Recommendation System for Pratilipis

1.Overview

The dataset consists of User level level information regarding any updates to their corresponding Pratilipis or stories.The updates are basically how much percent of the story they have read.There are other information such as the amount of reading time per Pratilipi,the date on which the Pratilipi was published and the author and the category of the Pratilipi.

Due to the absence of explicit feedback like Likes,or rating ,it is difficult to build a popularity based recommender system and this is the reason I have gone for building a recommender system based on implicit feedback.

Also explicit feedback can be scarce, while implicit feedback has many benefits. With recommender systems that are constructed on the back of implicit feedback, we may make adjustments as we go along, based on the actions of the user. In modern online recommender systems, implicit feedback is used to fine-tune the recommendation in real time based on each user's actions.

The idea is to annotate the read_percent variable as 1.But this can be confusing as our dataset consists only of

interactions of users with the stories.So essentially all the data points will have a 1 for its read_percent variable.

To give more insight into this ,I will be choosing samples for the training set in such a way that it consists of one interacted Pratilipi and 4 non-interacted ones per User ID. The ratio that has been chosen is 4:1(4 negative samples to one interacted sample).

The dataset was ranked based on the updated_at variable and split into the 75-25 % train test ratios respectively as mentioned in the problem statement.

2.Model

The model chosen over here is a Neural Collaborative Filtering model(MLP).The reason for choosing this is because it's a powerful baseline model.It is also the state of the art recommendation system for many companies.
Their primary benefits lie in (a) its performance, (b) its avoidance of arduous feature engineering, and (c) its independence from the popularity variable.
NCF is a very robust and accurate deep learning architecture for Recommender Systems.

Due to time constraints,I had trained the model only for one epoch and the hit ratio@10 was **`0.62 for 1000`** predictions.

There are other metrics to look at such as Average reciprocal hit rate (ARHR),Coverage,Diversity and Novelty

3.Alternatives

The other alternatives for Baseline are:
1.KNN-K nearest neighbours
2.BPR-Bayesian Personalized Ranking
3.eALS -Alternating Least Squares

4.Advantages of NCF over Alternatives:

1.The sampling ratio of a pointwise loss can be adjusted freely, but pairwise objective functions are limited to matching up exactly one negative case with a positive one.
2.Can get better as layers increase - Non-linearity
3.Effective output in lesser number of iterations.

5.Improvements:

1.Robust parameter optimization along with increase in hidden layers
2.Use GMF instead of MLP
3.Use NeuMF(GMF and MLP together)
4.Use df_Metadata embeddings for User and Item cold start recommendations(Like category ,author)

Other alternatives to the NCF include a) autoencoder-based ones, such as CDL and b) deep hybrid models for recommendation.