好的。我想要做一个hybrid movie recommender system, 这个data product可以实现：input a user id, given the historical user-item interaction data and item content data, the product can output the item id's of top 5 items recommended for the user.

Chatgpt: <https://chatgpt.com/share/671ddd8c-d4cc-8004-9e48-9e4b04fd942b>

Business Objective: provide personalized recommendation

1. Data

The data we have: Amazon Review 23’

Review data: We have timestamp for each user-item interaction

Product metadata: use title as the content

1. Model

The structure and logic of the model would be:

step 1: recall (candidate generation)

use matrix factorization to learn the user embedding matrix and item embedding matrix (with latent factors)

then retrieve the top 100 candidates for each user using FAISS ANN

step 2: ranking

use content-based filtering to compute the similarity scores between each candidate item and the user’s vector which is a weighted aggregation of all the item vectors the user has rated.

Item vector is extracted from the product title or description using tf-idf

Then select the top 5 items with highest similarity score as the final output

1. Steps:

* Train test split: split the user-item interaction based on timestamp,

leave one out: each user’s last interaction is selected as the test data

* Train the matrix factorization model using training set, and evaluate the model using test set

Next Step: Add negative sampling

* Benchmark: import the SVD model as baseline for matrix factorization, compare the RMSE with our matrix factorization model on our train/test data
* Save the user2idx, item2idx maps to pkl files, and also save the best trained models using torch.save(model.state\_dict(), 'mf\_model.pth') so that we can access the learned user\_embeddings and item\_embeddings which are saved in the model params

----- above is the train.py -------

----- below is the serve.py ------

* Employ FAISS ANN to retrieve the top 100 candidates of a given user id, based on the saved model embeddings and user2idx, item2idx maps
* Construct the content-based filtering model
* Evaluate the whole model using ranking metrics such as precision@k, map, based on the test set

1. Limitations and further study

* No online metrics (implicit) to test model, such as CTR, like, comments
* For fully cold start users,

1. Deploy

On flask api?

<https://ejmacias.medium.com/movie-recommendations-on-the-cloud-e63eea7973c7>

application:

input a user id

meta: matrix factorization embedding + ANN retrieve the top 100 candidates based on predicted ratings on items the user has yet interacted with -> extract the user’s profile vector based on the user’s interacted items, and the item vector of the top 100 candidates using tf-idf -> retrieve top 5 items using cosine similarity

output 5 ranked item ids

train.py:

1. train the matrix factorization embeddings, save the user and item embeddings to S3

serve.py:

1. access the given user id’s embeddings and its historical interactions, and use ANN to retrieve the top 100 candidates from the item embeddings
2. extract tf-idf vectors and generate the top 5 candidates