Book Recommender Status Report

Chris Barnett - cjbarnet@ucsc.edu, class 142

Introduction

The goal of this project is to assess the effectiveness of several collaborative filtering techniques, including one I have developed myself. The specific scenario this experiment will address is the recommendation of books to users based on their ratings of previous books. A successful algorithm should be able to accurately predict specific books that a user will like based on his/her prior preferences and the preferences of other users. The efficiency of the algorithm is also an important factor because a collaborative filtering algorithm will be far more useful if it can scale to the millions of items and users that are commonly dealt with in a modern web context.

Methodology/Plans

I will use the Book Crossing Dataset for this project, which is located at http://www.informatik.uni-freiburg.de/~cziegler/BX/ . The raw dataset contains 92,107 users, 270,170 books and 1,031,175 ratings, but I will use a cut-down version of the dataset that excludes users who have rated less than 6 books and books that have been rated by less than 2 users. That will leave 10,736 users, 40,183 books and 203,572 ratings.

For this machine learning problem, users are the instances and their ratings are the features. The targets we want to learn are the ratings a user would give to the books they have not rated yet. There is thus no hard line between which user ratings are features and which user ratings are the targets on which to train our algorithm. In order to facilitate training then, we will artificially divide up the user ratings data into features and targets by randomly dividing the data we have about each user in two; half will be considered features and half target values. We will similarly divide our data across users into 40% training, 40% validation and 20% testing data.

I will use the Lenskit Recommender Framework to support my experiment. I will run its implementation of the Slopeone and K-Nearest-Neighbor algorithms over my dataset and use the results to evaluate the relative effectiveness of my own algorithm.

My own algorithm is user-based and, in the simplest case, calculates the similarity factor between users by taking the dot product of their feature vectors. A book B's score with respect to user U_k is given by:

$$Y_{B,k} = \sum_{i}^{N_u} R_{B,i} S_{i,k}$$

Where:

$Y_{B,k} =$ The score of Book B with respect to User k
$N_u =$ The number of users
$R_{B,i} =$ User i's rating of Book B (or zero if User i has not rated Book B)
$S_{i,k} =$ The similarity factor between User i and User k.

The model can be made more complex by scaling various terms by power factors.

So, for example, instead of calculating the similarity between users by a simple dot product, an expression like the following would be used:

$$S_{i,k} = \left[\sum_{j}^{N_b} (R_{j,i} R_{j,k})^{\alpha} \right]^{\beta}$$

Where:

$S_{i,k} =$ The similarity factor between User i and User k.
$N_b =$ The number of books
$R_{j,i} =$ User i's rating of Book j (or zero if User i has not rated Book j)
$R_{j,k} =$ User k's rating of Book j
$\alpha =$ A scaling factor that needs to be learned. It acts as an indicator of the relative importance of two users' level of agreement over a single book on their overall similarity factor.
$\beta =$ Another scaling factor that needs to be learned. It affects the degree to which ratings are scaled by their user's similarity with the target user.

So there will be two parameters to learn: α and β.

I will compare the relative success of my algorithm against the Slopeone and KNN algorithms by evaluating each algorithm's accuracy at predicting the order in which a user would rank the "target" half of their rated books. Remember that each user's ratings will be split into two halves. One half will be the set of features that will be run through the machine learning algorithm. The other half will be the target values which the algorithm will need to predict. The output of each of the three algorithms (Slopeone, KNN and my own) will be a score for each book not in a user's feature set. These scores will provide a prediction of the order in which a user would rank the books in their target set. I will use Kendall's tau coefficient to calculate the

correlation between a user's actual ranking of their target set of books and the ranking predicted by each algorithm. The overall accuracy of an algorithm will be given by the average over all users in the testing set of the correlation between a user's actual ranking of their target set and the ranking predicted by the algorithm.

Progress/Problems

One of the greatest challenges I have faced so far is getting my data into a format that would be accepted by the Lenskit package. This was made difficult because the dataset uses string ISBN identifiers for books, but the Lenskit library requires integer identifiers. I was able to solve this problem once I managed to get my data into MySQL (which was also quite challenging) by creating a numeric id for every book using an auto-incrementing id on the books table when I created it. I then made the numeric ID into the primary key.

Another challenge I foresee is figuring out how to get the output I want out of the Slopeone and KNN algorithms in Lenskit. I have found Lenskit's documentation to be quite poor and so it may take a lot of experimenting and code exploration to figure out how it works well enough to use it in the way I want to.

I am sure I will also encounter various miscellaneous unforeseen challenges relating to getting different technologies to talk to each other and finding (and figuring out how to use) libraries to do such things as calculate the Kendall's Tau rank correlation coefficient between two rankings of books.

I have not yet figured out exactly what approach I will use to learn the parameters α and β for my algorithm. I suspect that I will need to settle for a relatively primitive approach because I don't think I will be up to implementing something like second degree approximation for this novel use case on my own. At any rate, this is definitely a major challenge that still remains to be solved.

At this stage I have my data ready in MySQL and in a format that is accepted by the Lenskit library. I have begun learning how to use Lenskit and planning how I will implement my algorithm on top of it as well as how I will get the results of the three algorithms I plan to use into a form that can easily be compared.