Project Design

Shichu Zhang z5547362

July 6, 2025

1 Scope

For this project, I'd like to complete a recommendation system like the homepage of Xiaohongshu(Rednote), which is a popular application in China. This system means to recommend high quality and personalized notes for the users, and raise the interactions like $save, like, comment, click, refer$ and $follow$.
The intended users of this application are the common users, generally in the ages of 15-40, interested in delicious food, cooking skills, traveling, pets, studying, fitness and so on. Just like the basic recommendation system of industrial social applications mainly consists of notes, I would like to present 10-20 notes relied on the users' preference each time the users refresh the homepage. Then we can evaluate the interest of users based on the time they scan the notes, whether they $click$ in the notes, $like, save, refer$ and $follow$ the authors of notes. Also, the users may touch the notes two seconds to give a feedback(generally negative) for the notes, and this would have a huge affect on the score of the recommended types.
After these actions, the users' preference will be updated in the recommendation system. As these social notes are send in high speed, the system is expected to update users' documents frequently.
To solve the cold start problem.
For new users, we will give them introductions, like asking "your preferred contents", then show some popular types like "study", "foods" and so on.
For new notes, we will check the tags the authors add, extract the images and texts in the notes, and then compare with similar notes.
This recommendation system will help to gain more adhesiveness of users, increase the time users spend on this application, and improve the advertisements planting in this application, which will bring back more cash flow.

2 Dataset

For this part, I would like to use the dataset of Rednote, but unluckily, Rednote doesn't provide any data from users, so I have to choose from similar applications.
I found that there are public datasets from kuaishou, youtube and instagram.

At first I would exclude youtube, as Rednote includes more textual and social notes. Then instagram and kuaishou have different advantages. Instagram has more social attribute, and the format of posts is also similar to Rednote. However, as a Chinese application, kuaishou fits the preferences of users and laws of China better.
After thinking, I would choose the dataset of kuaishou, which the dataset from GitHub:
https://github.com/jiaawe/kuaishou-recommender?
As the description, we can see that there are:
big_matrix.csv: user–item interaction records with timestamps and play durations
item_daily_features.csv: item metadata including upload date, video length, visibility etc.
social_network.csv: user–user following relationships
item_categories.csv: item category labels
kuairec_caption_category.csv: item titles and assigned category labels
Based on these datasets, we can found that almost all fields are useful, like the user_id, video_id, play_duration and watch_radio in the big_matrix.csv, video_id, video_duration, video_tag in the item_daily_features.csv, frient_list in social_network.csv, and the video_id, caption and category in kuairec_caption_category.csv.

3 Method

In this part, we would think about the methods for the recommendation system.
The basic content-based recommendation would use the caption, category and tags of the notes, then match the users' preference.
We can also use the collaborative filtering based on the user_id, video_id and watch_ratio.
In the user-based CF, similar notes will be recommended to the similar users, compared by their interaction vectors.
In the item-based CF, compute the cosine similarity between notes,then recommend based on users' history.
We can use hybrid recommendation to handle cold start by giving different weight for content-based recommendation and CF.

4 Evaluation

1. Recommendation Model
   In this part, we would consider the precision and recall of Top-N due to the user_id, video_id and watch_ratio in the big_matrix.csv. Then we can calculate the hit rate of the recommended notes.

2. Recommendation System
   In this part, we would consider the actions of users such as $like$, $save$, $follow$, $refer$

and negative feedback like *dislike*, *disgusted*.

3. Comparison between near-real time and dynamic Generally, both methods are important to this application, as we need to react due to the negative and positive feedback from users, and we should also catch the change of users' preference. For example, when a user gives a feedback as he is disgusted about a topic, we should immediately stop recommending the similar content notes, and we would increase the weight of some topics as the user start scanning some new topics.

4. Getting Feedback First we should consider the short-term feedback to the recommendation, as if the user *click*, *like*, *save*, *refer*, *follow* or *dislike* the notes, we would have a positive or negative feedback for those topics. Also, there are other scenes as they may pass or ignore some notes. Due to the industrial recommendation system, rather than passive feedback, that's kind of normal, as users won't click all notes in their homepage. Then we can update the users_docus based on these data.

   Then we should consider the long-term feedback. To get feedback of the personalized recommendation system, we need to figure out some efficient ways. For example, we should give users a questionnaire to ask whether they like recent recommendations, and give an adjustment for weight of topics if the user is not satisfied.
   Also, we should consider some special users who don't like personalized recommendation, as they feel being monitored, then we should reduce the weight of recommending even stop using it.