Categorization and Analysis of Yelp restaurant reviews

Niharika Purbey
Computer Science Department
Columbia University
np2544@columbia.edu

Aishwarya Iyer
Computer Science Department
Columbia University
ai2336@columbia.edu

Abstract—**The objective of this project is to use the Yelp reviews dataset to extract meaningful topics and their associated sentiment from Yelp restaurant reviews. We use topic modeling algorithm called LDA (Latent Dirichlet Allocation) and IBM's Alchemy API (Sentiment Analysis) to achieve this.**

Keywords-Big Data Analysis; Topic modeling; Latent Dirichlet Allocation; sentiment analysis

I. Introduction

Yelp is a crowd-sourced local business review and social networking site. Its reviews help customers find restaurant, bars, hair salons, stores, etc that suit their taste and preferences. Reviews are also a great way for business owners to identify what aspect of their business is working and what areas need improvement. However, since these reviews consist largely of unstructured text, identifying the relevant topics covered by the review can be time consuming. We attempt to address this issue by applying topic modeling techniques to identify what topics the review writers have touched upon in their review. Sentiment analysis is applied to determine the sentiment associated with each topic to obtain a granular break down of a businesses' strong and weak suits. Using these topics and their associated sentiment, a business owner can easily identify the issues that customers face with their business. Review topics or themes can also help users identify businesses that suit their personal preferences, such as “good service” and “cheap food”. We segregated and considered reviews that were marked as useful, then using LDA algorithm we extracted the main topics in each review for a particular business, then manually assigned relevant tags which make the business searchable. To demonstrate the real-world applications and usage of topic extraction, we built a web application that enabled users to select topics that interested them and returned all matching businesses.

II. Related Works

Yelp has sponsored the public “Yelp Dataset challenge” to discover innovative and useful ways in which they might use the large dataset available to them. Several papers have been submitted, several of which use LDA algorithm to extract subtopics. In one paper titled “Personalizing Yelp Star Ratings: a Semantic Topic Modeling Approach”[7], by Jack Linshi, in addition to extracting relevant topics from reviews, the author adds an additional tag of “GOODREVIEW” or “BADREVIEW” in the high rated and low rated reviews in order to extract topics by sentiment. For example, if there exists a “one star” review about restaurants that says, “the food was terrible”. Therefore, the sentiment attached to a tag is clearly known. This approach is similar to ours, although it has shortcomings and may not always be accurate. Some reviews may speak about both bad aspects and good, and broadly categorizing a tag according to the overall review might lead to some inaccuracies.

III. System Overview

For our project, we used 2 datasets from the Yelp Academic Dataset[1], the review dataset and the business information database. It consists of around 2.2M reviews for around 77K businesses. From this, we used only those reviews that belonged to a restaurant. For the purpose of training, we used around 77K reviews. Given below is the schema of the reviews dataset and the business information dataset.

```
review
{
'type': 'review',
'business_id': (encrypted business id),
'user_id': (encrypted user id),
'stars': (star rating, rounded to half-stars),
'text': (review text),
'date': (date, formatted like '2012-03-14'),
'votes': {(vote type): (count)},
}

business
{
'type': 'business',
'business_id': (encrypted business id),
'name': (business name),
'neighborhoods': [(hood names)],
'full_address': (localized address),
'city': (city),
'state': (state),
'latitude': latitude,
'longitude': longitude,
'stars': (star rating, rounded to half-stars),
'review_count': review count,
'categories': [(localized category names)]
'open': True / False (corresponds to closed,
not business hours),
'hours': {
(day_of_week): {
'open': (HH:MM),
'close': (HH:MM)
},
...
},
'attributes': {
(attribute_name): (attribute_value),
...
},
}
```

We preprocessed the data before using it. We iniatially removed all the stop words like “a”, “the” and found the part of speech of each word using nltk part of speech tagger [2]. We then found the noun lemmas of the words using WordNet Lemmatizer, this would be useful to generate the topics as the topics would generally be nouns.

After the preprocessing, we trained the data using LDA (Latent Dirichlet Allocation) algorithm. We used the gensim LDA model to train our system [3]. The training process took around 4 hours to complete.

We generated 50 topics from the training process. Given below are examples of some of the topics generated.

```
(11, '0.084*roll + 0.073*sushi + 0.037*crab +
0.032*shrimp + 0.026*fish + 0.025*tuna +
0.025*seafood + 0.024*chef + 0.018*salmon +
0.017*place')

(14, '0.049*time + 0.041*service + 0.038*customer +
0.024*day + 0.017*business + 0.015*work +
0.015*experience + 0.015*manager + 0.014*review +
0.014*hour')

(32, '0.078*kid + 0.033*fun + 0.032*husband +
0.031*child + 0.030*daughter + 0.028*mom +
0.025*je + 0.023*de + 0.017*parent + 0.016*family')
```

We then manually annotated the 50 topics based on our judgments. For the examples shown above, the general topics we gave were “seafood”, “service” and “kid-friendly” respectively.

For the testing phase, we considered a different set of reviews, and using the trained model, generated topics for a couple of restaurants. For the testing phase, we chose reviews of around 20 restaurants. For each topic that was generated, we also performed sentiment analysis on the reviews. We used IBM Alchemy API for this purpose [4]. Performing sentiment analysis on the reviews enabled us to get the sentiment associated with each topic. For example, if the topic was “service” and the sentiment associated with that review was “negative”, it means that the service of the restaurant was not good. Positive sentiment associated with that review would indicate good service.

We made a UI for the testing phase, wherein we provided a set of topics to the users. Based on the topic selected by the user, the system would generate a list of restaurants that are associated with the topic along with the sentiment associated with the topic for each restaurant.

IV. Algorithm

As mentioned, we used the Latent Dirichlet Allocation algorithm which is a generative statistical topic model for discovering the abstract "topics" that occur in a collection of documents [5].

This algorithm can be explained using a generative approach wherein we start with a collection of distribution on words (topics), and a distribution on topics for each document.

The algorithm works as follows:
We first generate each topic, which would be a distribution on words. For each document, we then generate a distribution on topics. For the nth word in the dth document, we allocate the word to a topic and then generate the word from the selected topic [6].

In our project, each document corresponds to a review in the dataset and each topic corresponds to a distribution on the words in the review.

The output would a set of topics that were generated. For example, we might have the global topics of “seafood” and “service” and a particular review could be 75% about food and 25% about customer service. For each word in the review, we first draw a topic based on the topic proportions for that particular review and then we draw a word based on the distribution of that particular topic.

To run this algorithm, we used the gensim module of LDA. We used this module for both, training and testing i.e LDA model estimation from a training corpus and inference of topic distribution on new, unseen reviews.

Gensim uses the stochastic optimization procedure for Latent Dirichlet Allocation. This optimization procedure has the advantage of scaling better to large datasets. The model is streamed i.e. the training documents may come in sequentially. It runs on constant memory with respect to the number of documents. The model is also distributed, wherein it can make use of a cluster of machines to hasten the model estimation process.

V. Software Package Description

The general flow of our project is data preprocessing, training and testing. Figure 1 gives a basic overview of our system.

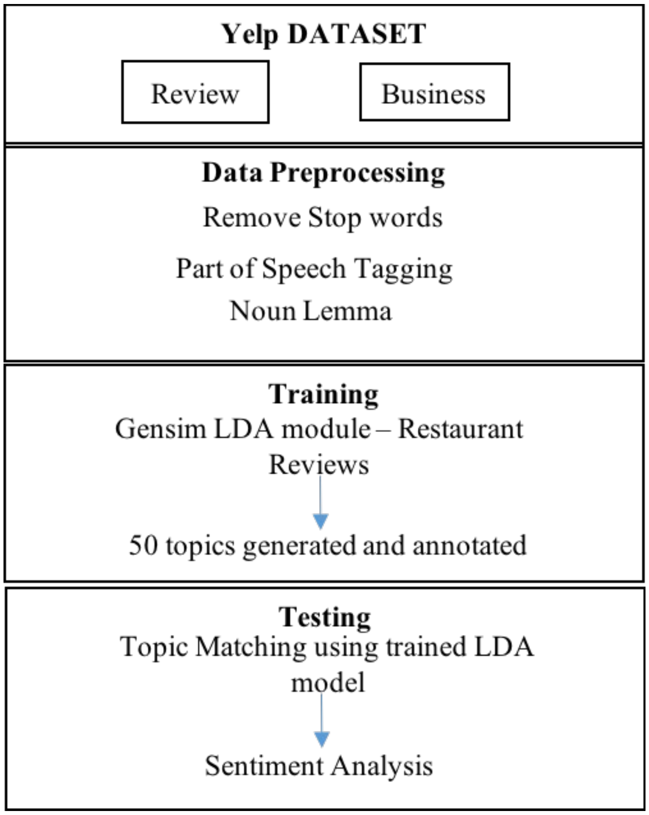


Figure 1: System Overview Design

settings.py: Contains information about the Mongo DB collections and the dataset files.

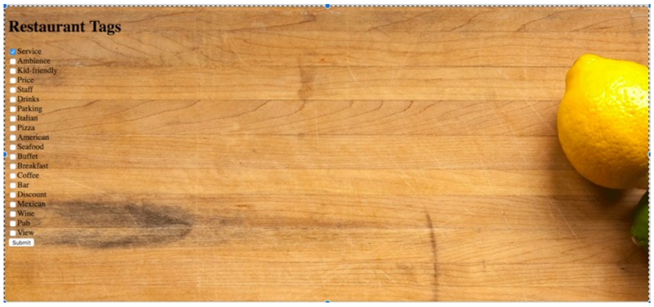
jsonExtraction.py: Basic data preprocessing and training. Firstly, we removed the stop words. This file contains code for part of speech tagging, noun lemmatizing and training the model using the LDA algorithm. Used NLTK libraries for part of speech tagging and noun lemmatizing and used gensim LDA module. Also stored, the part of speech tags and noun lemmas in MongoDB.

predict.py: This contains code to make prediction for the unseen reviews for a set of restaurants using the trained LDA model. It also stores the results in MondoDB.

sentiment_analysis.py: This code analyzes the reviews and generates sentiment and stores it in the DB. Used Alchemy API for the same.

server.py: Code to run the web front end of our system.

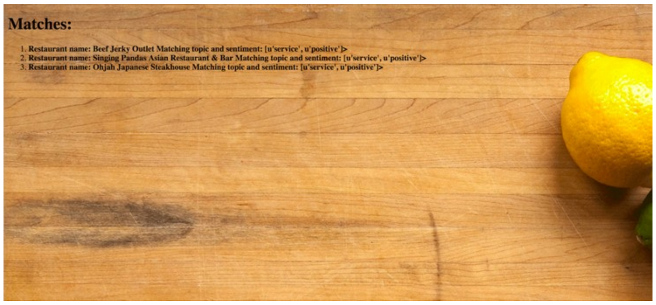
Given below are some of the screenshots of our UI.



Restaurant Tags

☑ Service
☐ Ambience
☐ Kid-friendly
☐ Price
☐ Staff
☐ Drinks
☐ Parking
☐ Italian
☐ Pizza
☐ American
☐ Seafood
☐ Buffet
☐ Breakfast
☐ Coffee
☐ Bar
☐ Discount
☐ Mexican
☐ Wine
☐ Pub
☐ View

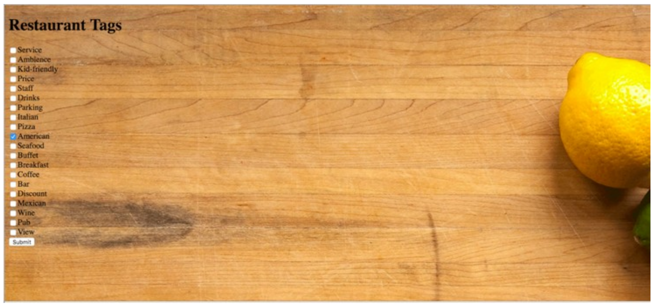
Figure 2: Selection of topic – “Service”



Matches:

1. Restaurant name: Beef Jerky Outlet Matching topic and sentiment: [u'service', u'positive']>
2. Restaurant name: Singing Pandas Asian Restaurant & Bar Matching topic and sentiment: [u'service', u'positive']>
3. Restaurant name: Ohjah Japanese Steakhouse Matching topic and sentiment: [u'service', u'positive']>

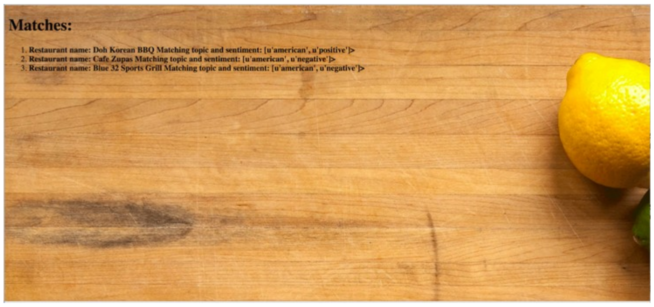
Figure 3: Restaurant Matches for the topic – “Service”



Restaurant Tags

☐ Service
☐ Ambience
☐ Kid-friendly
☐ Price
☐ Staff
☐ Drinks
☐ Parking
☐ Italian
☐ Pizza
☑ American
☐ Seafood
☐ Buffet
☐ Breakfast
☐ Coffee
☐ Bar
☐ Discount
☐ Mexican
☐ Wine
☐ Pub
☐ View

Figure 4: Selection of topic – “American”



Matches:

1. Restaurant name: Doh Korean BBQ Matching topic and sentiment: [u'american', u'positive']>
2. Restaurant name: Cafe Zupas Matching topic and sentiment: [u'american', u'negative']>
3. Restaurant name: Blue 32 Sports Grill Matching topic and sentiment: [u'american', u'negative']>

Figure 5: Restaurant Matches for the topic – “American”

As described above, figure 2 and 4 depict the selection of a ‘tag’ or a ‘topic’. This selection leads to the result page (figure 3 and 5) wherein we display the corresponding restaurant matches and also the sentiment associated with it.

VI. Experiment Results

We chose to test our system on reviews of about 20 restaurants not included in the training set.

Given below is the list of some of the topics that were generated.

1. Topics generated

Management, staff	juice	cake, cupcake, bakery
drinks	buffet, price, quality	view
pizza, pie	taco, Mexican	ambience
Bbq, lunch, sandwich, meat, American	order, service, delivery	price, discount
seafood, shrimp, fish, salmon	time, service	ice cream, dessert
location	kid-friendly	wine
Italian, pasta	club, bar, music	

2. Manually verifying topic accuracy

For each of these topics, we manually checked the review to determine whether the all important topics were captured. In 95% of the cases, all the important topics and nothing else were generated, in other cases a few unnecessary topics were generated as well. But since our method involves manually assigning tags, we can double check the accuracy of these topics.

VII. Conclusion

In conclusion, we were successfully able to generate relevant topics for reviews and analyze their sentiment to derive meaningful tags.

As future work, one could extend our idea of searching based on tags and add geo location filters as well to make the application more complete.

Contributions:
Following were the contributions to this project:
Aishwarya Iyer- Data Preprocessing
Niharika Purbey- Sentiment Analysis
Joint effort : Training, Testing, Web Application

Acknowledgment

We would like to thank Professor Ching-Yung Lin for providing us with invaluable knowledge and assisting us both during and after his work hours. We could not have completed this project without his guidance. W would also like to thank Eric Johnson, David Dhas and Sanjana Gopisetty for their patience and encouragement.

References

[1] Yelp Dataset - https://www.yelp.com/dataset_challenge

[2] NLTK Dcocumentation - http://www.nltk.org/

[3] Gensim_LDA_Module-https://radimrehurek.com/gensim/models/ldamodel.html

[4] IBM Alchemy API - http://www.alchemyapi.com/

[5] David M. Blei, Andrew Y. Ng, Michael I. Jordan, “Latent Dirichlet Allocation”, Journal of Machine Learning Research 3 (2003) 993-1022

[6] Latent_Dirichlet_Allocation-hhtp://blog.echen.me/2011/08/22/introduction-to-latent-dirichlet-allocation

[7] Jack Linshi, “Personalizing Yelp Star Ratings: a Semantic Topic Modeling Approach”