CSE158 Assignment 2

Wing Leung A16107662

Khoa Tran A14161521

Dave Jiang A15538726

QUESTION1:

- Data source: goodreads
- 200000 total reviews, 7169 unique books, 11357 unique users
  - Indicates that users on average made about 18 reviews
  - Each book reviewed approx 25.10 times
- Most Popular book: 362 reviews (slight outlier)
- Average rating given: 3.896
  - # of books above average = 144893
  - # of 4 star vs 5 star ratings VERY similar: 72436 vs 72457
  - Book with highest avg rating with over 25 reviews: 4.69
  - Book with lowest avg rating with over 25 reviews: 2.433
  - Average of MEAN ratings: 3.812 = similar to avg rating (should be)
  - # of books with avg rating over mean rating: 3254
  - # of books with avg rating less than mean rating: 3916

The dataset we chose to study contains entries of a particular user's rating for a specific book with data from Goodreads. In total, this dataset contains 200000 total ratings from 11357 unique users spanning 7169 unique book titles. On average, this indicates that each user made approximately 18 ratings and each book title received 25 ratings. With users and items both averaging a reasonable amount of interactions in which there can be overlaps amongst interactions between two users or two items, this became one of our motivations to build a model based on similarity (described later). Essentially, the greater the number of interactions an user or item has, the more data we have on it in which we are then able to use to compare with that of other users or items. In terms of the amount of ratings received, the most popular book had 362, which is a slight outlier for this measurement. Overall, the data was pretty well distributed in terms of the spread amongst the amount of ratings received books. This is important as the more evenly distributed a dataset is, the more likely it is for there to be an accurate representation (less heavily biased results, less outliers, more interactions per item and user). Taking into account all the ratings given, the average rating for a particular book was 3.896. More specifically, there were 144893 books that were rated higher than this average (four and five stars). One of the possible reasons for this is that more times than not, people tend to bother to rate things when they like something, and of course, give a positive rating. On the contrary, if a person has a bad experience with something, the lesser time they are willing to allocate to it let alone provide a rating or review for it. Thus, this can also help to explain why ratings of 1 and 2 (2434 and 7838) were given substantially less than that of 4 and 5. Perhaps one of the most interesting specs about this dataset is that the 144893 four and five star reviews are split by a difference of 21 (72436 four star, 72457 five star), only a 0.014 % difference. Realistically, this shows that there is not that much of a difference in people's minds between a four and five star book. Rather, four and five star ratings are viewed as more clustered together under the category of it being a good rating. When considering the average rating of individual books, we analyzed this data by taking the maximum and minimum averages only from books that received over the average number of ratings (25). This is to ensure that we are not at risk of possible outliers when it comes to this measurement. For example, a book that received 5 five star ratings will have an average of rating of 5. However, as the number of ratings for this book increases, it is unlikely for this book to continue receiving 5s on every rating. With this, the maximum and minimum average rating observed were 4.691 and 2.433, respectively. Lastly, it was observed that the average of all *mean* ratings (3.812) was very similar to just the average of all ratings (3.896), a 1.6% difference.

QUESTION 2:

Given a (user,book) pair as input, the primary predictive task of our model will be to decide if the user will read the book (1 if the user will read the book, 0 otherwise). Due to the binary nature of the classification task, we will stick with classification accuracy as our main method of measuring classification performance. Individual models will be evaluated by their performance on this metric on the validation set. Since the validation dataset is very balanced, we will not consider balanced error rate.Acting as the relevant baseline for our model is a simple popularity-based recommender system that determines if a user will read a book simply by checking if the book itself is popular. The highest performing baseline achieved for the predictive task was done by defining the most popular books as those with the highest number of interactions among the training set totaling up to 65% of all reviews (so the number of popular books is variable and dependent on the distribution of books in the training set). For reference, the popular prediction baseline had a classification accuracy of 65.38%.

Similarity-based recommender systems will comprise much of the experimentation and in particular, we will focus on utilizing Cosine Similarity, Jaccard Similarity, and Euclidean distance. Because we were able to explore the dataset and determine that the distribution of the data was already sufficient for being able to determine non-zero similarities between items and users, no further processing was necessary (a dataset with sparse interactions for each user and item would suffer drastically). Item-to-item and User-to-User similarity will both be considered unless we determine one to be clearly better performing.

Here are the equations of the three different similarity metrics we will use:

COSINE SIMILARITY:

$$\text{similarity} = \cos(\theta) = \frac{\mathbf{A} \cdot \mathbf{B}}{\|\mathbf{A}\|\|\mathbf{B}\|} = \frac{\sum\limits_{i=1}^{n} A_i B_i}{\sqrt{\sum\limits_{i=1}^{n} A_i^2}\sqrt{\sum\limits_{i=1}^{n} B_i^2}},$$

JACCARD SIMILARITY:

$$\text{Jaccard}(A, B) = \frac{|A \cap B|}{|A \cup B|}$$

EUCLIDEAN DISTANCE:

$$|U_i \setminus U_j| + |U_j \setminus U_i| = \|R_i - R_j\|$$

In addition, to incorporate both the similarity metric and the explicit feedback rating from the training set to estimate read prediction, collaborative filtering for rating prediction will be our last model.

Collaborative Filtering for Rating Prediction:

$$r(u,i) = \frac{1}{\underset{\text{Normalization constant}}{Z}} \sum_{\underbrace{j \in I_u \setminus \{i\}}_{\text{All items the user has rated other than } i}} r_{u,j} \cdot \mathrm{sim}(i,j)$$

$$Z = \sum_{j \in I_u \setminus \{i\}} \mathrm{sim}(i,j)$$

The rating prediction model will predict the user to read the book if the rating estimate exceeds some threshold.

QUESTION3:

Firstly, we will discuss the usefulness and implementations of a similarity-based recommender system built around either cosine similarity, jaccard similarity, or euclidean distance. If we consider the reading behavior of individual persons, it is often the case that people choose to read a book not necessarily because it is mainstream or popular, but because it aligns with the interests and topics that a reader has previously engaged in (someone who only reads fantasy books will likely to continue reading fantasy books, with popularity being only a secondary factor). Therefore, an effective approach to this problem is to simply compare a numerical value quantifying how similar the target book is to a book the user has consumed before, and judging a positive response (1) if that similarity exceeds some threshold.

However, our model will be dependent on the performance of three different similarity metrics that we need to optimize for performance. For each of these different similarity metrics, we first tested the performance of item-to-item and user-to-user comparison, and found that the former consistently gave more accurate predictions (e.g user-to-user similarity yielded a classification accuracy of 63.5% while item-item similarity scored 67.98% in one test run). So, we decided only to base our models around item-to-item similarity (items are compared by the set of users which interacted with them). Secondly, we tested the model's performance when we used mean similarity across all items interacted by the user against the model's performance when only the maximum similarity score was recorded, and found that mean similarity tended to score higher by an average 0.5-1% and thus decided to use mean similarity as a metric instead of max similarity across all models. Since the data does not take long to process and the results seemed to remain stable even for a large sample of 200k+, we figured that our model does not have any glaring issues with scalability and overfitting.

One alternative model we decided to experiment with was collaborative filtering for rating prediction because we figured that explicit ratings combined with the similarity metrics should provide more accurate information regarding a user's preferences and an item's popularity more than the similarity value itself. However, to optimize the rating prediction model, we decided to predict that a user would read a book if the predicted rating exceeds some threshold that is a function of the user's average rating for any game, since choosing an arbitrary value like rating > 4 tended to perform 0.5%-1% lower in terms of classification accuracy.

Although these models did all eventually end up performing better than a trivial (50%) predictor, there are a few key strengths and weaknesses which will inevitably explain the accuracy results we will report later for each of these models. Firstly, if we only consider the similarity-based models in isolation, it is clear that although these models will successfully identify books that are strongly correlated with the trends of a user's previous purchase history, it will fail to consider cases in which users read books simply because they are popular or trending. The model itself fails to capture the purchase habits of a typical reader (readers will often choose between two books that are related if one is more popular than the other). Therefore, to address this weakness, we will integrate popularity prediction into the similarity models with lower thresholds of popularity (so 28% of interactions instead of 65% of interactions in the baseline). So, if a book's similarity score exceeds a certain threshold or the book is popular enough, then a positive (1) prediction will be made, 0 otherwise.

Then, we will address the weaknesses of the rating prediction model. Although the rating calculation algorithm definitely scores better than a trivial predictor at a classification accuracy of 60.63%, its performance is still often poor when compared to the similarity or popularity based prediction models. One ostensible reason for this is that although rating is a somewhat good predictor of quality, it is not necessarily a strong predictor of how engaging or popular a book is. Highly rated-books can be extremely unpopular if the topic is niche enough, for example, and the low classification accuracy is likely an indication that the rating metric is simply not a strong enough metric for our read prediction task.

QUESTION4:

https://github.com/mick-zhang/GoodReads-Recommendation-using-Collaborative-Filtering/blob/master/Book%20Recommender%20System%20Github.ipynb?fbclid=IwAR0MHNe6jjas-9h_3YQbJrlPOuIwPOcNIDIgOPG6Rka94u-hY7stYb_ZJMU

When it comes to our dataset, I do not believe this specific dataset has been studied outside of our class, but we were able to find studies of very similar datasets that are also from GoodReads. One example (linked above) is of author Mick Zhang using a dataset whose main features are also user, item, and rating. In this study, Zhang is also trying to build a prediction / recommender system based on similarity between users using collaborative filtering and Euclidean distance. The size of the dataset Zhang uses also compares with that of ours in size, as Zhang's dataset consisted of roughly 220 thousand entries once filtered down compared to 200 thousand for ours. In his research, Zhang essentially has two steps when looking for user similarity, with the first being looking for "two users who rated the same book multiple times" and seeing how closely related to each other (Zhang, 2019). In the case where there does not exist a common book between users, Zhang takes the approach of comparing the Euclidean distance between one user and others who have "similarly rated books" (Zhang, 2019). As for the results, we couldn't necessarily compare them directly, as although our models had similar bases, they were used for different purposes in the end. What we can say, however, is that Zhang's results, in terms of his reported user similarity values, shows that the number of book similarities between users were quite sparse. This is evident through the similarity values holding only rounded numbers (i.e. 0.6), and not precise numbers (i.e. 0.63). Fortunately, this wasn't an issue for us as our data had enough overlap between a particular user's reviews with that of another, as on average, each user made 18 reviews. In general, however, this finding speaks to the difficulty of using a similarity based model on data that has very little overlap amongst interactions.

https://towardsdatascience.com/deep-learning-based-recommender-systems-3d120201db7e

Aside from traditional recommender system methods, more advanced, state-of-the-art methods used today involve deep learning. In the above article, author James loy used MovieLens 20M dataset to implement a deep learning based recommender systems. For this prediction task, the author tries to classify what type of movie the user will possibly watch using a neural network. At first, the author one hot encoded the usersID and movieID vectors as an input layer. For the embedding layer, the userID and MovieID vectors will be embedded to form an embedding layer. Finally, it combines both vectors and passes to a fully connected layer. Then the result of the fully connected layer will pass to an activation function(Sigmoid function) to classify whether the user watched that type of movie or not. After creating a neural network model, we can train the model and find an optimal weight in order to increase accuracy on the model. As for evaluating the models, Loy suggests a ‘Hit Ratio of 10’, in which out of 100 total items (99 in which the users haven’t interacted with and 1 that the user has), the number of times in which the 1 item the user has had interaction with appears in the top 10 recommendations divided by the number of total times recommendations were made for users. In terms of this metric, Loy’s recommender system scored 86%. When comparing our results, besides the fact that the datasets used are totally different, one analytical reason for why our accuracy was lower by nearly 20% is that we did not take this Hit Ratio of 10 approach. In essence, this limited our prediction model to only have one chance of either being right or wrong, rather than having ten with Loy’s approach. In practice, we believe that this Hit Ratio of Approach is much more sensible in terms of what would be an acceptable accuracy for prediction / recommender systems, where one would be satisfied with 1 out of 10 items being a correct recommendation.

https://www.ijcai.org/Proceedings/2019/0883.pdf?fbclid=IwAR0zKgxcqyUiMW37XUxSZrfO1TbLQU-G0bHjQ7sMUVo1r2o3xmfr3j_jo1Y

Sequential recommender systems are one of the hottest models in the AI field, this model is widely used in predicting whether the user will buy the associated item or not. Sequential recommendation systems are basically able to suggest items for the user that they might be interested in. It tries to understand the sequential user behaviors based on the user history. For example from the paper, Jimmy has done sequential actions in the past: booked a fight, booked a hotel and rented a car. Then the model would try to predict what Jimmy would do for next action based on previous actions. The next action may be visiting a tourist attraction via self-driving. For our dataset, we can also apply the sequential recommender systems to predict whether the user will read the book or not based on what the user has read in previous history.

QUESTION5:

Model	Classification Accuracy
Popularity (baseline)	.6538
Euclidean (item to item)	.6145
Popularity + Euclidean (item to item)	.6400
Cosine (item to item)	.6659
Popularity + Cosine (item to item)	.6771
Jaccard (item to item)	.6731
Popularity + Jaccard (item to item)	.6837
Collaborative filtering for rating prediction	.6063

For our results, we found that our baseline popularity model scored relatively well (.6538) compared to the other models that we implemented. We felt like this specific result made sense especially when it comes to books because a lot of times, readers tend to have read books if they are popular enough, regardless of what specific kind of book they might prefer individually. For this reason, we decided to test out a combination of this popularity threshold with each of the other models in which we implemented. Our highest scoring model was the combination of a Jaccard item-to-item similarity with the popularity baseline, scoring .6837, beating the baseline accuracy by approximately 3%. It is also important to note that this combination beat the naked Jaccard model by slightly over 1%. These results were also in line with our hypotheses about them to begin with. First off, we had expected the Jaccard similarity approach to work relatively well given the relatively large amount of average interactions for each item. This made it possible for there to be enough overlap between the interactions of different items so that we could identify similarities from them. Secondly, both the cosine and

Jaccard similarity models scored relatively similar, which indicates that the product of the magnitude of two feature vectors for two respective items was very similar to the number of features two items had in common. Lastly, it should be noted that both the naked cosine and Jaccard models scored higher than the popularity baseline, but both combinations scored higher than the naked models themselves. This supports our hypothesis that the baseline popularity model was an important factor to keep in our similarity models.

We also observed our worst performing similarity model to be that of Euclidean distance. Our main reason for this was that Euclidean distance offers a measurement that is not scaled. In turn, this type of model tends to skew in favor of items with more interactions, as the more interactions a pair of items have, the more total similarities they are likely to share with each other. The problem with this, both theoretically and in our model, essentially assumes that items with a small number of interactions are bad, as they are less likely to be recommended. However, this is not true, as an item can indeed be highly rated but simply have few ratings. Looking back, one way we could have improved upon our Euclidean model is by normalizing the euclidean values of each item to item comparison. This would've helped scale our threshold as well so that items with a high amount of interactions and those with a lower amount of interactions are weighted more equally.

Bibliography

- Citation: Zhang, Mick, Jun 12, 2019, GoodReads-Recommendation-using-Collaborative-Filtering, https://github.com/mick-zhang/GoodReads-Recommendation-using-Collaborative-Filtering/blob/master/Book%20Recommender%20System%20Github.ipynb?fbclid=IwAR0MHNe6jjas-9h_3YQbJrlPOuIwPOcNIDIgOPG6Rka94u-hY7stYb_ZJMU
- Loy, James. “Deep Learning Based Recommender Systems.” *Medium*, Towards Data Science, 19 Oct. 2020, https://towardsdatascience.com/deep-learning-based-recommender-systems-3d120201db7e
- Wang, Shoujin, et al. *Sequential Recommender Systems: Challenges, Progress and Prospects*. 2019, www.ijcai.org/Proceedings/2019/0883.pdf?fbclid=IwAR0zKgxcqyUiMW37XUxSZrfO1TbLQU-G0bHjQ7sMUVo1r2o3xmfr3j_jo1Y.
- McAuley, Julian, Fall 2019: Assignment1, Nov 2019, http://cseweb.ucsd.edu/classes/fa19/cse258-a/files/assignment1.pdf