Generating place recommendations for travelers

Team Members: Andy Gilbert (adgil), Andrew Hilger (amhilger)

Motivation and Overview

As vacation time becomes more precious, travelers have less time to plan their vacations. Services such as Yelp as useful for finding good restaurants and other attractions. However, these businesses are primarily reviewed by locals and don't necessarily account for the differing needs and tastes of travelers. For example, users in different cities have varying preferences, so visitors likely have different preferences from locals. As another example, places rated highly by locals may feel hostile to tourists. We expect that differentiating between reviews by locals and reviews by travelers will improve recommendations.

To make recommendations, we implemented collaborative filtering with sparse factor analysis. The model predicts business ratings by a user using a set of reviews by that user for other businesses, as well as other user's reviews of those businesses. We trained separate models for traveler reviews and local reviews. We evaluated these models' performance against a control model trained on both the tourist and local reviews. When evaluated on a test set, the trained model predicted local review scores with a mean absolute error (MAE) of 0.909 and traveler review scores with an MAE of 1.205, which was an improvement of 0.07 relative to the control model.

Dataset

We used the latest data release from the Yelp dataset challenge [1]. The dataset includes 1.18 million users, 4.73 million reviews, and 156 thousand businesses. The Yelp dataset does not label reviews as being "tourists" or "locals," nor does it provide any information about the user's home location. The business data includes each business's latitude/longitude and city/state, but this city/state data is not useful by itself for two reasons. First, the city and state address data are often inconsistent with the business's physical location given by (lat, long) coordinates. Second, a user from Palo Alto who "travels" to a business in Menlo Park shouldn't be considered a tourist.

To differentiate between tourists and locals, we associated each user and business with a metro area. We then identified "tourist" reviews written by users with a home metro area differing from the business's metro area. For our purposes, a metropolitan area is an economically and geographically connected region (e.g., the San Francisco Bay Area). The Yelp dataset consisted mainly of reviews from eleven different metro areas: Las Vegas, NV; Cleveland, OH; Pittsburgh, PA; Phoenix, AZ; Champaign, IL; Toronto, Canada; Stuttgart, Germany; Madison, WI; Montreal, Canada; Edinburgh, Scotland; and Charlotte, NC. We removed all users with less than 20 reviews given and businesses with less than 30 reviews received in order to have enough data for the collaborative filtering. This left us with 31 thousand businesses and 251 thousand users (~20% of original size in both cases).

Methods

<u>Clustering to Determine Metropolitan Areas</u>: The business data were filtered to remove businesses outside of states containing the 11 major metro areas in the dataset (<1% of the full dataset). We then performed k-means clustering to identify the metropolitan areas. The number of clusters *k* was set to 11 based on the expected number of metropolitan areas. Since the metro areas contained different numbers of businesses, the clustering was sensitive to the initialization of the cluster centroids. When two centroids were initialized too close to a city with a disproportionate number of businesses, that city captured two clusters. The initialization was tuned until the resulting cluster centroids corresponded to each visual cluster and no metropolitan area contained duplicate clusters. This was done by drawing one business randomly from each of the 11 states with

a major metro area as an initialization for the cluster positions. To cluster we used the scipy.cluster.kmeans2 function with 500 iterations.

Metropolitan-Area-Specific Features: We generated metropolitan-area specific features on a per-user basis. First, each review is linked to the corresponding user record and business record with the user ID and business ID in the review. We then aggregated reviews on a per-user basis to identify the following features for a user: (A) proportion of the users' reviews in each city; (B) proportion of the user's weeks on Yelp that the user has written a review in each metropolitan area, normalized by the user's total weeks on yelp. This latter feature avoided bias from users who reviewed more frequently while traveling than while at home. Since there were 11 metro areas, this yielded 22 features per user.

Clustering to Determine User Groups: We used k-means clustering with the metro-area-specific features to associate users with one of the metro areas. We used k = 11, the same number of clusters as in the first clustering step. We initialized one cluster for each metro area by initializing the proportion of reviews in that metro area to 100% and 0% for all others. We used scipy.cluster.kmeans2 with 600 iterations.

Collaborative Filtering: We implemented collaborative filtering using a sparse factor analysis as described in [2]. Given a set of reviews $Y_{n\times m}$ for m users who have reviewed n businesses, factor analysis identifies k latent factors that explain the observed reviews according to $Y = \Lambda X + N$ where each row of $\Lambda_{n\times k}$ represents the factor loading of a business, and each column $X_{k\times m}$ represents the factor preferences of a user. N represents additive noise.
We implemented expectation maximization in Python, The E step determines X for each user, and the M step determines Λ and an estimate of noise variance ψ for all the businesses. This model provides better scalability than a Pearson-correlation coefficient-based model or a personality-diagnosis model, which accounts for user-specific features [3].
We evaluate the model's performance using mean absolute error (MAE) between the actual star ratings and the star ratings predicted using the learned Λ and X, where the predicted star rating has a floor of 1 star and a ceiling of 5 stars.

Results

Business Location Filtering: Once we visualized the data, we realized the business location data is inconsistent. For example, many businesses were listed as being located in Alabama, in spite of having latitude/longitude coordinates in completely different states. This is likely because users/business owners set up a business's profile and are too lazy to correct the state drop-down menu from the default: Alabama. We assumed that the latitude/longitude coordinates were correct because these map directly the business's map location, which is the aspect most likely to be accurate given that users will report an inaccurate location before anything else.

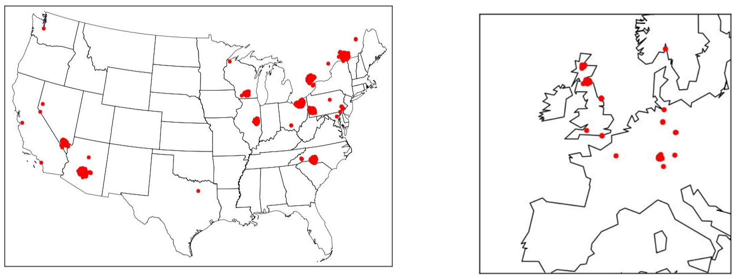

Figure 1: Initial dataset before filtering. Red dots are business locations.

Accordingly, we used the following filters: (1) removed businesses in states/countries that did not have a metropolitan area cluster. (2) Cross-referenced latitude/longitude with state and removed businesses where there was a mismatch. This latter filter caused problems with the Charlotte, NC dataset because the southwest corner of the metropolitan area is in South Carolina, so we allowed South Carolina data to pass the filter even though South Carolina doesn't contain a cluster centroid.

Business Clustering: After using the initialization strategy described above, the final results of business clustering (Fig. 2) looked very good based off of visual inspection and comparing latitude/longitude coordinates of clusters to city centers from Google Maps.

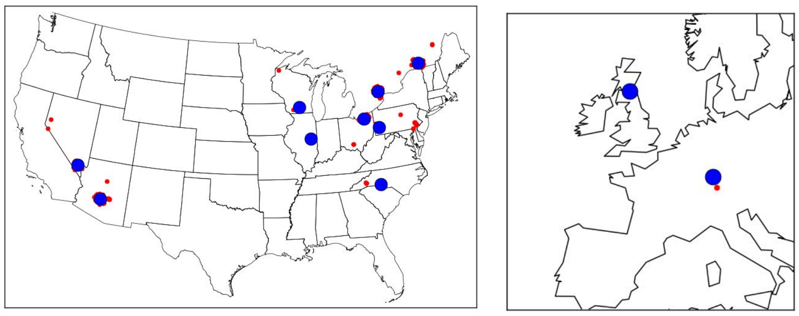

Figure 2: Clustering results on filtered business data. Blue dots are cluster centroids.

User Clustering: Similarly, user group clustering was sensitive to initialization, but after using the initialization strategy described above (initialization based on percentage of reviews in each metro area) the user clustering also worked well. The RMSE for Las Vegas was .025 and the total RMSE was .037. The histogram of error norms is shown in Fig. 3. As shown the vast majority of users are not far away from their cluster center with some outliers (more frequent travelers).

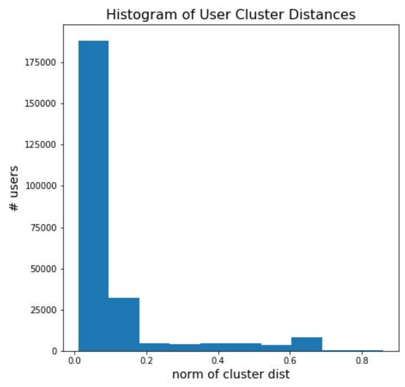

Figure 3: Histogram of distance norms between user groups and cluster centers.

Collaborative Filtering: To ensure computational feasibility, we considered only reviews of 30 thousand businesses in the Las Vegas metro area. After filtering only users with more than 20 reviews and businesses with more than 30 reviews we had 146,000 users, including 134,000 Las Vegas locals and 12,200 tourists from other metro areas. We trained using k = {3, 4, 5, 6, 8, 10, 12, 15} for the separate tourist and local data sets, and using k = {3, 4, 6, 8, 10} for the combined data sets.

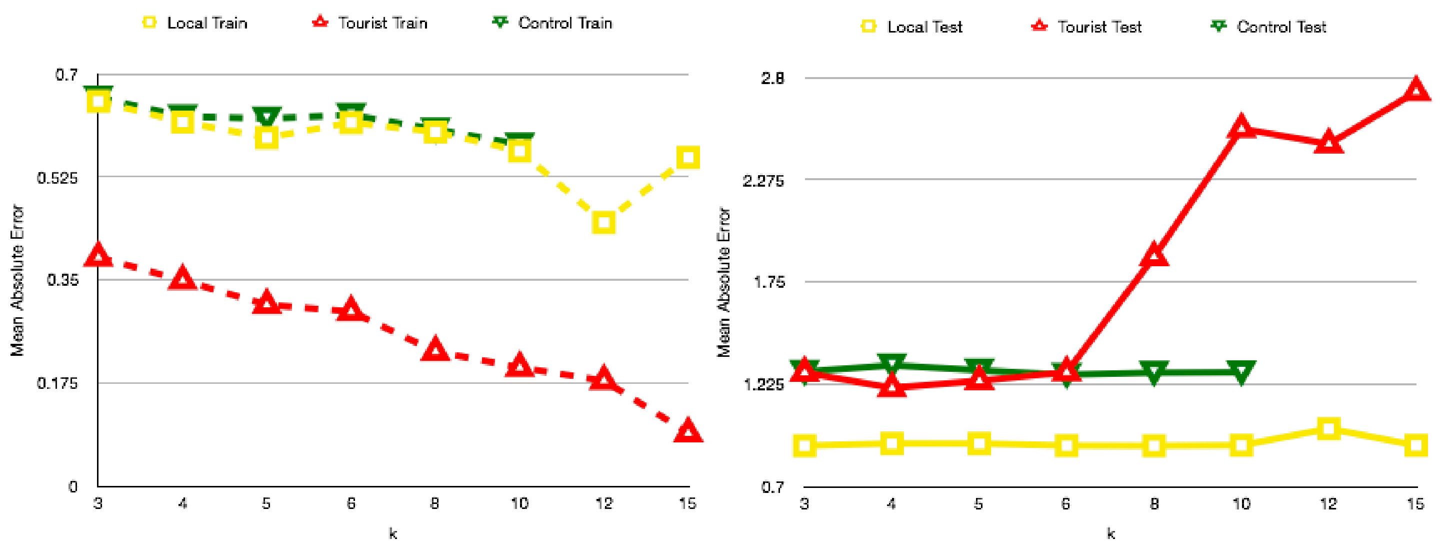

Figure 4: Mean absolute errors for training sets (left) and test sets (right) over range of hyperparameter k.

By varying the hyper-parameter k (the dimensionality of the user's modeled canonical preferences), we see that increasing the hyper-parameter beyond 6 reduces the training MAE while worsening the test MAE. In other words, excessive values of k encourage overfitting. Since the per iteration training complexity is O(mnk^2) [2], training with an excessively high k is particularly costly.

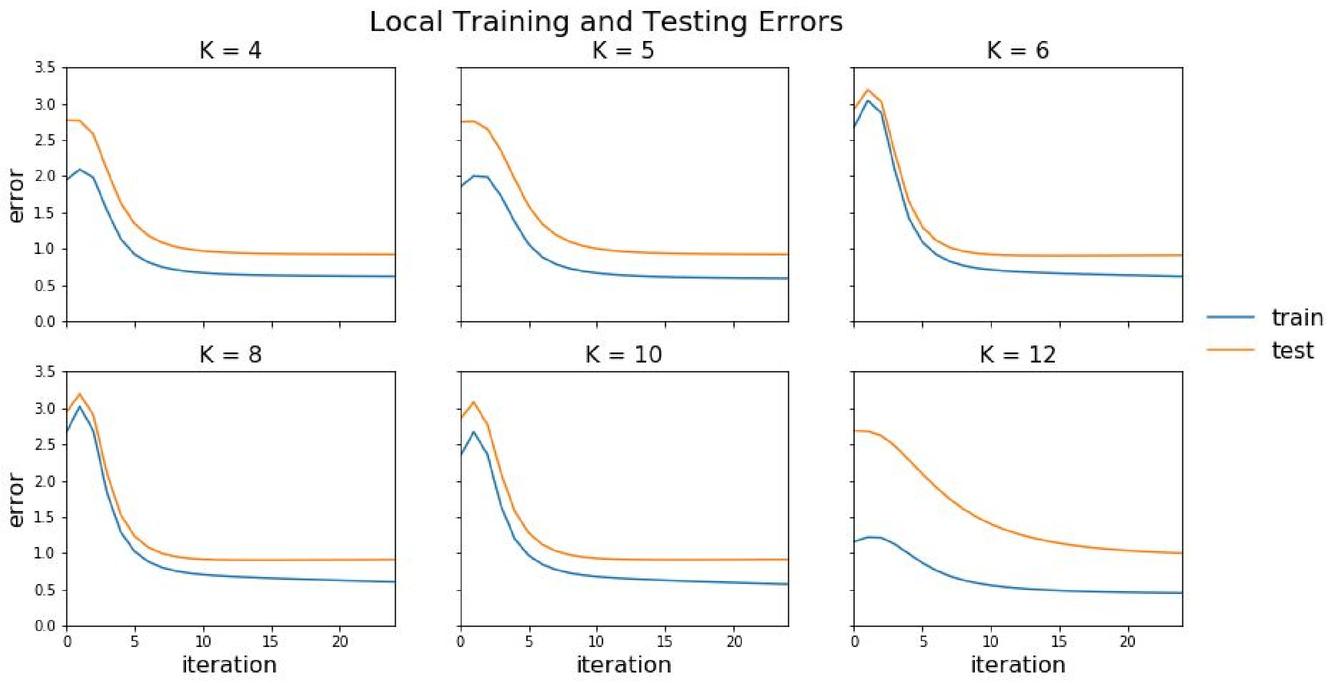

Figure 5: Learning curve MAE of training set (blue) and test set (yellow) for locals. The combined and tourist datasets had similar results.

The learning curves in Figure 5 illustrate that the model converged in 15-20 iterations for most of the data sets. Although the training MAE continued to decrease with more iterations, the test MAE stagnated or even worsened between iterations 20 through 25 indicating overfitting was occuring beyond this point. This pattern is replicated in learning curves for other values of k across the tourist only, local only, and control models.

Data Set (m = # users)	Optimal k	Training MAE (80% of n,m)	Test MAE (20% of n, m)
Tourists + Locals (control); m= 146k	6	0.6304	1.273
Tourists Only; m = 12k	4	0.3492	1.205
Locals Only; m = 134k	6	0.6174	0.909

Table 1: Collaborative filtering results for three training sets. Note that the combined dataset was tested on the same subset of test data as the tourists data set.

Conclusion

Tourist reviews were best modeled with k = 4 canonical preferences, while local reviews were best modeled with k = 6 canonical preferences. One interpretation is that tourists have less sophisticated preferences than locals. Alternatively, it is more advantageous to use lower k because there are fewer tourist reviews, so a lower k is required to prevent overfitting. Training separately on tourists and locals improved predictions for tourists relative to the control model, but the improvement was not significant (<0.1 improvement in MAE). The biggest limitation on the collaborative filtering model is lack of tourist reviews. Given access to the full dataset, we could train a more robust model that would likely be able to use a higher k value without overfitting (i.e. k = 6 instead of k = 4).

Given additional time, we would compare performance of tourist/local-aware collaborative filtering across all metro areas to investigate differences in tourists visiting different cities. We would also investigate training tourists separately by tourist home metro area to see whether tourist preferences vary across home metro areas. However, this would require significantly more reviews (at least a couple orders of magnitude).

Team Member Contributions: Andy worked on pre-processing and visualizing location data associated with businesses. Andrew wrote functions to link review data with corresponding user and business data, as well as generating metro-location specific features. Andy implemented clustering to link cities to metropolitan areas, completed debugging the metro-location specific features, and performed clustering of the users with the metro-location specific features. Andrew implemented the initial collaborative filtering code. We both debugged the collaborative filtering code, co-drafted the poster and this writeup, and presented the poster.

References

(1) “Yelp Dataset Documentation”. Yelp. https://www.yelp.com/dataset/documentation/json
(2) Canny, J. “Collaborative Filtering with Privacy via Factor Analysis.” *SIGIR* 2002,August 11-15, 2002, Tampere, Finland. <https://people.eecs.berkeley.edu/~jfc/papers/02/SIGIR02.pdf>.
(3) Su, X. and T. Khoshgoftaar. “A Survey of Collaborative Filtering Techniques” *Advances in Artificial Intelligence*, 2009, 421425 <https://www.hindawi.com/journals/aai/2009/421425/>
(4) “JSON to CSV Converter”. Scott Clark
https://github.com/Yelp/dataset-examples/blob/master/json_to_csv_converter.py