DATA 643: Recommender Systems

Final Project: Proposal

Walt Wells, Summer 2017

Overview

The DATA 643 Final Project will serve as an opportunity to study further and implement a recommender system at a medium scale.

I propose to implement a recommender system that utilizes the full Book-Crossing dataset (see #Data) . I will take an iterative approach to modeling and matrix factorization (see #Workflow) using a research grant on an available cloud resource (see #Resources). If there is time, the project will culminate in a user dashboard for rating and receiving book recommendations based on the final model.

Data

For this project we will use the full Book-Crossing Dataset.

From the site: "Collected by Cai-Nicolas Ziegler in a 4-week crawl (August / September 2004) from the Book-Crossing community with kind permission from Ron Hornbaker, CTO of Humankind Systems. Contains 278,858 users (anonymized but with demographic information) providing 1,149,780 ratings (explicit / implicit) about 271,379 books."

Publication Citation: Improving Recommendation Lists Through Topic Diversification, Cai-Nicolas Ziegler, Sean M. McNee, Joseph A. Konstan, Georg Lausen; Proceedings of the 14th International World Wide Web Conference (WWW '05), May 10-14, 2005, Chiba, Japan.

The csv .zip file has been downloaded and uncompressed and the resulting csvs are in a "BX-CSV-DUMP" folder. Since we will only use Collaborative Filtering models, we won't be doing anything with the user data - we'll just refer to the user IDs abstractly. We will utilize:

- The Utility Matrix: BX-Book-Ratings.csv
- For Images and Titles: BX-Books.csv

Deliverables

- *Definite:* A github repository with all data, code, interpretation, and visualization needed to understand and run the model
- *If time:* A ShinyR or Flexdashboard for a user to interact with the RecSys by rating n_1 items and receiving n_2 recommendations

Workflow

1) Data Prep
    - Setup resources / configure work environment
    - Import, clean data
    - Center and Scale
    - Review, select, and implement imputation technique
    - Alternatively, Binarize
2) Matrix Factorization
    - Review, select, and implement a matrix factorization technique

- Sample and test performance characteristics
- Initial runs may be samples of full data for ease of use

3) Evaluation
   - Determine goals of the system
   - Implement a technique(s) to achieve goals
4) Tuning and Optimization
   - Review, select, and test methods for improving performance
5) Finalize Model
   - Iterate over 2-5
   - Final factorization, factorized export
6) If Time: User Dashboard
   - Dashboard requests signals from user indicating book preferences
   - Develop method to deliver options to user (eg: pick one of 5, n times)
   - Develop method for recommending based on user input and factorized matrix. Be sure it considers a 'business goal' instead of just 'best ranking'
   - Push into production

Resources

Data Cleaning, NA Imputation, Matrix Factorization, Modeling, and Tuning will be done on a VM as part of an allocation grant from the Open Science Data Cloud. The OSDC offers services similar to commercial cloud providers like AWS, Azure and Google Compute, but is designed to serve the 'long tail' of the data science community by providing allocation grants to researchers in need of resources. I help manage the operations of the OSDC as part of my work with the Open Commons Consortium.

When stored as a sparse matrix, our Book Crossing Dataset is not too large and can be managed in the VM using ephemeral storage. We don't need to utilize block or object storage. I will port forward a Jupyter notebook running an R kernel through a proxy server and work in a browser on my local machine. Github is used to manage the code.

Future Studies (Out of Scope)

- Importing and harmonizing new user signals with existing dataset
- Providing a framework for user feedback on their recommendations
- Scalable Spark cluster for continuous iteration with full dataset

Near Term Questions

- If I'm not setting up infrastructure to manage incoming recommendataions and incorporate into the larger dataset, how to handle new recommendations? I roughly understand, but center/scale, any coercions done to user matrix will need to be reapplyed for correct comparison? Should I implement some kind of clustering/similarity study to manage new recommendations instead and compare to predictions for a similar users?
- Currently setup study environment in an m3.xlarge ubuntu VM (8cpu, 24GiB RAM, 10GiB Ephemeral Storage). Will this be enough for factorization?
- How big will my factored matrix be? Will it be small enough that I can export easily and run an Rshiny dashboard with it?