**Information Retrieval**

Assignment 3

Recommender System using Collaborative Filtering

Developers:

Smit Shah 2017A7PS0080H Saarthak Jain 2017A7PS0083H

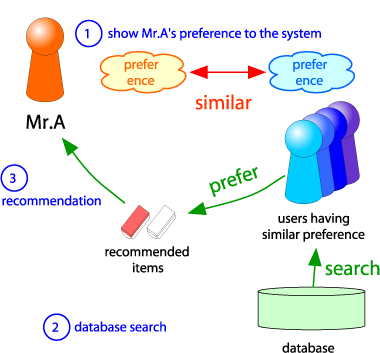
Dhruv Gupta 2017A7PS0108H Ayush Laddha 2017A8PS0717H

**Objective:**

This assignment involved the development of a mini recommender system like the one which Netflix uses, but on a smaller dataset. The aim of this assignment was to predict movie ratings of an unknown user given his history and ratings of other users.

We are working on a dataset which contains contain 1,000,209 anonymous ratings of approximately 3,900 movies made by 6,040 MovieLens users who joined MovieLens in 2000.The objective is to predict the rating of a user on a movie by item-item collaborative filtering and user-user collaborative filtering.

**Design Architecture**



*An illustration showing the major steps in Collaborative Filtering*

The main steps involved in user-user collaborative filtering involve finding similar users like target user X and using cosine similarity between them to predict the rating for target user X based on the rating of similar users.

Item-item collaborative filtering uses a similar approach but finds similarity between movies instead of users.

We have considered cosine distance as the distance measure to find similarity.

**Data pre-processing:**

The first step before creating a dataframe involved normalising the values given in ‘ratings.dat’.

We normalised the values to ensure that strict and generous raters are treated fairly.We subtracted the mean rating from every user rating to ensure this fairness.Those users who did not rate a movie X , we avoided subtracting the mean.

We use a dictionary of dictionary to store ratings of all movies given by a user and similarly another dictionary for storing ratings of all users given to a movie.

utility\_matrix : {movieid:{userid:rating}}

util\_user : {userid:{movieid:rating}}

Using a dictionary is similar to making a lists of lists which significantly saves memory rather than creating a 2D sparse matrix of zeros.

**Prediction of Ratings:**

Once we have our two dictionaries, we can use it for predicting ratings based on user-user and item-item collaborative filtering methods.

For every unknown rating in our testing data, we compute first the similarity of all movies in item-item collab filtering, take out the ones having positive similarity and compute the predicted rating using weighted average.



***sij***… similarity of items ***i***and ***j***

***rxj****…*rating of user ***u*** on item ***j***

***N(i;x)****…* set items rated by ***x*** similar to ***i***

Where

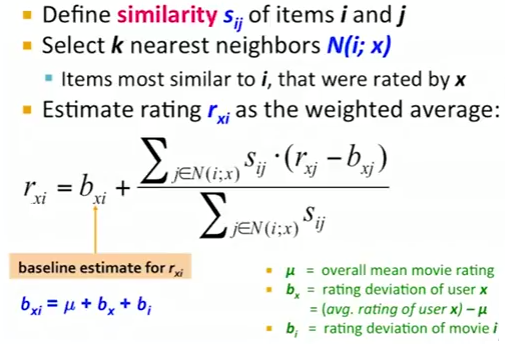
***sij***… similarity of items ***i***and ***j***

***rxj****…*rating of user ***u*** on item ***j***

***N(i;x)****…* set items rated by ***x*** similar to ***i***

and

**Baseline approach:**



Baseline approach tries to incorporate the strict and generous raters scenario, by adding the bias of each user x and movie. This improves our model to some extent because strict and generous raters are handled precisely.

**Results:**

Once we have our predictions , we use root mean square error to find the error in the model.

For running a test-dataset of size 10, we found an error of

Time for initialising: 8.539809703826904

Time for predicting test dataset: 34.25682711601257

RMSE: 0.9279232654244081 mae: 0.6645493923819997

RMSE Baseline: 0.6875922388001985 mae\_baseline: 0.2348152462094941

RMSE\_user: 10.519693298130246 mae\_user: 0.6489139909248206

For a 50-sized dataset:

Time for initialising: 8.421701908111572

Time for predicting test dataset: 193.31594491004944

RMSE: 1.0133719142929187

mae: 0.42760351363490307

RMSE Baseline: 0.7904798019118929 mae\_baseline: 0.12305976593812644

RMSE\_user: 56.265029183238205

mae\_user:

0.38543421607778655