

1. Introduction

The project uses the yelp dataset to perform mainly two tasks. These tasks were aimed at making us understand more about knowledge related to Information Retrieval, text and data mining. For this project, we used the Yelp data (text data + numeric data) to investigate two tasks. The dataset was very huge and we performed our tasks on the subset of the data. The original data included 42,153 businesses and 1,125,458 reviews out of which 14,257 business were the ones which had “Restaurants” as one of the categories. The Evaluation, Results and all the analysis is based on this data.

2. Task 1

2.1 Problem Statement

Category information is critical for yelp and it can be important for business recommendation. However, not all the businesses have high quality category metadata. For example, if some business is just labeled as “Restaurant”, the label is not very helpful and informative for the user. In this task, we had to predict the categories of businesses by using data mining, text mining or information retrieval algorithms.

2.2 Approach

Tf-idf stands for term frequency- inverse document frequency. This is used to determine the importance of a word to a document in a collection or corpus. The importance increases based on the number of times a word appears in the document but is offset by the frequency of word in the corpus or collection. This is often used by search engines in ranking a document relevant to a given user query.

Tf: Term Frequency, which measures how frequently a term occurs in the document.

$TF(t) = (\text{Number of times term } t \text{ appears in a document}) / (\text{Total number of terms in the document})$.

Idf: Inverse Document Frequency, which measures how important a term is. This is used to weigh down the frequent ones and scale up the rare ones.

$IDF(t) = \log_e(\text{Total number of documents} / \text{Number of documents with term } t \text{ in it})$.

Tf-Idf values are calculated for all the words in the reviews based on the word count obtained in the above collections.

We saved the tf-idf for a noun. The reason behind doing this is that tf-idf helps us to identify those words which are unique across the documents. For example, a word of an indian dish such as "Chicken Tikka" will not occur in any of the reviews which describe a restaurant which falls under different category such as Chinese or American. So this word is very important and tf-idf will help us identify such words and we can predict the categories easily.

Limitations of Tf-idf:

It computes document similarity based on the word count which may be slow for large vocabularies.

It does not makes use of semantic similarities between words.

We reduced this task to only those businesses, which had “Restaurants” as one of the categories and predicted the specific category such as “Indian”, “Chinese”, “Thai”, and “American” etc. We got 241 such categories from the dataset, which we considered.

The data from the yelp_academic_dataset_business JSON file is parsed to identify 'businessId', 'name' and 'categories' for businesses. This information is then stored in 'yelp_academic_d ataset_business' collection in mongodb.

Each row in collection corresponds to businessId, name and categories for a particular business.

The data for the business is obtained from 'yelp_academic_dataset_business.json' and 'yelp_academic_dataset_tip.json' files. These files are parsed and reviews for each business are identified

which has restaurants as one of the categories. This is done using the collection 'yelp_academic_dataset_business'. This review is then stored in 15 different text file based on the hash value of the business id.

Hash value is determined as follows:

Add the ASCII values of last two characters in the business id.

Sum mod 15.

This is then used to determine the name of the file.

Eg:- Let a represent sum of the ASCII values for last two characters.

Name of the file=a%15

The hash value determines the name of the file. The review is then given to the noun extraction function.

Libraries Used:

Apache OpenNLP

The Apache OpenNLP library is a machine learning based toolkit for the processing of natural language text. It supports the most common NLP tasks, such as tokenization, sentence segmentation, part-of-speech tagging, named entity extraction, chunking, parsing, and reference resolution. These tasks are usually required to build more advanced text processing services. OpenNLP also includes maximum entropy and perceptron based machine learning. We used this library to extract nouns from the reviews.

Standard Analyzer

Standard Analyzer perform standard filter, lower-case filter and stop filter using a list of English stop words. We use Standard Analyzer to remove the stop words, which do not contain information about the business categories.

English Analyzer

English analyzer performs EnglishPossessive filter, KeywordMarker filter and PorterStem filter. It converts word to its root form. This technique is a good practice whenever we are calculating the TF and IDF of a word since the count for a word should not be different for the same root. For example, running, runnable and run should not be treated as three different words but the root which is “run” should be stored.

The output of the noun extraction is then given to the analyzer. The analyzer used is Standard Analyzer and English Analyzer.

Standard Analyzer performs StandardTokenizer, StandardFilter, LowercaseFilter, and StopFilter. English Analyzer performs EnglishPossessive filter, KeywordMarker filter and PorterStemFilter.

The output text from the above process and the corresponding business id is then stored in the text file based on the hash value computed. This process is done for both the review and tip files.

For each text file generated data is stored in a hashmap with categories as key and value as hashmap with word as key and count as value. A hash map within a hash map—HashMap <String,HashMap <String,Integer>).This hashmap is traversed wherein a collection is created based on category and hashmap of value is stored for that category.This is done till all the 15 files are read.

This results in the creation of collections wherein each collection represents a category with word and wordcount as fields.

The reviews from the 'yelp_academic_dataset_review.json' file are extracted for a particular business. These reviews are then stored in a text file after extracting nouns and performing tokenization and stemming. This data processing step is same as the one explained above.

The result is then stored in a HashMap and sorted based on the tf-idf score. The top tf-idf scores are then used to determine categories of a business.

The reason for choosing only nouns from the reviews is that since we are doing aspect based sentiment analysis and we want to find information related to categories for a particular business the nouns in the sentences will speak about these categories. Our assumption is that the nouns convey more information about the food/dishes of the business.

Screenshot of the collections created in MongoDB:

```
> db.yelp_academic_dataset_business.findOne()
{
  "_id" : ObjectId("548aaaf3511e1f7cbbe550ad"),
  "business_id" : "vcNAWiLM4dR7D2nwwJ7nCA",
  "name" : "Eric Goldberg, MD",
  "categories" : [
    "Doctors",
    "Health & Medical"
  ]
}
```

Figure: yelp_academic_dataset_business collection

```

> db.yelp_academic_dataset_business_attributes.findOne()
{
  "_id" : ObjectId("548aabb8511ece9cded7c587"),
  "business_id" : "fXQmGQcLtMaRJyxsRfIe-w",
  "attributes" : {
    "Take-out" : "true",
    "Wi-Fi" : "free",
    "good_for" : {
      "dessert" : "false",
      "latenight" : "false",
      "lunch" : "false",
      "dinner" : "false",
      "breakfast" : "false",
      "brunch" : "false"
    },
    "Caters" : "true",
    "Noise Level" : "average",
    "Takes Reservations" : "false",
    "Delivery" : "false",
    "ambience" : {
      "romantic" : "false",
      "intimate" : "false",
      "classy" : "false",
      "hipster" : "false",
      "divey" : "false",
      "touristy" : "false",
      "trendy" : "false",
      "upscale" : "false",
      "casual" : "false"
    },
    "parking" : {
      "garage" : "false",
      "street" : "false",
      "validated" : "false",
      "lot" : "false",
      "valet" : "false"
    },
    "Has TV" : "false",
    "Outdoor Seating" : "true",
    "Attire" : "casual",
    "Alcohol" : "none",
    "Waiter Service" : "false",
    "Accepts Credit Cards" : "true",
    "Good for Kids" : "true",
    "Good For Groups" : "true",
    "Price Range" : "1"
  }
}

```

Figure: yelp_academic_dataset_business_attributes collection

```

null
> db.yelp_restaurants_categories.findOne()
{
  "_id" : ObjectId("548aaa1d511e0f523fa4c067"),
  "category" : [
    "Fondue",
    "Fish & Chips",
    "Karaoke",
    "Vietnamese",
    "French",
    "Cantonese",
    "Wineries",
    "Active Life",
    "Vegetarian",
    "Public Services & Government",
    "Shanghainese",
    "Arabian",
    "Ethiopian",
    "Delis",
    "Adult Entertainment",
    "Sandwiches",
    "Party & Event Planning",
    "Hot Dogs",
    "Irish",
    "Filipino",
    "Golf",
    "Laotian",
    "Cafeteria",
    "Gastropubs",
    "Desserts",
    "Brasseries",
    "Australian",
    "Soup",
    "Cocktail Bars",
    "Cheesesteaks",
    "Soul Food",
    "Asian Fusion",
    "Lounges",
    "Hospitals",
    "Belgian",
    "Burgers"
  ]
}

```

Figure: Unique categories in the dataset for businesses with “Restaurants” as one of the categories

```
>
> show collections
Active Life
Adult Entertainment
Afghan
African
Airports
American (New)
American (Traditional)
Amusement Parks
Apartments
Appliances
Arabian
Arcades
Argentine
Arts & Crafts
Arts & Entertainment
Asian Fusion
Australian
Auto Repair
Automotive
Bagels
Bakeries
Bangladeshi
Barbeque
Bars
Basque
Beauty & Spas
Bed & Breakfast
Beer Bar
Beer, Wine & Spirits
Belgian
Bistros
Bowling
Brasseries
Brazilian
Breakfast & Brunch
Breweries
British
Bubble Tea
Buffets
```

Figure: Collections list in mongoDB

```

> db.Spanish.findOne()
{
  "_id" : ObjectId("548bc539a053b98e459fa4db"),
  "file1" : {
    "restaur" : 2,
    "egg" : 2,
    "year" : 1,
    "beef" : 2,
    "tasti" : 1,
    "hubbi" : 1,
    "delicaci" : 1,
    "concern" : 1,
    "specialti" : 1,
    "corner" : 1,
    "aloha" : 1,
    "nichoacan" : 2,
    "appet" : 1,
    "worker" : 2,
    "antojito" : 1,
    "bistro" : 1,
    "favorit" : 1,
    "order" : 3,
    "serv" : 1,
    "birria" : 7,
    "goat" : 8,
    "version" : 1,
    "cours" : 1,
    "salsa" : 6,
    "seat" : 1,
    "jerki" : 1,
    "re" : 1,
    "carn" : 1,
    "daylight" : 1,
    "spot" : 2,
    "everyth" : 1,
    "style" : 1,
    "anyon" : 1,
    "i'd" : 1,
    "broth" : 1,
    "recommend" : 1,
    "pork" : 1,
    "i'm" : 1,
    "nichoacano" : 1,
    "locat" : 4,
  }
}

```

Figure: For one of the categories named “Spanish” out of 241 categories

Evaluation:

All the reviews for 41 different business id are extracted whose categories were already known. These reviews were then stored in text file after performing noun extraction, and performing tokenization and stemming using Standard Analyzer and English Analyzer. This data in the text file is then used to identify categories for business based on the Tf-idf score of word in the collection. The categories and sorted Tf-idf score for business is then stored in the text file for a particular business.

The Output.txt file in the Evaluation folder under Task 1 has the business id, categories and top categories predicted by the algorithm based on the Tf-idf score.

Precision is then calculated based on the categories of the business and top categories returned by the algorithm.

Precision was calculated based on the categories and predicted categories for top k tf-idf values.

For eg.

Let's say Business X has categories Pakistani, Indian, Buffets, Restaurants and the predicted categories based on the Tf-idf values are:

1-Restaurants

7- Indian

13- Buffets

22- Pakistani

Precision for top 20 Category=3/4 (The rank for Pakistani category in the result is 22 and not in top 20)

Precision for top 15 Category=3/4 (The rank for Pakistani category in the result is 22 and not in top 15)

Precision for top 10 Category=2/4 (The rank for Pakistani category in the result is 22, Buffet is 13 and both are not in top 10)

The below table describes the precision score averaged for 41 businesses.

	Top 20 Category	Top 15 Category	Top 10 Category
Precision	0.886829	0.829512	0.709756

The Precision score decreases as the number of top categories considered decreases.

The results generated can be viewed in the folder task1/evaluation/output.txt and task1/evaluation/evaluation_task1.xls

3. Task 2

3.1 Problem Statement

In this task, we are trying to do aspect based sentiment analysis. We classify the reviews into categories such as Food, Service and Ambience.

Each review submitted by the user does not convey the context which led reviewer to that experience. For Example: - Consider a review which is as follows:-

“Place X has the best food in town and the service is very fast. There is also an option for buffet during lunch hours.” (Rating – 4.5 stars).

Now from the review we cannot just tell from the rating why the user has given Place X a rating of 4.5 stars. Using such kind of reviews, we can extract more info about the basic features in a restaurant – food, service, price, discounts, ambience etc. By doing this, the user can understand why the reviewer has given whatever ratings he has given for a particular business. This information can indeed help other users to judge a particular business when the user does not have much time to spend on reading the reviews.

This categorization can also be used to rank the business. So our question becomes a multi- label classification problem since we are trying to classify a review into categories, which are pre-decided by extracting features from analyzing say, half the reviews in the dataset.

3.2 Approach

Stanford Core NLP

Stanford Core NLP provides a set of natural language analysis tools which can take raw text input and give the base forms of words, their parts of speech, whether they are names of companies, people, etc., normalize dates, times, and numeric quantities, and mark up the structure of sentences in terms of phrases and word dependencies, indicate which noun phrases refer to the same entities, indicate sentiment, etc. Stanford Core NLP is an integrated framework. Its goal is to make it very easy to apply a bunch of linguistic analysis tools to a piece of text. Starting from plain text, you can run all the tools on it with just two lines of code. It is designed to be highly flexible and extensible. With a single option you can change

which tools should be enabled and which should be disabled. Its analyses provide the foundational building blocks for higher-level and domain-specific text understanding applications.

We basically use this library to find the sentiment for a sentence and its score. It is also used to find the sentiment of the whole review. Stanford CoreNLP gives the sentiment as one of the following types:-

- Very Negative
- Negative
- Neutral
- Positive
- Very Positive

It also gives a score between 0-4 where 0 – Very Negative, 1 – Negative, 2- Neutral, 3-Positive and 4- Very Positive.

There is a limitation which we found while using this library.

Consider the following review:-

“Good dishes are coconut soup, vegetable korma, Malai Kofta, Baingan Bhartha, Chicken Tikka Masala, mango ice cream. The staff is wonderful and always keeps plates of naan a plenty. Maharaja is by far one of Madison's best restaurants”

For the above review, Stanford Core NLP gives the sentiment score 2 but if we replace the names of the dishes to “pizza”, it returns the sentiment score of 4. This means that its dictionary does not contains information outside the Oxford dictionary and other words such as above which are dishes name of Indian food is not in its dictionary and so it just assigns a random score. This is why we think, the actual reason might differ.

Task 2: Part 1

Scraping

We wrote a script in Python to extract all the food items, by category (eg: Chinese, Thai, American, and Indian). We populated the mongo collection for food category using the data scraped. We built manual keywords collection for Service and Ambience. The words inserted in the collections were analyzed after reading the reviews from “yelp_academic_dataset_reviews.json” and “yelp_academic_dataset_tip.json” in our test data.

Tokenization and Extraction

For extraction and tokenization, we basically use Apache OpenNLP to extract the nouns from the reviews. We then use Standard Analyzer to remove the punctuation marks, remove stop words and perform stemming on the data. After that, we use the English Analyzer to convert each noun into its root form and then run our algorithm which predicts the categories for businesses which have “Restaurants” as one of the category.

Approach

For generating ratings based upon the features we have proposed an approach which is discussed in the below sections. For evaluation purposes we had restricted our scope to 11 restaurants as we will be performing manual evaluation for our proposed method.

Data processing and storing review in a collection in MongoDB

We had selected all reviews of 11 restaurants and stored it in a json file. The file was collection of reviews and had information such as review text, user rating and so on.

For each review, we performed analysis by breaking them into sentences. For each sentence, we call the Stanford Core NLP and get the sentiment score for the sentence.

All the sentences in the review text are then stored with their corresponding sentiment scores in the mongodb collection named business_reviews_data. In the figure below we have listed how we stored a review in the collection. The fields 'stars' denote the rating given by the user with the review. 'nlpScore' field denotes the sentiment score of the entire review. Nouns recognized are the nouns picked up from the review. The fields - stars, nlpScore and nouns recognized are used for implementation of additional approaches that will be discussed later.

```
> db.business_reviews_data.findOne(<
<
  "_id" : ObjectId<"54910220a97cd8aa036061c1">,
  "businessId" : "JwUE5GmEO-sH1FuwJgKB1Q",
  "stars" : 4,
  "nlpScore" : 4,
  "categories" : "Food Service",
  "Nouns-recognized" : [
    "dinner",
    "selection",
    "food Open",
    "hours",
    "service",
    "night",
    "dish",
    "Fried Chicken Eggs Benedict"
  ],
  "Reviews" : {
    "Open 24 hours and provide nice service" : 4,
    "I usually go here after a night of partying" : 3,
    "Pretty good dinner with a nice selection of food" : 4,
    "My favorite dish is the Fried Chicken Eggs Benedict" : 2
  }
}>
>
```

Figure: Representation of a review record stored in the collection.

```

> db.Service.find().pretty()
{
  "_id" : ObjectId<"5487756e1c144447e81648b4">,
  "wordsList" : [
    "accept-cc",
    "lunch",
    "quick",
    "take-out",
    "experience",
    "wait staff",
    "smoking",
    "place",
    "by-appt-only",
    "kids",
    "alcohol",
    "area",
    "delivery",
    "parking-lot",
    "wifi",
    "take-reservations",
    "groups",
    "staff",
    "caters",
    "dinner",
    "people",
    "latenight",
    "coat check",
    "drive-thru",
    "slow",
    "parking-street",
    "brunch",
    "service",
    "wheelchair-accessible",
    "waiter",
    "breakfast",
    "valet"
  ]
}

```

Figure : Words list stored for a collection service.

Generate Rating

Here we select all the sentence of the reviews for a business from the collection along with their sentiment score. For every sentence we pick up the keywords (noun words – unigram, bigram). The keywords are then used to call a `findAllCategories` function which determines the category for that sentence. For instance, given a sentence “The service and staff are good”. Upon calling `findAllCategories`, the function would return service as the category.

We have created a collection of word list for categories - food, service, and ambience and stored in collections in the MongoDB that `findCategories` uses to determine the appropriate category. `findAllCategories` is a soft categorization and can assign a sentence two more than one category.

After categorizing all sentences in the reviews for a restaurant to the appropriate categories we then accumulate the sentiment scores, along with the count of all the sentences in reviews talking about a particular category. These results are stored in a `HashMap` - `review_scores`. Once the reading of all reviews for a business is completed the results are averaged and stored in a collection. In the figure below, for a restaurant the ratings generated across categories is shown.

```

> db.yelp_rank_restaurants.findOne()
{
  "_id" : ObjectId("54922c83673c03a2d03f15d9"),
  "business_id" : "jqo3Ljexof9sA8PhSTwEjA",
  "ourRating" : {
    "Service" : 2.764705882352941,
    "Ambience" : 2.8333333333333335,
    "Food" : 3.1904761904761907
  },
  "userRating" : {
    "Service" : 3.75,
    "Ambience" : 3.4,
    "Food" : 3.7333333333333334
  },
  "reviewRating" : {
    "Service" : 3.625,
    "Ambience" : 3.5,
    "Food" : 3.6
  }
}

```

Figure: Record for ratings generated by different approaches for a restaurant.

In the collection above, ourRating is the ratings generated for our proposed solution. The userRating and reviewRating fields in the collection will be discussed in additional approaches.

Additional Approaches

Professor Xiaozhong Liu had suggested us two additional approaches for generating ratings for restaurants.

Approach 1:

Make use of user rating instead of using Stanford Core NLP for sentiment analysis and assign user rating to service, food, ambience accordingly if mentioned in reviews.

Approach 2:

Perform sentiment analysis on the entire review and identify the categories the reviews are talking about and make use of it

Implementation of additional approaches

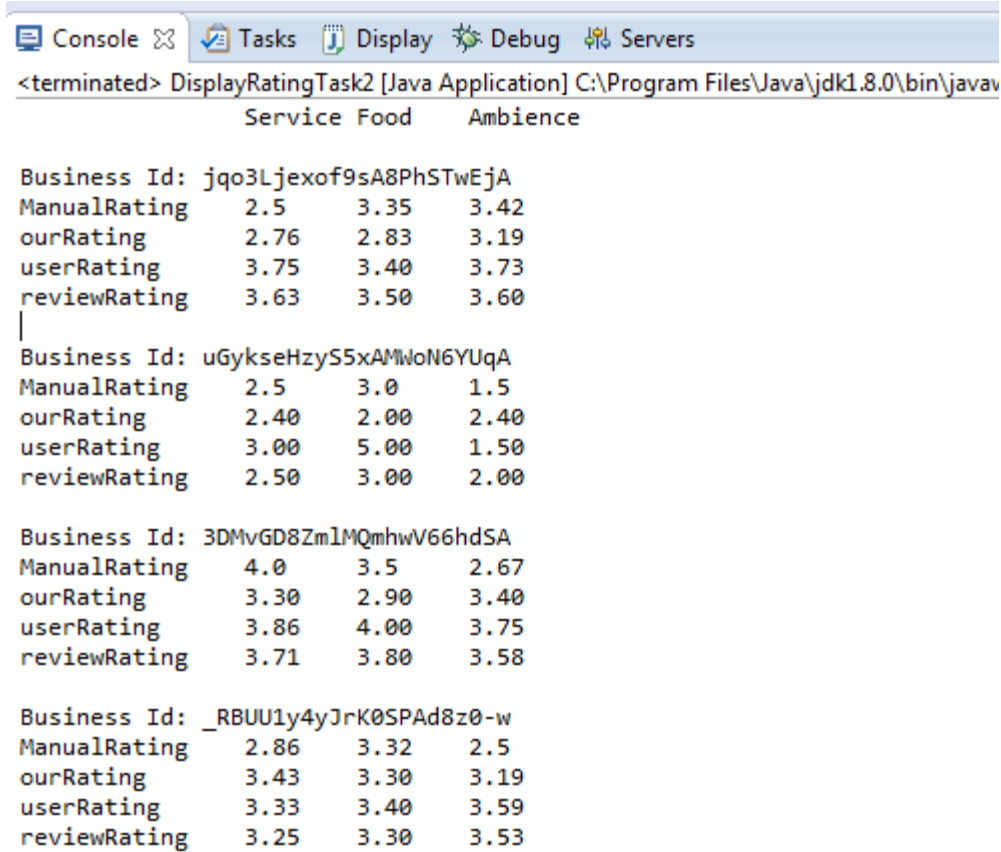
Implementation of both the approaches needed us to operate on the entire review, so we performed feature extraction on the entire review and stored in the collection with the fields names nouns recognized.

We analyzed the sentiment of the entire review and stored the score in the collection with the fieldname – nlpScore. We also added a field stars that the rating user has given in the review.

The nouns recognized are used to identify the categories been spoken about and the nlpRatings and userRatings are accumulated including the count of the reviews that belong to a category. These results are then aggregated and stored in the collection in the fields - userRating and reviewRating respectively.

Evaluation and Results

To compare the results of our proposed approach and suggested approaches we had randomly selected 11 restaurants and the 104 reviews were analyzed. We used an expert to annotate these reviews and the ratings for each category were aggregated. In this case, an expert was one of our team members.



```
<terminated> DisplayRatingTask2 [Java Application] C:\Program Files\Java\jdk1.8.0\bin\javaw
Service Food Ambience

Business Id: jqo3Ljexof9sA8PhSTwEjA
ManualRating 2.5 3.35 3.42
ourRating 2.76 2.83 3.19
userRating 3.75 3.40 3.73
reviewRating 3.63 3.50 3.60
|
Business Id: uGykseHzyS5xAMWoN6YUqA
ManualRating 2.5 3.0 1.5
ourRating 2.40 2.00 2.40
userRating 3.00 5.00 1.50
reviewRating 2.50 3.00 2.00

Business Id: 3DMvGD8ZmlMQmhwV66hdSA
ManualRating 4.0 3.5 2.67
ourRating 3.30 2.90 3.40
userRating 3.86 4.00 3.75
reviewRating 3.71 3.80 3.58

Business Id: _RBUU1y4yJrK0SPAd8z0-w
ManualRating 2.86 3.32 2.5
ourRating 3.43 3.30 3.19
userRating 3.33 3.40 3.59
reviewRating 3.25 3.30 3.53
```

Figure : Ratings generated for the restaurants.

The results generated can be viewed in the folder task2/output.txt .

The file which contains the reviews which were manually ranked by a human expert is “Ranking_Reviews_manually.txt”

For analysis we have taken an average of the ratings for service, food, ambience for all the restaurants.

Average Rating for restaurants			
Approach	Service	Food	Ambience
ManualRating	2.950909091	3.290909	2.72
ourRating	2.990909091	3.034545	3.02272727
userRating	3.481818182	3.779091	3.30272727
reviewRating	3.37	3.513636	3.29363636

Table 2: comparison of average rating generated for service, food, ambience with all the approaches.

For the considered data, if we assume the manual rating as a baseline for comparison an observation of the results suggests that our proposed hypothesis for generating results is better than user rating and review rating approaches.

Task 2: Part 2

In this task, we predict what kind of experience the user had after visiting the restaurant.

The experience can be of one of the following types:

- Positive Experience
- Very Positive Experience
- Very Negative Experience
- Neutral Experience
- Negative Experience

We think both the features (Part 1 and Part 2), which we have introduced, will help the user in a situation where the user does not have time to read and analyze all the reviews.

Tokenization and Extraction

For Task 2, the only difference between tokenization and extraction done in Task 1 is that in task 1, when we extract the nouns, we only extract one word. But in this task, we extract those bigrams or trigrams where there are consecutive nouns present in a sentence. For example, consider the following sentence:-

“The Butter Chicken is awesome.”

Now we will store Butter Chicken as one word and when we query the list of words stored in the Indian dishes wordlist, it will easily find that the sentence talks about Indian food and hence can assign a score for the category “Food”.

Approach

Our hypothesis in this task is that sentence-wise sentiment analysis is better than taking the whole review and giving it a sentiment score because our aim was to do aspect based sentiment analysis and we thought we would get useful information if we go sentence-wise. We also assumed that most of the time people tend to explain about one attribute in one sentence. Since we wanted to give score to different categories such as food, service and ambience, we thought we would be able to predict it accurately by analyzing the reviews sentence by sentence.

Our algorithm works in the following way:-

-From our dataset of reviews (test_reviews.json), we pick up the first review (using JSON parser to extract a review) and look at its business_id and store it in a HashMap.

-Then we store the business_id as the key and the value is another HashMap, which has the sentiment (one of the five types listed above).

-This sentiment is the key of the HashMap, which is stored as the value.

-Its value is an integer, which is nothing, but the counter, which after evaluating all the reviews will give us the count of sentences which, had sentiment of a particular type.

-Once all the reviews are evaluated we have a HashMap, which contains information about a business, and the how many sentences from all the reviews of that business convey which type of sentiment.

-Once we have this information, we can easily calculate the percentage of sentences with a sentiment type.

-We assume that this is the percentage of the users which have experienced such sentiments since our hypothesis is that sentence-wise sentiment analysis will give better and accurate results.

Results

The results are displayed as shown in the following image. The first column is the business id and besides it is the percentage of users which have had what kind of experience. The full results are provided in the file name “calculateExperience_output.txt”

```
jQo3Ljexof9sA8PhSTwEjA : {Neutral=20, Very negative=3, Negative=37, Positive=
21}

uGykseHzyS5xAMWoN6YUqA : {Neutral=7, Negative=30, Positive=6}

3DMvGD8Zm1MQmhwV66hdSA : {Neutral=12, Very negative=3, Negative=24, Positive=
32, Very positive=1}

_RBUU1y4yJrK0SPAd8z0-w : {Neutral=13, Very negative=2, Negative=42, Positive=
34, Very positive=4}

MKsb2VpLB-0UBODcInDsSw : {Neutral=6, Very negative=2, Negative=22, Positive=
11}

77ESrCo7hQ96VpCWwdvoXg : {Neutral=18, Negative=60, Positive=41, Very positive=
3}

KPoTixdjoJxSqRSEApSAGg : {Neutral=17, Very negative=1, Negative=64, Positive=
46, Very positive=3}

JwUE5GmEO-sH1FuWJgKB1Q : {Neutral=17, Negative=25, Positive=27, Very positive=
2}

rdAdANPNocvUtoFgcaY9KA : {Neutral=19, Very negative=1, Negative=39, Positive=
31, Very positive=5}

LYyGQgL60VKdV-p_90xmWQ : {Neutral=10, Very negative=4, Negative=48, Positive=
23}

UxMnY3Dxafucv5cXHyfBSA : {Neutral=18, Negative=41, Positive=11}
```

Fig 1 – Screenshot of results, which get displayed after evaluating the reviews.

4. Further Work

- Predict categories for business which have categories other than “Restaurants”.
- Find information about more categories such as price, worthiness, deals & discounts and display the results based on the reviews.
- Use the likes for a review and increase the score accordingly for the nouns extracted from those reviews.
- Ranking the service and ambience using already provided attributes (from attributes present in Screenshot named “yelp_academic_dataset_business_attributes” collection).

5. References

- Stanford CoreNLP [<http://nlp.stanford.edu/software/corenlp.shtml>]
- <http://www.ics.uci.edu/~vpsaini/>
- Apache OpenNLP [<https://opennlp.apache.org/>]