Project Design

Joey Zhou

z5548349

1. Scope

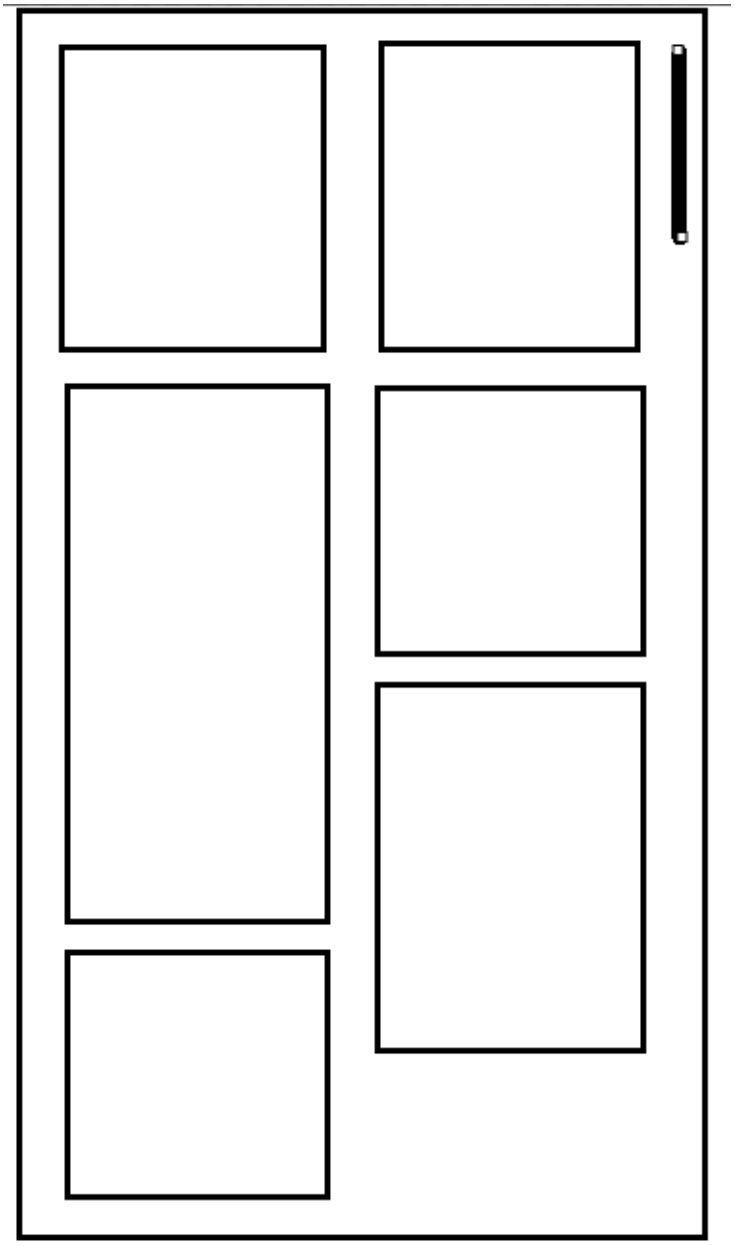
Scope Statement

The project shall deliver an E-commerce recommender system that provides similar products as well as popular ones to customers shopping online. The project shall be done in 5 weeks, starting from 6th July 2025. The system shall include Data Cleansing to get useful data from the dataset, core Recommender System model to provide accurate recommendations and evaluations to measure the performance of the system. The project shall only implant these functions based on the Amazon Reviews dataset and does not include any UI development.

UI, Interactions and evaluations

The UI is designed for mobile platforms. However, with a slight modification, the same recommended result can be applied to a webpage.

The page will be shown at the user's home page or after a user has made a purchase. The UI will be an infinite list of products which the user might like, each containing an image of the product, as well as the rating and price. The user can also long press the item to manually delete an item from the list.



The items shown here will initially be the most popular products among all users. After a user made some purchases, we will check other user's purchase history and recommend those the current user has not bought and might be interested in. We will also store the user's browsing history, if a user has viewed a lot of products and has not made a purchase, we consider the user a potential buyer and recommend similar items in here.

We will track if a user clicks on these items or not. If a user clicks on one item, we add more priority to these kinds of items and if the user ignores one item (scrolls down), we slightly decrease the priority. If the user manually removes one item, we significantly decrease priority for this kind of item.

It’s also important to track purchase histories and make changes to the results based on that. To be more precise, the system should analyze if the purchased product is something that one user will buy multiple times or not. For example, if it is a washing machine, then we will decrease the priority of washing machines. But if it is a snack, then we want to recommend more similar snacks to the user.

Discussion

The main goal of the system is to keep users browsing so they might find products they find interesting. Either it is a product the user has viewed but not purchased before, or it is something that is currently popular and the user might like. In this way, we can potentially encourage users to make a purchase they initially did not plan to make.

2. Database

The database used in this project will be a subset of Amazon Reviews 2023. The whole dataset contains 571.54M Reviews from 54.51M Users. It also includes product information on 48.19M products from 33 different categories. This project will choose Amazon_Fashion, Arts_Crafts_and_Sewing and Beauty_and_Personal_Care as the dataset.

The dataset consists of two types of information: User reviews and Product metadata.

Field	Type	Explanation
rating	float	Rating of the product (from 1.0 to 5.0).
title	str	Title of the user review.
text	str	Text body of the user review.
images	list	Images that users post after they have received the product. Each image has different sizes (small, medium, large), represented by the small_image_url, medium_image_url, and large_image_url respectively.
asin	str	ID of the product.
parent_asin	str	Parent ID of the product. Note: Products with different colors, styles, sizes usually belong to the same parent ID. The "asin" in previous Amazon datasets is actually parent ID. **Please use parent ID to find product meta.**
user_id	str	ID of the reviewer
timestamp	int	Time of the review (unix time)
verified_purchase	bool	User purchase verification
helpful_vote	int	Helpful votes of the review

Review Fields (https://huggingface.co/datasets/McAuley-Lab/Amazon-Reviews-2023)

The user review will be used in a collaborative filtering recommender system, and the useful fields will be rating, parent_asin and user_id.

It will also be used to determine if a product is popular recently, and for this case, the fields useful will be rating, parent_asin and timestamp.

As for the product metadata, average rating and price are useful information to show to the user.

title, features, description and details will be combined to build a document for a Content-Based recommender system.

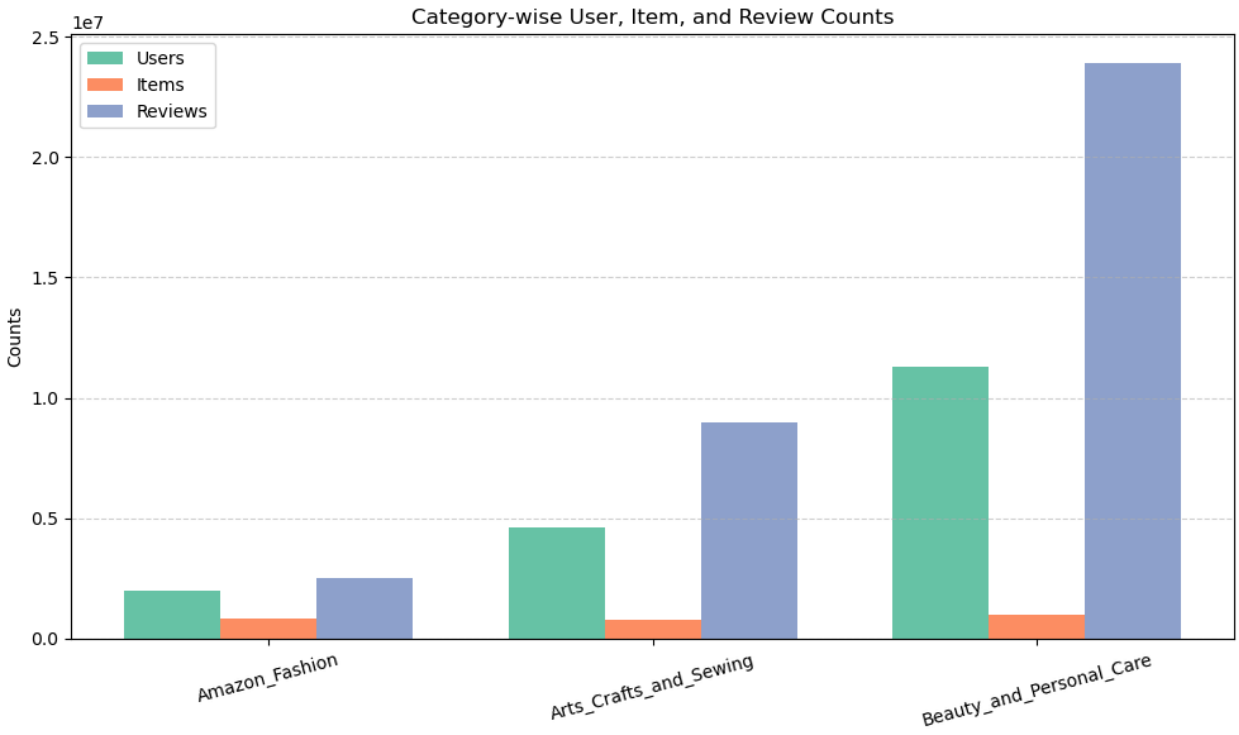
Image link will be used to extract images features and possibly be used with the Content-based recommender system.

The rest of the fields are either Null for many items such as categories or are not useful for this project such as rating_number. All these fields will be filtered during preprocessing.

Field	Type	Explanation
main_category	str	Main category (i.e., domain) of the product.
title	str	Name of the product.
average_rating	float	Rating of the product shown on the product page.
rating_number	int	Number of ratings in the product.
features	list	Bullet-point format features of the product.
description	list	Description of the product.
price	float	Price in US dollars (at time of crawling).
images	list	Images of the product. Each image has different sizes (thumb, large, hi_res). The “variant” field shows the position of image.
videos	list	Videos of the product including title and url.
store	str	Store name of the product.
categories	list	Hierarchical categories of the product.
details	dict	Product details, including materials, brand, sizes, etc.
parent_asin	str	Parent ID of the product.
bought_together	list	Recommended bundles from the websites.

Metadata Fields (https://huggingface.co/datasets/McAuley-Lab/Amazon-Reviews-2023)

As for data distribution, we can see although the three categories consist of similar numbers of items, the user number and review number are significantly different.



Data Distribution

Discussion

The main limitation of the dataset is that it lacks purchase information. Users who buy a product will not always necessarily write a review. We won't know which products a user buy together. This makes the dataset not the perfect choice for a collaborative filtering recommender system.

However, we are still able to get 3%-20% of users who post at least three reviews for different products. Since this is a large dataset, the useful number of reviews is still large enough for popular items. But this would be a problem for products with few reviews and the collaborative filtering system might not have enough information to provide accurate recommendations for these products.

The dataset is also extremely large. For example, the size of "raw_review_ Beauty_and_Personal_Care " is over 11G. Although filtering out unnecessary fields will significantly reduce the size of the dataset, this still makes the processing a challenge.

3. Methods

Content-Based recommender system

When a user has viewed a few similar items recently without purchasing any, or they have bought products that are likely to be bought again in the future, it might be a good idea to use a content-based recommender system to provide similar recommendations.

In this case, each user should have a profile for each category since we only care about similar products in the same category. The item metadata contains several fields that's useful: title, features, description and details. These fields can be combined together to create a document for each product. The user profile will contain all relevant product documents together. Then TF-IDF vectors can be used to find similar products for the user.

The result should be weighed among all categories, so the system provides products the user is interested in. This method will be likely to recommend more similar product to the user as the user is browsing the list, so as we have discussed above, after the user dislikes an item on the list or has made a purchase, the user profile should be updated.

Collaborative Filtering recommender system

As the database contains reviews, it's possible to build a user-based collaborative filtering recommender system. In this case, we care about recommending items that a similar user has also purchased.

The first step is to filter reviewers who have reviewed more than N items. The number N should be tested for the best results.

Then a rating table can be built based on users and products. This can be used to compare user and user similarities using KNN.

Considering the size of the dataset, it might be a good idea to use Neural Collaborative Filtering instead.

Including Images

Since the metadata contains image information, it is possible to use it to provide more accurate recommendations.

The idea is to extract image features by passing it to a ViT and combine this vector with the TF-IDF vector extracted from text information.

The result depends on how well a pretrained ViT can extract image features and the way to concat the features to the TF-IDF vector. However, since this method is applied to a

Content-Based recommender system, experiments can be done by applying this method to one category. If metrics are improved, then same method can be applied to all categories.

A Hybrid System

The final list will contain results from three parts.

The first part contains current popular items. This can be get by filtering the number of reviews within a certain timestamp range. We hope the user will like what everyone else likes. But since this is pure luck, the number of items should be strictly limited.

The second part contains recommendations based on previous views and purchases. Since the main goal is to keep the user browsing, a dynamic weight should be set so the user is browsing the ones they just clicked on.

The third part will be products recommended based on similar users. This should help the system keep providing different products to the user.

It's important to determine how much the system focuses on exploring new items. A weight needs to be set to balance the results from the three parts above, and the experiments should be done to find the optimal settings.

4. Evaluation

The main goal of the system is to keep the user browsing as long as possible while showing the user products they might potentially buy. So the following metrics will be used to evaluate the system.

NDCG@N

Although the list is theoretically infinitely long, realistically, only N (suppose 30) will be loaded each time. So we can apply NDCG@N to the result, measuring how well the model ranks the results.

Hit-Rate@N

The user can quickly browse the list by scrolling down but may get bored if no item catches their attention. We can assume that as long as the user clicks on one of the items per batch, they will be motivated to view more and the model is providing a good result.

Both NDCG@N and Hit-Rate@N will be measured based on a set of users that are either gathered from the dataset or mocked.

Staying Time/Browsing Number

These will be used to measure the performance of the system. The system will be considered successful if a user stays longer on the page, or they browse a greater number of products. However, as discussed above, it is possible for a user to quickly scrolling down without paying attention to the content. So Staying Time is preferred over Browsing Number.

The ratio of these two metrics can also be used to measure how much a user is engaged.

Purchased Number

The ultimate goal is to invite the user to buy more products. So keeping track if a user clicks on an item and makes a purchase is useful. However, since the system is mainly showing the user more potential options, we don't expect a great number of users will immediately buy anything (It's good enough for them to know something exists and make a purchase later). So this metric won't be used as the primary metric to evaluate the system.

Responding Time

The model will need to keep showing outputs as the user reaches the bottom of the list. Also, the user will be giving feedback (dislike, clicks on) and the model will be expected to use this feedback and update the user profile.

So the expected responding time should be near-real time and ideally limited to less than 1 second.

User Survey

A short survey given to real users can realistically reflect the performance of the system. The survey may contain the following questions.

1. Do you find items you like on the list?
2. How much are you interested in keeping browsing? (rating 1-5)
3. Accurate rate (seeing what you expected to be useful) (rating 1-5)
4. Refreshing rate (seeing what you didn't expect but useful) (rating 1-5)
5. Overall rating of the rage (rating 1-5)
6. Anything you would like to share?

This questionnaire will help balance the novelty and accuracy of the system. So the user is provided with a result that is helpful to them but also triggers their curiosity.