Recommendation Engine

Recommendation Engine - Item Based Collaborative Filtering

Recommender systems are active information filtering systems which personalize the information coming to a user based on his interests. These systems are used widely for recommending movies, articles, restaurants, places to visit, items to buy etc.Industry at large uses two different types of recommendation Engines - Content and Collaborative filtering based. This project is all about Collaborative filtering. Further Collaborative filtering is further divided into 2 type - User based and Item based.

This project deals with item based collaborative filtering algorithm. In item-based collaborative filtering, we consider set of items rated by the user and computes item similarities with the targeted item. Once similar items are found, and then rating for the new item is predicted by taking weighted average of the user's rating on these similar items.

Library

```
## Loading required package: SnowballC
```

Data

Implementing Item based Collaborative Filtering

This involves two steps: Calculating Similarity Function Predicting the targeted item rating for the targeted User.

Calculate item Similarity

We will calculate the similarity between co-rated items. We use cosine similarity or pearson-similarity to compute the similarity between items. The output produced by this is similarity matrix between Items.

Recommending

For recommending movies we are using the above similarity matrix. As first step, separate the non-rated movies and a weighted matrix is created by multiplying user similarity score (item_sim[,6]) with ratings given by other users.

In this most important step, we first predict the items which the user is not rated by making use of the ratings he has made to previously interacted items and the similarity values calculated in the previous step. First we select item to be predicted, we predict the rating for that particular movie by calculating the weighted sum of ratings made to movies similar to movie rating which were not rated or has NA. i.e We take the similarity score for each rated movie which was not rated or has NA and multiply with the corresponding rating and sum up all the for all the rated movies. This final sum is divided by total sum of similarity scores of rated items w.r.t the movie which was not rated or has NA.

```
## Warning in if (is.na(userRatings[, i])) {: the condition has length > 1 and
## only the first element will be used

## Warning in if (is.na(userRatings[, i])) {: the condition has length > 1 and
## only the first element will be used
```

```
## Warning in if (is.na(userRatings[, i])) {: the condition has length > 1 and
## only the first element will be used

## Warning in if (is.na(userRatings[, i])) {: the condition has length > 1 and
## only the first element will be used

## Warning in if (is.na(userRatings[, i])) {: the condition has length > 1 and
## only the first element will be used
```

	Just My Luck	Lady in the Water	Snakes on a Plane
Claudia Puig	3.0	2.787428	3.5
Gene Seymour	1.5	2.787428	3.5
Jack Matthews	NA	2.787428	4.0
Lisa Rose	3.0	2.787428	3.5
Mick LaSalle	2.0	2.787428	4.0
Toby	NA	2.787428	4.5

	Superman Returns	The Night Listener	You Me and Dupree
Claudia Puig	4.0	4.5	2.5
Gene Seymour	5.0	3.0	3.5
Jack Matthews	5.0	3.0	3.5
Lisa Rose	3.5	3.0	2.5
Mick LaSalle	3.0	3.0	2.0
Toby	4.0	NA	1.0

Conclusion

Calling above function gives the predicted values not previously seen values for movies in the original data set movie_ratings. This we can sort and recommend the top items.