Improving Restaurant Recommendations on Yelp

Gil Rosen
Graduate School of Business
Stanford University
gilrosen@stanford.edu

Tuan Nguyen
Department of Statistics
Stanford University
tuanminh@stanford.edu

1 Introduction

Yelp reviews, while useful as a general measure for a given business suffer from a common fate of most review systems; they are not segmented and as such review value of a business for any given individual is heavily diluted by other users with differing preferences. Furthermore, while star ratings are generally useful, there is an abundance of information available in the textual content of the rating itself and the sentiment information available therein.

This project aims to explore different methods for building custom business scoring and recommendation based on either star ratings alone, or a star rating and textual review combination. Accordingly, we devise and assess models which may predict, with appropriate accuracy, the likelihood with which a user will like a given business through a predicted star rating. Predictions will be based upon users' prior rating history on Yelp - star ratings and textual reviews. The most appropriate models would both be accurate in their prediction, as well as applicable to a large range of users.

2 Related work

Previous work by Jack Linshi demonstrated effectiveness of the common hypothesis that star rating is justified by its corresponding text review [2]. In particular, it has been shown that running Latent Dirichlet Allocation (LDA) model on corpuses conditioned on the star rating levels yield much more informative semantic and sentiment topical aspects with regard to a business. This project takes this idea further by investigating how one can identify a users preferences based on his/her existing reviews conditional on star ratings and take into account these preferences when making recommendations of a product/service/brand that are relevant for this specific user.

3 Technical Approach and Models

3.1 Approach

We leverage two approaches in building our model, a direct and indirect approach. In our direct approach we build a K means model from the corpus of reviews which provide user ids, business ids and ratings. Here we may directly identify the businesses that users like or dislike and create clusters of users with similar affinities. We may then assign a user to the cluster their prior rating history is closest to, and we may predict and recommend positively or negatively, businesses not visited by the user which are likewise strongly liked or disliked by others in the same cluster.

In our indirect approach we assume that there exist multiple latent variables represented as topics, which define the characteristics of a given user preference for businesses. We then use LDA in conjunction with the star rating to extract topics from the review from which we may generate user and business profiles. Users with topic vectors which closely match business vectors, both positively and negatively, can then be leveraged to anticipate user sentiment and predicted star rating for a given business.

3.2 K-Means Model

K means model is a clustering process whereby clusters are initially randomly chosen, and each record is matched to the cluster which is most similar - as defined by the minimum norm of the record and any given cluster. Once complete, the clusters are redefined as the mean of the records in each cluster and the records are then again matched to the most similar cluster. This is repeated until convergence.

To properly execute K means we first needed to define what a record would look like. As our aim was to recommend businesses for a given user, we defined a record as the total set of reviews for a given user across all restaurants - that is a matrix whose r rows represented each user, and which had d dimensions equal to the full set of up d possible restaurants. Each entry represented its star rating.

The end result would then be a set of reviewers, or users, whose affinities are most similar to one another, and where each cluster would represent the average affinities of that group for each restaurant. New users need only their restaurant history (or a portion thereof) to match with the most relevant cluster, and all other restaurants rated in the cluster which the user had not visited should be similarly rated by the user upon patronage.

Given that we withheld half of the users, upon training completion, we were able to take the test set of user records, remove the last 1500 businesses from their record set and match the users to a cluster based on the first 5300 businesses. We could then predict which other restaurants in the cluster they would like or dislike based on the cluster rating, and if they had indeed patronized those restaurants, we could compare the results and assess our prediction accuracy.

We leveraged Python to extract data from its JSON format and then Matlab to actually code and execute the K-Means algorithm. To normalize our data we subtracted 3 from the star rating yielding a score of 0 for those rated 3 stars, which was the same score we assigned to those businesses which had not been visited.

3.3 Latent Dirichlet Allocation

This section describes LDA model [1] as applied to our problem and the motivation to use reviews conditioned on star ratings as training data.

LDA is a generative probabilistic mixture model. The key assumption is that each review contains various topics, and words in the review are generated from those topics. Given a restaurant, its reviews contain a particular set of topics. We hypothesize that star ratings are justified by reviews, and therefore the ratings in aggregate determine a set of topics that describes a restaurant's features.

The generative process of LDA model is as follows:

- Given: Dirichlet distribution with K-d parameter vector α, where K is the number of topics (which can be determined experimentally)
- Given: Dirichlet distribution with V-d parameter vector η, where V is the number of vocabulary in the corpus
- For topic i from 1 to K
  - Draw a multinomial parameter vector $\phi_{t_i} \sim Dirichlet(\eta)$. This is the word distribution for topic i
- For review j from 1 to M
  - Draw a multinomial parameter vector $\theta \sim Dirichlet(\alpha)$. This is the topic distribution for review j
  - For word w in review j
    - Draw topic $t \sim Multinomial(\theta)$
    - Draw word $w \sim Multinomial(\phi_t)$

Since we also assume that review is justified by star rating, conditioning reviews on star rating level, i.e. segmenting reviews based on similar star ratings, would help reveal underlying sentiment aspects of the reviews. These sentiment aspects would be neutralized and lost if all reviews are considered together regardless of the star ratings.

```
{
"business_id": "_qopVQ6_Mz6W7-Pmbi56GQ",
"full_address": "1011 Washington Ave\nCarnegie, PA 15106",
"hours": {}, "open": true,
"categories": ["Automotive", "Auto Parts & Supplies"],
"city": "Carnegie",
"review_count": 3,
"name": "Advance Auto Parts",
"neighborhoods": [], "longitude": -80.0824517,
"state": "PA", "stars": 3.5,
"latitude": 40.3983526,
"attributes": {},
"type": "business"
}
```


Figure 1: Example of a business record

3.4 LDA Model training and testing

The training data as constructed segment reviews based on their star ratings and thus condition the reviews on the ratings. We use Python pandas library to perform the data segmentation. The training corpuses and dictionaries are constructed from the 1-star and 5-star review sets using the preprocessing utility in `gensim` package and the package's STOPWORDS to filter out frequently occurring words that do not contribute to either semantics or sentiment of a review. To build the models we use the `gensim` LdaModel to identify 20, 30, and 40 topics over 30 passes through the training corpuses.

Our prediction framework borrows idea from the Distributed Representation Model (Bengio et. al.). Specifically, we represent each restaurant by a topicvec which is a vector whose dimension is the same as that of the training dictionary. A topicvec is a vector whose entries are the sum probabilities of the topics of all the reviews from a restaurant reviews as output by our LDA models. Each user, similarly, is represented by a preference vector which is constructed the same way, using the user's reviews. A rating is computed using logistic function of a dot product between a user's topicvec and a restaurant's topicvec, scaled by a multiple of 5 to match the star rating scale. This can be justified by interpreting a star rating as a likelihood that a user will like the restaurant (on a scale of 1-5). Finally we compute the error using MSE between the predicted star rating and true star rating.

3.5 Dataset

Our dataset is constructed from Yelp Dataset Challenge dataset. Due to the limited time frame of the project, we focus on recommendation of restaurants only. From the Yelp dataset, we select business records whose category contains "Restaurants."

To ensure sufficient numbers of reviews for each restaurant in the training phase, we selected only businesses with at least 6 records for the K means approach and 50 records for the LDA approach. From the reviews set, we similarly restricted the set to reviewers with a minimum of 6 reviews for the K means approach and 50 reviews for the LDA approach.

While the original Yelp dataset contained 2.7M reviews across 86K businesses and 687K users, we trimmed it down to 33.7K reviews across 8.8K users across 6.8K restaurants and reviews which were either 1,2, or 4,5 stars for K means, and for the LDA approach we restricted ourselves to 300K 5 star reviews across 200K users and 32K businesses, and 50K 1 star reviews across 18K users and 5K businesses.

Finally we split the data into train and test sets. The K means model split the data evenly across reviewers into training and test sets, and for the test set we withheld 1/4 of the businesses for a given user to compare against predicted preferences.

For the LDA model we split the data into 75-25 training and test sets by business id and user id.

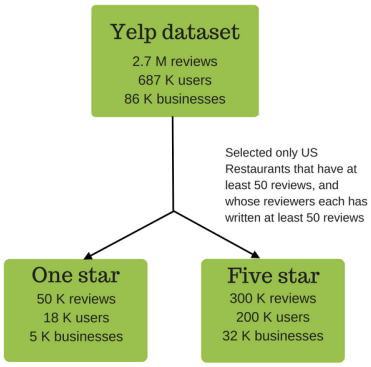

Figure 2: Data filtering process

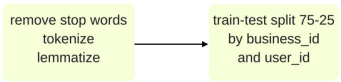

Figure 3: Data preparation process

4 Experimental Results

4.1 K Means Model

The number of clusters was determined experimentally as a function of time to convergence as well as overall accuracy. While trying cluster values of 30-70, convergence was achieved after 37 iterations for 30 clusters, through 57 iterations for 70 clusters. Timing however increased quite rapidly from 6.7 minutes for 30 clusters through 26.7 minutes for 70 clusters.

For datasets with fewer reviews per user, higher cluster numbers might increase the similarity across users of that same cluster, but at the same time would also reduce the number of potential businesses available to recommend as there would be fewer overlapping businesses in each group available for an accurate clustering, with still enough businesses that did not overlap allowing for prediction. Likewise for datasets with more reviews per user, we could afford to have more clusters with respect to accuracy, yet we suffered from performance which grew nearly exponentially.

50 clusters seemed an ideal compromise of performance with accuracy given a minimum of 6 reviews per user.

Once the cluster matrix was defined, we could then run the test set for user reviews against these clusters, matching on 78 percent of the business which users had reviews and then comparing the predicted reviews with their actual sentiments for the remaining 22 percent of businesses.

While there were of course numerous recommended businesses with predicted ratings that users had not patronized nor rated, and likewise numerous restaurants that users patronized and rated that had not been recommended one way or the other by their cluster, we found very impressive results for the overlap of predicted businesses which users did patronize.

For the overlap of businesses which we predicted a given positive or negative sentiment for a user in a cluster, that the user then indeed patronized, we had only a 3% error on average. That is if a user indeed visited and rated a business that their cluster had rated, the cluster provided correct sentiment 97% of the time.

Furthermore, of all the businesses a user visited, 22% were accurately predicted by their cluster, on average. This implied that even with a crude K means clustering, we could accurately predict over 20% of the businesses a user would visit and how they would feel about it thus succeeding in a more relevant and personalized rating system.

Below we see a figure representing for every user, the error in sentiment for the overlap of cluster predicted businesses and user rated business, as well as the % of total business actually patronized by a user that their cluster predicted both their patronage and their corresponding sentiment.

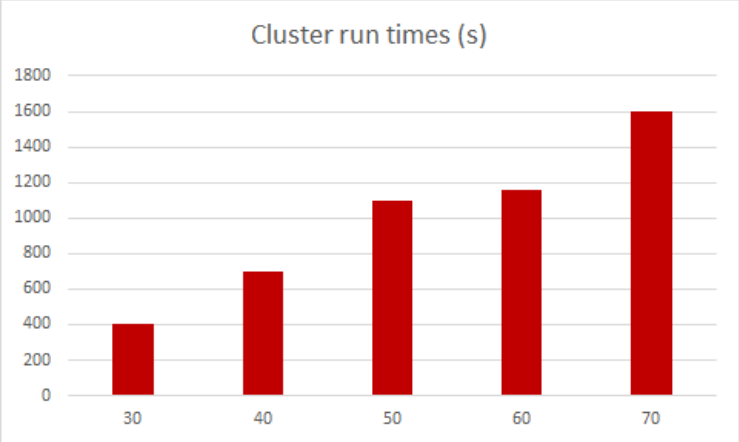

Figure 4: Cluster processing times

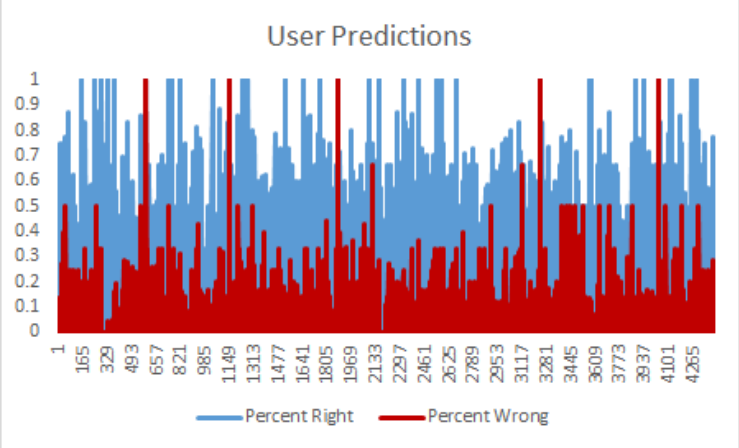

Figure 5: User review prediction and error

4.2 LDA Models

The number of topics k were determined experimentally. We trained LDA models using $k = 20, 30,$ and 40, each with 30 passes through the corpus.

LDA models converges quickly and we see stable results after 20 passes through the corpus. We use perplexity to determine the number of passes required. Perplexity measures how "perplexed" the trained model is on the testing set. As perplexity decreases, the model performs better on the test set. Figure 4 plots the perplexities of the three LDA models according to the passes through the data.

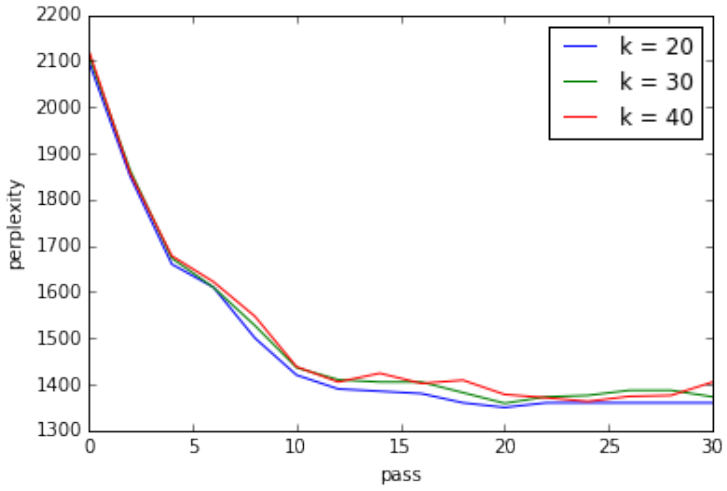

Figure 6: LDA perplexity plot for $k = 20, 30, 40$

Table 1 lists the topic words for 7 topics of those output by the 40-topic model. We note that the words capture well the negativity reflected by the one-star ratings, as evidenced by the presence of negative adjectives such as 'bad', 'terrible', 'horrible', etc. Table 2 lists the topic words for 7 topics of those output by 40-topic model. We note that the words capture well the positive reflected by the five-star ratings, as evidenced by the presence of positive adjectives such as 'good', 'amazing', 'great', etc. Figure 7 shows distribution of words for a sample topic from the 40-topic model.

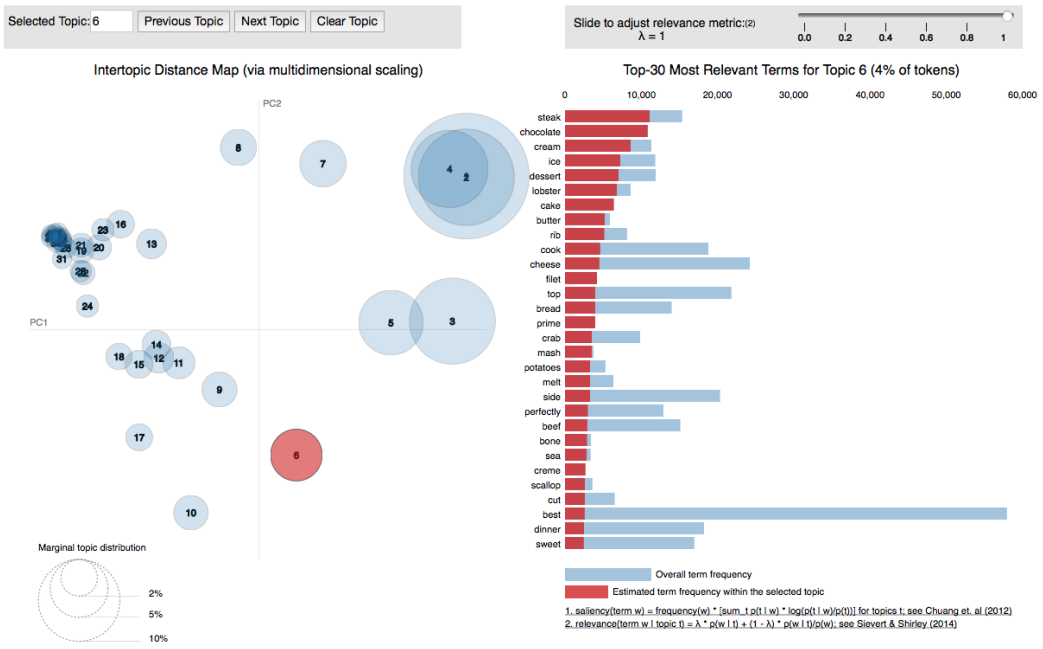

Figure 7: Sample word distribution

The topics generated by the LDA models indicate quite well features that can differentiate restaurants and users. Sample topics from our results describe Italian, Japanese, French, South American cuisines, among others.

Food	Bad service	Burger	Bad service	Bad service	Bad food	Japanese food
chicken	place	burger	food	order	wings	sushi
salad	table	fries	service	time	gyro	fish
sandwich	staff	bun	place	food	airport	place
dry	door	onion	bad	service	sauce	rolls
fries	dirty	bacon	worst	got	cockroach	buffet
lunch	customer	better	horrible	like	cuban	rice
meal	looked	hot	better	place	nasty	price
lettuce	left	pub	terrible	rude	smell	taste
cheese	bar	dog	money	wait	drenched	ordered
meat	going	frozen	location	want	msg	chef

Table 1: Topickmean words by the LDA Model on one star reviews

Good breakfast	Good Italian	Good bbq	Good bar	Good	Good food	Good Mexican
sandwich	pizza	bbq	place	great	salad	mexican
breakfast	good	wings	good	food	ordered	tacos
coffee	cheese	chicken	food	service	delicious	good
bread	italian	pork	bar	love	great	chips
eggs	best	good	great	best	fresh	salsa
toast	crust	fries	night	excellent	good	burrito
pancake	sauce	love	nice	amazing	cheese	best
eggs	love	ribs	happy	delicious	restaurant	taco
sub	fresh	best	service	awesome	cream	fish
fresh	pepperoni	mac	beer	prices	chicken	delicious

Table 2: Topic words by the LDA Model on five star reviews

Performance for LDA is evaluated using Mean Square Error between a user's rating from the test set and the predicted rating. $k = 30$ performed reasonably, while $k = 40$ which took quite longer to train, only improved marginally. We see that on average, the predicted star ratings using the LDA with 40 topics deviate from true ratings (which were only 5 stars since we're trying to predict which restaurants a user would like) about 1.32 star. If 2.5 star is chosen as a threshold to determine a boolean "like" or "not like", then using our model we would be able to predict the outcome successfully. Table 3 shows the results of the LDA models. The optimal leading α and η are the maxima from the parameter vectors α and η as described above.

Number of topics	Optimal leading α	Optimal leading η	MSE
20	0.029	0.032	1.9365
30	0.031	0.029	1.4473
40	0.027	0.033	1.3189

Table 3: LDA result summary

5 Conclusion

LDA models trained on reviews conditioned on star ratings generate topics that can accurately describe restaurants and users. We observe that $k = 40$ topics gives reasonably good performance, indicated by MSE of 1.3189. Improvements could be made in the future regarding the choice of number of topics to train, which we would like to have a more theoretically based way to determine. Given recent success of the word2vec models [3] in natural language modeling, future investigation should explore connection between our LDA topicvec and word2vec. This could give us directions to combine the interpretability of the topics output by LDA and the modeling flexibility achieved by word2vec.

K-means provides a valuable measure of predicting user preferences for establishments with little error. Given the much larger set of available establishments (dimensions) to those reviewed by any given user, this provides ample opportunity for accurate recommendations. Such recommendations

could improve user experience as well as provide an opportunity for monetization in appropriately marketing specific businesses to specific users. Furthermore we see that even with low numbers of reviews per user, 6, we are able to provide valuable predicted metrics.

Perhaps a combination of both LDA and K-means could in theory be used to present users with a more accurate and customized star rating for their specific preferences.

References

[1] David M. Blei, Andrew Y. Ng, and Michael I. Jordan. Latent dirichlet allocation. *J. Mach. Learn. Res.*, 3:993–1022, March 2003.

[2] Jack Linshi. Personalizing yelp star ratings: a semantic topic modeling approach. 2014.

[3] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *CoRR*, abs/1301.3781, 2013.