Rating Prediction Report

Mustafa Sepen

The Problem

Recommender systems help people find items of interest by making personalized recommendations according to their preferences. For example, a recommender system can make personalized recommendations of items such as movies, books, hotels, or music to people. In order to model users' preferences their past interactions such as product views, ratings, and purchases are used. In the recommender systems area of research (which is a subfield of machine learning and information retrieval) different algorithms have been developed which are currently being used by many large companies. In this project you will implement neighborhood based collaborative filtering (NBCF) algorithms in order to make predictions for movie ratings of people.

In this project, we tried to solve problem that problem step by step:

Input Reading and holding data:

Firstly, we handled the input reading part. We read input data via ifstereams functions. Then we tried holding data on RAM. Our first attempt we used the linked list to hold all the data. Then we calculated the top10 user and top 10 movie and printed with for loop. But we realized that using linked list in our algorithm, running time was too much. Then, we decided to change our method. So, we decided to using pointers and arrays. We created two classes. Those are: UserList and MovieList.

In UserList, we create 35184 objects. Those objects represent every user. And every object has an array which size 7331. This array holding movieID and Rank.

In MovieList we create 7331 objects. Those objects represent every movie. And every object has an array which size 35184. This array holding userID and Rank.

Those two class’s objects have Rating count integer. In every data we increase this movie and user’s rating count. Then we read those rating counts and find top 10 user and top 10 movie.

Apply UBCF/IBCF and making prediction:

We used cosine similarity for finding similar users and movie. Firstly, we used User Based Collaborative Filtering for making prediction, but our RMSE was very high. Then, we decided to change our similarity formula. We tried Pearson Similarity. In this formula, our RMSE didn’t decrease and running time increased too much.

Then we used cosine similarity again. But, we made a change. We used both User Based Collaborative Filtering and Item Based Collaborative Filtering. We could apply UBCF and IBCF via using two classes. We calculated top 20 similar user via Userlist class, and we calculated top 20 similar movie via MovieList.

For prediction, we took top similar user’s ranks in proportion to the similarity, top similar movie’s ranks in proportion to the similarity, movie average rank, user average rank. We sum it all up and divide it into four. This results our predicti