**Extracting Sub-Topics**

**From Yelp Reviews**

Course: **CSE 5334**

Professor

**Chengkai Li**

Associate Professor

Department of Computer Science and Engineering

The University of Texas at Arlington

Submitted

By

**Pavan Chowdari Ghanta**

Pavan.ghanta@mavs.uta.edu

**Objective and Overview:**

Yelp dataset contains ratings and reviews about various businesses and services. Yelp data (reviews and ratings) helps the users to choose from the list of available the best service or business. The reviews contains important details about the businesses. The users searching for a business of specific type have to go through all the reviews to evaluate the quality of the business. Sometimes, the search for services also depends on the time and situation. For ex., the users who are looking for a specific type of restaurant (calm environment, good ambience, wifi available) may have to read all the reviews to find the required information. Users may search for restaurants that are open till late night that can home deliver food. Some users may further need to know the amount of time they need to deliver the food. This kind of information is not directly available but can be extracted from the reviews. The businesses or services needs to be sub-categorized and are presented to the users so that they can find the appropriate information quickly. The sub-categories includes businesses with less check out time, parking availabilities, restaurants with quality service, coffee shops with peaceful environment and wifi availability, restaurants suitable for game day e.t.c., The businesses can be sub-categorized using already available category information and performing texual analysis of the reviews. This sub-categorization is useful in making the data more transparent to the users.

**Data Mining/analysis Tasks:**

The reviews text belonging to a business is collected and stored as single long text. All the stop words from the text are removed. The tfidf values of each word in all the reviews are calculated and normalized. For the sub-topic summary (list of words),tdidf values are calculated and normalized. Cosine similarity between each subtopic and all the reviews are calculated.

**Design of Methods:**

The following methods are designed and implemented:

* **Read\_businesses(filepath)**

Takes path of business file and reads the business details. While reading the businesses data,considering only the businesses of type restaurants.

* **Read\_Reviews(reviewsfile)**

Takes path of reviews file and reads the business reviews. The reviews belonging to same business are combined and stored as single text. The review text is tokenized and each business review tokens are stored.

* **TextPreprocess(reviewtext,wordsdict)**

Preprocesses the review text. The frequency of each word in the text is calculated and stored.

* **Calctfidf(reviewwordsdict)**

Calculates tfidf values of words in each review

* **Read\_Results():**

The best restaurants for each sub-topic are pre-calculated and stored in a document. This method reads the existing results for the subtopics.

* **Display():**

Displays various options which guides the programs execution.

* **Main()**

When full calculation of subtopics results is selected, this method is called.

For each subtopic, the cosine similarity between the subtopic and each business review is calculated. The business reviews with high similarity are displayed based on the selected subtopic.

**Implementation:**

Few subtopics are considered and stored in a dictionary. For each subtopic, summary is provided using list of words. Cosine similarity is calculated between subtopic (list of words) and each business review. Reviews with high cosine similarity are displayed as result.

* **Data Structures:**
  + Typestoconsider: List to store the types of businesses, whose reviews are considered for analysis. Only businesses of type present in list are considered for evaluation.
  + SUBTOPICSUMMARY: Dictionary to maintain the summary of the sub-topics. Sub-topics and their summary can be added to the dictionary. In the next full run of the program, new sub-topics are considered for analysis and best restaurants for the newly added sub-topics are calculated.
  + Datatoremovefromreview: List to maintain the unnecessary data that needs to be removed from the reviews.
  + Finalresults: Dictionary to store the subtopics and their respective best restaurants
  + Businessesdata: Dictionary to store the business data
* **Methods:**
  + **Display()**

Displays the pre-chosen sub-topics names. The best restaurants for the chosen sub-topics are already extracted from the reviews and stored in a file. User can select a sub-topic name to display the best restaurants in that category. There is also an option available for re-processing the reviews.

* + **Read\_Results():**

Reads the pre-process results from json file and stores in a dictionary.

* + **Read\_businesses(filepath)**

Each line is read from businesses file using json.loads and each business data is stored in a dictionary with business id as key and business data as value.

* + **Read\_Reviews(reviewsfile)**

Each line is read from businesses file using json.loads. Reviews belonging to same business are combined and stored in the dictionary with businessid as key. From each review, duplicate unwanted data (review\_id, user\_id, type, date) are removed.

* **TextPreprocess(reviewtext,wordsdict)**

Each review text is tokenized and stop words are removed using the library NLTK. The frequency of words in the text is calculated. Also the count of reviews in which the word occurs is also calculated.

* + **Calctfidf(reviewwordsdict)**

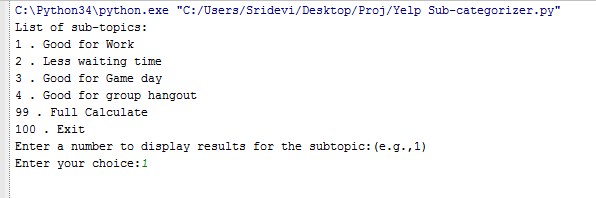
For each review words, tfidf values are calculated and stored in a dictionary with each word as key and tfidf as value.

* + **Main()**

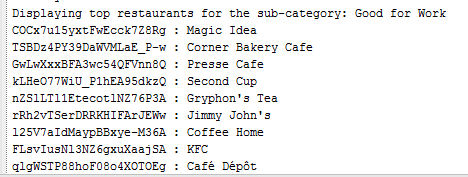
For each business review, cosine similarity with each subtopic is calculated. The values are stored in a dictionary with topic name as key and cosine similarity values of all businesses as values. The values are sorted and stored for each topic. For each subtopic, top 100 businesses having high cosine similarity values are displayed as results. This threshold value is configurable.

**Output:**

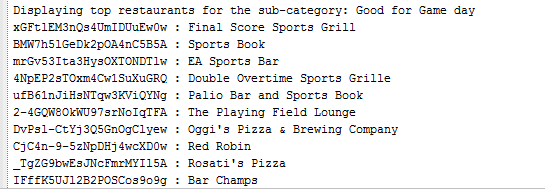
Screen shot 1:



Screen shot 2:



Screen shot 3:



**Steps to run the project:**

1) Copy the files Yelp Sub-categorizer.py and results.json to the directory having the yelp data

2) Run the python program

3) Select a subtopic to display the top 100 restaurants in that category.

4) To recalculate the values select the option 99.

**References:**

**http://www.ics.uci.edu/~vpsaini/**