COMP9727 Project Design Report
Furniture Recommendation System — Based on Amazon Reviews 2018 Dataset

Xutao Li

zID: z5455423

1 Project Scope

This project aims to build a recommendation system for **furniture** products, targeting individual consumers who frequently purchase furniture items on e-commerce platforms.

1.1 Use Scenarios

After logging into the desktop application (Electron+React), users can:

- Browse a personalized Top-N (e.g., 10-15) product list on the homepage;
- View two types of recommendations, "*You might also like*" and "*Frequently bought together*", when searching for or viewing any product detail page;
- Provide feedback on their preferences via "Like/Dislike" buttons.

1.2 Interaction and Feedback Collection

The system records users' explicit ratings and logs these interaction events in real-time for subsequent incremental model updates.

1.3 Interface Sketch

The interface is shown in Figure 1. Users can browse the personalized Top-N (e.g., 10-15) product list on the homepage, and click 'dislike' or 'like' to provide feedback.

1.4 Business Model

By increasing product exposure and conversion rates, advertising/commission fees are charged to third-party merchants; meanwhile, platform revenue is increased through additional sales (cross-selling) generated by recommendations.

1.5 Cold Start and Dynamic Updates

- **User Cold Start:** Use content-based initial profiles (e.g., style preferences selected during registration) + popular items list. *Rationale: For new users with no historical behavior, recommending based on their proactively provided demographic or preference information is an effective way to quickly establish an initial profile and avoid a "blank slate" experience. Popular items serve as a reliable supplement, ensuring a baseline quality for recommendations.*
- **Item Cold Start:** Use item metadata (title, category, price, brand, etc.) to generate vector representations and recommend based on similarity to existing items. *Rationale: Newly listed items lack user interactions, which collaborative filtering cannot handle. By utilizing their content attributes (like brand, material) to calculate similarity with older items, they can be recommended to users interested in similar products, solving the exposure problem for new items.*

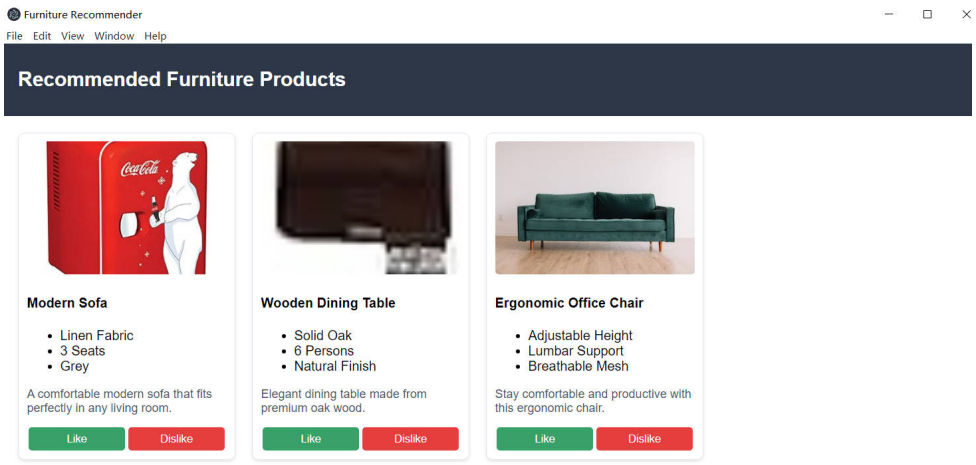

Figure 1: Interface Sketch

- **Model Update:** Daily offline batch retraining, combined with real-time streaming features (interactions within the last hour) for online re-ranking. *Rationale: Offline batch updates can train complex models using all historical data, ensuring core accuracy; while minute/hour-level online updates can quickly respond to recent changes in user interests, enhancing the timeliness and relevance of recommendations. This layered update strategy balances computational cost and timeliness.*

1.6 Competitive Analysis

- **Amazon:** As an e-commerce giant, Amazon's recommendation system is one of its core competitive advantages. Its most classic algorithm is **Item-to-Item Collaborative Filtering**. This method analyzes users' purchase history to calculate similarity between products. For example, if users who bought product A also frequently bought product B, then A and B are considered similar. The system recommends products similar to those the user has previously purchased or browsed. This method can handle massive data and provides interpretable recommendation results ("Because you bought A, we recommend B"), making it very suitable for e-commerce scenarios with huge product catalogs.

- **Netflix:** Netflix's recommendation system is a benchmark in the content distribution field. It primarily relies on **Matrix Factorization** technology, particularly popularized through the famous Netflix Prize competition. This technique decomposes the massive "user-movie" rating matrix into two low-dimensional matrices: one representing user preference features and another representing movie content features. By computing the dot product of these two matrices, it predicts user ratings for unwatched movies and makes recommendations accordingly. Today, Netflix employs more sophisticated hybrid models that integrate user behavior, metadata, and deep learning algorithms, even displaying personalized cover images for the same content based on different users' preferences.

- **Pinterest:** As a visual discovery engine, Pinterest's recommendation core is the **PinSage large-scale Graph Convolutional Network (GCN)**. It constructs all images (Pins) and boards into a massive heterogeneous graph. The PinSage [4] model learns embedding vectors for each node in the graph, which simultaneously encode both visual features and graph structural information. When a user shows interest in a Pin, the system can efficiently find millions of similar Pins in the embedding space for recommendations. This approach excels in handling visual content and leveraging graph structural information, sharing conceptual similarities with the LightGCN [1] planned for this project.

2 Datasets

2.1 Data Source

The dataset planned for this project is the "Home and Kitchen" subset from the Amazon Reviews Data 2018 [3]. This is a public and widely used real-world dataset for recommendation system research. Before use, this subset will be filtered using keywords from the product metadata (e.g., "furniture", "sofa", "table", "chair") to create a dataset focused on the furniture domain.

2.2 Data Quality and Limitations

The selected "Home and Kitchen" dataset contains millions of user reviews, covering a large number of users and products, which provides sufficient data for building robust recommendation models.

1. Data Quality: The dataset includes explicit user ratings (1-5 stars), detailed review texts, and timestamps, which are high-quality explicit and implicit feedback signals. Compared to the low-quality data samples mentioned in course tutorial 3, this dataset has richer fields and relatively less noise, supporting the implementation and evaluation of various recommendation algorithms.

2. The data comes from a real e-commerce platform, reflecting the actual purchasing and evaluation behaviors of real users, which has high practical significance.

3. The data is a historical snapshot, lacking real-time inventory/price changes, so temporal bias must be considered.

4. Review texts contain noise and HTML tags that need to be cleaned.

2.3 Data Usage Plan

- Construct "user-item-rating" triplets for the explicit feedback matrix; *Rationale: This is the core input required for building collaborative filtering models (especially matrix factorization), as ratings directly reflect the strength of users' explicit preferences.*

- Extract item metadata (brand, category, title) and TF-IDF vectors of review texts for use in content-based and hybrid models; *Rationale: Content features are key to solving the item cold-start problem and improving recommendation diversity and interpretability. TF-IDF is a mature and efficient text representation method.*

- Filter out low-activity users (<5 interactions) and low-interaction items to ensure data density. *Rationale: Users and items with too few interactions (e.g., only 1-2) make it difficult to learn reliable patterns and instead introduce noise and increase computational load. k-core filtering is a common method to increase data density and ensure model training effectiveness.*

2.4 Limitations of the Dataset

Although the dataset is of high quality, it still has some inherent limitations common to public datasets. Acknowledging these limitations is crucial for project evaluation:

Data Incompleteness: The dataset is only a snapshot of Amazon's data up to a specific period (2018). It does not include all users or all products, and it lacks richer interaction data such as user browsing, clicks, and add-to-cart events. This limits our understanding of complete user behavior.

Data May Be "Sanitized": Academic datasets are often cleaned and processed, potentially missing some of the complexities of real-world data, such as shilling attacks, outliers, and numerous missing fields. This can make the problem seem simpler than in a real-world scenario.

Limitations of Static and Offline Evaluation: Such datasets are mainly used for offline evaluation. Their fixed training/testing split encourages models to overfit to specific metrics on existing data. Excellent performance on a static dataset does not fully equate to a good recommendation system that meets dynamic user needs. For example, it cannot evaluate the presentation bias present in a real user interface or the long-term impact of recommendation diversity on user satisfaction.

2.5 Data Preprocessing Overview

To obtain high-quality furniture recommendation data suitable for model training, this project adopts the following three-step preprocessing strategy:

1. **Topic Filtering:** First, extract furniture-related records from the Home & Kitchen metadata and reviews based on the product `categories` path and title keywords (furniture, sofa, table, etc.).
2. **k-core Filtering:** Perform *k-core* filtering (default $k = 5$) on the extracted reviews, keeping users and items with at least 5 interactions, significantly reducing sparsity.
3. **Metadata Alignment:** Finally, based on the product `asin` in the filtered reviews, generate a corresponding subset of furniture metadata to support subsequent content feature extraction.

This process ensures the integrity and density of the training/evaluation data, while retaining a sufficient scale for deep model experiments.

2.6 Dataset Split Strategy

After the above preprocessing pipeline, we obtained a furniture recommendation dataset containing 160,738 reviews (covering 7,953 products and 24,285 users). To evaluate model performance, we need to split the dataset into training, validation, and test sets. We will split the dataset in an 8:1:1 ratio, i.e., 80% of the data for training, 10% for validation, and 10% for testing.

For collaborative filtering models, content-based models, and hybrid/deep models, the computational resources are all within acceptable ranges.

3 Methods

To address different scenario requirements and fully utilize the data, this project will implement and systematically compare the following three types of models:

1. **Matrix Factorization (MF) Collaborative Filtering:** As a mainstream algorithm validated by the industry, collaborative filtering discovers collective intelligence through users' historical behavior.
    - **Rationale for Selection:** Biased MF is chosen as a baseline because it is simple, robust, and can capture the inherent preference biases of users and items. We will further introduce LightGCN, a graph neural network model optimized for recommendation systems. By simplifying graph convolution operations, it can more efficiently learn high-order correlations in user-item interactions and is one of the current state-of-the-art collaborative filtering models.
    - **Optimization Strategy:** Incorporate a time-weighted decay factor in rating prediction, giving more weight to recent interactions than to older ones. This helps capture users' dynamically changing interests and improves the novelty of recommendations.
2. **Content-Based Recommendation:** To compensate for the inability of collaborative filtering to handle the cold-start problem.
    - **Rationale for Selection:** For new items with no interaction history or for users with sparse historical behavior, content-based recommendation is the only viable solution. By extracting the item's own textual attributes (title, description, etc.) and vectorizing them using TF-IDF, the similarity between items can be calculated. This method not only solves the cold-start problem but also provides intuitive recommendation explanations (e.g., "Because you liked A, we recommend the similar B").
3. **Hybrid/Deep Models:** Combine the advantages of collaborative filtering and content-based methods to achieve stronger predictive performance.
    - **Rationale for Selection:** NeuMF (Neural Matrix Factorization) is chosen as a representative model. By fusing traditional matrix factorization with a multi-layer perceptron (MLP), it can capture both linear and non-linear user-item relationships, potentially achieving higher accuracy than a single model. At the same time, we will explore a simple weighted fusion strategy, which linearly combines collaborative filtering scores with content similarity. Although simple,

it has low computational overhead and strong interpretability, making it a practical solution that balances effectiveness and efficiency.

The model development milestones are shown in Table 1.

Table 1: Timeline and Milestones

Week	Task
5	Data download, cleaning, and EDA; user profiling/item feature engineering
6	Baseline (popular items), collaborative filtering (MF) implementation and evaluation
7-8	Content-based model, NeuMF [2] training and tuning, offline metric comparison
9	GUI prototype development, online evaluation API integration
10	User study (questionnaire + usability testing), result analysis, report writing, and project presentation preparation

4 Evaluation

To comprehensively measure the performance of the recommendation system, the evaluation will be conducted from both offline and online perspectives, covering aspects such as accuracy, ranking quality, and user experience.

4.1 Offline Metrics

Offline evaluation uses historical data to backtest the model and is the primary means for rapid algorithm iteration and parameter tuning.

- **Ranking Accuracy (AUC, NDCG@K):**
  - **Rationale for Selection:** AUC (Area Under ROC Curve) measures the model's ability to rank all "user-item" pairs, reflecting the overall ranking quality, and is insensitive to imbalanced positive/negative samples. NDCG@K (Normalized Discounted Cumulative Gain) focuses more on the ranking quality of the top K recommended items, giving higher rewards for higher-rated (more relevant) items placed at the top, which is closer to the real-world scenario where users only care about the top of the list.
- **Top-N Hit Rate (Precision@K, Recall@K):**
  - **Rationale for Selection:** The final form of a recommendation system is to present a list to the user, so evaluating the quality of this list is crucial. Precision@K focuses on "how many of the recommended items are liked by the user," while Recall@K focuses on "how many of the items liked by the user were recommended." Together, they measure the effectiveness of the recommendation list.
- **Diversity and Novelty (Diversity, Novelty):**
  - **Rationale for Selection:** A good recommendation system should not only recommend homogeneous popular items. The diversity metric (Intra-list Diversity) is used to measure the dissimilarity among items in a recommendation list to prevent users from feeling monotonous. Novelty (measured by the average popularity of recommended items) encourages the model to recommend some long-tail, non-popular items, helping users discover surprises. These two metrics are important supplements for measuring user experience and long-term satisfaction.

For these metrics, the priority order is:

1. NDCG@K
2. AUC

3. Top-N Hit Rate (Precision@K, Recall@K)

4. Diversity and Novelty (Diversity, Novelty)

During the model development phase, NDCG@K and Precision@K are typically considered more practically meaningful than AUC and Recall metrics because they better reflect users' real experience in the product interface.

4.2 Online/User Study

An improvement in offline metrics does not fully equate to an improvement in real user satisfaction. Therefore, a small-scale user study will be conducted through a prototype system to verify the actual effectiveness of the model. A/B testing will be conducted by recruiting 3-5 students to participate in a simulated browsing process in the GUI:

1. **Business Metrics (CTR, Add-to-Cart Rate):**
   - **Rationale for Selection:** Click-Through Rate (CTR) and add-to-cart rate are direct evidence of whether the recommendation results are attractive to users and can promote conversions. They are core metrics for evaluating the business value of a recommendation system.
2. **Subjective Satisfaction Questionnaire:**
   - **Rationale for Selection:** By having users rate the accuracy, diversity, novelty, and overall usability of the recommendations on a Likert 5-point scale, we can quantify the subjective experience that offline metrics cannot capture.
3. **System Performance Metrics (Response Time):**
   - **Rationale for Selection:** Users have a very low tolerance for latency. Verifying that the system can return recommendation results within an acceptable time (e.g., <200ms) is fundamental to ensuring a good user experience.

4.3 Resources and Scalability

- Offline training uses a GPU (NVIDIA RTX 4060);
- Online inference model parameters are <50MB;
- Data updates use daily batch retraining.

5 Conclusion and Feasibility

The Amazon Reviews 2018 dataset is large-scale, domain-specific, and well-structured. By combining collaborative filtering, content-based, and deep hybrid models, high-quality personalized recommendations can be achieved in the home furnishing e-commerce scenario. This design fully considers cold-start issues, real-time requirements, and multi-dimensional evaluation. The project plan is feasible to be completed and a runnable prototype delivered within the semester.

References

[1] X. He, K. Deng, X. Wang, Y. Li, Y. Zhang, and M. Wang, "LightGCN: Simplifying and Powering Graph Convolution Network for Recommendation," in Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval, Virtual Event, China, 2020, pp. 639–648.

[2] X. He, L. Liao, Z. Zhang, L. Nie, J. Hu, and T. S. Chua, "Neural collaborative filtering," in Proceedings of the 26th International Conference on World Wide Web, pp. 173–182, 2017.

[3] G. Linden, B. Smith, and J. York, "Amazon. com recommendations: Item-to-item collaborative filtering," IEEE Internet Comput., vol. 7, no. 1, pp. 76–80, Jan. 2003.

[4] J. McAuley, C. Targett, Q. Shi, and A. van den Hengel, "Image-based recommendations on styles and substitutes," in Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 43–52, 2015.