TERM PROJECT: RECOMMENDATION SYSTEM

ESRA NUR ERSAN

HANYANG UNIVERSITY

Department of Computer Science &

Software Engineering

9325020182

Collaborative Filtering

In our daily lives we all watch movies or dramas,right? But sometimes we have a lot of difficulties to choose which one we should watch and that's exactly what recommendation system is for. These recommendation systems predict what we could like according to our previous behaviors. One of the simple models for these systems are collaborative filtering models that based on people like similar things or has similar taste. There are 2 famous techniques for this algorithm called memory based approach and model based approach.Memory based approach finds similar users based on cosine or pearson similarity and take the weighted average of ratings.This approach is non scalable but easy to create and the outcomes are pretty explainable.However,the model based approach use machine learning algorithms like PCA,Matrix Factorization,Neural Nets,SVD etc. to find ratings of unrated items.Even though the results are not accurate enough due to hidden factors,this approach can deal with missing data.From my point of view,memory based approach is more convenient to go with so I implemented it in my project.

DETAILED DESCRIPTION OF MY CODE

ReadFile() : For this function I have created a dictionary that will keep the rating information of user.Since our dataset includes user id,item id ,rating and timestamp we can read the lines except timestamp.Also,for the missing items(user or item).

UserInfo() : Check every user within a loop to save their information according to their attributes.

CosineSimilarity() : To find the items correlated by both users that we are trying to find the cosine similarity for.If it doesn't meet the threshold (min_correlation) return similarity 0.

UserSimilarity() : To calculate similarity and keep the calculated one in user_simillarity {} dict

NearestNeighbors() : Check situation among users to find their neighbors by copying the key of the top neighbors to the user's neighbor

Prediction(): Checking through the neighbors if user has no neighbors return None if neighbor has rated item include the neighbor for prediction.

Recommender() : recommendations and result for each set

How to execute my program

#/[recommender.py] [inputfile.txt] [training_file] [test_file]