Predicting User Ratings and Creating Business Recommendations on Yelp

William Zeng
wizeng@
stanford.edu

Shivaal Roy
shivaal@
stanford.edu

Michelle Guo
mguo95@
stanford.edu

December 12, 2016

1 Introduction & Motivation

Yelp is a user-to-business review platform where users are able to rate businesses, primarily restaurants, that they have visited. With 3 million claimed businesses, 150 million monthly users, and over 100 million reviews, there is a rich amount of information that can be utilized to provide information to users about nearby businesses. Users often reference Yelp when trying to choose which restaurant to eat at, making a restaurant's rating and previous reviews a major factor in the decision. However, another part of Yelp that influences users' decisions on where to go are restaurant recommendations provided by the app. Thus, creating pertinent restaurant recommendations for users is in the best interests of both businesses and users.

2 Problem Definition

We sought to leverage information-rich features of the Yelp Challenge Dataset, such as the bipartite user-to-business graph created from reviews and the information available about a specific business, in order to provide an accurate recommendation system for Yelp users. Specifically, we attempt to solve two problems: 1) predict the existence of a link (review) between a user and restaurant and 2) recommend a restaurant to a user given minimal information about the user's preferences (partial cold start problem).

3 Related Work

3.1 Evaluation of Item-Based Top-N Recommendation Algorithms (Karypis 2001)

Karypis [1] presents an item-based model of recommendation when most of the commercial recommendation systems were predominately user-based. Item-based recommendation systems first

compute item similarity by looking at how items are related in a user history, and then recommend items most similar to the user's current items. The author shows that item-based recommendation was up to 28 times faster than user-based recommendation, and provided recommendations that were up to 27% better. In a Yelp context, this paper's results infer that it would be more effective to recommend a restaurant based off a user's past reviewed restaurants than finding similar users and recommending common restaurants they visited.

Since restaurant recommendation is the primary objective of this project, this paper provides a good high-level overview of a recommendation algorithm that leverages information about the users and items (restaurants).

3.2 Supervised Random Walks: Predicting and Recommending Links in Social Networks (Backstrom & Leskovec 2011)

In this paper, Backstrom and Leskovec [3] propose an algorithm for predicting links in a network. Specifically, they look at combining information about the network structure with information stored in the nodes and edges of the graph at some time t in the graph to predict edges between nodes at some later time t'. They develop a supervised random walk algorithm, where they guide a random walk through the graph by assigning probabilities to edges in the graph. They aim to directly learn the function that assigns weights to edges. They formulate a constrained optimization problem, where they seek to find the optimal weights of w of the edge strength function $f_w(\psi_{uv})$ by solving

$$\min_w F(w) = ||w||^2$$

with the restraint

$$\forall d \in D, l \in L : p_l < p_d$$

Here D is the set of *destination nodes* that a given node s has links to. L is the set of *no-link nodes* that s does not have links to. p_l is the probability assigned to an edge going from s to a node in L, and p_d is the probability assigned to an edge going from s to a node in D. Thus the function should assign higher probabilities to edges going to nodes in D than to edges going to nodes in L. This initial problem is a hard optimization problem, so the optimization problem is relaxed to

$$\min_w F(w) = ||w||^2 + \lambda \sum_{d \in D, l \in L} h(p_l - p_d)$$

where $h(\cdot)$ is a loss function. $h(\cdot) = 0$ when $p_l - p_d < 0$, and $h(\cdot) > 0$ when $p_l - p_d > 0$.
They manually define features that incorporate both network structure (e.g. number of common neighbors) and node and edge attribute information (e.g. cosine similarity between title's of papers written). Since this problem is not convex, they use several different starting points to find a good solution. The constructed supervised random walk algorithm is not limited to link prediction, and can be applied to many applications that require learning to rank nodes in a graph, such as recommendations and missing links.

Because the Yelp dataset has a fair amount of structure to it (users, restaurants, and links between the two groups), this paper provides a promising potential method that we could implement on our data to recommend restaurants to a user.

3.3 Topic-Sensitive PageRank (Haveliwala 2002)

Haveliwala [2] introduces a modified version of the original PageRank algorithm which precomputes a set of PageRank vectors based on a set of representative topics, allowing us to compute relative importance of Web pages with respect to a target set of preferential topics. For each query, the relevance of each topic to the query is computed, and the output PageRank vector is a weighted linear combination of the topic PageRank vectors based on the relevance. Haveliwala shows that more accurate results can be achieved by computing query-specific importance score, while still maintaining fast querying speed due to still having the precomputation from PageRank.

In this paper, each topic PageRank vector is created through a personalized PageRank. In Yelp, since we're looking to recommend restaurants relevant to a given user, we can treat that user as a topic and perform a personalized PageRank on each user. Here, since we only have one topic for each set of recommendations, we dont need to linearly combine multiple topic-specific PageRank vectors.

4 Models & Algorithms

4.1 Data Collection

The Yelp Challenge Dataset contains business and user details, reviews, friendship network information, check-ins, and tips. We discard the friendship network, check-ins, and tips, and instead only focused on businesses, users, and reviews. Since this dataset covers several cities, we filtered the businesses to include only those that are in Las Vegas and that are restaurants.

We further narrowed the dataset by only considering reviews above a certain number of stars and users who've written above a certain threshold number of reviews. We're only able to leverage positive reviews when making recommendations, and we are able to make better inferences and have a clearer graph structure when we only consider users who have written many reviews. We experimented with different combinations of minimum numbers of stars (`min_stars`) and minimum review counts (`min_reviews`); for example, after filtering on reviews 4 stars and above and users with more than 100 of those reviews, we were left with 12077 reviews written by 76 users on 2475 restaurants. After this filtering, we created a bipartite, user-to-business directed graph, where an edge between user u and business b corresponds to u writing a positive review on b.

4.2 Models

In our milestone, we implemented an items-based recommender system, which essentially provides the top k restaurant recommendations to users based on their past history of reviews. The recommendations were generated by first computing the average feature vector for all the restaurants in the user's review set, and then comparing the distance between this average feature vector with all other restaurants in the graph, where the feature set was a vector of the attributes of each business as booleans or categorical variables. Some attributes included the cuisine(s) of the restaurant, the price range, and different crowds that the restaurant was 'Good For'. However, we found that the accuracy of this system was quite poor. For each user, we removed one of their reviewed

restaurants randomly, and checked to see if this deleted link was contained among the top 10 recommendations that we provided for that user. With this evaluation metric, we predicted a deleted link only 0.8% of the time averaged over all users. We realized that this approach didn't utilize the graph structure between users and businesses at all, which we believed would improve our accuracy, so we decided to use a personalized PageRank algorithm to provide recommendations.

We utilized the Yelp network structure (provided by the bipartite user to business graph) to better understand users' restaurant preferences. From this understanding, we can provide business recommendations to users based on restaurants they have already reviewed. All three of our implementations utilize modifications of the PageRank algorithm, which is applicable as a recommendation algorithm because the restaurants with the top k highest PageRank scores can be the top k recommendations given to the user. By using PageRank, we could even provide recommendations to users without any previous reviews, which is applicable in a partial or complete cold start problem where we don't have much information in the user. Our first method predicts missing links in a user-business graph using a personalized PageRank algorithm in order to provide recommendations. The second converted the user-business graph into a weighted, undirected, business-to-business graph, and then used power-iteration to determine the PageRank scores. Finally, the third method tackles the cold-start problem by providing recommendations for restaurants in a specific genre without any information about the user.

4.3 Algorithms

In our first method, we offered 10 recommendations for each user, which were the 10 restaurants with the highest PageRank scores. We ran a personalized PageRank with random walk by using the set of restaurants the user reviewed previously as the teleport set. In this approach, we kept the original user-to-business bipartite graph. Since it's bipartite, the random walk goes from user to business then from business to user, but it only scored the business nodes. For evaluation, we deleted an edge between a user and a business they reviewed, thus removing it from the teleport set, and checked to see whether the 10 recommendations encompassed the removed business.

In our second method, we wanted to convert the original user-business graph into a direct business-business graph. To do this, we created an edge between two restaurants if there was a user who reviewed both of them, with the weight of the edge corresponding to the number of users who reviewed both restaurants. We created an adjacency matrix, where the entry in row i and column j indicates the number of users who reviewed both i and j. The teleport set for a user is all the restaurants they reviewed. By factoring the teleport set into the adjacency matrix and normalizing each row, we get the transition probabilities matrix. We then run power-iteration on this matrix to determine the PageRank scores and thus the top 10 recommendations. One benefit of this approach is by using power-iteration, the PageRank vector is guaranteed to converge. We evaluate this approach the same as the first one. We remove a user's review, which deletes one weight from the adjacency matrix, then run the algorithm. We then check if the deleted business is within the top 10 recommendations given by the algorithm.

Finally, we wanted to address the cold-start problem where we don't have much information

about the user, including any past reviews. We specifically framed it as a new Yelp user trying to find a restaurant in a specific genre. For example, if a new user was searching for Korean food, we want to return some Korean restaurant recommendations without any prior knowledge of the user. To do this, we ran our PageRank algorithm in our first part on the user-business graph with random teleportation since there were no previous reviews. We then sorted the restaurants by decreasing PageRank scores and filtered by the genre in the query, and return the top 10 results. For evaluation, since there are no previous reviews to compare against, we will manually evaluate the quality of each recommendation based on the rating, number of reviews, and review content for each restaurant.

5 Results and Findings

5.1 Initial Findings

We ran our first algorithm on three sets of data. The first dataset filtered reviews with only 3 stars and above and users with over 50 reviews. The second dataset also only considered reviews of 3 stars and above, but looked at users with at least 100 reviews. The last considered reviews of 4 stars and above and also only users with at least 100 reviews. The accuracies reported are actually an average of 50 trials for each value of n_iters and alpha.

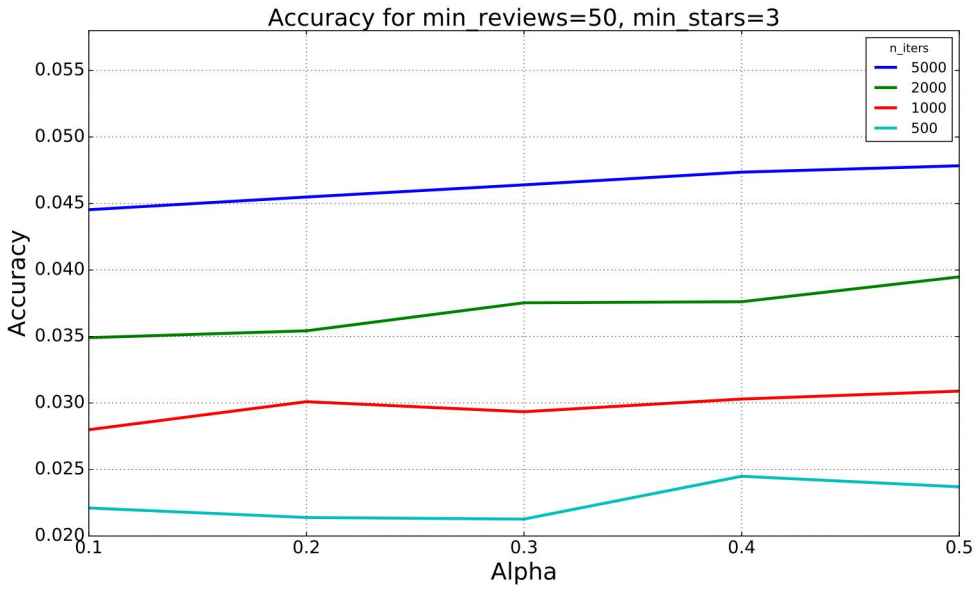

Figure 1: Accuracy for 50 minimum reviews and 3.0 minimum star rating

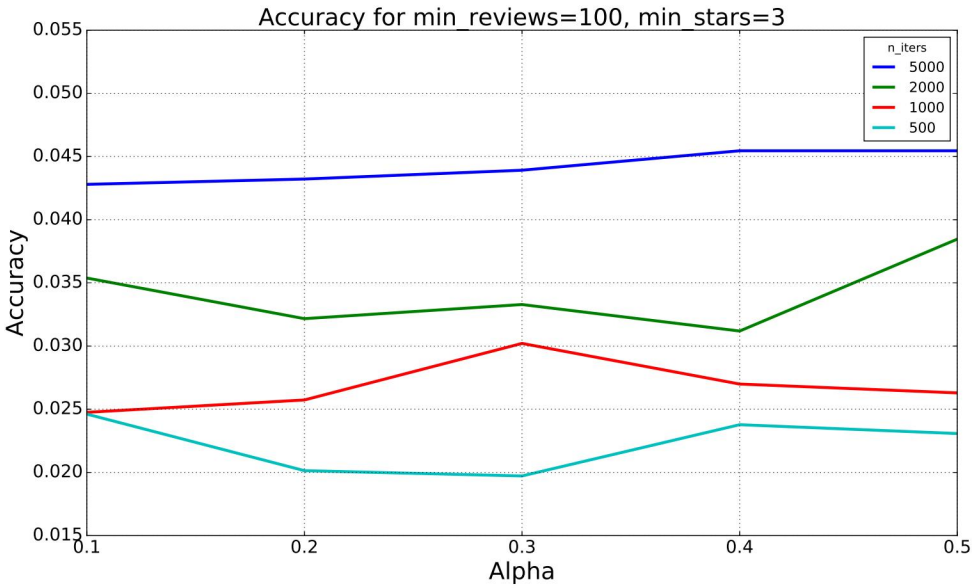

Figure 2: Accuracy for 100 minimum reviews and 3.0 minimum star rating

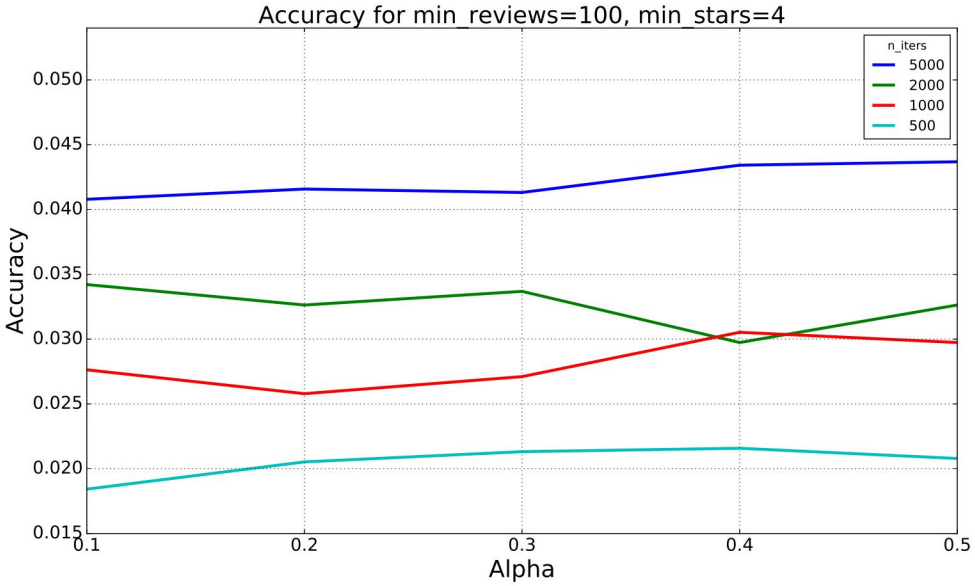

Figure 3: Accuracy for 100 minimum reviews and 4.0 minimum star rating

Across these graphs we see that increasing the number of iterations for our personalized PageRank algorithm increased accuracy greatly, while changing the alpha (probability of teleporting to a node in the teleport set) only marginally changed the accuracy. The increase we see with number of iterations can be explained by the fact that we are better learning the true probability distribution of a user going to a restaurant they have not seen before. Ideally we would find the number of iter-

ations where accuracy no longer increases, but even at 5000 iterations, our program took multiple hours to run and so was not feasible to explore higher iteration counts.

Given the large number of restaurants in the Las Vegas area, predicting the missing link 5% of the time with 10 recommendations is pretty good. Since the edges are deleted at random, it is possible that the edge deleted was to a restaurant that the user doesn't normally go to and isn't super popular, which would make it hard to predict that missing link.

Our top 5 recommendation results for the partial cold start problem are shown in Table 1. When providing Italian cuisine recommendations to a Yelp user who lacks experience in reviewing Italian cuisine, we provide reasonable recommendations by using the user's reviewed restaurants as the starting preference set. We obtain Italian restaurant recommendations with high star ratings and large review counts, showing that our approach is capable of solving a partial cold start problem reasonably well.

Table 1: Partial Cold Start for Italian Cuisine

Business	# Stars	# Reviews
Settebello Pizzeria Napoletana	4.0	256
Cugino's Italian Deli & Pizza	4.5	460
Nora's Italian Cuisine	4.0	839
The Bootlegger Italian Bistro	4.0	542
Bambino's East Coast Pizzeria	4.5	153

5.2 Difficulties

The primary limitation of this project is that there is inherently less graph structure in the Yelp dataset than with a social network. Users don't necessarily review new restaurants that are related to past restaurants they visited, since they could just be trying different top rated restaurants of different genres in totally different locations. Whereas in a social network most new links form with a friend of a friend, the same clustering is not seen in a Yelp dataset as much.

Furthermore, it was somewhat difficult to come up with the evaluation metric. There isn't a clear-cut way to measure the accuracy of recommendations as there is with most machine learning problems where there is a defined output. Instead, we tried to evaluate the accuracy both by hand and with the missing link prediction problem, where we deleted an edge and tried to have the algorithm predict it.

5.3 Future Work

There is a lot of very rich metadata in the nodes and edges of the graph we didn't utilize in our PageRank algorithm, which only looked at the graph structure. In terms of users, we could utilize the friend network they have with other users. For edges, which are the reviews, we would like to incorporate the Hidden Factors and Topics model, in which we are able to connect latent textual topics from user reviews and latent factors in order to predict ratings for businesses that a user has

not yet reviewed. This model utilizes the actual content of the reviews themselves. Finally, there is a lot of data about the business that could determine similarity between businesses, such as the geographic proximity of the two restaurants, the difference in the feature sets that provide details about each restaurant, etc. The number of check-ins a user has to a restaurant could also be utilized.

For the HFT model, we believe that this is one step further beyond our main goal, as predicting ratings will implicitly allow us to provide restaurant recommendations to users. The HFT model proposed in McAuley and Leskovec focused on Latent Dirichlet Allocation (LDA), however we are interested in trying to use probabilistic Latent Semantic Analysis (pSLA) model, which is an approach used in Sivic et al. Although Sivic et al. showed that the two models provide similar results, pLSA is much simpler to implement and would be interesting to experiment with in combination with latent-factor recommender systems in the context of HFT.

We can also consider common usage behavior of Yelp to add more real-world features to our model. Most avid Yelpers generally consult Yelp before going to a restaurant, and thus usually won't end up going to poorly-reviewed businesses. Thus, for people who often use Yelp, going to a restaurant and writing a positive experience are correlated. However, another class of Yelp users may not use the app often and may not consult it before visiting a restaurant. If they have such an extremely poor experience at the restaurant, then they are likely to look it up on Yelp afterwards and write a poor review. In real-life use cases, Yelp users can normally be divided into those who consistently write Yelp reviews versus those who only write them when they have an extremely poor or good experience. Thus, we can take this into consideration when analyzing a Yelper's review history and define their profile as one of the two above-mentioned categories.

References

[1] G. Karypis. Evaluation of Item-Based Top-N Recommendation Algorithms. Proceedings of the tenth international conference on Information and knowledge management. 247-254, 2001.

[2] T. H. Haveliwala. Topic-Sensitive PageRank. 11th International World Wide Web Conference, 2002.

[3] L. Backstrom, J. Leskovec. Supervised Random Walks: Predicting and Recommending Links in Social Networks. In Proc. WSDM, 2011.

Michelle: paper writing, centroid algorithm implementation, cold start implementation
Shivaal: paper writing, plotting analysis graphs, pagerank implementation
Will: paper writing, data processing, creating the algorithms