Assignment Instructions: User-User Collaborative Filtering

Overview

In this assignment, you will implement user-user collaborative filtering using a spreadsheet. (We are using only basic spreadsheet operations that would work with Google's free Drive-based spreadsheet, Excel, or any other common spreadsheet program. We'll help you find the correct operations.)

Instructions

Part 1 - Without Normalization

First, you will implement user-user collaborative filtering without normalization.

1. Start by downloading the starting spreadsheet. This is a 25 user x 100 movie matrix of ratings selected from the class data set. The spreadsheet has three sheets in it (this is not supposed to be an exercise in spreadsheet tricks; as a result, we've already given you a significant start). 1) The first sheet is a ratings matrix with movies as rows and users as columns, 2) The second sheet is a ratings matrix with movies as columns and users as rows, and 3) The third sheet is the start of your correlations matrix.
2. Open the sample matrix in your favorite spreadsheet program. Note that the matrix contains a significant number of missing values -- do not replace these with zeroes, they are correctly missing.
3. Complete the user-by-user correlations matrix. To check your math, note that the correlation between users 1648 and 5136 is 0.40298, and the correlation between users 918 and 2824 is -0.31706. All correlations should be between -1 and 1, and the diagonal should be all 1's (since they are self-correlations).
4. Identify the top 5 neighbors (the users with the 5 largest, positive correlations) for users 3867 and 89. For example, if the target user were #3712, the closest neighbors are 2824 (corr: 0.46291), 3867 (corr: 0.400275), 5062 (corr: 0.247693), 442 (corr: 0.22713), and 3853 (corr: 0.19366). Don't forget to exclude the target user (corr: 1.0) from your possible selections.
5. Create a new worksheet in your spreadsheet, and use it to compute the predictions for each movie for users 3867 and 89 by taking the correlation-weighted average of the ratings of the top-five neighbors (for each target user) for each movie. The formal formula for correlation-weighted average is $\frac{\sum_{n=1}^{5} r_n w_n}{\sum_{n=1}^{5} w_n}$. Remember, you will need to make sure that your weight for each contributed rating is the user-user correlation when that neighbor has rated the movie, but 0 when the neighbor has not rated the movie).
6. Submit 12 values for this assignment as indicated below. You will be submitting the top three movie IDs and the predicted ratings (to three decimal places) for each user. In a real recommender system, we'd be excluding movies the user has already rated, but do not do this here. Indeed, you should look to see what the user's rating (if any) is for the top-recommended movies. For example, if the user ID was 3712, the correct submission would be:

Top Movie: 641 Prediction: 5.000

2nd Movie: 603 Prediction: 4.856

3rd Movie: 105 Prediction: 4.739

And if the user ID were 3525, there would be a three-way tie among:

Movies: 238, 194, and 38 (all with a prediction of 5.000)

A few tips we hope you'll find helpful:

1. Your calculations will be easier if you use some of the built-in functions that spreadsheets have to help. These include SUMPRODUCT (which computes dot-products) and SUMIF (which can be used to compute the total weight). It is also helpful to understand two special paste functions: paste values (which copies cell values without copying the underlying formulas) and paste transpose (which can help re-orient your data).
2. Your intermediate goal should be to have a sheet for each target user with a user-by-rating table and a set of user weights (for the top-five neighbors) to allow you to compute all 100 predictions for the target user.
3. Once you have all the predictions, you can always copy them (copy the values) and sort them to find the top three results.
4. Don't worry about values that result in division by zero -- those are cases where no neighbors rated the movie.
5. It will take 95% of the work to get a single instance of this correct. Use the test cases to make sure you have your computations correct. Then compute the cases you'll submit.

Part 2 - Normalization

Next, you will repeat the computation but this time you will normalize the scores.

1. Repeat step 5 from part 1. This time, however, use the normalization formula: $\bar{r}_u + \frac{\sum_{n=1}^{5}(r_n - \bar{r}_n)w_n}{\sum_{n=1}^{5} w_n}$
2. Submit 12 values for this assignment as indicated below. You will be submitting the top three movie IDs and the predicted ratings (to three decimal places) for users 3867 and 89. In a real recommender system, we'd be excluding movies the user has already rated, but do not do this here. Remember, you do not need to re-compute the correlations, just use the existing correlations but normalize the ratings being averaged by subtracting each neighbor's mean rating from each of their ratings (and add the target user's mean back into the total).

For example, if the user ID was 3712, the correct submission would be:

Top Movie: 641 Prediction: 5.900

2nd Movie: 603 Prediction: 5.546

3rd Movie: 105 Prediction: 5.501

Note that it is possible to have movie predictions outside the five-star scale; in this case this is because 3712 rates movies very high to begin with, and his/her neighbors have wider ranges (and think these movies are very high in those ranges).

Also, these top movie predictions are interesting in other ways. The top movie was only rated by one neighbor, but was so highly rated (1.4 above mean) that it stands out.

And if the user ID were 3525, the top three would be:

Top Movie: 238 Prediction: 4.760

2nd Movie: 424 Prediction: 4.663

3rd Movie: 134 Prediction: 4.585

Note how only one of these three is the same as from the top three without normalization! And notice that the (perhaps surprisingly) strong prediction for 134 is again the result of having just a single neighbor's rating (and that being a neighbor who really liked the movie).

NOTE: This assignment is being automatically graded. While we will set a tolerance for rounding, please follow rounding conventions. 4.76449 should be entered as 4.764, 4.76450 should be entered as 4.765.