Report: Recommendation System based on Yelp Reviews

OBJECTIVE:

Recommending restaurants to people based on the similarity of reviews and categories between restaurants.

INTRODUCTION:

Yelp, an American Public Company with its website, Yelp.com is a crowd-sourced local business review and social networking site. The site has pages devoted to individual locations, such as restaurants and schools, where Yelp users can submit their review of the product or service using a one-to-five-stars rating system. The Yelp dataset is made publicly available for personal, educational and academic purposes.

- *Data link:* https://www.yelp.com/dataset
- *Format of the Data:* json file
- *Structure of the Data:* The dataset consists of 6 json files, namely - business.json, user.json, review.json, checkin.json, tips.json and photos.json. Since our focal point under consideration here is restaurants, we'll be working with only Restaurant reviews.
- *Features used:* To build the system we are considering the following **features** from the dataset:

Variable Name	***Description***
business_id	Business ID
text	Review given by user
name	Restaurant name
categories	The category in which respective restaurant fall
stars	Rating given on the basis of five- star rating system by the user

APPROACH:

Our goal is to recommend restaurants based on similar reviews and almost identical categories. Using bag-of-words representation for reviews, we get TF-IDF matrix which is then used to get cosine-similarity matrix. Also, we represent categories using one-hot encoding and calculate the category similarity between two restaurants using cosine similarity. Finally, we calculate a score matrix which is nothing but the average of the cosine-similarity matrices of TF-IDF and categories. This score matrix is then used to retrieve the top 5 similar restaurants.

All in all, our model takes into account concepts like MapReduce, TF-IDF (Term frequency – Inverse Document Frequency), Cosine similarity and Bag of words to find the similarities between restaurant reviews and categories.

MapReduce is a processing technique and a program model for distributed computing based on Java. The MapReduce algorithm contains two important tasks, namely Map and Reduce. Map takes a set of data and converts it into another set of data, where individual elements are broken down into tuples (key/value pairs). On the other hand, the Reduce task takes the output from a map as an input and combines those data tuples into a smaller set of tuples. As the sequence of the name MapReduce implies, the Reduce task is always performed after the Map job.

TF-IDF stands for **Term Frequency and Inverse Document Frequency**. TF-IDF helps in evaluating importance of a word in a document. It is a statistical measure that evaluates how relevant a word is to a document in a collection of documents. This is done by multiplying two metrics: how many times a word appears in a document , and the inverse document frequency of the word across a set of documents.

Cosine similarity is a measure of similarity between two non-zero vectors. It measures the similarity between two vectors of an inner product space. The cosine of the angle between two vectors determines whether two vectors are pointing in roughly the same direction. Cosine value ranges from -1 to 1. So, if two vectors make an angle 0, then cosine value would be 1, which in turn would mean that the sentences are closely related to each other. If the two vectors are orthogonal, i.e. cos 90, then it would mean that the sentences are almost unrelated.

Bag-of-words is a representation of text that describes the occurrence of words within a document. It involves two things, a vocabulary of known words and a measure of the presence of known words. It is called a "*bag*" of words, because any information about the order or structure of words in the document is discarded. The model is only concerned with whether known words occur in the document, not where in the document.

BUILDING THE RECOMMENDATION SYSTEM:

Note: Due to the sheer size of the dataset, we are taking into consideration only a subset of data.

Steps followed to build the recommendation system:

1. *Extracting required variables from the dataset:* Since we have various features present in our dataset, to extract reviews pertaining to restaurants we will be considering two files – reviews.json and business.json. The reviews data consists of reviews from all the business and the business data consists of information on diverse businesses. We extract the business IDs of restaurants with the help of category feature as all the restaurants consist of one category named 'Restaurant'. With the help of the business ID, we apply an inner join to extract all the reviews concerning restaurants. This results in a dataset having reviews of all the restaurants along with their respective business IDs.

2. *Preprocessing and Cleaning of Data:* After extracting the data, we club all of the reviews of one business together. Next, we clean the dataset by looking for any null or duplicate values. If any null or duplicate value is present, we drop them. We also check the uniqueness of individual business ID and we drop the duplicates, if any are found. After dropping all the missing and duplicate values, we clean the textual data by removing extra spaces, stop-words and applying stemming.

Stemming *is definitely the simpler than Lemmatization. With Stemming, words are reduced to their word stems. A word stem need not be the same root as a dictionarybased morphological root, it just is an equal to or smaller form of the word. Lemmatization is computationally expensive and much more intensive than Stemming. Hence, we thought it would be appropriate to use Stemming instead of Lemmatization given our limited resources.*

3. *Calculating Term Frequency - Inverse Document Frequency:*

$$Term\ Frequency: \frac{Number\ of\ times\ term\ t\ appears\ in\ the\ document}{Number\ of\ terms\ in\ the\ document}$$

$$Inverse\ Document\ Frequency: \log\left(\frac{Number\ of\ documents}{Number\ of\ documents\ with\ term\ t}\right)$$

We calculate TF-IDF value of each word occurring in the reviews and create a sparse matrix. *A* ***sparse matrix*** *or sparse array is a matrix with very few non zero elements.*

*TF – IDF = Term Frequency * Inverse Document Frequency*

4. *One hot encoding for Categories:*

We are applying one hot encoding for 'category' feature to show presence and absence of a particular category corresponding to a business ID. Here, we are creating a sparse matrix again, to represent the one hot encoded value for categories.

5. *Calculating Cosine Similarity:*

$$\text{similarity} = \cos(\theta) = \frac{\mathbf{A} \cdot \mathbf{B}}{\|\mathbf{A}\|\|\mathbf{B}\|}$$

To compute the similarity score between restaurants, we are applying cosine similarity on the TF-IDF values of reviews as well as on the one hot encoded values of categories.

6. *Calculating Average score:*

The final similarity score between every pair of restaurants is an average of cosine similarity score of reviews and categories.

7. *Recommending top 5 Restaurants on the basis of score:*

On the basis of the above-mentioned average score, we recommend top 5 restaurants.

RESULTS:

- *User input:*

Search restaurants...

name: " italian "

index_id	name	categories
0	prego cucina italiana	Wine Bars, Bars, Nightlife, Restaurants, Italian
1	polo's italian cuisine	Restaurants, Bars, Wine Bars, Italian, Nightlife
2	olive garden italian restaurant	Salad, Bars, Restaurants, Nightlife, Wine Bars, Italian, Soup
3	scaddabush italian kitchen & bar	Wine Bars, Nightlife, Restaurants, Bars, Beer, Wine & Spirits, Italian, Food, Specialty
4	d'angelo's italian ristorante	Restaurants, Caterers, Event Planning & Services, Italian, Venues & Event Spaces
5	stone & vine urban italian	Pizza, Restaurants, Italian, Wine Bars, Bars, Nightlife
6	olive garden italian restaurant	Bars, Wine Bars, Italian, Salad, Nightlife, Restaurants, Soup
7	olive garden italian restaurant	Bars, Salad, Wine Bars, Nightlife, Italian, Restaurants, Soup
8	annie's gourmet italian	Bars, Nightlife, Venues & Event Spaces, Local Services, Restaurants, Couriers & Delivery
9	rinaldi's italian deli - scottsdale	Restaurants, Italian, Delis
10	buca di beppo italian restaurant	Food, Caterers, Italian, Event Planning & Services, Pizza, Restaurants
11	times square neighborhood italian restaurant	Italian, Restaurants
12	crust simply italian	Food, Pizza, Caterers, Italian, Restaurants, Beer, Wine & Spirits, Event Planning & Serv
13	la tavola italiana	Italian, Restaurants, Pizza
14	stancato's italian restaurant	Restaurants, Pizza, Caterers, Event Planning & Services, Italian
15	olive garden italian restaurant	Italian, Restaurants, Nightlife, Wine Bars, Salad, Bars, Soup
16	miele's italian and banquet hall	Restaurants, Event Planning & Services, Italian, Food, Desserts, Venues & Event Spaces,
17	fanucce's italian imports and pizzeria	Pizza, Restaurants, Italian
18	sauce italian bistro & bar	Restaurants, Food, Food Delivery Services, Nightlife, Bars, Italian, Sports Bars, Pizza

The user mentions a key word based on which we display the top 20 search results. The user can then use the index value to select the final restaurant based on which the recommendation system will suggest the top 5 restaurants.

- *Select the index:*

Select the index_id...

id: 6

This displays the selected restaurant:

name	categories	ratings
olive garden italian restaurant	Bars, Wine Bars, Italian, Salad, Nightlife, Restaurants, Soup	2.5

- *Output:*

The recommended restaurants are:

name	categories	ratings
marco's pizza	Restaurants, Pizza	3.5
ferris steak house	Steakhouses, Restaurants	3.5
chinato	Italian, Restaurants	4.0
swigs pub & grill	American (Traditional), Chicken Wings, Restaurants, Nightlife, Pubs, Sports Bars, Bars	2.5
bistro honda	Japanese, Restaurants	4.0