EMSE6586 Final Project-User Analysis Part

For MySQL database it’s easier to query back all the unique reviewers (since there’s a specific reviewer table, but the insertion took a lot more time comparing to MongoDB, which I could directly dump the whole json file in, only took me less than 10 mins)

Therefore I made a list of all the users and shuffled it and sampled 5000 out of it. And use the 5000 unique reviewer ID to query the products they reviewed and the ratings they gave (for both MySQL and MongoDB)

My philosophy: make a list of the reviewers and their reviews, in which reviewers’ ID are the keys and their reviews are the values. However, reviews is also a dictionary, in which the asin is the keys and the rating(s) is the value (My initial thought was that a person could rate the same book multiple times so I made the value a list of ratings but it seems like that didn’t happen—one reviewer only reviewed one book once), I choose dictionary because I want to intuitively see what a certain/specific user has reviewed and the ratings.

MySQL is much faster when executing this task. It took MySQL several hours, but MongoDB multiple days to give me back the requested dictionary

Then I could directly go into the dictionary and look at the reviewers, the books they reviewed and the ratings they gave.

If I want to see the rating pattern of a specific reviewer, I could directly request the values corresponding to the reviewer ID as the key in the dictionary and directly look into the values which is the rating they gave to the books they reviewed. And some people didn’t give many ratings and within the few ratings they gave pretty high rating to every book, maybe they are just more prone to give high ratings. Some people have high average ratings, but within their ratings, they also gave mediocre or even low rating to several books, which might indicate that they are more objective. For those reviewers it’s more meaningful to look into the

Improvement or deviation:

Could do the same to a specific book, use book asin as the key and the reviews as the values. Within the values, use the reviewer ID as the keys and the rating as the value, and if a book’s average rating is low it means that many people don’t like it, thus it might not be well written or popular. In this case, for recommender systems, this book should be ranked lower

Could link asin databse (if accessible, but it’s not included in the data set) to the database, so you can look into the books that an objective reviewer gave bad rating and find the book and see what’s it about.

Could also analyze if a reviewer is more objective when rating, so that his/her rating scores are more representative of the quality of books or his or her reviews. In this case, if a reviewer is used to giving high ratings or low ratings, his or her reviews probably should be ranked lower in the review page while the more objective user’s rating should be ranked higher in the review page so that when someone is looking through review page of a book to see if he/she wants to buy it and read it, he/she could read the more objective reviews first to make more objective judgement about the book.