Project Proposal: Recommendation System

Kevin Zhang, Ken Sun, Austin Mease

October 24, 2015

1 Overview

1.1 Motivation

We chose this topic because recommendation systems are applicable to many different kinds of systems, especially on the web. Many retailers can benefit from recommendation systems to increase user satisfaction by quickly matching customers with the products that they will likely be most interested in. This area of machine learning and pattern recognition has recently emerged as a very hot and lucrative topic. In fact, Netflix held a competition where they awarded 1 million dollars to the team that could increase the accuracy of their recommendation system by 10 percent or more. We wish to explore this topic in detail in order to gain knowledge of the techniques used for designing recommendation systems.

1.2 Initial investigations

We have learned that collaborative filtering is the most widely used strategy for designing modern recommendation systems. By analyzing user's past preferences and behaviors, collaborative filtering can look at relationships between users and products to identify potential user-item interactions. After establishing these associations, the system can easily produce tailor made suggestions for each user. The two most popular techniques for collaborative filtering include neighborhood methods (item and/or user based) and latent factor models (matrix factorization). We wish to explore both methods in our project.

We have also identified some interesting challenges in designing recommendation systems. In particular, the cold start problem in collaborative filtering models means that the system will have difficulty drawing inferences for users and items for which there is a paucity of data. However, some systems may leverage implicit feedback, in which the system deduces user preferences indirectly through additional related data like search, browsing, and purchase history. The system may also supplement explicit data, like a user's item ratings, with implicit data, like a user's demographics, to make suggestions more accurate. Recommendation systems must also deal with the sparcity of rating matricies. For instance, a single user may have only rated 25 out of a catalog of 100,000 products. The flexibility of matrix factorization models can help deal with such issues.

Another interesting variation we have found deals with pre-existing user and item biases that are independent of any user-item interaction. For exam-

ple, some users may be more critical than others or public opinion may drive the perception of a particular item. To deal with this, a recommendation system may attribute some portion of a rating to biases. The system can then correct for these biases so that it only considers the portion of the rating that contributes to a user-item interaction. A simple approximation considers global average rating, item bias, user bias, and user-item interaction. A related extension deals with differences in these factors over time. For instance, item/user biases and user preferences may change over time. By considering each of these factors as a function of time, a system a can effectively address temporal changes such as items going in and out of popularity and evolutions in a user's tastes. Again, a matrix factorization approach lends itself to this problem because it can accommodate these types of factors in the data. As such, we are highly considering incorporating this problem as an extension in our project.

1.3 Machine Learning and Data Analysis Tasks

We will present students with a ratings matrix where the rows represent users, columns represent items, and individual entries represent specific ratings. The main machine learning technique we will use is collaborative filtering, which bases recommendations on the similarities between different users or different items. Due to its simplicity and intuitiveness, we will begin our project by presenting a neighborhood-based collaborative filtering approach. We will then discuss a more effective method using latent factor models, or matrix factorization. We might also include extensions that deal with biases, temporal dynamics, and dimensionality reduction of the ratings matrix. We will ask students to build a predictors using each method, find the rate of error of their predictors, and then explore the benefits of each method. We may also ask students to perform cross validation.

2 Core Concepts

2.1 Concepts and Tools

To discuss item-based collaborative filtering, we will introduce the idea of similarity measures to identify items that are similar to our target item. Since there are many different similarity measures including cosine-based, pearson-based, and adjusted cosine-based similiarity, we will choose one for students to implement. After creating such a model, we can make a prediction using a weighted sum of similar items to the target item that the user has also rated.

For latent factor models, we expect to use matrix factorization techniques to decompose a user-item matrix into separate user feature and item feature matrices. This decomposition yields a mapping between users and items onto a latent feature space. Latent features may not have obvious meanings but can characterize some underlying properties of the users and items, revealing abstract characteristics (eg. character complexity, comedic value of the movie, etc.) of the data that may not be extractable through other methods. We can then approximate a particular rating (and predict unknown ratings) by using the dot product of a user feature vector with an item feature vector.

In order to avoid the issue of sparse ratings matrices, we will perform the

decomposition using only known ratings while using factorization methods like Stochastic Gradient Descent or Alternating Least Squares to avoid overfitting to the training data. Alternatively, we may also use traditional SVD and PCA to reduce the dimsionality of ratings matrices that are sparse. Using PCA, we can also project the ratings matrix onto a smaller dimensional subspace that is more manageable and easier to visualize.

3 Resources

3.1 Papers/Tutorials

- http://www2.research.att.com/ volinsky/papers/ieeecomputer.pdf
- http://arxiv.org/pdf/1503.07475.pdf
- http://www.slideshare.net/sscdotopen/latent-factor-models-for-collaborative-filtering
- http://grouplens.org/site-content/uploads/Item-Based-WWW-2001.pdf
- http://www.quuxlabs.com/blog/2010/09/matrix-factorization-a-simple-tutorial-and-implementation-in-python/

3.2 Datasets

- https://gist.github.com/entaroadun/1653794