Homework 3

You should submit a single file containing your code ("homework3.py") to the autograder

All data files used in this homework are the same as those used in Assignment 1. Start by downloading the Assignment 1 files by following the link in the Assignment 1 spec.

Rating prediction

1. Compute the global average rating in the training set, as well as the validation MSE for a model that always predicts that value.
2. Implement the update equations described on Slide 83 of the recommendation lecture slides. The autograder will run the model for a single iteration.
3. Improve upon the above solution (e.g. by running the code for multiple iterations, regularizing differently, etc.). Improvement will be measured on the validation set.

Read prediction

Since we don't have access to the test labels, we'll need to simulate validation/test sets of our own. So, we'll split the training data ('train_Interactions.csv.gz') as follows: (1) Reviews 1-190,000 for training (2) Reviews 190,001-200,000 for validation (3). For the first assignment, you'll want to upload to Gradescope for testing only when you have a good model on the validation set. This homework will help you to build such a validation set correctly, which will significantly speed up your testing and development time.

4. Although the runner builds a validation set, it only consists of positive samples. We also need examples of user/item pairs that *weren't* read. For each (user,book) entry in the validation set, sample a negative entry by randomly choosing a book that that user hasn't read.
5. The Assignment 1 baseline file provides a simple implementation (which is repeated in the homework stub). Evaluate the performance (accuracy) of the baseline model on the validation set you have built. Modify the strategy (in the function "improvedStrategy") to improve upon its performance, e.g. by setting a different popularity threshold.
6. A stronger baseline than the one provided might make use of the Jaccard similarity (or another similarity metric). Given a pair (u, b) in the validation set, consider all training items b ′ that user u has read. For each, compute the Jaccard similarity between b and b′, i.e., users (in the training set) who have read b and users who have read b ′. Predict as 'read' based on the condition provided (a combination of Jaccard similarity and a popularity threshold). Your solution will be marked correct if it matches the reference 95% of the time, so there is room for some error.

Category prediction

The stub contains code to build training/validation sets consisting of 9,000/1,000 reviews (a small fraction of the complete dataset). We want to build features that represent common words. The stub starts by removing punctuation and capitalization, and finding the 500 most common words across all reviews ('review_text' field) in the training set. See the 'text mining' lectures for code for this process.

7. Build bag-of-words feature vectors by counting the instances of these 500 words in each review. The 501st dimension should be the offset term ("1")
8. Try to improve upon the performance of the above classifier by using different dictionary sizes, or changing the regularization constant C passed to the logistic regression model.