Recommendation system for the entrepreneurs to be successful (Yelp Dataset)

Sharmodeep Sarkar
Northeastern University
Boston, MA
sarkar.s@husky.neu.edu

Sarita Joshi
Northeastern University
Boston, MA
joshi.sar@husky.neu.edu

Abstract

Social media is an attractive data source for creating informal information visualizations that reflect people's lives and daily experience. User data aggregated by services such as Twitter, Instagram, and other social media sites has provided useful insights in various dimension. We plan to work with the Yelp dataset. Yelp is a great example of a huge database of reviews on businesses ranging from food, restaurants, shopping, entertainment. Generally, one tends to consume this extensive data only by looking at the star rating for a business in specific domain and ignores other useful information. Moreover, if a business stakeholder is willing to analyze the market trend based on this internet search and review service (Yelp.com), he has to go over all the reviews in a specific region to get an idea about the demand. (demand and supply practice) This strategy seems implausible with a high probability of failure. Thus, we aim to develop a trained software that can help us achieve better results with regards to this problem and provide a chance of successful business.

.

1. Introduction

Yelp, arguably the Internet's largest social platform influences the choice of users. A user registered at Yelp explorers' businesses around based on the star rating. It appears to influence the user's choice and depicts user's judgement. Similar, for a business stakeholder, user review after the service is a crucial factor in terms of market trend and competition. Touting its average 135 million monthly users, dominance in local search engine results, and ability to shut down businesses overnight, Yelp is force to be reckoned with. Currently, a Yelp's star rating for a business is the mean of all star ratings given to that business. However, it is necessary to consider the implications of representing an entire business by a single star rating. What if one user cares about only food, but a restaurant's page has a 1-star rating with reviewers complaining about poor service that ruined their delicious meal? The user may likely continue to search for other restaurants, when the 1-star restaurant may have been ideal. Thus, for such a massive dataset plays a profound role on a business, we often have simpler structures such as topics in document or a title for user area of interest. By breaking these reviews down into latent subtopics using LDA, we are then able to predict a restaurant's star rating per hidden topic. Our end goal is to design a recommendation system for business owners that considers such important hidden variables with its composition in each review text and comes up with a aggregate star rating which is composed on these important factors. Stating all these, we plan to come up with a supervised learning model that can then predict the success rate for a specific business in a specific location or with preference to specific topic.

1.1. Related work

There are several methods for learning abstract topics in a collection of text. LDA, Latent Dirichlet Allocation is a standard known method to discover hidden topics. There are many other topic modelling methods that appears to be useful in studying such useful documents and its impact. While there are number of works that explore the relationship between Yelp review texts and star ratings, most consider text and ratings separately. A relatively new topic model, the pachinko allocation model (PAM), which models correlation between topics and between words, also appears to be a promising method for studying Yelp review text as well. One related study, for example, uses a traditional LDA to discover hidden topics, and then uses these hidden topics to predict star ratings by averaging the star ratings of all reviews for businesses that contained a specific topic.

Our work involves unsupervised learning, but the novelty of our work lies in considering the weights of these hidden topics based on the result of topic modelling. Unlike previous works, we choose the representation of each topic based on various algorithms such as levenshtein distance, k-medoids so that the approximation is as true as possible. While generating abstract topics, we even

consider the lemma of the word context, semantics of the review text and then calculate the business star based on these hidden topics weights. We then use this crucial information to design a recommendation system for a business stakeholder that may plan to look for competing businesses around, best suitable place based on specific topic for business expansion or help them to know the overall business star rating based on aggregation of such useful information and even help them know a business' best feature and worst feature.

2. Problem Addressed

We preprocess the review text i.e. 158,000 Yelp reviews using LDA, POS tagger, using a bayes approach to get all the nouns from the reviews and use a lemmatizer to consolidate words belonging to the same stem or cluster categories based on the patterns in the data. This step is important because we want the LDA to give out topics which are just nouns and not any other POS. We then store the relevant information (business id, Review id, review text, nouns) in python pandas' data frame. Next, we feed these to our LDA to get the latent topics. Each of the reviews are tagged with some number of topics. Now, to calculate the stars for each topic we average the review rating of all the reviews belonging to a specific sub-topic. If the results are not satisfactory we also use weighted average using the percentage that topic made up of the respective review or use positive and negative weights of neighbor words to those words in the review relating to our given topic. Finally, we cross reference the geographic locations of the business ids and each of their subtopic ratings to get a sense of the demand and delivery of each of these latent topics in the geographical regions of the cities, within the scope of given dataset. Thus, we attempt to provide a feasible system to the business owners by providing them information about abstract hidden topics with appropriate novelty measures, recalculate the Yelp business star for these n number of businesses and further recommends business owners based on various factors.

3. Dataset

This project is performed with the data from the Yelp Dataset Challenge. This dataset includes business, review, user, and check-in data in the form of separate JSON flat-files. A business object includes information about the type of business, location, rating, categories, and business name, as well as contains a unique business id. A review object has a rating, review text, and is associated with a specific business id and user id. We mainly deal with these two types of JSON data objects. There are overall 55,000 business information provided in the dataset and over 158,000 corresponding reviews. Based on this dataset, we pay attention to the overall semantics of the overall review text and then specialize on specific topic. The business data expand across 4 countries, 10 cities. The overall data size exceeds 5GB.

Business object

```
{
  '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),
  will be adequate.

  '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),
      ...
  },
  }
```

Review object

```
{
  '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)},
}
```

4. Algorithms

We used Python scripts. We designed our own LDA algorithm, compared the results with the Genism Python Library which is a topic modeling tool for documents. We have used D3.js for getting useful visual insights.

4.1. Latent Dirichlet Allocation

Latent Dirichlet Allocation (LDA) is a Bayesian generative model for text. It is used as a topic model to discover the underlying topics that are covered by a text document. LDA assumes that a corpus of text documents cover a collection of K topics. Each topic is defined as a multinomial distribution over a word dictionary with |V | words drawn from a Dirichlet $\beta k \sim$ Dirichlet(η). Each document from this corpus is treated as a bag of words of a certain size, and is assumed to be generated by first picking a topic multinomial distribution for the document $\theta d \sim$ Dirichlet(α). Then each word is assigned a topic via the distribution θd, and then from that topic k, a word is sampled from the distribution βk. θd for each document can be thought of as a percentage breakdown of the topics covered by the document. The topic distribution of a corpus from the LDA model can be found in numerous ways that converges to the most likely parameters (word distributions per topic and topic distributions per word) We did a comparative study using the Python Library Gensim used for topic modelling and the LDA algorithm implemented by us a small subset of data for Madison city. Our design choice involved taking the top 10,000 words in the corpus after all preprocessing i.e. stopwords removal, stemming, lemmatizing. For 20 topics, we opted for top 20 frequently occurring words for the first iteration and all words for the second iteration.

4.2. Topic Modelling Example: (City Madison)

Words	Distribution	Frequency	Total Words in Class 0
topic: 0 (64139 words)			
beer	0.123841	9334	64139
selection	0.043456	3275	64139
place	0.04262	3212	64139
bar	0.039967	3012	64139
food	0.019828	1494	64139
game	0.01862	1403	64139
tap	0.015834	1193	64139
curd	0.015171	1143	64139
list	0.013658	1029	64139
night	0.010541	794	64139
menu	0.009015	679	64139
bartender	0.008856	667	64139
wine	0.008697	655	64139
lot	0.007542	568	64139
brew	0.007516	566	64139
burger	0.007277	548	64139
friend	0.006959	524	64139
atmosphere	0.006866	517	64139
drink	0.006746	508	64139
town	0.006428	484	64139

Table 1: Topic 0 obtained for the smaller dataset, with 20 words.

Note: We ran it as a sample for one of the city to provide statistical evidences. It success effectively for the complete dataset as well.

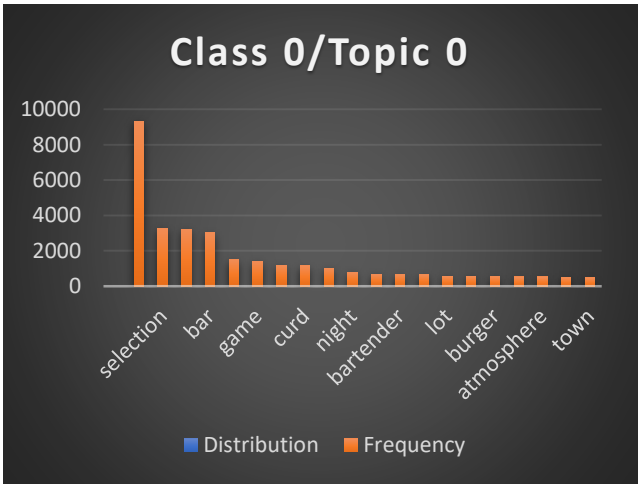

Figure 1: Topic-wise distribution across bag of words for topic 0

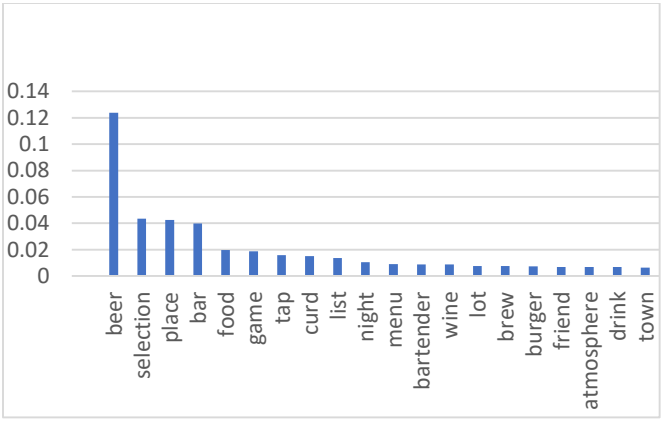

Figure 2: Word distribution in topic 0

Figure 3: Word Cloud tag for the words in Topic one.

Based on the above plotted graphs, we can see that the word "Beer" precedes over all other words in the current bag of words. Adding further to the cloud tag, we implemented few novelty approaches such as levenshtein distance, k-medoids so that the approximation is as accurate as possible.

After, a constant number of iterations i.e. say initial value of 100, we ran Cluster AffinityPropagation algorithm (from sklearn) that will iterate and generate a leader for each group of bag of words (All the 20 topics that we obtained as an output of LDA algorithm)

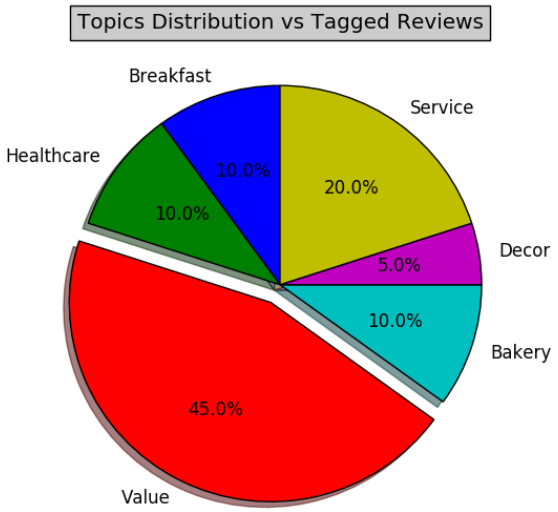

Figure 4: This is a reference plot for understanding the word clustering for small data points. 'The Topics are the Cluster Leaders found by the Cluster Affinity Propagation algorithm (from sklearn). The graph shows the percentage of reviews tagged (by running out tagging system against our generated LDA model) in each of these Topics.)

Based on human cognition i.e. manually clustering the leaders and a comparative study among other clustering algorithm i.e. K-means (with varying k value to iterate and get an adequately better result), K-medoid, levenshtein distance and affinity propagation clustering algorithm, we got the best match leaders from the affinity propagation algorithm. Based on the dimensionality reduction implemented via LDA in the above step at document level, we can represent complex heavy textual information in simple vectorial format. Also, as preprocessing we do keep track of the semantics of the textual information which comes up at a minor risk of few semantics misplaced.

4.3. Predicting Topic Stars

Now, after running LDA unsupervised algorithm we generated a model based on the input review text (N= 8320, city: Madison), found 20 topics with bag of words representing each topic and a phi value generated for each word distribution topic-wise.

$\varphi_{k=1\ldots K,w=1\ldots V}$	probability (real number between 0 and 1)	probability of word *w* occurring in topic *k*
$\varphi_{k=1\ldots K}$	*V*-dimension vector of probabilities, which must sum to 1	distribution of words in topic *k*

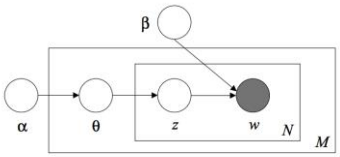

Figure 1: Graphical model representation of LDA. The boxes are "plates" representing replicates. The outer plate represents documents, while the inner plate represents the repeated choice of topics and words within a document.

where $p(z_n|\theta)$ is simply θ_i for the unique i such that $z_n^i = 1$. Integrating over θ and summing over z, we obtain the marginal distribution of a document:

$$p(\mathbf{w}|\alpha,\beta) = \int p(\theta|\alpha)\left(\prod_{n=1}^{N}\sum_{z_n} p(z_n|\theta)p(w_n|z_n,\beta)\right)d\theta. \quad (3)$$

Finally, taking the product of the marginal probabilities of single documents, we obtain the probability of a corpus:

$$p(D|\alpha,\beta) = \prod_{d=1}^{M}\int p(\theta_d|\alpha)\left(\prod_{n=1}^{N_d}\sum_{z_{dn}} p(z_{dn}|\theta_d)p(w_{dn}|z_{dn},\beta)\right)d\theta_d.$$

Based on these values after the train step, we took random number of review text for the same city, validated the trained model across multiple subsets for varying iterations.

Example of a test sample: "I enjoy this place everytime I go in there. Its family owned and has consistently good food. Large soda selection with awesome crab rangoon. My favorite dish is the Happy family which has beef, pork, chicken, crab, and veggies in brown sauce."

After evaluating the result i.e. the topic to which this unseen text document belongs, we found that resultant topic as Topic 1 with bag of words as below:

```
'food: 0.049616 (3654) (62419)'
'meat: 0.038823 (2859) (62419)'
'place: 0.028355 (2088) (62419)'
'sauce: 0.025178 (1854) (62419)'
'menu: 0.024554 (1808) (62419)'
'side: 0.019978 (1471) (62419)'
'pork: 0.019462 (1433) (62419)'
'chicken: 0.014290 (1052) (62419)'
'beef: 0.014127 (1040) (62419)'
'flavor: 0.012959 (954) (62419)'
'rib: 0.012511 (921) (62419)'
```

```
'meal: 0.010746 (791) (62419)'
'portion: 0.010325 (760) (62419)'
'bean: 0.009891 (728) (62419)'
'restaurant: 0.009687 (713) (62419)'
'option: 0.009212 (678) (62419)'
'item: 0.009171 (675) (62419)'
'price: 0.008560 (630) (62419)'
'quality: 0.008234 (606) (62419)'
'order: 0.007895 (581) (62419)'
```

Figure 5: Output on a new document. Topic 1, consisting of above words with highest distribution for food.

Running the affinity propagation for this set of words, we tagged this topic as "food"
The overall accuracy prediction for 10 set of unseen documents, we received a utterly wrong tag for only one document. For 70% of it, the results were considerable and for the rest we had a 50-50 case.

Thus, we went ahead with the star prediction, hidden topic-wise instead of the aggregated business star. Based on the python library TextBlob, we then calculated the sentiment score for each of the reviews [-1, +1] i.e. [worst, best]. Then we calculated the topic-wise score for each of the reviews using the below formula. [MAX_STAR = 5 for Yelp]

```
topic_star = row['sentiment_score']*row['stars']*score * MAX_STAR
```

Based on this topic-wise star calculation, we then calculated the topic-wise business rating that was the average of the topic-wise scores for n number of reviews for a given business Id.
These topic-wise scores were then aggregated and averaged over the total number of topics [K for the LDA] to get the predicted business rating. [star rating]

Results:

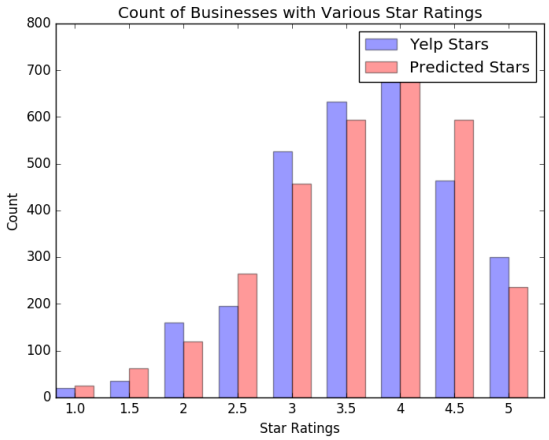

Fig 6: Count Distribution of predicted and Yelp stars

4.4. Predicting Topic-wise Business Stars

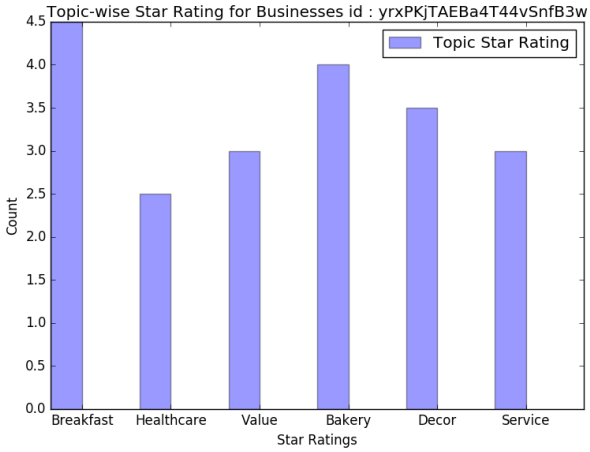

Fig 7: Topic-wise star rating for business id= "yrxPKjTAEBa4T44vSnfB3w" (whose one of the reviews has been used as a test sample for Figure 5)

4.5. Business recommendation system

After implementing the algorithms from step 4.1- 4.4, we calculated K extra features (K = number of topics) By analyzing the dataset we found that for a business id, the city, state, categories, review counts and the K topic-wise stars are the most informative features for predicting the overall star rating based on any classification algorithm. For a comparative study, between the stars generated from step 4.4 and the stars predicted using a SVM classifier, we used both different kernels viz. linear, polynomial, sigmoid and rbf.

Accuracy: Based on the dense dataset that we have for the city Madison, we found achieve a reasonably good amount of accuracy for a linear and polynomial kernel i.e. > 90% and less than 33% for sigmoid and rbf kernel.

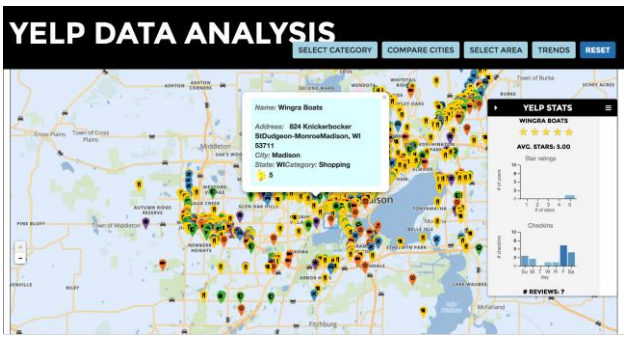

Fig: 8 Dense data visualization for Madison city, Yelp

SVM classifier with a linear or polynomial kernel gives a good observation as compared to other kernels. Refer Fig. 8 for the Comparison.

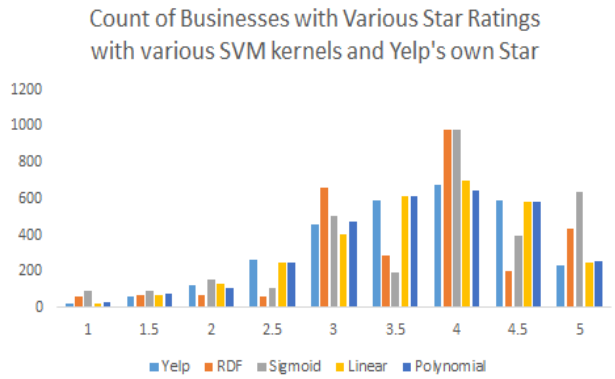

Fig 8: Count of businesses, city Madison with various star ratings and different algorithms.

These visualizations clearly portray the star prediction observed after the LDA and our business star calculation formula for the given dataset.

Based on the latent topics extracted, the topic-wise star ratings conferred and the over-all star ratings calculated from the topic-wise stars, the stake holders can use the system to recommend the business owners the most important parameters (the topics mined from the user reviews) to run a successful business at a particular location and of a particular category. For example, If the latent topics extracted are Service, Décor, Music etc. that means these are the topics the customers are talking about (reviewing on Yelp). Hence to run a successful business, the owner must be aware of his position (analogous to Star Rating) in all these different fields. The Yelp Star in the dataset is just an over-view of the business whereas the Topic-wise stars given a much better insight as to how the business is doing with regards to various parameters that are currently in demand.

This system can also be used for only Tagging a huge corpus of reviews

5. Conclusion

Using this system, we have successfully extracted K (K=20 for our evaluation) topics from the review. The affinity propagation algorithm seemed to work quite decently (manual evaluation) to get us the Topic names (cluster exemplar) from the bag of words for each Topic generated by the LDA algorithm. For the topic-wise star rating we used a novel formula based on the phi-value of the LDA for the words in the review, the sentiment score of the review (using python's TextBlob library) and the review overall

star rating from the dataset. The topic-wise star ratings seemed to work pretty well (refer results section). Now for the over-all star rating for a business we tried two approaches viz: Multiclass SVM classifier with various kernels and a simple mathematical formula for the based on aggregation of star ratings for a specific topic for a specific business. Comparing the Yelp Business Stars with the Calculated Business Stars we see that SVM with linear and polynomial kernel i.e. > 90% and less than 33% for sigmoid and rbf kernel. The aggregation approach for the overall Business-Stars proved to be 70-75% accurate depending on the topics mined on different iterations. Finally using this system and extending the available Yelp dataset to have topic-wise business star ratings, the business owners can have a much deeper insight for their existing business or chances of expansion depending on location and category. For example, one can easily answer questions like which is the best place Y for an entrepreneur to open a restaurant of category X1 provided he offers best services in features X2…Xn .

6. Future Work

Currently we use TextBlob for the sentiment analysis of the user reviews. But this is an important aspect as a review may be regarding Food but it might be saying bad things about the Food, hence Food-star must be low. We have implemented the negative scoring for negative reviews based on the score provided by TextBlob but this can be made more robust by taking a bigram and trigram approach and have some contextual information while analyzing the reviews. This implementation is entirely an unsupervised approach. Hence the evaluation of the results was mostly done manually except for the over-all business star predictions. In future if we would have a topic-wise star rating from Yelp, the evaluation of the system can be done more rigorously.

References

[1] https://www.cs.princeton.edu/~blei/papers/BleiNgJordan2003.pdf.

[2] http://www.yelp.com/html/pdf/YelpDatasetChallengeWinner_ImprovingRestaurants.pdf

[3] http://www.datalab.uci.edu/papers/distributed_topic_modeling.pdf.

[4] http://nlp.stanford.edu/software/tmt/tmt-0.4/

[5] http://www.andrewng.org/portfolio/latent-dirichlet-allocation-2/.

[6] D. Cai, Q. Mei, et all. ”Modeling Hidden Topics on Document Manifold.” Department of Computer Science, University of Illinois. CIKM 2008.

[7] https://www.yelp.com/dataset_challenge

[8] https://people.cs.umass.edu/~wallach/courses/s11/cmpsci791ss/readings/griffiths02gibbs.pdf

[9] https://faculty.cs.byu.edu/~ringger/Fall2013-CS679/lectures/Lecture13-gibbslda.pptx

[10] http://scikit-learn.org/stable/modules/generated/sklearn.cluster.AffinityPropagation.html

[11] https://www.npmjs.com/package/affinity-propagation