9727 project design

Name:Lixian Deng
Zid:z5556881

1. Scope

For this design, the domain of this recommender system is movie recommendation, we can help normal movie enjoyers using an online platform to find some films based on their preferences.

Each recommendation session will present 10 movies at a time to the user. These movies will shown on website or mobile browsing. It will include the titles, thumbnails, short descriptions, and interactive buttons.

[🎞 Movie 1] [🎞 Movie 2] [🎞 Movie 3] ... |

Cover1 cover2 cover3 (can click)

👍 👎 Add 👍 👎 Add 👍 👎 Add

Rating(from 1★ to 5★)

Comment1 comment2 comment3

The system will like this with a really easy and quick feedback buttons(👍 👎), and we can use "add" for a future watchlist, when we click the cover of the movie, it will show some information like rating, genre, or a small clip of trailer. And the comment is below the movies. This interaction can have feedback both in explicit and implicit, which will be saved by the system for updating the user preferences and improving the accuracy of recommendations.

For the Dynamic Scenario, we can collect user's feedback in regular time, then retraining the recommendation model for update. The time can be a week due to a short time or a much longer time can't show the real taste of a user.

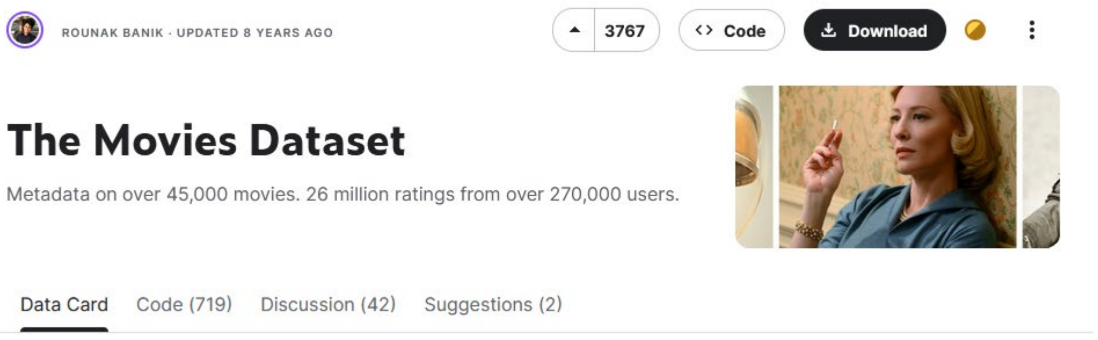
To address the cold Start Problems, I thought using metadata like keywords was especially useful for this scenarios. For example, we can recommend some popular movies or content based on genre, keywords, and cast when it comes to a new user, for a new added movie, we can use metadata files from the database to find the movies with similar content profiles and put it into the relevant streams to the user.

Also this system can have some potential business applications like putting some ads with the recommendations, or create a premium sub to offer more personal features like AI-Recommendations or more watchlist,

2. Database

For this project, I use this "The Movies Dataset"

https://www.kaggle.com/datasets/rounakbanik/the-movies-dataset



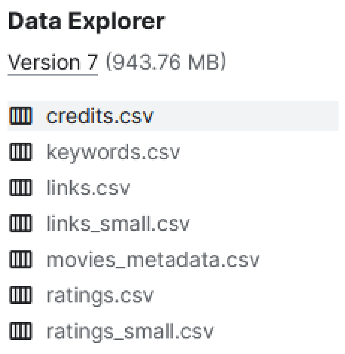
ROUNAK BANIK · UPDATED 8 YEARS AGO

The Movies Dataset

Metadata on over 45,000 movies. 26 million ratings from over 270,000 users.

These files contain metadata for all 45,000 movies listed in the Full MovieLens Dataset. The dataset consists of movies released on or before July 2017. Data points include cast, crew, plot keywords, budget, revenue, posters, release dates, languages, production companies, countries, TMDB vote counts and vote averages. And also has has files containing 26 million ratings from 270,000 users for all 45,000 movies. Ratings are on a scale of 1-5 and have been obtained from the official GroupLens website.

The dataset has some CSV files:



Data Explorer

Version 7 (943.76 MB)

- credits.csv
- keywords.csv
- links.csv
- links_small.csv
- movies_metadata.csv
- ratings.csv
- ratings_small.csv

I think these files are useful for our system, we have a great amount of ratings, and we can learn latent user to predict user preferences.

We also can use some information like genres, overview in movies_metadate.csv and the keywords.csv to compute movie similarity through TF-IDF or embeddings.

The user profiles can be built based on their rated or commented movies and matched to similar content.

With both operations, we can combine them to generate a much more personalized recommendations.

Although the movie database is such a huge one, but maybe it still has some shortcomings. Maybe many movies have few ratings, this will affect the performance and may require filtering out some low-activity users or movies.

3. Method

For a better performance of our system using The Movies Dataset, I use these methods to making the system more accurate and robust.

a. Collaborative Filtering using Matrix Factorization
   we can build a user-item matrix using the ratings.csv file, in this matrix each row represents a user, each column represents a movie. By applying SVD we can learn latent representations of users and movies in a shared embedding space.
   It can capture the implicit relationships between users and items and learn user preferences even without explicit metadata.

b. Content-Based Filtering using Metadata + TF-IDF
   The movies_metadate.csv and the keywords.csv and credits.csv give us information about each movie including: cast, crew, genres, keywords, overview and so on. We will represent each movie as a feature vector combining the cast/crew embeddings, one-hot encoded genres and TF-IDF vectorized keywords. The user profiles will be built by aggregating feature vectors of movies they have rated.

c. Hybrid Recommender System

   To combine the advantages of both methods, We aim to combine both methods via a hybrid system, which merges collaborative and content-based predictions.

   Firstly, We combine prediction scores using a weighted average or linear model, then tune the weight α using validation data or grid search, so we can adjust the balance between CF(Collaborative Filtering) and CB(Content-Based Filtering) to optimize performance.

   It's standard practice in large-scale recommender systems and can mitigate weaknesses of individual methods, This improves robustness and coverage for new users and items.

4. Evaluation

For the offline model evaluation, we can use a variety of top-N recommendation metrics, such as precision@10, Recall@10, NDCG@10, MAP.

These metrics will be computed per user and then averaged across users to reflect personalization quality. The precision can show accuracy but penalizes diversity, Recall can favor inclusion but may include some noise, the NDCG(Normalized Discounted Cumulative Gain) can balance the precision and position relevance. When we choosing the best model to use, we can use the model that optimize NDCG while has an acceptable Recall.

For an online system evaluation, we can collect the interactions from users by clicking 👍/👎, ADD through the stream of 10 personalized movie recommendations. The click rate and add-watchlist rate can help us to assess user satisfaction and engagement.

We can also gather qualitative feedback through a short questionnaire, such as "How would you rate your overall satisfaction (1–5)?" "Would you use our system again?" This feedback will help us improve movies recommendation accuracy and movie viewer interface design.