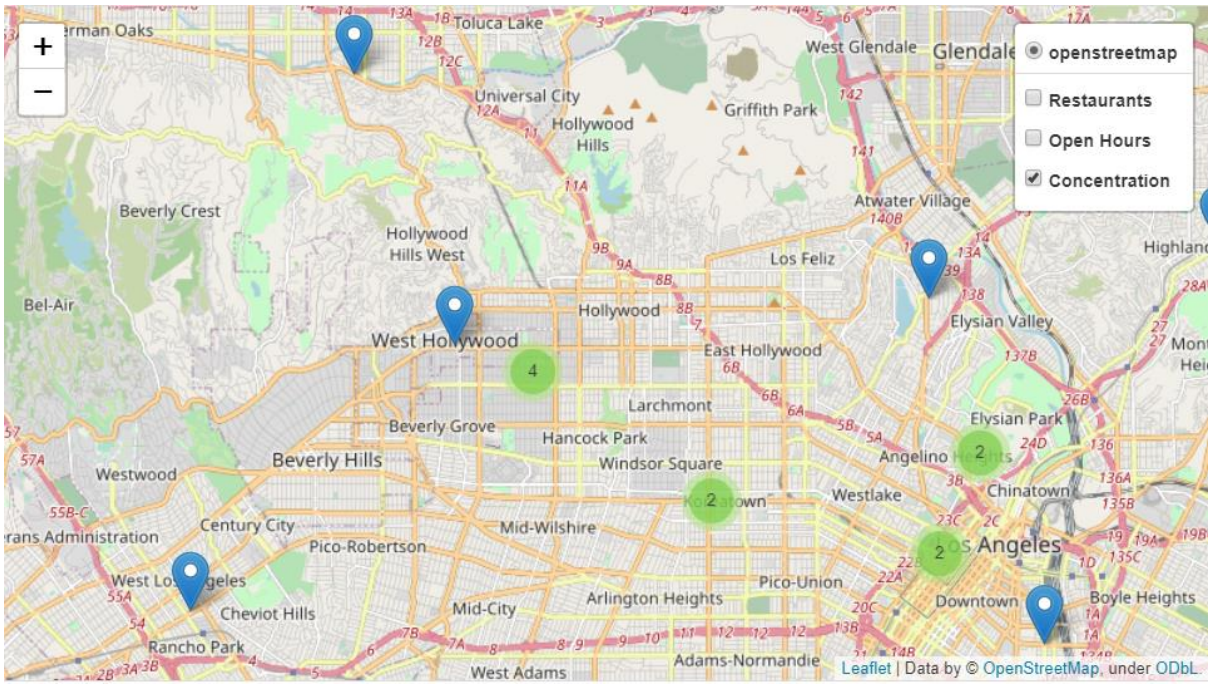


In the third layer ‘Concentration’, it shows the concentration of the famous burger restaurants in LA.

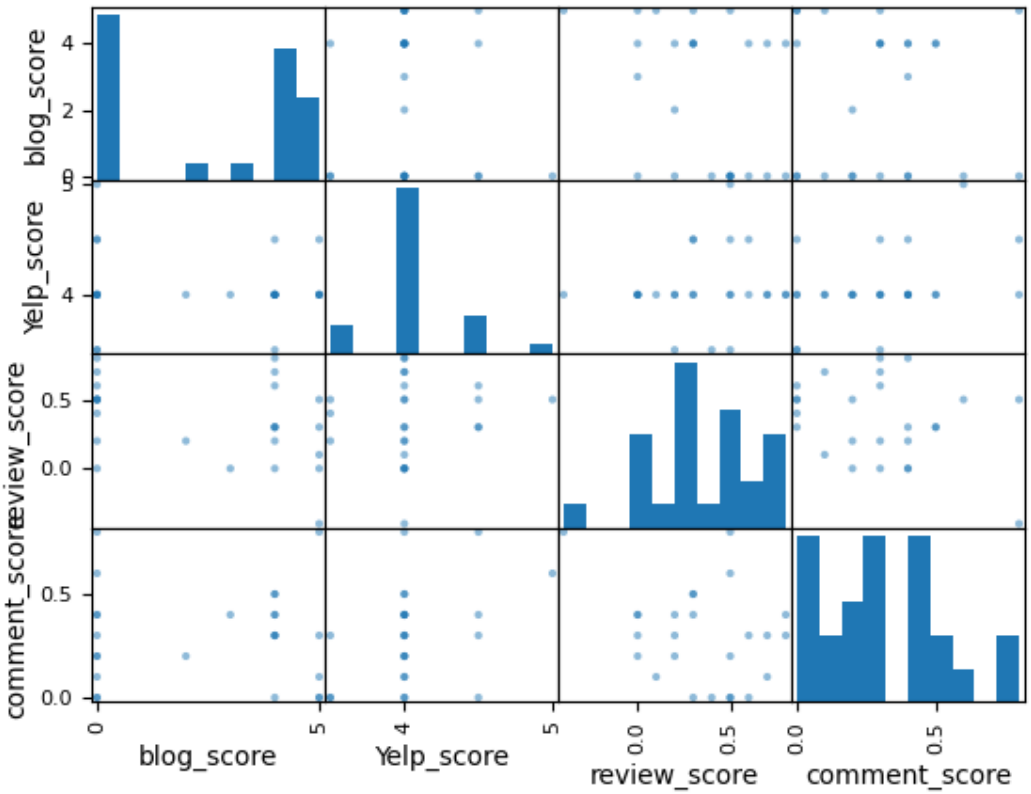


In the above picture, it is possible to notice that famous burger restaurants are concentrated on the West Hollywood area.

When you click the number, you can see more specific concentration of the restaurants like the picture below.



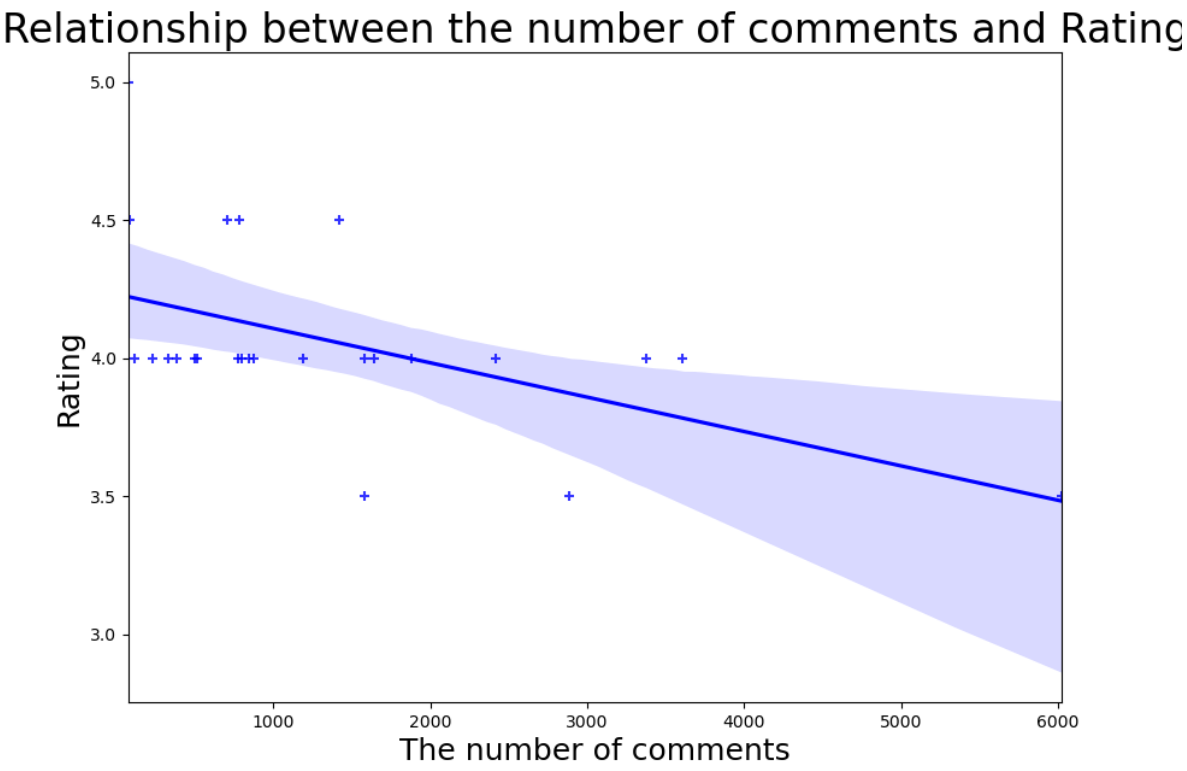
My second output is 'Scatter_plot_matrix.png'. It is a scatter plot matrix of 4 different scores of the restaurants : blog_score(Rated score from 'www,timeout.com'), yelp_score(Rated score from yelp api), review_score(sentiment score of the review from 'www.timeout.com'), comment_score(sentiment score of the comments of the visitors scraped from ylep api).



As some of the restaurants had 'no rating' in 'www.timeout.com' and I converted it to score 0, in first histogram, there are many 0 scores. As I scraped famous restaurants (which means they are good restaurants), the scores are concentrated and distributed in high scores. Especially, in yelp, all the scores were same or above 3.5.

Before having the result, I expected dots in the scatter plot are distributed on y=x line as they are all scores of the same restaurants. However, I could not find specific relation from this scatter plot matrix. I expect if there were more restaurants, I can watch better relation from the scatter plot matrix.

My last output is regression plot of the number of comments and rating (rated score from yelp api).



As I thought high number of comments means there are many visitors and it means the restaurant is good, I expected positive relation between rating and the number of comments. However, my result showed negative relation between them. In my opinion, current app users tend to leave a lot of complaints rather than praise on the app. Therefore, this negative relation occurred.

To sum up, I made a gourmet map of La hamburgers with the information from the blog www.timeout.com, Yelp api, government weather api. As a result, by making a map displaying the concentration of LA hamburgers, I could find a pattern that famous hamburger restaurants are concentrated on West Hollywood area. In addition, I made a scatterplot matrix between various scores of the restaurants including scraped score from yelp and timeout.com, sentiment score of the review and comments. However, as I selected only famous restaurants, the number of the data was too small to discover specific pattern from the scatterplot matrix. Finally, I made a regression plot between the yelp score and the number of comments. It showed negative relation which was opposite from my expectation.