DSCI-351 Recommender Systems **Kevin Shi, Christopher Fluta, Dren Gara**
Final Project **August 30, 2024**

Introduction: Dataset Description & Type of Recommender Systems Built

Our dataset for this project is the book-crossing dataset with user review ratings of books from Kaggle. There are 1,031,175 rows and 19 columns in the dataset ('Unnamed: 0', 'user_id', 'location', 'age', 'isbn', 'rating', 'book_title', 'book_author', 'year_of_publication', 'publisher', 'img_s', 'img_m', 'img_l', 'Summary', 'Language', 'Category', 'city', 'state', 'country'). The dataset contains 92,107 unique users and 241,090 unique book titles with ratings ranging from 0-10.

We decided to perform user-based collaborative filtering with KNN and item-based collaborative filtering by creating a rating matrix from user information (user_id) and ratings of items (books). Since our dataset has user information and item features such as author, summary, category, and publisher, we decided to also build a content-based filtering system utilizing TF-IDF. This content-based system recommends items that are similar to ones the user has liked in the past, based on features such as author, summary, category, and publisher.

Method(s)/Model(s) Utilized

Data Cleaning

For user-based and item-based collaborative filtering, due to the large computational expense of processing the full dataset, we reduced the dataset size by only keeping users from the USA and users who rated books in English. We also excluded users with less than 100 ratings and books with less than 20 ratings. The rating matrix shape was (813, 1768) with 813 unique users and 1,768 unique books. We also filled in missing values with zero.

For content-based filtering, we kept only the books from the USA and books written in English. Only content-relevant columns were kept: title, author, summary, category, publisher, isbn, year of publication, and language. Duplicate entries of book title, isbn, and summary were dropped since some books have multiple versions in publication, so there can be two versions of a book with the exact same name but different publishers.

User-Based Collaborative Filtering with KNN (w/ interpretation & evaluation)

The purpose of this method is to recommend books to a user based on users' nearest neighbors' ratings. This involved first identifying similar users by locating the nearest neighbors of the target user using the KNN model while excluding the target user from its own neighbor list. Next, ratings of a user's nearest neighbors were retrieved, and then the weighted average ratings for books the user has not yet rated were calculated. Finally, recommendations for books the user has not yet rated were generated based on the highest average ratings among neighbors.

When interpreting the results of this system, after specifying a target user id, the number of neighbors to consider, and the number of recommendations to be returned, the system returns the n similar users to the target user and their distances to the target user, along with the number of book recommendations specified with ratings for each recommended book.

For evaluation of the system's performance, we classified ratings into binary labels based on a threshold of 7 (1 if ≥ 7, else 0) to compute evaluation metrics such as accuracy, precision, recall, F1-score, and RMSE, and fitted the KNN model using a train-test split approach for each number of neighbors from 1 to 10. RMSE increases as the number of neighbors increases, so perhaps smaller values of n produce more accurate results. Precision, or the proportion of recommended books that are actually good or liked, decreases as the number of neighbors increases. Recall generally stays low across all the neighbors, meaning that the proportion of all good books that are actually recommended is very low.

Since accuracy and RMSE usually do not measure the true effectiveness of the system, we measured Spearman Rank Correlation and R-score (utility). For Spearman Rank Correlation, scores are all very high for the top-10 and top-20 recommendations, indicating that the rankings produced by our system are very close to the true rankings, regardless of the number of neighbors. As we increase the number of recommendations from 10 to 20, the system might include books that are less relevant or have lower predicted ratings compared to the top items, so this might have slightly reduced the correlation. For R-score, there is an increase from 0.7053 to 0.7352 when moving from 1 to 2 neighbors, which suggests that considering the ratings of a second neighbor improves the accuracy of the predictions. From n=3 to n=9, there are many fluctuations with the scores, meaning that recommendations are influenced by a balance of closely similar and somewhat dissimilar neighbors. The R-score drops sharply at n=10 to 0.3508, indicating that by including too many neighbors, the model is incorporating the preferences of users who are not very similar to the target user, leading to poorer recommendation accuracy.

Item-Based Collaborative Filtering (w/ interpretation & evaluation)

The item-based collaborative filtering approach recommends books by analyzing similarities between items based on user ratings. This method is effective for suggesting books similar to those a user has already enjoyed. We created an item-user matrix, where rows represent books and columns represent users, and used the K-Nearest Neighbors (KNN) algorithm with cosine similarity to measure how closely related books are. The model then recommends books that are most similar to those the user has rated highly. In practice, if a user liked 'The Da Vinci Code,' the system might recommend other mystery or thriller novels with similar ratings patterns.This approach works well for users with clear genre preferences.

The model's performance was evaluated using accuracy, precision, recall, and F1-score. We found that accuracy and precision decrease as the number of neighbors increases. For instance, when recommending books similar to 'Harry Potter and the Goblet of Fire,' accuracy dropped from 0.6190 with one neighbor to 0.4375 with ten neighbors. Recall remained high at 1.0000, but the F1-score, which balances precision and recall, also decreased. Overall, the item-based model is effective in identifying relevant books, though there is a trade-off between precision and diversity. Future improvements could focus on optimizing the number of neighbors and incorporating additional features like genre or author similarity to enhance recommendation accuracy.

Content-Based Filtering (w/ interpretation & evaluation)

The content-based filtering method uses TF-IDF vectorization and cosine similarity to recommend books based on their features. This approach is ideal for finding books similar in content alone. Our system takes a book title as input and returns the 10 most similar books in the database. It employs exact and partial matching, using the most similar book title if an exact match isn't found. We also drop duplicates in titles, ISBNs, and summaries. We focus on book_author, summary, category, and publisher as they are the most meaningful features, providing insight into a book's topic, content, and audience. Since we've filtered for English-language books, we avoid issues with non-English entries and other alphabets.

The system provides accurate recommendations; for instance, books within a series recommend others in the series, and genre-specific books suggest similar ones. The partial match system works well, as seen with the Harry Potter series. We also implemented a diversity score, showing high diversity in recommendations, with an average score of 0.92 across five books. Higher diversity scores indicate more varied results, and the system excels in balancing relevance with diversity without needing user profiles. There is room for improvement in our system, however. The Harry Potter series reveals a recommendation anomaly where multiple versions of the same book are treated as different entries due to variations in titles, ISBNs, and summaries. Implementing a canonical title system could standardize these entries. Another improvement could be incorporating serendipity—introducing unexpected but attractive recommendations to enhance user experience. The system may also face coverage issues in less-popular genres and struggle with higher-dimensional data as the dataset grows, necessitating revisions to TF-IDF vectorization.

Conclusion: Limitation and Future Improvement

For our collaborative filtering system, some limitations include the large scale computation requirement of using the full dataset (e.g. we only considered users from USA and books written in English), challenges with sparsity due to insufficient user ratings, the cold start problem, and the lack of novelty for item-based CF recommendations. Some ideas for future improvement include broadening the system to consider users from different countries and books in different languages, filling in missing values with the mean of the particular user or book ratings (instead of filling in with 0), and using user studies for system evaluation.

For our content-based filtering system, some limitations include the typical lack of novelty and serendipity associated with content-based systems, the similarity score not serving as a comprehensive measure, the consideration of books only written in English, and the handling of multiple versions of the same book being an oversight. Some ideas for future improvement include incorporating user profiles and/or combining with a collaborative filtering method into a hybrid system (particularly user-based implementation), and implementing a canonical title system to handle multiple publications of the same book.