Data 643 - Project 2

Recommendation Systems

Team Members: Angus Huang, Nathan Cooper, Pavan Akula

06/18/2018

**Prompt:**

For assignment 2, start with an existing dataset of user-item ratings, such as our toy books dataset, MovieLens, Jester [http://eigentaste.berkeley.edu/dataset/] or another dataset of your choosing. Implement at least two of these recommendation algorithms:

• Content-Based Filtering

• User-User Collaborative Filtering

• Item-Item Collaborative Filtering

Organization:

1. Each member builds a different filtering system.
2. Chose MovieLens small Data Set (1 Mb): https://grouplens.org/datasets/movielens/
3. Meet regularly 3 times since project started.

Assignments:

* Angus: User-User Collaborative Filtering
* Pavan: Content-Based Filtering
* Nathan: Item-Item Collaborative Filtering

**Introduction:**

For this project, the team divide the work by each member building a different style of recommendation system. We then subsequently meet in three separate occasions to discuss our progress and share our findings. By focusing each recommendation system, we are able to deep dive into very detailed level of building each system. We then discussed our results and gave advice on how to proceed.

**Data:**

We selected the small dataset on the MovieLens website, <https://grouplens.org/datasets/movielens/>. The Exploratory data analysis in the ‘Item-Item’ file shows that the data is sparse, as to be expected in a user-item matrix. We also find that most of the movies have fewer than 50 ratings with a mode around 10 viewings. The users had an average of 149 ratings, however there was a very strong right-skew with some ‘power-users’ having ratings numbering in the thousands. That said, 95% of users had at least 20 ratings. We handled the preprocessing of the data separately based on the requirements of the analysis that we used.

**Methodology:**

The item-item collaborative filtering used both cosine similarity and Euclidean distance to measure similarity in movies. The data was trimmed to prevent bias due to low numbers of ratings per movie and per viewer. Based on the exploratory data analysis, these limits were set at greater than 5 ratings per movie, and greater than 20 views per user. The data was scaled by a simple linear transformation so the 1-5 scale was set to 0 - 𝛑/2 to be more amenable to cosine. Note in this context, 0 means not rated. We find that the cosine similarity gave results that seemed to thematically match e.g., The ‘burbs (1989) matches with So I Married an Axe Murderer (1993), both are Dark Comedies with suspense and horror elements to their plots. The Euclidean distance did not return results that matched as well.

For user-based recommendation system, we use one of the few python recommendation packages, “surprise” to calculate similarities between users. The package also has built-in “grid search” algorithm for searching the best parameters for the best RMSE and MAE values. We have learned SVD, single value decomposition, is very useful in dimension reduction that can help for large matrix manipulation.

For content based filtering we have used *Term Frequency Inverse Document Frequency* (tfidf), a popular technique to evaluate how important a word is to a document in a collection or corpus. In our case, a movie is classified into a various genre.

For example: Movie *Toy Story (1995)* is classified into *adventure, animation, children, comedy, and fantasy.* This movie may be recommended for users who want to watch *adventure* or to a family with *children*. In short, it can be recommended to users with various tastes.

**Conclusions:**

We have used three different measures to identify the similarity between movies *Cosine similarity, Pearson's Correlation Coefficient* and *Euclidean Distance*. Based on the results for content-based recommendation system, Euclidean distance similarity measure is not suitable. Pearson Correlation and Cosine similarity methods yield better results.